Local Coordinate Concept Factorization for Image Representation

Haifeng Liu, *Member, IEEE*, Zheng Yang, Ji Yang, Zhaohui Wu, *Senior Member, IEEE*, and Xuelong Li, *Fellow, IEEE*

***Abstract*— Learning sparse representation of high-dimensional data is a state-of-the-art method for modeling data. Matrix factorization-based techniques, such as nonnegative matrix factorization and concept factorization (CF), have shown great advantages in this area, especially useful for image representation. Both of them are linear learning problems and lead to a sparse representation of the images. However, the sparsity obtained by these methods does not always satisfy locality conditions. For example, the learned new basis vectors may be relatively far away from the original data. Thus, we may not be able to achieve the optimal performance when using the new representation for other learning tasks, such as classification and clustering. In this paper, we introduce a locality constraint into the traditional CF. By requiring the concepts (basis vectors) to be as close to the original data points as possible, each datum can be represented by a linear combination of only a few basis concepts. Thus, our method is able to achieve sparsity and locality simultaneously. We analyze the complexity of our novel algorithm and demonstrate the effectiveness in comparison with the state-of-the-art approaches through a set of evaluations based on real-world applications.**

***Index Terms*— Data representation, dimensionality reduction, image clustering, matrix factorization.**

I. Introduction

LOW-DIMENSIONAL data representation is a fundamental problem in many real-world applications such as pattern recognition, computer vision, image processing, and so on [1]–[9]. Especially, linear data representations, such as vector quantization (VQ), principal component analysis (PCA), independent component analysis (ICA), sparse coding [10], [11], nonnegative matrix factorization (NMF) [12], [13] and concept factorization (CF) [14], have been widely used in data analysis tasks.

Among these methods, matrix factorization has been frequently used in linear data representation. Some clustering objective functions can be written as matrix factorization objectives. Given a data matrix $\mathbf{X}$, the above algorithms aim to find two or more matrix factors whose product is a good approximation to the original matrix. One factor can be interpreted as a matrix with a number of cluster prototypes as its columns, which reveals the latent semantic structure, and the other factor can be considered as the coefficients (also referred to as encodings) that show the nearest cluster prototypes. In real applications, the dimension of the found cluster prototypes is usually much smaller than that of the original data matrix. This gives rise to a compact representation of the data points, which can facilitate other learning tasks such as clustering and classification.

Among these matrix factorization methods, NMF [12], [13], [15], [16] is distinguished from others in that it enforces the constraint that the factor matrices must be nonnegative. That is, all elements must be equal to or greater than zero. The CF model is a variation of NMF in that each cluster is expressed by a linear combination of the data points and each data point is represented by a linear combination of the cluster centers. The major advantage of CF over NMF is that the NMF algorithm can only be performed in the original feature space of the data points, but the CF method can be performed in any data representation space, so that it can be kernelized and the powerful idea of the kernel method can be applied [14]. Both NMF and CF map the data from high-dimensional space to a low-dimensional space and obtain a sparse encoding of the data. However, the sparsity obtained by these methods does not always satisfy locality conditions. Since the local points share the greatest similarity, it would be more natural to represent the basis vectors using a few nearby anchor points, which leads to a more efficient representation of the data.

Recently, Cai *et al.* [17] proposed a locally consistent CF approach to encode the geometrical information of the data space, which can extract the document concepts with respect to the intrinsic manifold structure. This method is able to discover the local geometrical structure, but does not always satisfy the locality conditions as we mentioned before. To add sparseness constraint to the matrix factorization, Hoyer [18] showed how explicitly incorporating the notion of sparseness improves the inferred decompositions. However, there is no work that includes locality and sparsity constraints simultaneously to the best of our knowledge.

Manuscript received February 22, 2013; revised September 1, 2013; accepted October 1, 2013. This work was supported in part by the National Basic Research Program of China (973 Program) under Grant 2013CB336500, in part by the National Natural Science Foundation of China under Grant 61379071, Grant 91120302, and Grant 61125106, in part by Zhejiang Provincial Natural Science Foundation of China under Grant Y12F020150, in part by Qian Jiang Talented Program of Zhejiang Province under Grant 2011R10055, in part by the Doctoral Fund of Ministry of Education New Faculty Program under Grant 20100101120067, and in part by the Shaanxi Key Innovation Team of Science and Technology under Grant 2012KCT-04. *(Corresponding author: Z. Wu.)*

H. Liu, Z. Yang, J. Yang, and Z. Wu are with the College of Computer Science, Zhejiang University, Hangzhou, Zhejiang 310058, China (e-mail: haifengliu@zju.edu.cn; yzzju@zju.edu.cn; carolyj1991@zju.edu.cn; wzh@zju.edu.cn).

X. Li is with the Center for OPTical IMagery Analysis and Learning, State key Laboratory of Transient Optics and Photonics, Xi'an Institute of Optics and Precision Mechanics, Chinese Academy of Sciences, Xi'an 710119, China (e-mail: xuelongli@opt.ac.cn).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TNNLS.2013.2286093

In this paper, we introduce a novel matrix factorization algorithm, called local-coordinate CF (LCF), which imposes a locality constraint into the original concept factorization method. By requiring the concepts (basis vectors or cluster prototypes) to be as close to the original data points as possible, each datum can be represented by a linear combination of only a few nearby basis concepts, thus achieving sparsity and locality simultaneously. To achieve this goal, we incorporate the idea of local coordinate coding [19] into the original CF and propose a new objective function. To solve the corresponding optimization problem, we use an iterative multiplicative algorithm to find an optimal solution efficiently.

The rest of this paper is organized as follows. Section II reviews the background of matrix factorization and the related work is introduced in Section III. Section IV introduces our LCF approach and detailed analysis of the algorithm is provided in Section V. A variety of experimental results are presented in Section VI. Finally, we provide some concluding remarks in Section VII.

II. Background

Matrix factorization is an important topic in the mathematical discipline of linear algebra. A wide variety of methods of doing so have been developed over the decades by incorporating different constraints. Among these algorithms, the most popular ones include singular value decomposition, PCA, VQ, and ICA. In particular, NMF and CF have been shown to be very useful in many data analysis applications such as image processing [12], face recognition [20], document clustering [14], [21], [22], bioinformatics [23]–[25], and blind source separation [26].

Suppose we have n data points $\{\mathbf{x}_i\}_{i=1}^n$. Each data point $\mathbf{x}_i \in \mathbb{R}^m$ is m dimensional and is represented by a vector. The vectors are placed in the columns and the whole data set is represented by a matrix $\mathbf{X} = [\mathbf{x}_1, \ldots, \mathbf{x}_n] \in \mathbb{R}^{m\times n}$. NMF aims to find an $m \times k$ matrix $\mathbf{U}$ and a $k \times n$ matrix $\mathbf{V}$ where the product of these two factors is an approximation to the original matrix, represented as

$$\mathbf{X} \approx \mathbf{UV}.$$

Each column vector of $\mathbf{U}$, $\mathbf{u}_i$, can be regarded as a basis and each data point $\mathbf{x}_i$ is approximated by a linear combination of these k bases, weighted by the components of $\mathbf{V}$: $\mathbf{x}_i \approx \sum_{j=1}^k \mathbf{u}_j v_{ji}$.

The specialty of NMF is that it enforces that all entries of the factor matrices must be nonnegative. One limitation of NMF is that the nonnegative requirement is not applicable to applications where the data involve negative numbers. The second is that it is not clear how to effectively perform NMF in the transformed data space so that the powerful kernel method can be applied.

CF is proposed to address the above problems while inheriting all the strengths of the NMF method. In the CF model, we rewrite the NMF model by representing each basis vector (cluster center) $\mathbf{u}_j$ by a linear combination of the data points $\mathbf{u}_j = \sum_i w_{ij}\mathbf{x}_i$, where $w_{ij} \geq 0$. Let $\mathbf{W} = [w_{ij}] \in \mathbb{R}^{n\times k}$. CF tries to decompose the data matrix to satisfy the following condition:

$$\mathbf{X} \approx \mathbf{XWV}.$$

Using the Frobenius norm to qualify the approximation, CF tries to minimize the following objective function:

$$\mathcal{O} = \|\mathbf{X} - \mathbf{XWV}\|^2. \tag{1}$$

CF relaxes the condition on the data matrix that every element has to be nonnegative and hence the applicability of the technique is expanded. Ding *et al.* [27] proposed a convex-NMF algorithm, which performs the same factorization as CF. Other than the convex NMF, Ding *et al.* [27] also proposed a number of new variations on the theme of NMF. In the semi-NMF algorithm, only one matrix factor is restricted to contain nonnegative entries, while relaxing the constraint on the basis vectors.

