Knowl Inf Syst (2015) 43:103–126
DOI 10.1007/s10115-013-0715-x

REGULAR PAPER

Graph-based local concept coordinate factorization

Ping Li · Jiajun Bu · Lijun Zhang · Chun Chen

Received: 20 October 2011 / Revised: 16 May 2013 / Accepted: 29 November 2013 /
Published online: 20 December 2013

Abstract Ubiquitous data are increasingly expanding in large volumes due to human activities, and grouping them into appropriate clusters is an important and yet challenging problem. Existing matrix factorization techniques have shown their significant power in solving this problem, e.g., nonnegative matrix factorization, concept factorization. Recently, one state-of-the-art method called locality-constrained concept factorization is put forward, but its locality constraint does not well reveal the intrinsic data structure since it only requires the concept to be as close to the original data points as possible. To address this issue, we present a *graph-based local concept coordinate factorization* (GLCF) method, which respects the intrinsic structure of the data through manifold kernel learning in the warped Reproducing Kernel Hilbert Space. Besides, a generalized update algorithm is developed to handle data matrices containing both positive and negative entries. Since GLCF is essentially based on the local coordinate coding and concept factorization, it inherits many advantageous properties, such as the locality and sparsity of the data representation. Moreover, it can better encode the locally geometrical structure via graph Laplacian in the manifold adaptive kernel. Therefore, a more compact and better structured representation can be obtained in the low-dimensional data space. Extensive experiments on several image and gene expression databases suggest the superiority of the proposed method in comparison with some alternatives.

Keywords Manifold kernel learning · Local coordinate coding · Graph Laplacian · Concept factorization · Clustering

P. Li (✉) · J. Bu · C. Chen
Zhejiang Provincial Key Laboratory of Service Robot, College of Computer Science,
Zhejiang University, Hangzhou 310027, China
e-mail: patriclouis.lee@gmail.com

L. Zhang
Department of Computer Science and Engineering, Michigan State University,
East Lansing, MI 48824, USA

1 Introduction

An increasing number of data have emerged in our daily life during the past decades, and it is foreseeable that their quantities will sharply rise up to a much higher level in the near future. To deal with these data with high dimensions usually brings about a big challenge. To this end, there have existed many techniques to solve this issue. Among them, matrix factorization techniques have attracted lots of attention in pattern recognition, data mining, computer vision, etc. In this work, we focus on this very topic. Our basic goal is to learn a better structured data representation, thus alleviating clustering [4] or classification tasks [15]. Generally, a given matrix can be decomposed into two or more components, whose product is expected to be a good approximation to the original matrix. One component is regarded as the vector space spanned by the basis vectors that reflect the semantic data structure, and the others are coefficient vectors encoded by linear combinations of basis vectors.

In many real-world applications, we want to project the original data onto a much lower-dimensional data space, in which the dimension is fundamentally dominated by that of the bases vectors. Principally, there are many methods capable of achieving this goal, such as *principal component analysis* Abdi and Williams (2010) [1,15], *nonnegative matrix factorization* (NMF) Gong and Zhang (2012) [12,19,28], *concept factorization* (CF) [35], and *locality-constrained concept factorization* (LCF) [25]. The obtained data representation is more compact with less redundancy, which is beneficial for the learning algorithms, e.g., linear regression, clustering, or classification. Due to its property of reducing dimensions, matrix factorization techniques are closely related with dimensionality reduction [18] and subspace learning [33] in the field of data mining.

However, most of these methods do not respect the manifold geometrical structure, which can be encoded by graph Laplacian [2,3] in the data space. Specially, LCF is one state-of-the-art algorithm proposed very recently [25], which imposes the locality constraints on the traditional concept factorization. On the one hand, this locality constraint only requires the concept to be as close to the original data points as possible, and each sample is approximated by the linear combination with its salient data points, which fails to reveal the intrinsic data structure. On the other hand, the update rules of LCF do not adapt to data matrices with negative entries, which limits its applicability. To address these issues, we in this paper present a novel matrix factorization algorithm, called *graph-based local concept coordinate factorization* (GLCF). Extensive experiments on several real-world databases validate the effectiveness of the proposed GLCF approach.

It is worth highlighting the contributions of this work as follows.

- Our proposed method considers the manifold geometrical structure in local concept coordinate factorization via graph-based kernel learning in the warped *Reproducing Kernel Hilbert Space* (RKHS), which can reflects the underlying geometry of the data.
- A more generalized multiplicative update algorithm for GLCF is developed, so as to handle arbitrary input data matrix, i.e., regardless of its entries being positive or negative.
- Since GLCF is essentially based on the local coordinate coding and concept factorization, it naturally inherits many advantageous properties. More importantly, through taking into account the intrinsical data structure via graph-based kernel learning in the kernel space, GLCF not only enhances the local coordinate coding ability but also provides better structured data representation, which facilitates the succeeding learning tasks.

The remainder of this paper is organized as follows. Section 2 briefly reviews some related work toward matrix factorization and manifold learning. Section 3 introduces the proposed GLCF approach involving the manifold kernel learning, and a generalized update algorithm is also presented. Experimental results are reported in Sect. 4 with detailed analysis. Finally, we provide some concluding remarks in Sect. 5.

2 Related work

In this section, we review some related matrix factorization approaches in the first part and manifold learning methods in the second part.

Given one specific pattern or input data matrix, it is believed that either of them reside on a data space with much lower dimensions. In other words, an observation can be regarded as a linear or nonlinear combination of just a few hidden or latent variables. The well-known nonnegative matrix factorization (NMF) technique often yields a sparse data representation [19], which is encoded by merely a small fraction of components. Its extended version concept factorization (CF) [21,35] is different from NMF, since each base of CF is simply a linear combination of all data points, each of which is the linear combination of the data centers. Previous studies have shown that NMF learns a parts-based representation allowing only additive combinations, and CF is capable of revealing the desired semantic concepts on any data regardless of the signs (i.e., it works on both positive and negative data). To this end, many extensions or refinements on NMF and CF have emerged, such as convex NMF [32], quadratic NMF [37], robust NMF [39], efficient NMF methods using the gradient method [13], discriminative concept factorization [17], and sparse concept coding [8]. As mentioned earlier, the intrinsic dimension is usually much smaller than that of the original data, and many existing matrix factorization methods (e.g., NMF and CF) often give rise to a compact representation and a sparse coding of the data [5,16]. In [6,9], authors employed the locality preserving property in nonnegative matrix factorization, which uses the Euclidean distance [6] or KL-divergence [9] to evaluate the similarity on the hidden topics. Ding et al. [11] introduced an orthogonal nonnegative matrix-factorization-based clustering method, which makes use of the orthogonal constraints to give more rigorous clustering interpretations. A recently developed learning method named *local coordinate coding* has attracted great attention, in which a nonlinear function can be approximated by a global linear function based on local coding [38]. In particular, each data point is locally approximated by a linear combination of nearby anchor points and the linear weights constitute its local coordinate coding. Liu et al. [25] proposed locality-constrained concept factorization (LCF), which considers the locality constraints in revealing the underlying concepts. But, LCF suffers from some major drawbacks. First, LCF does not consider the manifold geometrical structure in the data space and its locality constraints fail to reveal the intrinsic data structure well. Second, it can only handle nonnegative data in its current form since its update rules fail to handle the data matrix with negative entries. Motivated by this, we propose a novel graph-based local concept coordinate factorization method, which considers the manifold geometrical structure in the kernel space via graph-based learning. Moreover, a more generalized update algorithm that can deal with both positive and negative data entries is developed in our work.

In recent years, much attention has been paid on the situation that data points are drawn from sampling a probability distribution, which supports on or near to a submanifold of the ambient Euclidean space [3,22,40]. Here, a submanifold with d dimensions refers to a subset of a m-dimensional Euclidean space. Actually, it is difficult to fill the human-generated text documents uniformly in high-dimensional Euclidean spaces, which inspires researchers to consider the intrinsic manifold structure when deriving the new low-dimensional data space. To consider the underlying manifold geometry, there are a number of manifold learning methods, such as Laplacian eigenmap [2] and locally linear embedding [27]. All these algorithms share a common characteristic that the nearby points tend to have similar labels, which is often called *local invariance* [14]. Graph-based manifold learning methods have received wide acceptance and gained huge success in various applications, such as clustering [5,6], categorization [7], and ranking [10,34]. In [7], active learning is performed in the manifold kernel space, and the most representative and informative data points are selected by minimizing the expected error. In [6], the graph structure is considered as a regularizer in nonnegative matrix factorization. In [10,34], the manifold structure of the data space is well respected when ranking data points.

3 Our GLCF approach

This section is primarily devoted to our graph-based local concept coordinate factorization approach. First, we briefly describe manifold kernel learning, which reveals the semantic structure in the warped RKHS. Then, the objective functions for KLCF and GLCF are both introduced with a generalized update algorithm. Finally, we summarize the main procedures of GLCF and provide an analysis on its computational complexity.

3.1 Manifold kernel learning

Let $\mathcal{X}$ be a compact domain on a manifold, then a complete Hilbert space $\mathcal{H}$ of functions $\mathcal{X} \to \mathbb{R}$ with the inner product $\langle \cdot, \cdot \rangle_{\mathcal{H}}$ is a RKHS if the point evaluation functionals are bounded by

$$|f(\mathbf{x})| \le C \|f\|_{\mathcal{H}},$$

where C is a constant, $\mathbf{x} \in \mathcal{X}$ is a m dimensional data vector, $f \in \mathcal{H}$. Notice that RKHS here is appropriately warped by a general scheme when applied a manifold or Euclidean space. If we deform the norm $\|\cdot\|_{\mathcal{H}}$, it gives rise to a new RKHS $\tilde{\mathcal{H}}$ with $\tilde{k}(\cdot, \cdot)$ as its kernel. Let $\mathcal{V}$ be a linear space containing a positive semi-definite inner product in its quadratic form, and $S : \mathcal{H} \to \mathcal{V}$ be a bounded linear operator. Define $\tilde{\mathcal{H}}$ as the spaces of serial functions from original $\mathcal{H}$ with a modified inner product, which has been proved to be a RKHS as well [30]. The mathematical formulation is

$$\langle f, g \rangle_{\tilde{\mathcal{H}}} = \langle f, g \rangle_H + \langle Sf, Sg \rangle_{\mathcal{V}}. \tag{1}$$

Given data points $(\mathbf{x}_1, \ldots, \mathbf{x}_n)$, define $S : \mathcal{H} \to \mathbb{R}^n$ to be the evaluation map $S(f) = (f(\mathbf{x}_1), \ldots, f(\mathbf{x}_n))$, $\mathbf{f} = (f(\mathbf{x}_1), \ldots, f(\mathbf{x}_n)) \in \mathcal{V}$, and $\mathbf{g} = (g(\mathbf{x}_1), \ldots, g(\mathbf{x}_n)) \in \mathcal{V}$. Let $\mathbf{M}$ be a symmetric and positive semi-definite matrix, then we have

$$\|Sf\|_{\mathcal{V}}^2 = \mathbf{f}^t \mathbf{M} \mathbf{f}, \quad \langle Sf, Sg \rangle_{\mathcal{V}} = \langle \mathbf{f}, \mathbf{g} \rangle = \mathbf{f}^t \mathbf{M} \mathbf{g}.$$

For the data vector $\mathbf{x}$, we define

$$\mathbf{k_x} = (\mathbf{k}(\mathbf{x}, \mathbf{x}_1), \ldots, \mathbf{k}(\mathbf{x}, \mathbf{x}_n))^T,$$

and then the reproducing kernel in $\tilde{\mathcal{H}}$ is expressed by

$$\tilde{\mathbf{k}}(\mathbf{x}, \mathbf{z}) = \mathbf{k}(\mathbf{x}, \mathbf{z}) - \mathbf{k}_{\mathbf{x}}^T \cdot (\mathbf{I} + \mathbf{MK})^{-1} \cdot \mathbf{M} \cdot \mathbf{k_z}, \tag{2}$$

where the vectors $\mathbf{k_x}$ and $\mathbf{k_z}$ contain a group of kernel functions (e.g., linear kernel, Gaussian kernel), and they are virtually corresponding to $\mathbf{f}$ and $\mathbf{g}$ in the $\mathcal{V}$ space.

Without the loss of generality, we assume n_q data points are utilized to derive the linear space $\mathcal{V}$, then it is readily to rewrite the above formulation in its matrix form as shown by

$$\tilde{\mathbf{K}}_{\tilde{\mathcal{H}}} = \mathbf{K}_{\mathcal{H}} - \mathbf{K}_{\mathcal{H}_{qn}}^T (\mathbf{I}_{\mathcal{H}_q} + \mathbf{M}_{\mathcal{H}_q}\mathbf{K}_{\mathcal{H}_q})^{-1}\mathbf{M}_{\mathcal{H}_q}\mathbf{K}_{\mathcal{H}_{qn}}, \tag{3}$$

where the matrices $\mathbf{K}_{\mathcal{H}} \in \mathbb{R}^{n\times n}$, $\mathbf{K}_{\mathcal{H}_{qn}} \in \mathbb{R}^{n_q\times n}$, and $\mathbf{K}_{\mathcal{H}_q} \in \mathbb{R}^{n_q\times n_q}$ are all in $\mathcal{H}$. Here, $\mathbf{I}$ is an identity matrix with the same size of $\mathbf{K}_{\mathcal{H}_{qn}}$. We refer to $\tilde{\mathbf{K}}_{\tilde{\mathcal{H}}} \in \mathbb{R}^{n\times n}$ as the kernel matrix in the warped RKHS.

3.2 The objective function

Our proposed GLCF method is fundamentally based on the manifold learning [3,7] and local coordinate coding [38].

Given n input data points, each of them is treated as a column vector with m dimensions, and these entries form a whole data matrix denoted by $\mathbf{X} \in \mathbb{R}^{m\times n}$. Concept factorization is aimed to find two data matrices $\mathbf{W} \in \mathbb{R}^{n\times p}$ and $\mathbf{V} \in \mathbb{R}^{n\times p}$, whose entries are nonnegative. The much lower intrinsic dimensions can be estimated from the underlying concepts in the data space. Each concept is a linear combination of the entire data points, and each data point is a linear combination of all the concepts. Thus, the original data representation can be approximately written in the following form [35]:

$$\mathbf{X} \approx \mathbf{X}\mathbf{W}\mathbf{V}^T. \tag{4}$$

In concrete, the concept coordinate coding can be defined as a concept pair (γ, C), where C is a set of anchor points with d dimensions, and γ is a map of $\mathbf{x} \in \mathbb{R}^d$ to $[\gamma_v(\mathbf{x})]_{v\in C} \in \mathbb{R}^{|C|}$ such that $\sum_v \gamma_v(\mathbf{x}) = 1$. It induces the following physical approximation of $\mathbf{x}$ in $\mathbb{R}^d : \gamma(\mathbf{x}) = \sum_{v\in C} \gamma_v(\mathbf{x})v$. The corresponding concept coding norm is $\|\mathbf{x}\|_\gamma = \sqrt{\sum_{v\in C} \gamma_v(\mathbf{x})^2}$. It has been proved that the complexity of local coordinate coding depends on the intrinsic manifold dimensionality if the data lie on a manifold [38].

For the local concept coordinate coding system, the bases vectors in $\mathbf{U} = \mathbf{W}\mathbf{X}$ are modeled as the anchor points, and each row of the coefficient matrix $\mathbf{V}$ contains the corresponding coordinates for each data point. We assume each original data point is sufficiently close to only a small number of anchor points. When the locally sparse constraints are enforced on the traditional concept factorization, it gives rise to minimizing the objective function of LCF [25], as shown by

$$J_{LCF} = \|\mathbf{X} - \mathbf{X}\mathbf{W}\mathbf{V}^T\|^2 + \lambda \sum_{i=1}^{n}\sum_{k=1}^{p} |v_{ik}| \|\sum_{j=1}^{n} w_{jk}\mathbf{x}_j - \mathbf{x}_i\|^2, \tag{5}$$

where the two coefficient matrices $\mathbf{W}$ and $\mathbf{V}$ are nonnegative, the term $\lambda \geq 0$ is a smoothing parameter that controls the regularization.

After a series of linear algebra operations (e.g., the property $\|\mathbf{A}\|^2 = \mathrm{Tr}(\mathbf{A}^T\mathbf{A})$, $\mathrm{Tr}(\mathbf{AB}) = \mathrm{Tr}(\mathbf{BA})$), we get

$$\begin{aligned}\mathcal{R}1 &= \|\mathbf{X} - \mathbf{X}\mathbf{W}\mathbf{V}^T\|^2 \\ &= \mathrm{Tr}\big((\mathbf{X} - \mathbf{X}\mathbf{W}\mathbf{V}^T)^T(\mathbf{X} - \mathbf{X}\mathbf{W}\mathbf{V}^T)\big) \\ &= \mathrm{Tr}\big((\mathbf{I} - \mathbf{W}\mathbf{V}^T)^T\mathbf{X}^T\mathbf{X}(\mathbf{I} - \mathbf{W}\mathbf{V}^T)\big),\end{aligned} \tag{6}$$

where $\mathbf{I}$ is an identity matrix.

$$\begin{aligned}\mathcal{R}2 &= \sum_{i=1}^{n}\sum_{k=1}^{p}|v_{ik}|\|\sum_{j=1}^{n}w_{jk}\mathbf{x}_j-\mathbf{x}_i\|^2\\&=\sum_{i=1}^{n}\|(\mathbf{x}_i\mathbf{1}^T-\mathbf{XW})\mathbf{\Lambda_i}^{\frac{1}{2}}\|^2\\&=\sum_{i=1}^{n}\mathrm{Tr}\big((\mathbf{x}_i\mathbf{1}^T-\mathbf{XW})\mathbf{\Lambda_i}(\mathbf{x}_i\mathbf{1}^T-\mathbf{XW})^T\big)\\&=\sum_{i=1}^{n}\mathrm{Tr}(\mathbf{x}_i\mathbf{1}^T\mathbf{\Lambda_i}\mathbf{1}\mathbf{x}_i^T-2\cdot\mathbf{x}_i\mathbf{1}^T\mathbf{\Lambda_i}\mathbf{XW}^T+\mathbf{XW}\mathbf{\Lambda_i}\mathbf{W}^T\mathbf{X}^T)\\&=\sum_{i=1}^{n}\mathrm{Tr}(\mathbf{1}^T\mathbf{\Lambda_i}\mathbf{1}\mathbf{x}_i^T\mathbf{x}_i-2\cdot\mathbf{1}^T\mathbf{\Lambda_i}\mathbf{W}^T\mathbf{X}^T\mathbf{x}_i+\mathbf{W}\mathbf{\Lambda_i}\mathbf{W}^T\mathbf{X}^T\mathbf{X}),\end{aligned}\tag{7}$$

where $\mathbf{1}$ is a column vector with all ones, $\Lambda_i = diag(|v_{i1}|,\ldots,|v_{ip}|)\in\mathbb{R}^{p\times p}$.