III. Related Work

The original NMF emphasizes the desideratum of sparsity. However, the experiments have shown that NMF factors are not necessarily sparse. Reference [28] discussed the conditions for obtaining parts-of-whole representations by NMF. To address this limitation, several schemes have been developed on top of NMF to add the sparsity constraint [18], [20], [29]. Ding *et al.* [27] also pointed that convex NMF has a nice property that the factored matrices tend to be very sparse [27]. Gao *et al.* proposed a variational regularized 2-D NMF method, which is developed under the framework of maximum a posteriori probability and is adaptively fine tuned using the variational approach [30]. This method can naturally impose sparsity constraint and incorporate prior information into the basis features.

In addition, the bases obtained by NMF are spatially global, whereas local bases would be preferred. Stan *et al.* [20] proposed local NMF (LNMF) to achieve a localized NMF representation by adding more constraints to enforce spatial locality. Wang *et al.* [31] proposed a novel subspace method using Fisher linear discriminant analysis, called Fisher NMF, which can produce both additive and spatially localized basis images as LNMF.

There is also some other work, which tries to combine NMF and classification methods such as support vector machines. For example, Zoidi *et al.* [32] proposed multiplicative update rules for concurrent NMF and maximum margin classification. The idea is to perform NMF, while ensuring that the margin between the projected data of the two classes is maximal. Usually, NMF requires the entire data set to reside in the memory and thus cannot be applied to large-scale or streaming data sets. To address this issue, Guan *et al.* [33] proposed an online NMF method, which performs in an incremental fashion via robust stochastic approximation.

The aforementioned approaches have been applied to various real applications. However, the basis vectors learned from these approaches may be far away from the data points or the encodings may not be sparse. Therefore, the basis vectors may not be optimal to represent the data points. In the following, we introduce our approach, which explicitly requires the basis

vectors to be as close to the original data points as possible. Hence, it can achieve sparsity and locality simultaneously.

IV. Local-Coordinate CF

In this section, we introduce our LCF algorithm, which takes the locality constraint as an additional requirement. The algorithm presented in this paper is fundamentally motivated from the concept of local coordinate coding [19].

A. Objective Function

First, we introduce the concept of coordinate coding.

Definition 1: A coordinate coding is a pair (γ, C), where $C \subset \mathbb{R}^d$ is a set of anchor points, and γ is a map of $\mathbf{x} \in \mathbb{R}^d$ to $[\gamma_v(\mathbf{x})]_{v\in C} \in R^{|C|}$ such that $\sum_v \gamma_v(\mathbf{x}) = 1$. It induces the following physical approximation of $\mathbf{x}$ in $\mathbb{R}^d$: $\gamma(\mathbf{x}) = \sum_{v\in C} \gamma_v(\mathbf{x})v$.

According to this definition, the CF model can be considered as a coordinate coding where the basis vectors $\mathbf{u}_j = \sum_i w_{ij}\mathbf{x}_i$ are a set of anchor points, and each column of $\mathbf{V}$ contains the coordinates for each data point with respect to the anchor points. To add the local sparse constraint to the traditional CF, we require that each original data point should be sufficiently close to only a few anchor points. This can be achieved by introducing the following term to measure the locality and sparsity penalty:

$$\mathcal{R} = \sum_{k=1}^{K} |v_{ki}| \, \|\mathbf{u}_k - \mathbf{x}_i\|^2 = \sum_{k=1}^{K} |v_{ki}| \left\| \sum_{j=1}^{N} w_{jk}\mathbf{x}_j - \mathbf{x}_i \right\|^2. \quad (2)$$

The above constraint incurs a heavy penalty if $\mathbf{x}_i$ is far away from the anchor point $\mathbf{u}_k$ while its coordinate v_{ki} with respect to $\mathbf{u}_k$ is large. Therefore, by minimizing $\mathcal{R}$, we essentially try to formalize our intuition that $\mathbf{x}_i$ is close to the anchor points $\mathbf{u}_k$ as much as possible, otherwise its coordinate with respect to $\mathbf{u}_k$ tends to be zero.

With the locality constraint, our LCF algorithm reduces to minimize the following objective function:

$$\mathcal{O} = \|\mathbf{X} - \mathbf{XWV}\|^2 + \lambda \sum_{i=1}^{N}\sum_{k=1}^{K} |v_{ki}| \left\| \sum_{j=1}^{N} w_{jk}\mathbf{x}_j - \mathbf{x}_i \right\|^2. \quad (3)$$

The $\lambda \geq 0$ is a regularization parameter. It is easy to observe that LCF will incur a heavy penalty if $\mathbf{x}_i$ is far away from the anchor point $\mathbf{u}_k$ while its new coordinate v_{ki} with respect to $\mathbf{u}_k$ is large. By minimizing our objective function, only a few coefficients v_{ki} are nonzero. Thus, we actually try to represent $\mathbf{x}_i$ by only a few nearby anchor points $\mathbf{u}_k$. In this way, we preserve the sparse and local structures simultaneously.

B. Algorithm

We introduce an iterative algorithm to find a local minimum for the optimization problem. The objective function can be rewritten as follows:

$$\begin{aligned}\mathcal{O} &= \|\mathbf{X} - \mathbf{XWV}\|^2 + \lambda \sum_{i=1}^{N}\sum_{k=1}^{K} |v_{ki}| \left\| \sum_{j=1}^{N} w_{jk}\mathbf{x}_j - \mathbf{x}_i \right\|^2 \\ &= \|\mathbf{X} - \mathbf{XWV}\|^2 + \lambda \sum_{i=1}^{N} \left\| (\mathbf{x}_i\mathbf{1}^T - \mathbf{XW})\mathbf{D}_i^{1/2} \right\|^2 \quad (4)\end{aligned}$$

where $\mathbf{D}_i = \mathrm{diag}(|\mathbf{v}_i|) \in \mathbb{R}^{K\times K}$. Using the matrix property $\mathrm{Tr}(\mathbf{AB}) = \mathrm{Tr}(\mathbf{BA})$, $\|\mathbf{A}\|^2 = \mathrm{Tr}(\mathbf{A}^T\mathbf{A})$ and $\mathrm{Tr}(\mathbf{A}) = \mathrm{Tr}(\mathbf{A}^T)$, we have

$$\begin{aligned}\mathcal{O} &= \mathrm{Tr}\Big((\mathbf{X} - \mathbf{XWV})(\mathbf{X} - \mathbf{XWV})^T \\ &\qquad + \lambda \sum_{i=1}^{N} (\mathbf{x}_i\mathbf{1}^T - \mathbf{XW})\mathbf{D}_i(\mathbf{x}_i\mathbf{1}^T - \mathbf{XW})^T\Big) \\ &= \mathrm{Tr}\Big(\mathbf{XX}^T - 2\mathbf{XWVX}^T + \mathbf{XWVV}^T\mathbf{W}^T\mathbf{X}^T + \lambda\sum_{i=1}^{N} (\mathbf{x}_i\mathbf{1}^T\mathbf{D}_i\mathbf{1}\mathbf{x}_i^T \\ &\qquad - 2\mathbf{x}_i\mathbf{1}^T\mathbf{D}_i\mathbf{W}^T\mathbf{X}^T + \mathbf{XWD}_i\mathbf{W}^T\mathbf{X}^T)\Big). \quad (5)\end{aligned}$$

Let ψ_{jk} and ϕ_{ki} be the Lagrange multiplier for constraints $w_{jk} \geq 0$ and $v_{ki} \geq 0$, respectively. We define matrix $\Psi = [\psi_{jk}]$ and $\Phi = [\phi_{ki}]$, then the Lagrange $\mathcal{L}$ is

$$\begin{aligned}\mathcal{L} = \mathrm{Tr}\Big(&\mathbf{XX}^T - 2\mathbf{XWVX}^T + \mathbf{XWVV}^T\mathbf{W}^T\mathbf{X}^T \\ &+ \lambda \sum_{i=1}^{N} (\mathbf{x}_i\mathbf{1}^T\mathbf{D}_i\mathbf{1}\mathbf{x}_i^T - 2\mathbf{x}_i\mathbf{1}^T\mathbf{D}_i\mathbf{W}^T\mathbf{X}^T \\ &+ \mathbf{XWD}_i\mathbf{W}^T\mathbf{X}^T)\Big) + \mathrm{Tr}(\Psi\mathbf{W}^T) + \mathrm{Tr}(\Phi\mathbf{V}^T). \quad (6)\end{aligned}$$

Define $\mathbf{K} = \mathbf{X}^T\mathbf{X}$ and a column vector $\mathbf{a} = \mathrm{diag}(\mathbf{K}) \in \mathbb{R}^N$. Let $\mathbf{A} = (\mathbf{a}, \ldots, \mathbf{a})^T$ be a $K \times N$ matrix whose rows are $\mathbf{a}^T$. Define a column vector $\mathbf{b} = \mathrm{diag}(\mathbf{W}^T\mathbf{KW}) \in \mathbb{R}^K$. Let $\mathbf{B} = (\mathbf{b}, \ldots, \mathbf{b})$ be a $K \times N$ matrix whose columns are $\mathbf{b}$. The partial derivatives of $\mathcal{L}$ with respect to $\mathbf{W}$ and $\mathbf{V}$ are as follows:

$$\frac{\partial \mathcal{L}}{\partial \mathbf{W}} = 2\mathbf{KWVV}^T - 2\mathbf{KV}^T + \lambda\sum_{i=1}^{N}(-2\mathbf{X}^T\mathbf{x}_i\mathbf{1}^T\mathbf{D}_i + 2\mathbf{KWD}_i) + \Psi \quad (7)$$

$$\frac{\partial \mathcal{L}}{\partial \mathbf{V}} = 2\mathbf{W}^T\mathbf{KWV} - 2\mathbf{W}^T\mathbf{K} + \lambda(\mathbf{A} - 2\mathbf{W}^T\mathbf{K} + \mathbf{B}) + \Phi. \quad (8)$$

Using the Karush–Kuhn–Tucker conditions [34] $\psi_{jk}w_{jk} = 0$ and $\phi_{ki}v_{ki} = 0$, we obtain the following equations:

$$\begin{aligned}(\mathbf{KWVV}^T)_{jk}w_{jk} - (\mathbf{KV}^T)_{jk}w_{jk} + \lambda\Big(\sum_{i=1}^{N}\mathbf{KWD}_i\Big)_{jk}w_{jk} \\ - \lambda\Big(\sum_{i=1}^{N}\mathbf{X}^T\mathbf{x}_i\mathbf{1}^T\mathbf{D}_i\Big)_{jk}w_{jk} = 0 \quad (9)\end{aligned}$$

$$\begin{aligned}2(\mathbf{W}^T\mathbf{KWV})_{ki}v_{ki} - 2(\mathbf{W}^T\mathbf{K})_{ki}v_{ki} \\ + \lambda(\mathbf{A} - 2\mathbf{W}^T\mathbf{K} + \mathbf{B})_{ki}v_{ki} = 0. \quad (10)\end{aligned}$$

The above equations lead to the following update rules:

$$w_{jk} \leftarrow w_{jk}\frac{\left(\mathbf{K}\mathbf{V}^T + \lambda\sum_{i=1}^{N}\mathbf{X}^T\mathbf{x}_i\mathbf{1}^T\mathbf{D}_i\right)_{jk}}{\left(\mathbf{K}\mathbf{W}\mathbf{V}\mathbf{V}^T + \lambda\sum_{i=1}^{N}\mathbf{K}\mathbf{W}\mathbf{D}_i\right)_{jk}} \tag{11}$$

$$v_{ki} \leftarrow v_{ki}\frac{2(\lambda+1)\left(\mathbf{W}^T\mathbf{K}\right)_{ki}}{(2\mathbf{W}^T\mathbf{K}\mathbf{W}\mathbf{V} + \lambda\mathbf{A} + \lambda\mathbf{B})_{ki}}. \tag{12}$$

In summary, the iterative updating algorithm to perform the LCF is described as follows.

1) Initialize the matrix variables $\mathbf{W}$ and $\mathbf{V}$ randomly.
2) Update $\mathbf{W}$ using (11).
3) Update $\mathbf{V}$ using (12).
4) Repeat steps 2 and 3 until the objective function converges.

The obtained $\mathbf{W}$ and $\mathbf{V}$ are the solutions for the objective function.

C. Convergence of the Algorithm

We have the following theorem regarding the above iterative updating rules. Theorem 1 guarantees the convergence of the iterations in (11) and (12) and therefore the final solution will be a local optimum.

Theorem 1: The objective function $\mathcal{O}$ in (3) is nonincreasing under the update rules in (11) and (12). The objective function is invariant under these updates if and only if $\mathbf{W}$ and $\mathbf{V}$ are at a stationary point.

To prove Theorem 1, we use an auxiliary function similar to that used in the expectation–maximization algorithm. To make the proof complete, we restate the definition of auxiliary function and its property, which will be used to prove the algorithm convergence.

Definition 2: $G(x, x')$ is an auxiliary function for $F(x)$ if the conditions

$$G(x, x') \geq F(x), \qquad G(x, x) = F(x)$$

are satisfied.

Lemma 1: If G is an auxiliary function, then F is nonincreasing under the update

$$x^{t+1} = \arg\min_{x} G(x, x'). \tag{13}$$

Proof: $F(x^{t+1}) \leq G(x^{t+1}, x^t) \leq G(x^t, x^t) = F(x^t)$. ■

The equality $F(x^{t+1}) = F(x^t)$ holds only if x^t is a local minimum of $G(x, x^t)$. By iterating the updates in (13), the sequence of estimates will converge to a local minimum $x_{\min} = \arg\min_x F(x)$. Next, we will define an auxiliary function for our objective function and use Lemma 1 to show that the minimum of the objective function is exactly our update rule, thereby Theorem 1 is proved.

First, we prove the convergence of the update rule in (11). Considering any element w_{ab} in W, we use $F_{w_{ab}}$ to denote the part of $\mathcal{O}$, which is only relevant to w_{ab}. Since the update is essentially element wise, it is sufficient to show that each $F_{w_{ab}}$ is nonincreasing under the update step of (11). We prove this by defining the auxiliary function G for $F_{w_{ab}}$ as follows.

Lemma 2: The function

$$\begin{aligned} G\left(w, w_{ab}^{(t)}\right) = {} & F_{w_{ab}}\left(w_{ab}^{(t)}\right) + F'_{w_{ab}}\left(w_{ab}^{(t)}\right)\left(w - w_{ab}^{(t)}\right) \\ & + \frac{(\mathbf{K}\mathbf{W}\mathbf{V}\mathbf{V}^T)_{ab} + \lambda\sum_{i=1}^{N}(\mathbf{K}\mathbf{W}\mathbf{D}_i)_{ab}}{w_{ab}^{(t)}} \\ & \times\left(w - w_{ab}^{(t)}\right)^2 \end{aligned} \tag{14}$$

is an auxiliary function for $F_{w_{ab}}$, the part of $\mathcal{O}$, which is only relevant to w_{ab}.

Proof: Since $G(w, w) = F_{w_{ab}}(w)$ is obvious, we only need to show that $G\left(w, w_{ab}^{(t)}\right) \geq F_{w_{ab}}(w)$. Compare $G\left(w, w_{ab}^{(t)}\right)$ with the Taylor series expansion of $F_{w_{ab}}(w)$

$$\begin{aligned} F_{w_{ab}}(w) = {} & F_{w_{ab}}\left(w_{ab}^{(t)}\right) + F'_{w_{ab}}\left(w_{ab}^{(t)}\right)\left(w - w_{ab}^{(t)}\right) \\ & + \frac{1}{2}F''_{w_{ab}}\left(w - w_{ab}^{(t)}\right)^2. \end{aligned} \tag{15}$$

We only need to show that

$$\left(\mathbf{K}\mathbf{W}\mathbf{V}\mathbf{V}^T\right)_{ab} + \lambda\sum_{i=1}^{N}\left(\mathbf{K}\mathbf{W}\mathbf{D}_i\right)_{ab} \geq \frac{1}{2}w_{ab}^{(t)}F''_{w_{ab}}.$$

It is easy to check that

$$\begin{aligned} F'_{w_{ab}} = {} & \left(\frac{\partial\mathcal{O}}{\partial\mathbf{W}}\right)_{ab} = \left(2\mathbf{K}\mathbf{W}\mathbf{V}\mathbf{V}^T - 2\mathbf{K}\mathbf{V}^T\right)_{ab} \\ & + \lambda\sum_{i=1}^{N}\left(-2\mathbf{X}^T\mathbf{x}_i\mathbf{1}^T\mathbf{D}_i + 2\mathbf{K}\mathbf{W}\mathbf{D}_i\right)_{ab} \\ F''_{w_{ab}} = {} & 2(\mathbf{K})_{aa}\left(\mathbf{V}\mathbf{V}^T\right)_{bb} + 2\lambda\sum_{i=1}^{N}(\mathbf{K})_{aa}(\mathbf{D_i})_{bb}. \end{aligned} \tag{16}$$