In the above formulations, $\mathbf{X}^T\mathbf{X}$, $\mathbf{X}^T\mathbf{x}_i$ and $\mathbf{x}_i^T\mathbf{x}_i$ are forms of standard kernels, of which each entry is nothing but the inner product between the data points [35], which can be denoted by a mercer kernel $\mathcal{K}(\cdot,\cdot)=\langle\phi(\cdot),\phi(\cdot)\rangle$. Now we assume the data are mapped into the high-dimensional space RKHS using some mapping function $\phi(\cdot)$. For each data point, it is denoted by $\mathbf{x}\to\phi(\mathbf{x})$ and all the data points form a whole $\phi(\mathbf{X})$. Thus, the kernel LCF aims to minimize

$$\begin{aligned}J_{KLCF} &= \mathrm{Tr}\big((\mathbf{I}-\mathbf{WV}^T)^T\langle\phi(\mathbf{X}),\Phi(\mathbf{X})\rangle(\mathbf{I}-\mathbf{WV}^T)\big)+\lambda\sum_{i=1}^{n}\mathrm{Tr}(\mathbf{1}^T\mathbf{\Lambda_i}\mathbf{1}\langle\phi(\mathbf{x}_i),\phi(\mathbf{x}_i)\rangle\\&\quad-2\cdot\mathbf{1}^T\mathbf{\Lambda_i}\mathbf{W}^T\langle\phi(\mathbf{X}),\phi(\mathbf{x}_i)\rangle+\mathbf{W}\mathbf{\Lambda_i}\mathbf{W}^T\langle\phi(\mathbf{X}),\phi(\mathbf{X})\rangle)\\&=\mathrm{Tr}\big((\mathbf{I}-\mathbf{WV}^T)^T\mathcal{K}(\mathbf{X},\mathbf{X})(\mathbf{I}-\mathbf{WV}^T)\big)+\lambda\sum_{i=1}^{n}\mathrm{Tr}\big(\mathbf{1}^T\mathbf{\Lambda_i}\mathbf{1}\mathcal{K}(\mathbf{x}_i,\mathbf{x}_i)\\&\quad-2\cdot\mathbf{1}^T\mathbf{\Lambda_i}\mathbf{W}^T\mathcal{K}(\mathbf{X},\mathbf{x}_i)+\mathbf{W}\mathbf{\Lambda_i}\mathbf{W}^T\mathcal{K}(\mathbf{X},\mathbf{X})\big).\end{aligned}\tag{8}$$

To extract the latent concepts consistent with the manifold geometry, it is natural to take advantage of the manifold adaptive kernel [30] for the KLCF. As mentioned above, manifold structure can be discovered using the graph Laplacian associated with the data points; hence, we use it to encode the geometrical structure of the data. Assume data points are modeled as a nearest neighbor graph G with n nodes, and an edge with the assigned weight is put between two nearby nodes. The weight matrix $\mathbf{S}$ on the graph can be designed in multiple ways, among of which a simple one is to adopt the binary weights (i.e., the weight takes 1 if the two vertices are connected with each other). Then, we build this sparse matrix $\mathbf{S}$ according to

$$\mathbf{S}_{ij}=\begin{cases}1, & \text{if } \mathbf{x}_i\in\mathcal{N}(\mathbf{x}_j) \text{ or } \mathbf{x}_j\in\mathcal{N}(\mathbf{x}_i);\\0, & \text{otherwise},\end{cases}\tag{9}$$

where $\mathcal{N}(\mathbf{x})$ denotes the k nearest neighbors of the data point $\mathbf{x}$. The graph Laplacian is defined as $\mathbf{L}=\mathbf{D}-\mathbf{S}$, where $\mathbf{D}$ is a diagonal matrix with elements $\mathbf{D}_{ii}=\sum_j\mathbf{S}_{ij}$. This way, we can substitute $\mathbf{M}$ with a graph Laplacian $\mathbf{L}$. So for convenience, we utilize all available data points to derive the linear space $\mathcal{V}$ in the warped RKHS (i.e., $n_q=n$), and compact

Eq. (3) into its concise form

$$\tilde{\mathbf{K}}_{\mathcal{M}} = \mathbf{K} - \mathbf{K}^T(\mathbf{I} + \mathbf{LK})^{-1}\mathbf{LK}, \tag{10}$$

where $\tilde{\mathbf{K}}_{\mathcal{M}}$ indicates that this kernel matrix is in a manifold RKHS. In practice, to control the role of the graph Laplacian, it is sensible to associate a constant γ with $\mathbf{L}$ (i.e., $\gamma\mathbf{L}$). In this work, we take the default value $\gamma = 1$, which means the graph Laplacian remains still in $\tilde{\mathcal{H}}$.

Substituting the manifold adaptive kernel $\tilde{\mathbf{K}}_{\mathcal{M}}$ into the Eq. (8), we obtain the objective function of the proposed GLCF approach formulated as

$$\begin{aligned} J_{GLCF} = \mathrm{Tr}\big((\mathbf{I} - \mathbf{WV}^T)^T\tilde{\mathbf{K}}_{\mathcal{M}(:,:)}(\mathbf{I} - \mathbf{WV}^T)\big) + \lambda\sum_{i=1}^{n}\mathrm{Tr}\big(\mathbf{1}^T\mathbf{\Lambda_i}\mathbf{1}\tilde{\mathbf{K}}_{\mathcal{M}(i,i)} \\ - 2\cdot\mathbf{1}^T\mathbf{\Lambda_i}\mathbf{W}^T\tilde{\mathbf{K}}_{\mathcal{M}(:,i)} + \mathbf{W}\mathbf{\Lambda_i}\mathbf{W}^T\tilde{\mathbf{K}}_{\mathcal{M}(:,:)}\big). \end{aligned} \tag{11}$$

3.3 Generalized update algorithm

Assume that all elements of the manifold kernel $\tilde{\mathbf{K}}_{\mathcal{M}}$ are nonnegative, it can readily obtain the multiplicative update rules using the Karush–Kuhn–Tucker (KKT) conditions and Lagrangian multipliers as follows (details can be referred to [25])

$$w_{jk} \leftarrow w_{jk}\frac{(\tilde{\mathbf{K}}_{\mathcal{M}(:,:)}\mathbf{V} + \lambda\sum_{i=1}^{N}\tilde{\mathbf{K}}_{\mathcal{M}(:,i)}\mathbf{1}^T\mathbf{\Lambda_i})_{jk}}{(\tilde{\mathbf{K}}_{\mathcal{M}(:,:)}\mathbf{WV}^T\mathbf{V} + \lambda\sum_{i=1}^{N}\tilde{\mathbf{K}}_{\mathcal{M}(:,:)}\mathbf{W}\mathbf{\Lambda_i})_{jk}} \tag{12}$$

$$v_{ki} \leftarrow v_{ki}\frac{2(\lambda + 1)(\tilde{\mathbf{K}}_{\mathcal{M}(:,:)})_{ki}\mathbf{W}}{(2\mathbf{VW}^T\tilde{\mathbf{K}}_{\mathcal{M}(:,:)}\mathbf{W} + \lambda\mathbf{A} + \lambda\mathbf{B})_{ki}}, \tag{13}$$

where $\mathbf{A} = [\mathbf{a}, \ldots, \mathbf{a}] \in \mathbb{R}^{n\times p}, \mathbf{a} = diag(\tilde{\mathbf{K}}_{\mathcal{M}}),\ \mathbf{B} = [\mathbf{b}, \ldots, \mathbf{b}]^T \in \mathbb{R}^{n\times p}, \mathbf{b} = diag(\mathbf{W}^T\tilde{\mathbf{K}}_{\mathcal{M}}\mathbf{W})$. Similar to LCF, it is true that our objective function J_{GLCF} is nonincreasing and invariant if and only if $\mathbf{W}$ and $\mathbf{V}$ are both at a stationary point under these update rules. However, in many real situations, the manifold kernel matrix might have negative entries, which make the above update rules fail. It is really desirable to design a generalized update algorithm that can handle arbitrary components of the kernel matrix. To address this issue, we resort to a powerful theorem proposed in [29] and deduce our generalized update algorithm following the fashion in [5].

Denote a nonnegative vector with m components by $\mathbf{z}$, a symmetric positive definite matrix by $\mathbf{H}$, any vector with the same dimension as $\mathbf{z}$ by $\mathbf{b}$, we consider a function in the nonnegative quadratic formulation [29] like

$$f(\mathbf{z}) = \frac{1}{2}\mathbf{z}^T\mathbf{Hz} + \mathbf{b}^T\mathbf{z}, \tag{14}$$

then it is easy to derive the solution $\mathbf{z}^*$ that minimizes this function by the following iterative update rule

$$z_i \leftarrow z_i\left[\frac{-b_i + \sqrt{b_i^2 + 4(\mathbf{H}^+\mathbf{z})_i(\mathbf{H}^-\mathbf{z})_i}}{2(\mathbf{H}^+\mathbf{z})_i}\right], \tag{15}$$

where $\mathbf{H}^+$ and $\mathbf{H}^-$ are both nonnegative matrices satisfying $\mathbf{H} = \mathbf{H}^+ - \mathbf{H}^-$ with the entries specified by

$$\mathbf{H}_{ij}^{+} = \begin{cases} \mathbf{H}_{ij}, & \text{if } \mathbf{H}_{ij} > 0, \\ 0 \quad , & \text{otherwise.} \end{cases} \quad \mathbf{H}_{ij}^{-} = \begin{cases} |\mathbf{H}_{ij}|, & \text{if } \mathbf{H}_{ij} < 0, \\ 0 \quad , & \text{otherwise.} \end{cases}$$

It can be observed that the objective of GLCF in Eq. (11) is a quadratic formulation of $\mathbf{W}$ or $\mathbf{V}$, respectively, when the other is fixed. Thus, we can intuitively apply the above update principle in Eq. (15) into our GLCF approach, requiring the $\mathbf{H}$ and $\mathbf{b}$ be identified. Following the deductions in [35], we can derive the generalized update algorithm for our GLCF method. To be more concise, we simply substitute $\tilde{\mathbf{K}}_{\mathcal{M}}$ with $\mathbf{K}$.

To yield the update rules for $\mathbf{W}$, the variable $\mathbf{V}$ should be fixed. Take the first-order and second-order derivatives w.r.t. $\mathbf{W}$, we could, respectively, obtain the variable $\mathbf{z}$ and the matrix $\mathbf{H}$ for the multiplicative update rule of $\mathbf{W}$. Similarly, the multiplicative update rule for $\mathbf{V}$ can be obtained when fixing $\mathbf{W}$. To achieve the consistency with $\mathbf{H}$ in Eq. (15), the kernel matrix $\mathbf{K}$ is also decomposed into two nonnegative parts shown by

$$\mathbf{K}_{ij}^{+} = \begin{cases} \mathbf{K}_{ij}, & \text{if } \mathbf{K}_{ij} > 0, \\ 0 \quad , & \text{otherwise.} \end{cases} \quad \mathbf{K}_{ij}^{-} = \begin{cases} |\mathbf{K}_{ij}|, & \text{if } \mathbf{K}_{ij} < 0, \\ 0 \quad , & \text{otherwise.} \end{cases}$$

First, we fix the matrix $\mathbf{V}$, and take the first-order derivative of the quadratic form $J(\mathbf{W})$ (i.e., the objective function of GLCF) with respect to $\mathbf{W}$ at $\mathbf{W} = 0$, leading to

$$\left.\frac{\partial J}{\partial w_{jk}}\right|_{\mathbf{W}=0} = -2(\mathbf{KV})_{jk} - 2\lambda \left(\sum_{i=1}^{N} \mathbf{K}_{(:,i)} \mathbf{1}^T \mathbf{\Lambda_i} \right)_{jk}. \tag{16}$$

Then, taking the second-order derivative of $J(\mathbf{W})$ w.r.t. $\mathbf{W}$, we have

$$\frac{\partial^2 J}{\partial w_{jk} \partial w_{mn}} = 2\mathbf{K}_{jm} (\mathbf{V}^T \mathbf{V})_{nk} + 2\lambda \sum_{i=1}^{N} \mathbf{K}_{jm} (\mathbf{\Lambda_i})_{nk}. \tag{17}$$

So the generalized update rule for $\mathbf{W}$ can be expressed by

$$w_{jk} \leftarrow w_{jk} \frac{-(\mathbf{R_w})_{jk} + \sqrt{(\mathbf{R_w})_{jk}^2 + 4\mathbf{P}_{jk}^{+}\mathbf{P}_{jk}^{-}}}{2\mathbf{P}_{jk}^{+}}, \tag{18}$$

where $\mathbf{R_w} = -\mathbf{KV} - \lambda \sum_{i=1}^{N} \mathbf{K}_{(:,i)} \mathbf{1}^T \mathbf{\Lambda_i} \in \mathbb{R}^{n\times p}$, $\mathbf{P}^{+} = \mathbf{K}^{+}\mathbf{W}\mathbf{V}^T\mathbf{V} + \lambda \sum_{i=1}^{N} \mathbf{K}^{+}\mathbf{W}\mathbf{\Lambda_i} \in \mathbb{R}^{n\times p}$, $\mathbf{P}^{-} = \mathbf{K}^{-}\mathbf{W}\mathbf{V}^T\mathbf{V} + \lambda \sum_{i=1}^{N} \mathbf{K}^{-}\mathbf{W}\mathbf{\Lambda_i} \in \mathbb{R}^{n\times p}$.

Similarly, when the matrix $\mathbf{W}$ is held on, we take the first-order and second-order derivatives of the quadratic form $J(\mathbf{V})$ with regard to $\mathbf{V}$, leading to

$$\left.\frac{\partial J}{\partial v_{ki}}\right|_{\mathbf{V}=0} = -2(\lambda + 1)(\mathbf{KW})_{ki} + \lambda(\mathbf{A} + \mathbf{B})_{ki}. \tag{19}$$

$$\frac{\partial^2 J}{\partial v_{ki} \partial v_{mn}} = 2\delta_{ni} (\mathbf{W}^T \mathbf{KW})_{km}, \tag{20}$$

where δ_{ni} equals 1 if $i = n$ and equals 0 otherwise. In consequence, the generalized update rule for $\mathbf{V}$ can be given by

$$v_{ki} \leftarrow v_{ki} \frac{-(\mathbf{R_v})_{ki} + \sqrt{(\mathbf{R_v})_{ki}^2 + 4\mathbf{Q}_{ki}^{+}\mathbf{Q}_{ki}^{-}}}{2\mathbf{Q}_{ki}^{+}}, \tag{21}$$

where $\mathbf{R_v} = -(\lambda + 1)\mathbf{KW} + (1/2)\lambda(\mathbf{A} + \mathbf{B}) \in \mathbb{R}^{n\times p}$, $\mathbf{Q}^{+} = \mathbf{V}\mathbf{W}^T\mathbf{K}^{+}\mathbf{W} \in \mathbb{R}^{n\times p}$, $\mathbf{Q}^{-} = \mathbf{V}\mathbf{W}^T\mathbf{K}^{-}\mathbf{W} \in \mathbb{R}^{n\times p}$.

Algorithm 1 Graph-based local concept coordinate factorization (GLCF)

```
Input:
    An $m$-dimensional data matrix $\mathbf{X}$ with $n$ instances, the number of learned bases $p$, the number of nearest neighbors $k$, the tradeoff parameter $\lambda$.
Output:
    Two coefficient matrices $\mathbf{W}$ and $\mathbf{V}$.
1: Initialize: Generate $\mathbf{W}$ and $\mathbf{V}$ with random data rescaled to [0,1].
2: Encode the data geometrical structure. The locally geometrical structure in the data space is encoded by Laplacian graph, i.e., $\mathbf{L} = \mathbf{D} - \mathbf{S}$. The weight matrix $\mathbf{S}$ is yielded in the manner of Eq. (9) with $k$ nearest neighbors.
3: Graph-based kernel learning. Learn the manifold adaptive kernel $\tilde{\mathbf{K}}_{\mathcal{M}}$ in the warped RKHS using Eq. (10).
4: Learn the coefficient matrices. Update the two coefficient matrices $\mathbf{W}$ and $\mathbf{V}$ using the generalized multiplicative algorithm in Eqs. (18) and (21) respectively.
5: Repeat the update process until convergence.
```

3.4 The graph-based local concept coordinate factorization algorithm

In summary, our proposed graph-based local concept coordinate factorization (GLCF) algorithm mainly consists of three components, i.e., encoding the data geometry using graph Laplacian, learning the manifold adaptive kernel, and updating the coefficient matrices $\mathbf{W}$ and $\mathbf{V}$. Detailed descriptions can be found in Algorithm 1. Through this algorithm, a new low-dimensional data space spanned by the atoms of the coefficient matrix $\mathbf{V}$ can be easily captured, so as to facilitate the tasks of clustering or classification.

Here, we give some brief analysis on the time complexity of the proposed GLCF algorithm. First, it needs $O(n^2k)$ to construct the k nearest neighbor graph for graph Laplacian. Second, it requires $O(n^2)$ to build the data-dependent kernel matrix $\mathbf{K}$. Third, the manifold adaptive kernel costs $O(n^3)$. Fourth, updating $\mathbf{W}$ and $\mathbf{V}$ desires $O(n^2p)$. If the algorithm converges in t iterations, the total computational complexity is $O(n^2k + n^2 + n^3 + tn^2p)$. Since p is often a very small number to ensure the local preserving property of GLCF and it often requires only a few iterations for convergence, the overall cost using big O is $O(n^3)$. If we utilize $n_q (k < n_q < n)$ data points to derive the linear space $\mathcal{V}$ in the warped RKHS, the computational complexity can be further reduced to $O(n^2n_q)$. Extensive experiments were conducted on several databases to show its effectiveness in the following section.

4 Experiments

In this section, we investigate the clustering performance of the proposed method on several image and gene expression data. First, we give the descriptions about the data sets, evaluation metrics, and parameter settings. Then, the clustering results of GLCF in comparison with six different algorithms are reported with some interesting analysis. Finally, we conduct the model selection, explore the convergence property, and show some visualizations of the clustering results.

4.1 Data corpora

We employ three image databases and six gene expression data sets [41] in the experiments. Some samples of the image databases are given in Fig. 1. Descriptions of them are given stated as below.