Therefore, we have

$$\begin{aligned} & (\mathbf{K}\mathbf{W}\mathbf{V}\mathbf{V}^T)_{ab} + \lambda\sum_{i=1}^{N}(\mathbf{K}\mathbf{W}\mathbf{D}_i)_{ab} = \sum_{k}(\mathbf{K}\mathbf{W})_{ak}\left(\mathbf{V}\mathbf{V}^T\right)_{kb} \\ & + \lambda\sum_{i=1}^{N}\sum_{k}(\mathbf{K}\mathbf{W})_{ak}(\mathbf{D}_i)_{kb} \geq (\mathbf{K}\mathbf{W})_{ab}\left(\mathbf{V}\mathbf{V}^T\right)_{bb} \\ & + \lambda\sum_{i=1}^{N}(\mathbf{K}\mathbf{W})_{ab}(\mathbf{D}_i)_{bb} \geq \sum_{k}(\mathbf{K})_{ak}w_{kb}^{(t)}\left(\mathbf{V}\mathbf{V}^T\right)_{bb} \\ & + \lambda\sum_{i=1}^{N}\sum_{k}(\mathbf{K})_{ak}w_{kb}^{(t)}(\mathbf{D}_i)_{bb} \geq w_{ab}^{(t)}\Big((\mathbf{K})_{aa}\left(\mathbf{V}\mathbf{V}^T\right)_{bb} \\ & + \lambda\sum_{i=1}^{N}(\mathbf{K})_{aa}(\mathbf{D}_i)_{bb}\Big) \geq \frac{1}{2}w_{ab}^{(t)}F''_{w_{ab}}. \end{aligned}$$

Thus, $G\left(w, w_{ab}^{(t)}\right) \geq F_{w_{ab}}(w)$. ■

Then, we define an auxiliary function for the update rule in (12). Similarly, let $F_{v_{ab}}$ denote the part of $\mathcal{O}$ relevant to v_{ab}. Then, the auxiliary function regarding v_{ab} is defined as follows.

Lemma 3: Function

$$G\left(v, v_{ab}^{(t)}\right) = F_{v_{ab}}\left(v_{ab}^{(t)}\right) + F'_{v_{ab}}\left(v_{ab}^{(t)}\right)\left(v - v_{ab}^{(t)}\right) + \frac{\left(\mathbf{W}^T\mathbf{X}^T\mathbf{X}\mathbf{W}\mathbf{V} + \frac{1}{2}\lambda\mathbf{A} + \frac{1}{2}\lambda\mathbf{B}\right)_{ab}}{v_{ab}^{(t)}}\left(v - v_{ab}^{(t)}\right)^2 \tag{17}$$

is an auxiliary function for $F_{v_{ab}}$, the part of $\mathcal{O}$, which is only relevant to v_{ab}.

Proof: Similarly, $G(v, v) = F_{v_{ab}}(v)$ is obvious, we just compare $G\left(v, v_{ab}^{(t)}\right)$ with the Taylor series expansion of $F_{v_{ab}}(v)$ to show that $G\left(v, v_{ab}^{(t)}\right) \geq F_{v_{ab}}(v)$. The Taylor series expansion of $F_{v_{ab}}(v)$ is as follows:

$$F_{v_{ab}}(v) = F_{v_{ab}}\left(v_{ab}^{(t)}\right) + F'_{v_{ab}}\left(v_{ab}^{(t)}\right)\left(v - v_{ab}^{(t)}\right) + \frac{1}{2}F''_{v_{ab}}\left(v - v_{ab}^{(t)}\right)^2. \tag{18}$$

We only need to show that

$$\left(\mathbf{W}^T\mathbf{X}^T\mathbf{X}\mathbf{W}\mathbf{V} + \frac{1}{2}\lambda\mathbf{A} + \frac{1}{2}\lambda\mathbf{B}\right)_{ab} \geq \frac{1}{2}v_{ab}^{(t)}F''_{v_{ab}}. \tag{19}$$

From the definition of $\mathbf{A}$ and $\mathbf{B}$, it is easy to check that $\mathbf{A} \geq 0$ and $\mathbf{B} \geq 0$. Thus, we have

$$\left(\mathbf{W}^T\mathbf{X}^T\mathbf{X}\mathbf{W}\mathbf{V} + \frac{1}{2}\lambda\mathbf{A} + \frac{1}{2}\lambda\mathbf{B}\right)_{ab} \geq \left(\mathbf{W}^T\mathbf{X}^T\mathbf{X}\mathbf{W}\mathbf{V}\right)_{ab} = \sum_k \left(\mathbf{W}^T\mathbf{X}^T\mathbf{X}\mathbf{W}\right)_{ak} v_{kb} \geq v_{ab}\left(\mathbf{W}^T\mathbf{X}^T\mathbf{X}\mathbf{W}\right)_{aa}. \tag{20}$$

In addition, since

$$F''_{v_{ab}} = 2\left(\mathbf{W}^T\mathbf{K}\mathbf{W}\right)_{aa} \tag{21}$$

so we have

$$\left(\mathbf{W}^T\mathbf{X}^T\mathbf{X}\mathbf{W}\mathbf{V} + \frac{1}{2}\lambda\mathbf{A} + \frac{1}{2}\lambda\mathbf{B}\right)_{ab} \geq \frac{1}{2}v_{ab}^{(t)}F''_{v_{ab}} \tag{22}$$

and finally obtain $G\left(v, v_{ab}^{(t)}\right) \geq F_{v_{ab}}(v)$. ■

With the above lemmas, we give the proof of Theorem 1.

Proof of Theorem 1: From Lemma 2, we know that $G\left(w, w_{ab}^{(t)}\right)$ is an auxiliary function for $F_{w_{ab}}$, and from Lemma 3, we know that $G\left(v, v_{ab}^{(t)}\right)$ is an auxiliary function for $F_{v_{ab}}$. According to Lemma 1, we can obtain the update rules by solving $w^{(t+1)} = \arg\min_w G\left(w, w_{ab}^{(t)}\right)$ and $v^{(t+1)} = \arg\min_z G\left(v, v_{ab}^{(t)}\right)$, respectively. To solve these optimization problems, we need to obtain

$$G'\left(w, w_{ab}^{(t)}\right) = F'_{w_{ab}}\left(w_{ab}^{(t)}\right) + 2\frac{\left(\mathbf{K}\mathbf{W}\mathbf{V}\mathbf{V}^T\right)_{ab} + \lambda\sum_{i=1}^N \left(\mathbf{K}\mathbf{W}\mathbf{D}_i\right)_{ab}}{w_{ab}^{(t)}} \times\left(w - w_{ab}^{(t)}\right) \tag{23}$$

$$G'\left(v, v_{ab}^{(t)}\right) = F'_{v_{ab}}\left(v_{ab}^{(t)}\right) + 2\frac{\left(\mathbf{W}^T\mathbf{X}^T\mathbf{X}\mathbf{W}\mathbf{V} + \frac{1}{2}\lambda\mathbf{A} + \frac{1}{2}\lambda\mathbf{B}\right)_{ab}}{v_{ab}^{(t)}} \times\left(v - v_{ab}^{(t)}\right). \tag{24}$$

TABLE I
PARAMETERS USED IN COMPLEXITY ANALYSIS

Parameters	Description
m	number of features for each data point
n	number of data points
k	number of basis
l	number of labeled data points
c	number of classes

It is easy to check that

$$F'_{w_{ab}} = \left(\frac{\partial\mathcal{O}}{\partial\mathbf{W}}\right)_{ab} = \left(2\mathbf{K}\mathbf{W}\mathbf{V}\mathbf{V}^T - 2\mathbf{K}\mathbf{V}^T\right)_{ab} + \lambda\sum_{i=1}^N\left(-2\mathbf{X}^T\mathbf{x}_i\mathbf{1}^T\mathbf{D}_i + 2\mathbf{K}\mathbf{W}\mathbf{D}_i\right)_{ab}$$

$$F''_{v_{ab}} = \left(\frac{\partial\mathcal{O}}{\partial\mathbf{V}}\right)_{ab} = \left(2\mathbf{W}^T\mathbf{K}\mathbf{W}\mathbf{V} - 2\mathbf{W}^T\mathbf{K}\right)_{ab} + \lambda\left(\mathbf{A} - 2\mathbf{W}^T\mathbf{K} + \mathbf{B}\right)_{ab}. \tag{25}$$