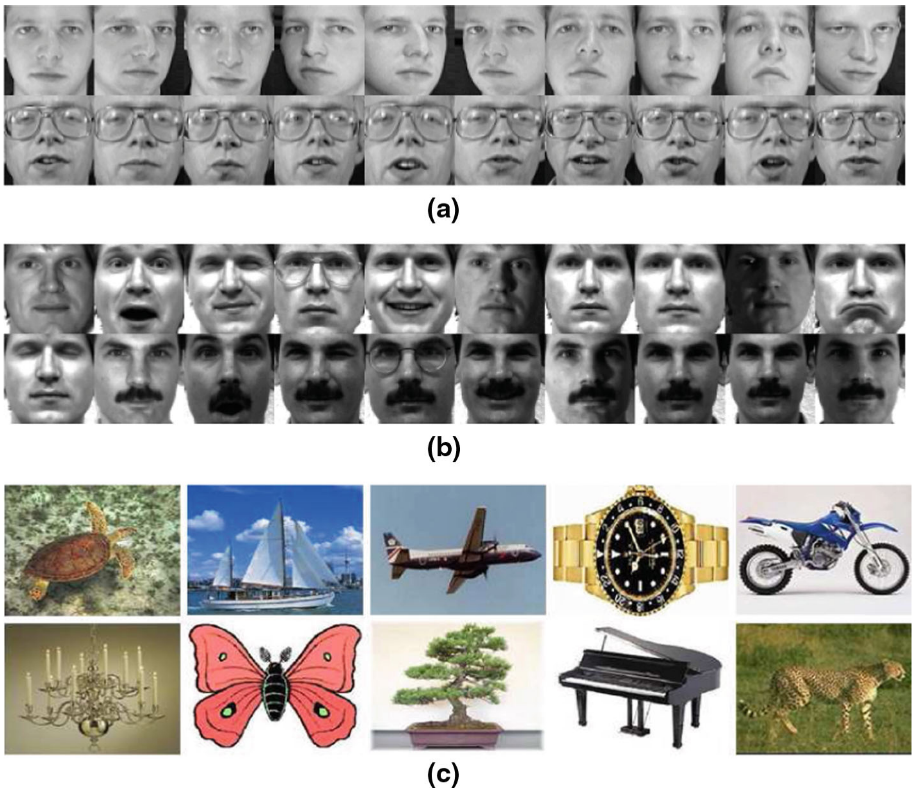

Fig. 1 Sample images from the image databases. **a** ORL; **b** Yale; **c** Caltech

The ORL facial data[1] contain 400 images referring to 40 subjects, and each subject has 10 images under different situations, such as varied lighting and different expressions (i.e., open/closed eyes, yes/not smiling). The Yale database is constructed at the Yale Center for Computer Vision and Control.[2] It contains 165 gray scale images involving 15 distinctive individuals. Each individual subject has 11 images with different facial expressions or configurations (i.e., center-light, left-light, right-light, with/without glasses, happy, sad, normal, surprised, wink, and sleepy). For ORL and Yale, original images were all normalized in scale and orientation to make the two eyes be aligned at the same horizontal position. In this test, we employ the cropped images with 64×64 pixels, and each pixel has 256 gray levels. Thus, every facial image is represented by a 4096-dimensional vector.

The Caltech image database is a subset of Caltech101[3] with top most 15 categories except the BACKGROUND Google. Each category contains 88 to 800 images. The size of each image is roughly 300×200 pixels. Since the low-level pixels cannot well reflect the image, we extract five kinds of features from each image using a common feature extraction toolkit,[4] including color histogram/moments, edge histogram, gabor wavelets transform, local binary

[1] http://www.cl.cam.ac.uk/research/dtg/attarchive/facedatabase.html.

[2] http://cvc.yale.edu/projects/yalefaces/yalefaces.html.

[3] http://www.vision.caltech.edu/Image-Datasets/Caltech101/.

[4] http://appsrv.cse.cuhk.edu.hk/~jkzhu/felib.html.

Table 1 Statistics of the data sets

Data sets	Type	Samples	Features	Classes	negfea
ORL	face	400	4,096	40	N
Yale	face	165	4,096	10	N
Caltech	image	3,855	809	15	Y
genALL	gene	248	4,184	6	N
genGCM	gene	198	5,334	14	Y
genHBC	gene	22	3,226	3	N
genLYM	gene	62	4,026	3	Y
genMLL	gene	72	4,194	3	Y
genNCI	gene	60	1,123	9	Y

pattern, and GIST [42]. Thus, each image can be represented by a 809-dimensional feature vector. Note that the obtained feature vectors contain a few negative entries.

Gene expression refers to the production level of protein molecules defined by a gene [41] and becomes of vital importance in genetics and molecular biology. Usually, gene expression data contain a large number of genes, leading to sparse features. Thus, to obtain a low-dimensional data representation of the gene data makes much sense. In this test, we examine six kinds of gene expression data, including genALL (it covers six subtypes of acute lymphoblastic leukemia), genGCM (it consists of 198 human tumor samples of fifteen types), genHBC (it involves 22 hereditary breast cancer samples), genLYM (it has three most prevalent adult lymphoid malignancies), genMLL (it contains three leukemia classes), and genNCI (it spans nine classes and is used to examine the variation in gene expression). Details can be referred to [41]. Among them, two data sets are nonnegative, while the rest ones contain negative features.

The statistics of these data are shown in Table 1, where "negfea" denotes there exist negative entries in the feature vector space.

4.2 Parameter settings

For all the data sets except KM, we applied the traditional k-means in the new transformed space and repeated 20 times with different randomly initial centers. The clustering results were recorded in terms of the minimum objective function value. As a preprocessing step, the samples in the ORL and Yale databases are normalized to unit Euclidean length. No more processing are conducted on the remaining data sets, because their features have been extracted from the original data and normalization might deteriorate the clustering performance on them.

For the graph-based methods, the number of nearest neighbor was empirically set to 5 to achieve the local consistency. For EGD, we adopted the code provided by the author in [31]. There are some parameters in GNMF, LCF, and GLCF, and we search the parameters in a broad range from 10^{-3} to 10^{3}. The results of the optimal parameters were recorded. In particular, to have an overview of the performances on different number of clusters on the data, we randomly selected $\{2, 4, 6, 8, 10, 12, 14, 16, 18, 20\}$ splits from ORL and $\{2, 4, 6, 8, 10, 12, 14\}$ splits from Yale to examine the clustering performance of the compared algorithms. For all methods, we repeated 20 test runs to obtain the averaged results. In particular, we observe that the compared methods yield more favorable results on ORL and Yale when k equals 1. The reported results could be better if the parameters are further tuned in a sensible grid for the data sets.

4.3 Evaluation metrics

We use two popular evaluation metrics to examine the clustering performance, i.e., the accuracy (AC) and the normalized mutual information (NMI) [6,20,23,25,35]. Usually, the clustering result is evaluated by comparing the obtained cluster label of each sample with its groundtruth label. Given an image $\mathbf{x}_i$, let a_i and g_i be the obtained cluster label and the label provided by the corpus in respective, then AC is defined as

$$\mathrm{AC} = \frac{\sum_{i=1}^{n} \delta(g_i, map(a_i))}{n}, \tag{22}$$

where we denote n by the total number of images in the corpus, $\delta(\cdot,\cdot)$ is a delta function that equals one if the two entries are the same and equals zero otherwise. The permutation mapping function $map(a_i)$ maps each obtained cluster label a_i to the equivalent label in the corpus. The Kuhn–Munkres algorithm [26] can be used to find the best mapping.

For NMI, let C and C' denote the sets of clusters from the ground truth and the algorithms, respectively, then the mutual information between them is expressed by [24,36]

$$\mathrm{MI}(C, C') = \sum_{c_i \in C, c'_j \in C'} p(c_i, c'_j) \cdot \log_2 \frac{p(c_i, c'_j)}{p(c_i) \cdot p(c'_j)}, \tag{23}$$

where $p(c_i)$ and $p(c'_j)$ are assigned by the probabilities that a sample arbitrarily selected from the database belongs to the clusters c_i and c'_j, respectively. And $p(c_i, c'_j)$ is the joint probability that the arbitrarily chosen sample belongs to the cluster c_i and c'_j at the same time. To simplify comparisons among different cluster sets, we use the normalized mutual information NMI, which is formulated as

$$\mathrm{NMI}(C, C') = \frac{\mathrm{MI}(C, C')}{\max(H(C), H(C'))}, \tag{24}$$

where $H(C)$ and $H(C')$ denote the entropies of the cluster sets C and C', respectively. Obviously, $\mathrm{NMI}(C, C')$ takes values ranging from zero to one. In particular, if the clusters in the two sets are identical, NMI takes one; if the two sets are completely independent, NMI becomes zero.

4.4 Performance comparisons

In the experiments, we investigate seven algorithms, including a k-means baseline, an empirical geodesic distance method, and five matrix-factorization-based clustering techniques. They are listed as follows:

- Traditional k-means clustering method (**KM**). It performs clustering in the original data space.
- Empirical geodesic distance (**EGD**). Here, we adopt the ISOMAP [31] to derive the transformed space, where the geodesic distances on a weighted graph are incorporated with metric multidimensional scaling.
- Nonnegative matrix factorization (**NMF**) [28]. It is one of the most popular matrix factorization techniques for nonnegative data points.
- Graph-regularized nonnegative matrix factorization (**GNMF**) [6], which preserves the locally geometrical information through a regularizer.
- Concept factorization (**CF**) [35], which is an extension of NMF.

- CF with the locality constraint (**LCF**) [25]. This is a state-of-the-art variant of CF.
- Graph-based local concept coordinate factorization (**GLCF**), which is our newly proposed method.

As mentioned in the preceding part, clustering was performed in the low-dimensional space derived from the embedding method and the matrix factorization approaches.

4.5 Results

The clustering results of the compared algorithms on ORL and Yale are, respectively, shown in Tables 2 and 3. The drawings in Fig. 2 depict the corresponding comparison curves in terms of clustering accuracy. Notice that the number of clusters in Fig. 2 is actually p, i.e., the number of classes involved in the test data split. With the increase of the number of classes, it becomes more difficult to conduct clustering for the compared algorithms. Tables 4 and 5 show the comparison results of different algorithms on Caltech and six gene expression data sets in terms of the averaged results and the standard deviations. The best results are boldfaced. The paired t-tests were conducted at the significance level of 0.05 on Caltech and six gene expression data as indicated by "•".

Observed from these tables and figures, several interesting points can be summarized in the following.