Replacing the corresponding terms in (23) and (24) with the equations in (25), and then setting $G'\left(w, w_{ab}^{(t)}\right)$ and $G'\left(v, v_{ab}^{(t)}\right)$ to zero, we have

$$\left(\mathbf{K}\mathbf{W}\mathbf{V}\mathbf{V}^T - \mathbf{K}\mathbf{V}^T\right)_{ab} + \lambda\sum_{i=1}^N\left(-\mathbf{X}^T\mathbf{x}_i\mathbf{1}^T\mathbf{D}_i + \mathbf{K}\mathbf{W}\mathbf{D}_i\right)_{ab} + \frac{\left(\mathbf{K}\mathbf{W}\mathbf{V}\mathbf{V}^T\right)_{ab} + \lambda\sum_{i=1}^N\left(\mathbf{K}\mathbf{W}\mathbf{D}_i\right)_{ab}}{w_{ab}^{(t)}}\left(w - w_{ab}^{(t)}\right) = 0 \tag{26}$$

$$\left(2\mathbf{W}^T\mathbf{K}\mathbf{W}\mathbf{V} - 2\mathbf{W}^T\mathbf{K}\right)_{ab} + \lambda\left(\mathbf{A} - 2\mathbf{W}^T\mathbf{K} + \mathbf{B}\right)_{ab} + 2\frac{\left(\mathbf{W}^T\mathbf{X}^T\mathbf{X}\mathbf{W}\mathbf{V} + \frac{1}{2}\lambda\mathbf{A} + \frac{1}{2}\lambda\mathbf{B}\right)_{ab}}{v_{ab}^{(t)}}\left(v - v_{ab}^{(t)}\right) = 0. \tag{27}$$

Through simplification and keeping the nonnegativity, we can obtain

$$w_{ab}^{(t+1)} = w_{ab}^{(t)}\frac{\left(\mathbf{K}\mathbf{V}^T + \lambda\sum_{i=1}^N\mathbf{X}^T\mathbf{x}_i\mathbf{1}^T\mathbf{D}_i\right)_{ab}}{\left(\mathbf{K}\mathbf{W}\mathbf{V}\mathbf{V}^T + \lambda\sum_{i=1}^N\mathbf{K}\mathbf{W}\mathbf{D}_i\right)_{ab}} \tag{28}$$

$$v_{ab}^{(t+1)} = v_{ab}^{(t)}\frac{2(\lambda+1)\left(\mathbf{W}^T\mathbf{K}\right)_{ab}}{\left(2\mathbf{W}^T\mathbf{K}\mathbf{W}\mathbf{V} + \lambda\mathbf{A} + \lambda\mathbf{B}\right)_{ab}} \tag{29}$$

which are exactly the same updates as in (11) and (12). Therefore, the objective function $\mathcal{O}$ in (3) is nonincreasing under these updates. □

V. ALGORITHM ANALYSIS

A. Computational Complexity Analysis

The computational complexity is an important metric to evaluate the quality of an algorithm. Especially, for our LCF algorithm, which uses an iterative update rule to find the optimal solution, the algorithm efficiency depends on both the computational complexity of each update step and the convergence rate. In this section, we discuss the computational complexity of the updating algorithms comparing with standard NMF and CF.

To precisely analyze the computational complexity of our multiplicative updating algorithm, we counted the number of arithmetic operations (including addition, multiplication, and division) for each update step in the algorithm. We listed

in Table I the parameters used in the counting and summarized the numbers of each operation for each algorithm in Table II. Due to the introduction of locality constraint, our LCF algorithm needs $2n^2k + 2nk^2$ more operations for addition and multiplication compared with traditional CF algorithm. However, the big O of both algorithms is the same.

In addition to the multiplicative updating, both CF and LCF need to compute the kernel matrix $\mathbf{K}$, which requires $O(n^2m)$ operations. Suppose the multiplicative updates for NMF, CF, and LCF stop after t_1, t_2, and t_3 iterations, individually, the overall computational complexity for these algorithms will be $O(t_1mnk)$, $O\big(t_2\big(n^2k + n^2m\big)\big)$, and $O\big(t_3\big(n^2k + n^2m\big)\big)$.

B. Connection With Gradient Descent Method

Other than the multiplicative update method to find the matrix factors, others have suggested gradient descent algorithms [35], [36] to solve the optimization problem. The algorithm is also called alternating nonnegative least squares or projected gradient. Reference [37] also shows that the project gradient method converges faster than the multiplicative update method.

Using gradient descent method, the additive update rules for the problem of (3) are as follows:

$$w_{ij} \leftarrow w_{ij} + \delta_{ij}\frac{\partial \mathcal{O}}{\partial w_{ij}}, \qquad v_{ij} \leftarrow v_{ij} + \gamma_{ij}\frac{\partial \mathcal{O}}{\partial v_{ij}}.$$

δ_{ij} and γ_{ij} are the parameters to control the step size of gradient descent. This algorithm always takes a step in the direction of the negative gradient, the direction of steepest descent. As long as they are sufficiently small, the updates should reduce $\mathcal{O}$.

We set

$$\delta_{ij} = -\frac{w_{ij}}{2\big(\mathbf{KWVV}^T + \lambda\sum_{i=1}^{N}\mathbf{KWD}_i\big)_{ij}} \tag{30}$$

$$\gamma_{ij} = -\frac{z_{ij}}{\big(2\mathbf{W}^T\mathbf{KWV} + \lambda\mathbf{A} + \lambda\mathbf{B}\big)_{ij}} \tag{31}$$

then we can obtain

$$\begin{aligned}
&w_{ij} + \delta_{ij}\frac{\partial \mathcal{O}}{\partial w_{ij}} = w_{ij} - \frac{w_{ij}}{2\big(\mathbf{KWVV}^T + \lambda\sum_{i=1}^{N}\mathbf{KWD}_i\big)_{ij}} \\
&\big(\big(2\mathbf{KWVV}^T - 2\mathbf{KV}^T\big)_{ij} + \lambda\sum_{i=1}^{N}\big(-2\mathbf{X}^T\mathbf{x}_i\mathbf{1}^T\mathbf{D}_i + 2\mathbf{KWD}_i\big)_{ij}\big) \\
&= w_{ij}\frac{\Big(\mathbf{KV}^T + \lambda\sum_{i=1}^{N}\mathbf{X}^T\mathbf{x}_i\mathbf{1}^T\mathbf{D}_i\Big)_{ij}}{\Big(\mathbf{KWVV}^T + \lambda\sum_{i=1}^{N}\mathbf{KWD}_i\Big)_{ij}}v_{ij} + \gamma_{ij}\frac{\partial \mathcal{O}}{\partial v_{ij}} \\
&= v_{ij} - \frac{v_{ij}}{\big(2\mathbf{W}^T\mathbf{KWV} + \lambda\mathbf{A} + \lambda\mathbf{B}\big)_{ij}} \\
&\quad\times\big(2\mathbf{W}^T\mathbf{KWV} - 2\mathbf{W}^T\mathbf{K} + \lambda\big(\mathbf{A} - 2\mathbf{W}^T\mathbf{K} + \mathbf{B}\big)\big)_{ij} \\
&= v_{ij}\frac{2(\lambda+1)\big(\mathbf{W}^T\mathbf{K}\big)_{ij}}{\big(2\mathbf{W}^T\mathbf{KWV} + \lambda\mathbf{A} + \lambda\mathbf{B}\big)_{ij}}
\end{aligned}$$

which are the multiplicative update rules in (11) and (12).

Many works have pointed out the fact that the multiplicative update algorithm can be considered as a gradient descent method [13], [38]. However, the factorization result and the convergence rate are dependent on the choice of the step size δ_{ij} and γ_{ij}. Without a careful choice for the step size parameters, a little can be said about the convergence of gradient descent methods.

The advantage of the multiplicative updating rules is the guarantee of the nonnegativity of the matrix factor and also the convergence to a local optimum.

C. Algorithm for Negative Data Matrices

The iterative update algorithm described in Section IV-B only works for the nonnegative data matrix. When the data matrix contains negative data point, the Lagrange method would not work. In this section, we leverage the following theorem proposed in [39] to find a solution for the general case.

Theorem 2: Define the nonnegative general quadratic form as

$$Q(Y) = \frac{1}{2}Y^T\mathbf{A}Y + c^TY \tag{32}$$

where Y is a m-dimensional nonnegative vector, $\mathbf{A}$ is a symmetric semipositive definite matrix, and c is an arbitrary vector. Let $\mathbf{A} = \mathbf{A}^+ - \mathbf{A}^-$, where $\mathbf{A}^+$ and $\mathbf{A}^-$ are two symmetric matrices defined As follows:

$$\mathbf{A}^+_{ij} = \begin{cases}\mathbf{A}_{ij} & \text{if } \mathbf{A}_{ij} > 0;\\ 0 & \text{otherwise}\end{cases} \qquad \mathbf{A}^-_{ij} = \begin{cases}|\mathbf{A}_{ij}| & \text{if } \mathbf{A}_{ij} < 0;\\ 0 & \text{otherwise}\end{cases}$$

then, the solution Y that minimizes $Q(Y)$ can be obtained by the following iterative update:

$$y_i \leftarrow y_i\frac{-c_i + \sqrt{c_i^2 + 4(\mathbf{A}^+Y)_i(\mathbf{A}^-Y)_i}}{2(\mathbf{A}^+Y)_i}. \tag{33}$$

Fixing $\mathbf{V}$, our objective $\mathcal{O}$ in (5) becomes a quadratic form of W. Thus, we can apply the above theorem to minimize the objective function by identifying the corresponding $\mathbf{A}$ and c term in $\mathcal{O}(\mathbf{W})$. The two coefficients for the quadratic form of $\mathcal{O}(\mathbf{W})$ can be obtained by taking the second-and first-order derivatives with respect to W at $W = 0$, respectively

$$\frac{\partial^2\mathcal{O}}{\partial w_{ij}\partial w_{kl}} = 2k_{ik}\big(\mathbf{VV}^T\big)_{lj} + 2\lambda\sum_{i=1}^{N}k_{ik}(\mathbf{D_i})_{lj} \tag{34}$$

$$\left.\frac{\partial\mathcal{O}}{\partial w_{ij}}\right|_{\mathbf{W}=0} = -2\big(\mathbf{KV}^T\big)_{ij} - 2\lambda\sum_{i=1}^{N}\big(\mathbf{X}^T\mathbf{x}_i\mathbf{1}^T\mathbf{D}_i\big)_{ij}. \tag{35}$$

Let $\mathbf{K} = \mathbf{K}^+ - \mathbf{K}^-$, where $\mathbf{K}^+$ and $\mathbf{K}^-$ are symmetric matrices whose elements are all positive

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

Substituting $\mathbf{A}$ and c in (33) with the above terms, respectively, we obtain the multiplicative updating solution for computing each element w_{ij} of $\mathbf{W}$

$$w_{ij} \leftarrow w_{ij}\frac{c_{ij} + \sqrt{c_{ij}^2 + \mathbf{P}^+_{ij}\mathbf{P}^-_{ij}}}{2\mathbf{P}^+_{ij}} \tag{36}$$

TABLE II
COMPUTATIONAL OPERATION COUNTS FOR EACH ITERATION IN NMF, CF, AND LCF

	Computational complexity for each update			
	addition	multiplication	division	overall
NMF	$2mnk + 2(m+n)k^2$	$2mnk + 2(m+n)k^2 + (m+n)k$	$(m+n)k$	$O(mnk)$
CF	$4n^2k + 4nk^2$	$4n^2k + 4nk^2 + 2nk$	$2nk$	$O(n^2k)$
LCF	$6n^2k + 6nk^2$	$6n^2k + 6nk^2 + 2nk$	$2nk$	$O(n^2k)$

where $c_{ij} = \left(\mathbf{K}\mathbf{V}^T\right)_{ij} + \lambda \sum_{i=1}^{N} \left(\mathbf{X}^T\mathbf{x}_i\mathbf{1}^T\mathbf{D}_i\right)_{ij}$, $\mathbf{P}^+ = \mathbf{K}^+\mathbf{W}\mathbf{V}\mathbf{V}^T + \lambda \sum_{i=1}^{N} \mathbf{K}^+\mathbf{W}\mathbf{D_i}$ and $\mathbf{P}^- = \mathbf{K}^-\mathbf{W}\mathbf{V}\mathbf{V}^T + \lambda \sum_{i=1}^{N} \mathbf{K}^-\mathbf{W}\mathbf{D_i}$. When all data are nonnegative, $\mathbf{P}^-$ becomes zero and the solution will be

$$w_{ij} = w_{ij} \frac{\left(\mathbf{K}\mathbf{V}^T + \lambda \sum_{i=1}^{N} \mathbf{X}^T\mathbf{x}_i\mathbf{1}^T\mathbf{D}_i\right)_{ij}}{\left(\mathbf{K}\mathbf{W}\mathbf{V}\mathbf{V}^T + \lambda \sum_{i=1}^{N} \mathbf{K}\mathbf{W}\mathbf{D}_i\right)_{ij}} \tag{37}$$

which is the same form as (11).

Similarly, fixing $\mathbf{W}$, we obtain the two coefficients for the quadratic form of $\mathcal{O}(\mathbf{V})$

$$\left.\frac{\partial \mathcal{O}}{\partial v_{ij}}\right|_{\mathbf{V}=0} = -2(\lambda+1)\mathbf{W}^T\mathbf{K} + \lambda(\mathbf{A}+\mathbf{B}) \tag{38}$$

$$\frac{\partial^2 \mathcal{O}}{\partial v_{ij}\partial v_{kl}} = 2\mathbf{W}^T\mathbf{K}\mathbf{W}. \tag{39}$$

The update rule for $\mathbf{V}$ is as follows:

$$v_{ij} \leftarrow v_{ij} \frac{c_{ij} + \sqrt{c_{ij}^2 + 4\mathbf{Q}_{ij}^+\mathbf{Q}_{ij}^-}}{2\mathbf{Q}_{ij}^+} \tag{40}$$

where $c_{ij} = 2(\lambda+1)(\mathbf{W}^T\mathbf{K})_{ij} - \lambda(\mathbf{A}+\mathbf{B})_{ij}$, $\mathbf{Q}^+ = 2\mathbf{W}^T\mathbf{K}^+\mathbf{W}\mathbf{V}$, and $\mathbf{Q}^- = 2\mathbf{W}^T\mathbf{K}^-\mathbf{W}\mathbf{V}$.

When all data are nonnegative, $\mathbf{Q}^-$ becomes zero and the solution will be

$$v_{ij} = v_{ij} \frac{2(\lambda+1)\left(\mathbf{W}^T\mathbf{K}\right)_{ij} - \lambda(\mathbf{A}+\mathbf{B})_{ij}}{2\left(\mathbf{W}^T\mathbf{K}\mathbf{W}\mathbf{V}\right)_{ij}}. \tag{41}$$

VI. EXPERIMENTS

In this section, we demonstrate the effectiveness of our algorithm through the experiments on image clustering. We compare the results with five other related methods on several data sets. The algorithms that we evaluated are listed as follows.

1) Traditional K-means (**Kmeans**).
2) **NMF** [13].
3) **CF** [14].
4) NMF with sparseness constraints (**SparseNMF**) [18].
5) Nonnegative local coordinate factorization(**NLCF**) [29].
6) Our proposed **LCF**.

In our experiments, we would like to test clustering performance of our proposed algorithm with different numbers of clusters. Therefore, the evaluations were conducted with the cluster number ranging from two to 10. For each given cluster number k, we randomly chose k clusters and ran the test ten times, and the final scores were obtained by calculating the average and variance over the ten test runs. Since the clustering results of the evaluated methods depend on the initialization, each test run consisted of ten subruns with different initializations and we chose the best result to report. Note that, each data point has only one label.