- The proposed GLCF method systematically outperforms other methods. This is mainly attributed to the fact that the local coordinate coding and the locally geometrical structure are both well respected so as to strengthen the sparsity of the new data representation and the local consistency of the underlying concepts. Moreover, manifold kernel learning is performed in the warped RKHS to better extract the latent concepts.
- EGD and GNMF perform better than NMF, CF, and LCF. This is because that both of them take advantage of the manifold information, which improves the clustering performance. Particularly, EGD employs the geodesic distance information on a weighted graph and GNMF employs the locally geometrical structure of the data space.
- Compared to EGD and GNMF, the superimposed local coordinate constraint is enforced on each concept for GLCF to preserve the local manifold structure and to obtain a sparse low-dimensional data representation. Moreover, the manifold kernel learning in the warped RKHS can better adapt to some data distributions with nonlinear correlations. Hence, GLCF is expected to achieve more promising results as supported by the records in the tables.
- Compared to NMF and GNMF, GLCF can handle negative entries in the input data space as shown in Tables 4 and 5. Since in many real-world applications, the extracted features cannot be guaranteed to be nonnegative, especially in the biology field, it is a significant advantage of GLCF to deal with the negative data in some situations. Therefore, it justifies the wide applicability and good interpretability of the developed GLCF method.
- GNMF consistently improves the performance of NMF, which suggest the efficacy of the manifold regularizer. In most cases, LCF outperforms CF, which demonstrates that the locality constraint imposed on concept factorization indeed takes effect.

Note that the results of NMF, CF, and LCF on ORL and Yale are better than those in [25], this might be due to the reason that we use the cropped images of 64×64 pixels, while the latter uses the images of 32×32 pixels with less image information. Moreover, since it has been shown that SparseNMF [16] performs worse than LCF in [25], we do not report its results here.

Table 2 Clustering performances on different splits of the ORL database

p	Accuracy							Normalized mutual information						
	KM	EGD	NMF	GNMF	CF	LCF	GLCF	KM	EGD	NMF	GNMF	CF	LCF	GLCF
2	0.923	0.958	0.890	0.925	0.923	0.918	**0.958**	0.772	0.863	0.694	0.803	0.758	0.761	**0.870**
4	0.814	0.849	0.775	0.825	0.823	0.828	**0.949**	0.751	0.784	0.689	0.761	0.743	0.751	**0.911**
6	0.738	0.800	0.740	0.790	0.772	0.768	**0.876**	0.712	0.765	0.735	0.778	0.748	0.753	**0.866**
8	0.715	0.781	0.731	0.763	0.696	0.753	**0.834**	0.740	0.811	0.751	0.776	0.730	0.777	**0.862**
10	0.635	0.740	0.697	0.732	0.686	0.709	**0.816**	0.693	0.768	0.744	0.782	0.740	0.752	**0.855**
12	0.625	0.685	0.680	0.718	0.647	0.674	**0.776**	0.703	0.744	0.752	0.790	0.721	0.741	**0.840**
14	0.598	0.692	0.650	0.709	0.617	0.670	**0.737**	0.695	0.758	0.738	0.779	0.700	0.743	**0.828**
16	0.569	0.648	0.643	0.650	0.573	0.616	**0.708**	0.684	0.731	0.745	0.762	0.684	0.723	**0.809**
18	0.549	0.664	0.626	0.671	0.586	0.628	**0.694**	0.680	0.747	0.736	0.790	0.695	0.732	**0.814**
20	0.558	0.637	0.628	0.669	0.557	0.604	**0.707**	0.702	0.736	0.747	0.795	0.688	0.725	**0.819**
Avg.	0.672	0.745	0.706	0.745	0.688	0.717	**0.805**	0.713	0.770	0.733	0.782	0.721	0.746	**0.847**

The best record on each split is highlighted in boldface

Table 3 Clustering performances on different splits of the Yale database

p	Accuracy							Normalized mutual information						
	KM	EGD	NMF	GNMF	CF	LCF	GLCF	KM	EGD	NMF	GNMF	CF	LCF	GLCF
2	0.811	**0.847**	0.814	0.832	0.798	0.807	0.841	0.527	**0.562**	0.482	0.531	0.476	0.457	0.500
4	0.584	0.711	0.610	0.668	0.594	0.599	**0.742**	0.424	0.517	0.437	0.510	0.407	0.446	**0.563**
6	0.560	0.586	0.589	0.609	0.577	0.614	**0.624**	0.492	0.508	0.517	0.520	0.496	0.532	**0.556**
8	0.522	0.557	0.563	0.590	0.573	0.538	**0.634**	0.495	0.515	0.515	0.556	0.525	0.512	**0.591**
10	0.515	0.560	0.553	0.575	0.536	0.530	**0.611**	0.520	0.545	0.537	0.551	0.517	0.531	**0.615**
12	0.513	0.536	0.566	0.562	0.547	0.525	**0.574**	0.543	0.532	0.565	0.570	0.541	0.537	**0.612**
14	0.509	0.507	0.535	0.534	0.531	0.529	**0.570**	0.554	0.537	0.549	0.578	0.542	0.554	**0.615**
Avg.	0.574	0.615	0.604	0.624	0.594	0.592	**0.656**	0.508	0.531	0.515	0.545	0.500	0.510	**0.579**

The best record on each split is highlighted in boldface

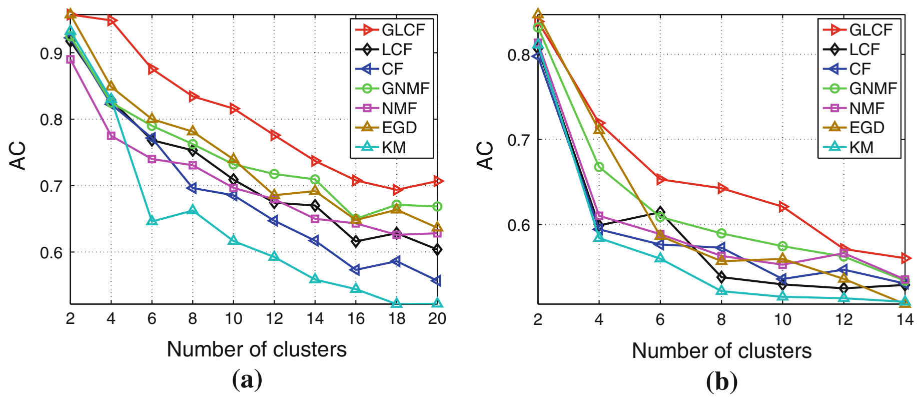

Fig. 2 Clustering performance comparison of different algorithms in terms of accuracy. **a** ORL; **b** Yale

4.6 Model selection

For our GLCF algorithm, there are two parameters: the trade-off parameter λ and the number of nearest neighbors k. As mentioned earlier, we fix $k = 1$ for ORL and Yale while $k = 5$ for the rest data. Thus, here we only explore the influences of λ on the clustering performance. Take ORL and Yale for example, we randomly choose six clusters from them for testing.

In detail, we fix the number of nearest neighbors to be 1 and search the optimal parameter from a wide range of $[10^{-3}, 10^{3}]$. Specially, the grid between 0 and 1 is divided into smaller bins so as to capture better parameters for λ. Besides, we conduct the model selection for LCF under the same settings. Selection results are drawn by curves in Fig. 3. As vividly depicted in these figures, GLCF performs favorably well when λ is no more than 0.5 and it deteriorates sharply when λ is over 1. Similar behaviors can be observed on other data sets. In summary, the trade-off parameter λ plays a crucial role in the proposed GLCF algorithm.

4.7 Convergence study

As we know, the generalized update rules lead to the two coefficient matrices $\mathbf{W}$ and $\mathbf{V}$. It has been shown the optimization of the objection function J_{GLCF} can be solved through iteratively updating these rules. For the purpose of examining the convergence, we have conducted some tests on ORL and Yale for NMF, CF, LCF, and GLCF. GNMF behaves similarly as NMF, so we do not depict its curves. Their convergence curves are depicted in Fig. 4 for ORL and Yale. In these figures, the horizontal axis represents the number of iterations, while the vertical axis denotes the log value of the objective function.

As can be seen from the graphs, these algorithms have fast convergence rates, since the local minima can be achieved after a very few iterations using the update rules. These methods have similar observations on the rest data sets.

4.8 Visual clustering results

To demonstrate the advantages of GLCF intuitively, we illustrate the clustering results of NMF, CF, LCF, and GLCF in terms of 3D graphs. Take ORL and Yale for example, we ran-

Table 4 Clustering accuracy (mean ± std) on Caltech and gene expression data sets

Data sets	KM	EGD	NMF	GNMF	CF	LCF	GLCF
Caltech	0.453 ± 0.028	0.505 ± 0.012	n/a	n/a	0.354 ± 0.015	0.488 ± 0.024	**0.531 ± 0.021** ●
genALL	0.461 ± 0.057	0.538 ± 0.004	0.441 ± 0.031	0.482 ± 0.037	0.359 ± 0.019	0.477 ± 0.030	**0.561 ± 0.000** ●
genGCM	0.380 ± 0.032	0.357 ± 0.0257	n/a	n/a	0.308 ± 0.016	0.365 ± 0.033	**0.408 ± 0.020** ●
genHBC	0.368 ± 0.020	0.366 ± 0.010	0.364 ± 0.000	0.374 ± 0.000	0.358 ± 0.017	0.384 ± 0.000	**0.427 ± 0.022** ●
genLYM	0.952 ± 0.051	0.969 ± 0.004	n/a	n/a	0.647 ± 0.059	0.811 ± 0.085	**1.000 ± 0.000** ●
genMLL	0.730 ± 0.030	0.821 ± 0.000	n/a	n/a	0.408 ± 0.023	0.665 ± 0.008	**0.833 ± 0.000**
genNCI	0.420 ± 0.035	0.473 ± 0.023	n/a	n/a	0.318 ± 0.034	0.371 ± 0.030	**0.497 ± 0.020**

"n/a" denotes the algorithm cannot handle negative entries, "●" indicates our method outperforms the rest at the significance level of 0.05, and the best results are highlighted in boldface

Table 5 Normalized mutual information (mean ± std) on Caltech and gene expression data sets