TABLE III
STATISTICS OF THE TWO DATA SETS

dataset	size(N)	dimensionality(M)	number of classes(K)
Yale	165	1024	15
ORL	400	1024	40

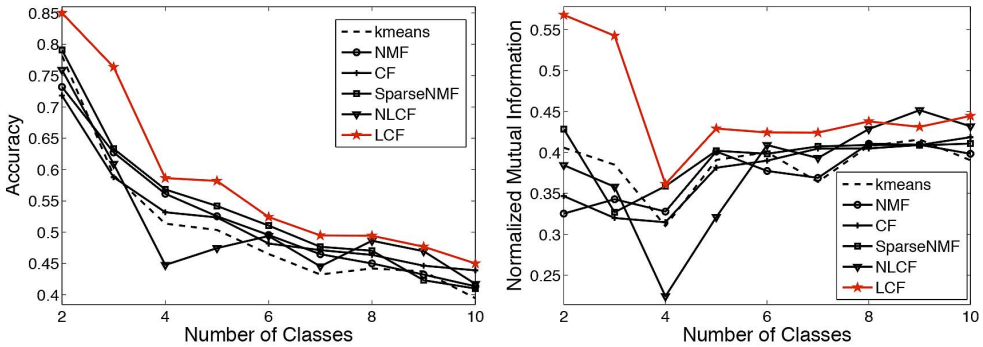
In the experiments, the parameters were set to be the values that each algorithm can achieve its best results via cross validation. However, there is one exception, which is that we gave no constraint on the basis matrix of Sparse NMF and just set the sparsity penalty weighting of the coefficient matrix via cross validation. This is because we just paid our attention on the sparsity of the representations. For our LCF algorithm, the regularization parameter is set to be $\lambda = 0.3$.

A. Data Sets

The experiments are conducted on two data sets. One is the Yale database, and the other is Cambridge ORL face database. The important statistics of these data sets are described as follows (also summarized in Table III).

1) The Yale database contains 165 gray scale images of 15 individuals. All images demonstrate variations in lighting condition (left-light, center-light, and right-light), facial expression (normal, happy, sad, sleepy, surprised, and wink), and with/without glasses.
2) The ORL database contains ten different images of each of 40 distinct subjects, thus 400 images in total. For some subjects, the images were taken at different times, varying the lighting, facial expressions (open/closed eyes, smiling/not smiling), and facial details (glasses/no glasses). All the images were taken against a dark homogeneous background with the subjects in an upright, frontal position (with tolerance for some side movements).

In all the experiments, images are preprocessed so that faces are located. Original images are first normalized in scale and orientation such that the two eyes are aligned at the same position. Then, the facial areas were cropped into the final images for clustering. Each image is of 32×32 pixels with 256 gray levels per pixel. All images for one individual or subject are treated as one cluster in our experiments.



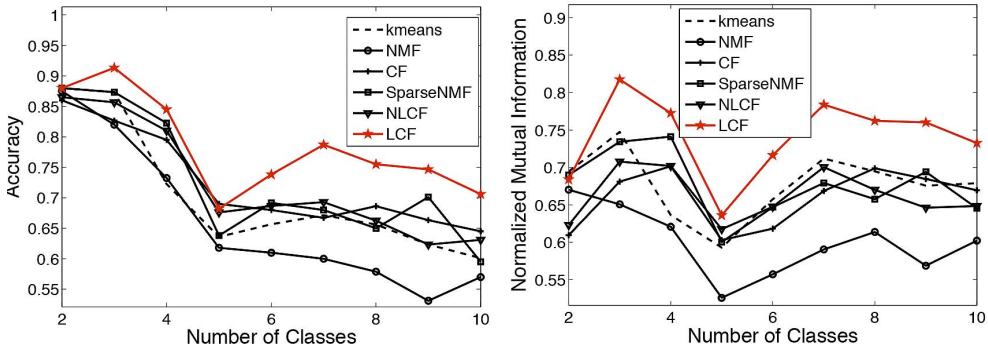
(a) (b)

Fig. 1. Clustering performance on the Yale database. (a) Accuracy versus number of clusters. (b) Mutual information versus number of clusters.

TABLE IV

CLUSTERING RESULTS COMPARISON ON THE YALE DATABASE

k	Kmeans	NMF	CF	SparseNMF	NLCF	LCF
	Accuracy(%)					
2	78.18 ± 16.86	73.18 ± 18.00	71.82 ± 20.21	79.09 ± 17.51	75.91 ± 18.07	$\mathbf{85.00 \pm 17.01}$
3	59.09 ± 12.14	62.73 ± 11.42	58.79 ± 9.70	63.33 ± 9.91	60.91 ± 16.34	$\mathbf{76.36 \pm 17.29}$
4	51.36 ± 7.20	56.14 ± 5.75	53.18 ± 9.55	56.82 ± 7.87	44.77 ± 3.06	$\mathbf{58.64 \pm 9.52}$
5	50.36 ± 3.36	52.55 ± 5.30	52.36 ± 5.74	54.18 ± 6.94	47.45 ± 6.43	$\mathbf{58.18 \pm 7.18}$
6	46.52 ± 6.85	49.55 ± 8.91	48.18 ± 8.10	51.06 ± 8.63	49.39 ± 7.91	$\mathbf{52.42 \pm 10.74}$
7	43.25 ± 6.42	46.49 ± 5.97	47.14 ± 5.92	47.66 ± 5.29	44.55 ± 6.67	$\mathbf{49.48 \pm 4.84}$
8	44.20 ± 3.78	45.00 ± 3.56	46.36 ± 2.91	47.05 ± 2.84	48.64 ± 4.97	$\mathbf{49.43 \pm 5.69}$
9	43.74 ± 7.03	43.23 ± 4.33	44.65 ± 6.32	42.32 ± 4.50	46.97 ± 7.83	$\mathbf{47.68 \pm 5.71}$
10	39.45 ± 4.49	41.36 ± 5.77	43.91 ± 5.19	41.00 ± 3.34	41.73 ± 4.27	$\mathbf{45.00 \pm 7.50}$
Avg.	50.68 ± 7.57	52.25 ± 7.67	51.82 ± 8.18	53.61 ± 7.43	51.15 ± 8.40	$\mathbf{58.02 \pm 9.50}$
	Normalized Mutual Information(%)					
2	40.58 ± 39.97	32.53 ± 38.16	34.66 ± 43.79	42.85 ± 40.46	38.44 ± 41.93	$\mathbf{56.78 \pm 39.45}$
3	38.49 ± 20.50	34.28 ± 16.27	31.99 ± 20.51	32.70 ± 15.86	35.80 ± 27.14	$\mathbf{54.24 \pm 30.02}$
4	31.00 ± 6.92	32.78 ± 9.04	31.48 ± 11.96	35.88 ± 10.04	22.43 ± 4.97	$\mathbf{36.13 \pm 9.92}$
5	39.08 ± 7.99	40.11 ± 8.46	38.13 ± 8.51	40.21 ± 7.76	32.10 ± 10.45	$\mathbf{42.91 \pm 10.41}$
6	40.03 ± 11.26	37.74 ± 10.81	38.99 ± 11.40	39.83 ± 11.50	40.89 ± 10.75	$\mathbf{42.43 \pm 11.94}$
7	36.39 ± 6.24	36.89 ± 6.36	40.46 ± 5.32	40.73 ± 5.37	39.32 ± 6.70	$\mathbf{42.41 \pm 4.47}$
8	40.78 ± 4.09	41.07 ± 3.64	40.48 ± 2.26	40.90 ± 3.08	42.85 ± 4.13	$\mathbf{43.78 \pm 3.70}$
9	41.58 ± 7.28	40.94 ± 5.24	40.90 ± 7.06	40.86 ± 3.54	$\mathbf{45.15 \pm 6.71}$	43.12 ± 5.09
10	39.04 ± 5.58	39.83 ± 4.59	41.85 ± 4.96	41.07 ± 2.63	43.18 ± 6.44	$\mathbf{44.45 \pm 7.32}$
Avg.	38.55 ± 12.20	37.35 ± 11.40	37.66 ± 12.86	39.45 ± 11.14	37.80 ± 13.25	$\mathbf{45.14 \pm 13.59}$



(a) (b)

Fig. 2. Clustering performance on the ORL database. (a) Accuracy versus number of clusters. (b) Mutual information versus number of clusters.

B. Evaluation Metrics

We use two metrics to evaluate the clustering performance [14]. One metric is accuracy (AC) and the other is the normalized mutual information ($\widehat{\text{MI}}$). The result is evaluated by comparing the cluster label of each sample with the label provided by the data set.

The metric is used to measure the percentage of correct labels obtained. Given a data set containing n images, for each sample image, let l_i be the cluster label we obtained by applying different algorithms and r_i be the label provided by the data set. The is defined as follows:

$$\text{AC} = \frac{\sum_{i=1}^{n} \delta\big(r_i, \text{map}(l_i)\big)}{n} \tag{42}$$

where $\delta(x, y)$ is the delta function that equals one if $x = y$ and equals zero otherwise, and $\text{map}(l_i)$ is the mapping function that maps each cluster label l_i to the equivalent label from the data set. The best mapping can be found using the Kuhn–Munkres algorithm [40].