Data sets	KM	EGD	NMF	GNMF	CF	LCF	GLCF
Caltech	0.474 ± 0.014	0.512 ± 0.000	n/a	n/a	0.375 ± 0.013	0.496 ± 0.012	**0.533 ± 0.017** ●
genALL	0.373 ± 0.077	0.414 ± 0.027	0.340 ± 0.051	0.339 ± 0.072	0.312 ± 0.009	0.269 ± 0.093	**0.500 ± 0.000** ●
genGCM	0.474 ± 0.024	0.474 ± 0.018	n/a	n/a	0.285 ± 0.012	0.432 ± 0.031	**0.490 ± 0.014** ●
genHBC	0.194 ± 0.010	0.194 ± 0.014	0.191 ± 0.000	0.191 ± 0.000	0.213 ± 0.038	0.191 ± 0.000	**0.231 ± 0.018**
genLYM	0.887 ± 0.082	0.854 ± 0.014	n/a	n/a	0.537 ± 0.024	0.664 ± 0.078	**1.000 ± 0.030** ●
genMLL	0.552 ± 0.020	0.611 ± 0.000	n/a	n/a	0.324 ± 0.010	0.461 ± 0.025	**0.624 ± 0.000** ●
genNCI	0.418 ± 0.035	0.485 ± 0.031	n/a	n/a	0.298 ± 0.035	0.364 ± 0.034	**0.505 ± 0.025**

"n/a" denotes the algorithm cannot handle negative entries, "●" indicates our method outperforms the rest at the significance level of 0.05, and the best results are highlighted in boldface

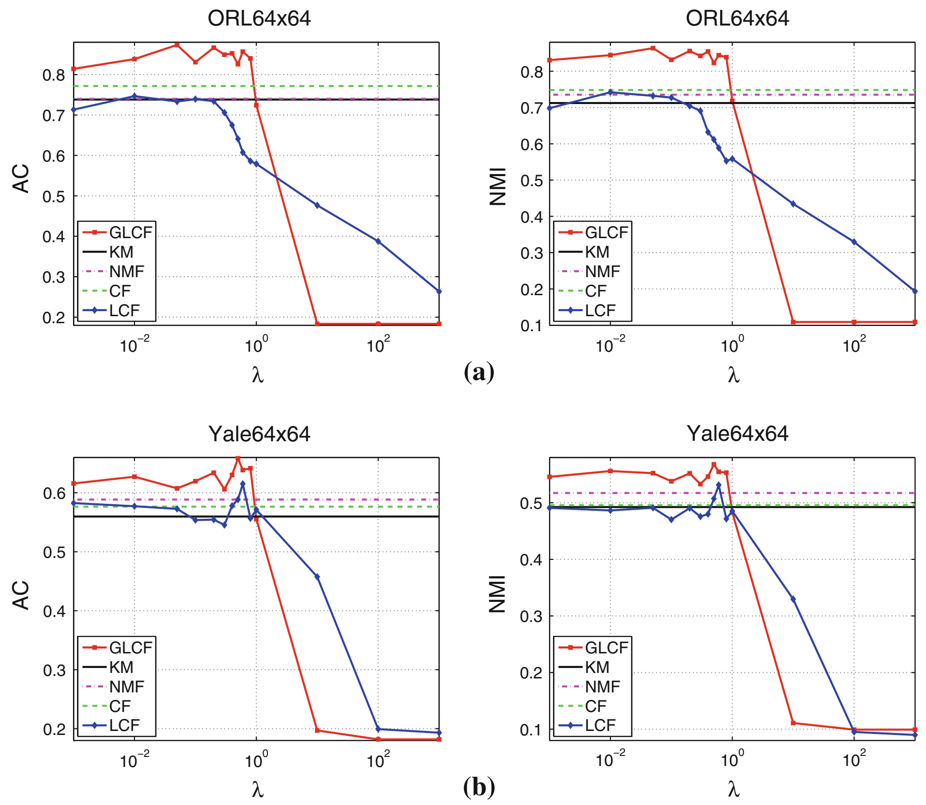

Fig. 3 Clustering performance of GLCF with the different λ. **a** ORL; **b** Yale. In each subfigure, the *left panel* reflects the clustering accuracy, and the *right panel* denotes the normalized mutual information

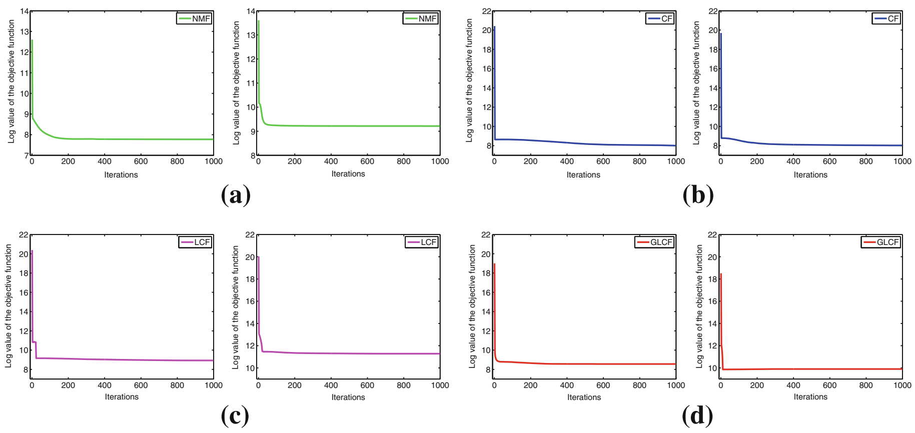

Fig. 4 Convergence performances of different algorithms. **a** NMF; **b** CF; **c** LCF; **d** GLCF. The *left panel* in each subfigure denotes the result on ORL, and the *right panel* in each subfigure denotes the result on Yale

domly select three classes from them, respectively. The visualizations of the clustering results are displayed in Fig. 5. In these graphs, the three clusters are remarked by different colorful shapes and we also highlight the centroid using one cross with a circle. As vividly shown in

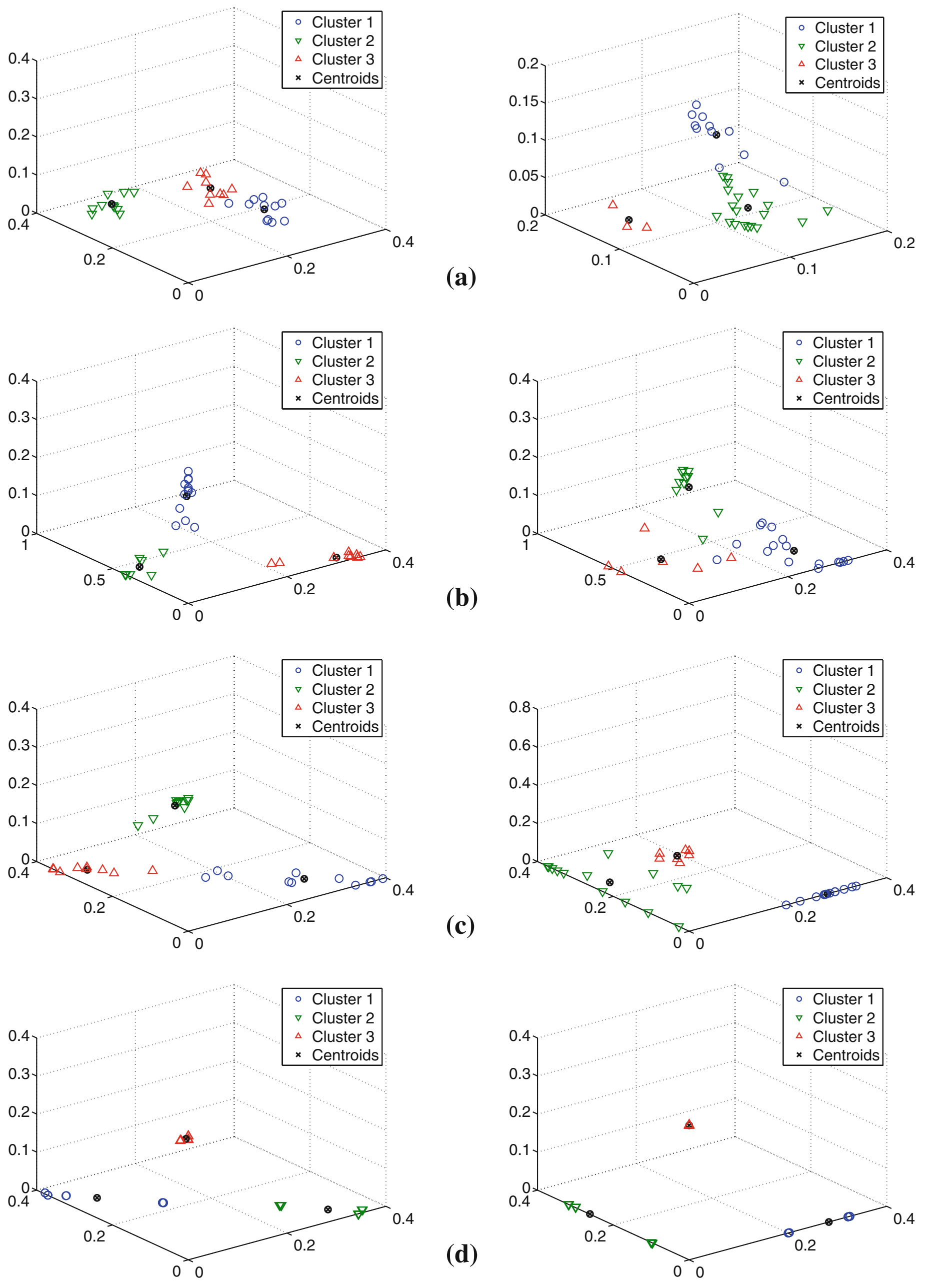

Fig. 5 Visual clustering results of different algorithms for randomly selected three clusters. **a** NMF; **b** CF; **c** LCF; **d** GLCF. The *left panel* in each subfigure denotes the result on ORL, and the *right panel* in each subfigure denotes the result on Yale

these figures, the data points within the same cluster are more compact and concentrated by GLCF and also those within different clusters are scattered far away, which further justifies the advantages of our method.

5 Conclusion

In this paper, we present a novel matrix factorization method called *graph-based local concept coordinate factorization* (GLCF). It aims to preserve the locally geometrical structure via graph Laplacian in the warped RKHS and ensure the sparseness of the new representations by adding the local coordinate constraints. The obtained new data representation can better preserve the manifold structure in the low-dimensional data subspace spanned by the atoms of the coefficient matrix $\mathbf{V}$. Furthermore, it makes the learned bases vector closer to the concept centers, and each data point is a linear combination of only a few components. We develop a generalized multiplicative update rules to optimize the objective function, and it has been shown this optimization framework is able to handle negative entries in the data matrix from both the theoretical and empirical viewpoints. A number of experiments were conducted on several image and gene expression databases. Results have validated the effectiveness of the proposed GLCF approach.

Acknowledgments We thank the anonymous reviewers for their valuable comments and suggestions which greatly improve the quality of the paper. This work was supported in part by National Natural Science Foundation of China under Grants 91120302, 61222207, 61173185, and 61173186, National Basic Research Program of China (973 Program) under Grant 2013CB336500, the Fundamental Research Funds for the Central Universities under Grant 2012FZA5017, and the Zhejiang Province Key S&T Innovation Group Project under Grant 2009R50009.

References

1. Abdi H, Williams LJ (2010) Principal component analysis. Wiley Interdiscip Rev: Comput Stat 2(4):433–459
2. Belkin M, Niyogi P (2002) Laplacian eigenmaps and spectral techniques for embedding and clustering. Adv Neural Inf Process Syst 2:585–592
3. Belkin M, Niyogi P, Sindhwani V (2006) Manifold regularization: a geometric framework for learning from labeled and unlabeled examples. J Mach Learn Res 7:2399–2434
4. Bouguila N, Ziou D (2012) A countably infinite mixture model for clustering and feature selection. Knowl Inf Syst 33(2):351–370
5. Cai D, He X, Han J (2011) Locally consistent concept factorization for document clustering. IEEE Trans Knowl Data Eng 23(7):902–913
6. Cai D, He X, Han J, Huang TS (2011) Graph regularized nonnegative matrix factorization for data representation. IEEE Trans Pattern Anal Mach Intell 33(8):1548–1560
7. Cai D, He X (2012) Manifold adaptive experimental design for text categorization. IEEE Trans Knowl Data Eng 24(4):707–719
8. Cai D, Bao H, He X (2011) Sparse concept coding for visual analysis. In: Proceedings of the 24th IEEE conference on computer vision and pattern recognition, pp 2905–2910
9. Cai D, He X, Wang X, Bao H, Han J (2009) Locality preserving nonnegative matrix factorization. In: Proceedings of the 21st international joint conference on artificial intelligence, pp 1010–1015
10. Cheng X, Du P, Guo J, Zhu X, Chen Y (2013) Ranking on data manifold with sink points. IEEE Trans Knowl Data Eng 25(1):177–191
11. Ding C, Li T, Peng W, Park H (2006) Orthogonal nonnegative matrix t-factorizations for clustering. In: Proceedings of the 12th ACM international conference on knowledge discovery and data mining, pp 126–135
12. Gong P, Zhang C (2012) Efficient nonnegative matrix factorization via projected Newton method. Pattern Recognit 45(9):3557–3565
13. Guan N, Tao D, Luo Z, Yuan B (2012) NeNMF: An optimal gradient method for nonnegative matrix factorization. IEEE Trans Signal Process 45(9):3557–3565
14. Hadsell R, Chopra S, LeCun Y (2006) Dimensionality reduction by learning an invariant mapping. In: Proceedings of the 19th IEEE conference on computer vision and pattern recognition 2:1735–1742
15. Hastie T, Tibshirani R, Friedman J, Franklin J (2009) The elements of statistical learning: data mining, inference, and prediction

16. Hoyer PO, Dayan P (2004) Non-negative matrix factorization with sparseness constraints. J Mach Learn Res 5:1457–1469
17. Hua W, He X (2011) Discriminative concept factorization for data representation. Neurocomputing 74(18):3800–3807
18. Kim Y, Chung C, Lee SG, Kim D (2011) Distance approximation techniques to reduce the dimensionality for multimedia databases. Knowl Inf Syst 28(1):227–248
19. Lee DD, Seung HS (1999) Learning the parts of objects by non-negative matrix factorization. Nature 401(6755):788–791
20. Li Z, Wu X, Peng H (2010) Nonnegative matrix factorization on orthogonal subspace. Pattern Recognit Lett 31(9):905–911
21. Li P, Chen C, Bu J (2012) Clustering analysis using manifold kernel concept factorization. Neurocomputing 87:120–131
22. Li P, Bu J, Chen C, Wang C, Cai D (2013) Subspace learning via locally constrained A-optimal nonnegative projection. Neurocomputing 115:49–62
23. Li P, Bu J, Chen C, He Z, Cai D (2013) Relational multi-manifold co-clustering. IEEE Trans Cybern. 43(6):1871–1881
24. Liu H, Wu Z, Cai D, Huang TS (2012) Constrained nonnegative matrix factorization for image representation. IEEE Trans Pattern Anal Mach Intell 34(7):1299–1311
25. Liu H, Yang Z, Wu Z (2011) Locality-constrained concept factorization. In: Proceedings of the 22nd international joint conference on artificial intelligence, pp 1378–1383
26. Lovász L, Plummer M (1986) Matching theory. In: North Holland, Akadémiai Kiadó
27. Roweis ST, Saul LK (2000) Nonlinear dimensionality reduction by locally linear embedding. Science 290(5500):2323–2326
28. Seung D, Lee L (2001) Algorithms for non-negative matrix factorization. Adv Neural Inf Process Syst 13:556–562
29. Sha F, Lin Y, Saul LK, Lee DD (2007) Multiplicative updates for nonnegative quadratic programming. Neural Comput, MIT Press 19(8):2004–2031
30. Sindhwani V, Niyogi P, elkin M (2005) Beyond the point cloud: from transductive to semi-supervised learning. In: Proceedings of the 22nd international conference on machine learning, pp 824–831
31. Tenenbaum JB, De Silva V, Langford JC, Bauckhage C (2000) A global geometric framework for nonlinear dimensionality reduction. Science 290(5500):2319–2323
32. Thurau C, Kersting K, Wahabzada M, Bauckhage C (2011) Convex non-negative matrix factorization for massive datasets. Knowl Inf Syst 29(2):457–478
33. Tzimiropoulos G, Zafeiriou S, Pantic M (2012) Subspace learning from image gradient orientations. IEEE Trans Pattern Anal Mach Intell 34(12):2454–2466
34. Xu B, Bu J, Chen C, Cai D (2012) A Bregman divergence optimization framework for ranking on data manifold and its new extensions. In: Proceedings of the 26th AAAI conference on artificial intelligence, pp 1190–1196
35. Xu W, Gong Y (2004) Document clustering by concept factorization. In: Proceedings of the 27th annual international ACM SIGIR conference on research and development in information retrieval, pp 202–209
36. Xu W, Liu X, Gong Y (2003) Document clustering based on non-negative matrix factorization. In: Proceedings of the 26th annual international ACM SIGIR conference on research and development in information retrieval, pp 267–273
37. Yang Z, Oja E (2012) Quadratic nonnegative matrix factorization. Pattern Recognit 45(4):1500–1510
38. Yu K, Zhang T, Gong Y (2009) Nonlinear learning using local coordinate coding. Adv Neural Inf Process Syst 22:2223–2231
39. Zhang L, Chen Z, Zheng M, He X (2011) Robust non-negative matrix factorization. Frontiers Electr Electron Eng China 6(2):192–200
40. Zhang Z, Wang J, Zha H (2012) Adaptive manifold learning. IEEE Trans Pattern Anal Mach Intell 34(2):253–265
41. Zhu S, Wang D, Yu K, Li T, Gong Y (2010) Feature selection for gene expression using model-based entropy. IEEE/ACM Trans Comput Biol Bioinform 7(1):25–36
42. Zhu J, Hoi SCH, Lyu MR, Yan S (2008) Near-duplicate keyframe retrieval by nonrigid image matching. In: Proceedings of the 16th ACM international conference on multimedia, pp 41–50

Author Biographies

Ping Li is currently pursuing the Ph.D. degree in Computer Science at Zhejiang University. He received the MS degree in Information and Communication Engineering from Central South University, China, in 2010. His research interests include machine learning, data mining, and multimedia analysis.

Jiajun Bu received the BS and Ph.D. degrees in Computer Science from Zhejiang University, China, in 1995 and 2000, respectively. He is a professor in College of Computer Science, Zhejiang University. His research interests include embedded system, data mining, information retrieval, and mobile database.

Lijun Zhang received the BS degree in Software Engineering and Ph.D. degree in Computer Science from Zhejiang University, China, in 2007 and 2012, respectively. Currently, he is a postdoctoral researcher in the Department of Computer Science and Engineering at Michigan State University, USA. His research interests include machine learning, optimization, and data mining.

Chun Chen received the BS degree in Mathematics from Xiamen University, China, in 1981, and his MS and Ph.D. degrees in Computer Science from Zhejiang University, China, in 1984 and 1990, respectively. He is a professor in College of Computer Science, Zhejiang University. His research interests include information retrieval, data mining, computer vision, computer graphics, and embedded technology.