In clustering applications, mutual information is used to measure how similar two sets of clusters are. Given two sets

TABLE V
CLUSTERING RESULTS COMPARISON ON THE ORL DATABASE

k	Kmeans	NMF	CF	SparseNMF	NLCF	LCF
	Accuracy(%)					
2	88.00 ± 17.64	87.50 ± 17.64	86.00 ± 16.09	88.00 ± 17.49	86.50 ± 16.13	$\mathbf{88.00 \pm 17.78}$
3	87.33 ± 10.73	82.00 ± 11.27	82.67 ± 12.36	87.33 ± 10.93	85.67 ± 10.86	$\mathbf{91.33 \pm 9.80}$
4	72.25 ± 13.53	73.25 ± 8.37	79.50 ± 9.34	82.25 ± 7.94	81.00 ± 7.92	$\mathbf{84.50 \pm 9.92}$
5	63.60 ± 11.62	61.80 ± 7.40	$\mathbf{69.00 \pm 8.26}$	63.80 ± 7.77	67.60 ± 7.26	68.20 ± 8.27
6	65.67 ± 7.04	61.00 ± 7.57	68.00 ± 7.74	69.17 ± 5.83	68.67 ± 6.66	$\mathbf{73.83 \pm 6.99}$
7	67.14 ± 13.19	60.00 ± 6.23	66.71 ± 7.06	68.00 ± 6.07	69.29 ± 10.23	$\mathbf{78.71 \pm 11.00}$
8	65.63 ± 7.57	57.88 ± 5.76	68.63 ± 6.29	65.00 ± 6.98	66.25 ± 5.53	$\mathbf{75.50 \pm 6.20}$
9	62.33 ± 7.44	53.11 ± 4.55	66.33 ± 7.32	70.11 ± 6.35	62.33 ± 7.67	$\mathbf{74.67 \pm 8.18}$
10	60.10 ± 7.23	57.00 ± 6.15	64.50 ± 6.17	59.50 ± 7.79	63.10 ± 6.49	$\mathbf{70.60 \pm 6.62}$
Avg.	70.23 ± 10.66	65.95 ± 8.33	72.37 ± 8.96	72.57 ± 8.57	72.27 ± 8.75	$\mathbf{78.37 \pm 9.42}$
	Normalized Mutual Information(%)					
2	$\mathbf{69.45 \pm 42.12}$	67.02 ± 40.95	60.95 ± 40.11	68.99 ± 42.71	62.31 ± 39.21	68.38 ± 41.04
3	74.75 ± 17.48	65.08 ± 15.94	68.11 ± 18.03	73.43 ± 17.50	70.77 ± 16.96	$\mathbf{81.76 \pm 17.12}$
4	63.64 ± 16.27	62.09 ± 9.01	70.15 ± 11.54	74.12 ± 9.17	70.20 ± 11.11	$\mathbf{77.26 \pm 13.09}$
5	59.30 ± 12.75	52.59 ± 9.10	60.30 ± 8.93	60.03 ± 9.42	61.76 ± 8.72	$\mathbf{63.61 \pm 10.83}$
6	65.75 ± 6.22	55.72 ± 5.45	61.83 ± 7.03	64.70 ± 7.13	64.75 ± 6.41	$\mathbf{71.63 \pm 8.32}$
7	71.19 ± 10.56	59.06 ± 6.53	66.86 ± 7.62	67.92 ± 7.45	70.04 ± 8.65	$\mathbf{78.39 \pm 10.46}$
8	69.48 ± 6.22	61.38 ± 4.54	69.89 ± 4.74	65.77 ± 5.71	67.01 ± 5.66	$\mathbf{76.23 \pm 5.03}$
9	67.56 ± 7.50	56.88 ± 5.32	68.47 ± 5.68	69.42 ± 5.19	64.62 ± 5.50	$\mathbf{76.01 \pm 7.92}$
10	67.89 ± 6.54	60.23 ± 6.54	66.96 ± 5.40	64.55 ± 6.59	64.85 ± 6.16	$\mathbf{73.25 \pm 6.10}$
Avg.	67.67 ± 13.96	60.01 ± 11.49	65.95 ± 12.12	67.66 ± 12.32	66.26 ± 12.04	$\mathbf{74.06 \pm 13.32}$

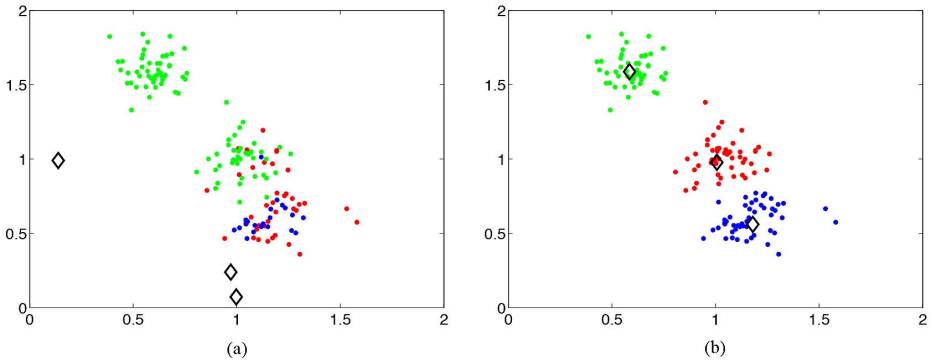


Fig. 3. Experiments on learning the overcomplete basis. Black diamonds: bases learned by each algorithm. (a) NMF clustering results. (b) LCF clustering results.

of image clusters $\mathcal{C}$ and $\mathcal{C}'$, their mutual information metric $\mathrm{MI}(\mathcal{C}, \mathcal{C}')$ is defined as follows:

$$\mathrm{MI}(\mathcal{C}, \mathcal{C}') = \sum_{c_i \in \mathcal{C}, c'_j \in \mathcal{C}'} p(c_i, c'_j) \cdot \log \frac{p(c_i, c'_j)}{p(c_i) \cdot p(c'_j)} \tag{43}$$

where $p(c_i)$, $p(c'_j)$ denote the probabilities that an image arbitrarily selected from the data set belongs to the clusters c_i and c'_j, respectively, and $p(c_i, c'_j)$ denotes the joint probability that this arbitrarily selected image belongs to the cluster c_i as well as c'_j simultaneously. $\mathrm{MI}(\mathcal{C}, \mathcal{C}')$ takes values between zero and $\max(H(\mathcal{C}), H(\mathcal{C}'))$, where $H(\mathcal{C})$ and $H(\mathcal{C}')$ are the entropies of $\mathcal{C}$ and $\mathcal{C}'$, respectively. It reaches the maximum $\max(H(\mathcal{C}), H(\mathcal{C}'))$ when the two sets of image clusters are identical and it becomes zero when the two sets are completely independent. One important character of $\mathrm{MI}(\mathcal{C}, \mathcal{C}')$ is that the value keeps the same for all kinds of permutations. In our experiments, we use the normalized metric $\widehat{\mathrm{MI}}(\mathcal{C}, \mathcal{C}')$, which takes values between zero and one

$$\widehat{\mathrm{MI}}(\mathcal{C}, \mathcal{C}') = \frac{\mathrm{MI}(\mathcal{C}, \mathcal{C}')}{\max(H(\mathcal{C}), H(\mathcal{C}'))}. \tag{44}$$

C. Clustering Results

Fig. 1(a) and (b) shows the plots of accuracy and normalized mutual information versus the number of clusters for different algorithms on the Yale data set. As can be observed, our proposed LCF algorithm consistently outperforms all the other algorithms. The detailed clustering results are shown in Table IV. The last row shows the average accuracy (normalized mutual information) over k. Comparing with the best algorithm other than our proposed LCF, i.e., SparseNMF, our algorithm LCF achieves 4.41% improvement in accuracy and 5.69% improvement in normalized mutual information.

Fig. 2(a) and (b) shows the graphical clustering results for the ORL data set. LCF obtains the best result for most of the cases. SparseNMF fails to consider the locality condition, and in some cases, performs even worse than Kmeans. Table V shows the detailed clustering accuracy and normalized mutual information. Comparing with the best algorithm other than our proposed LCF algorithm, i.e., SparseNMF, LCF achieves 5.8% improvement in accuracy. For normalized mutual information, LCF achieves 6.4% improvement.

D. Learning the Overcomplete Basis

Usually, matrix factorization methods are used for dimension reduction in many applications. However, [41] shows that in some cases, it is desirable to learn the overcomplete basis. Here, we evaluate the performance of our algorithm in this aspect. To show the performance for learning the overcomplete basis of our proposed algorithm, 150 data points

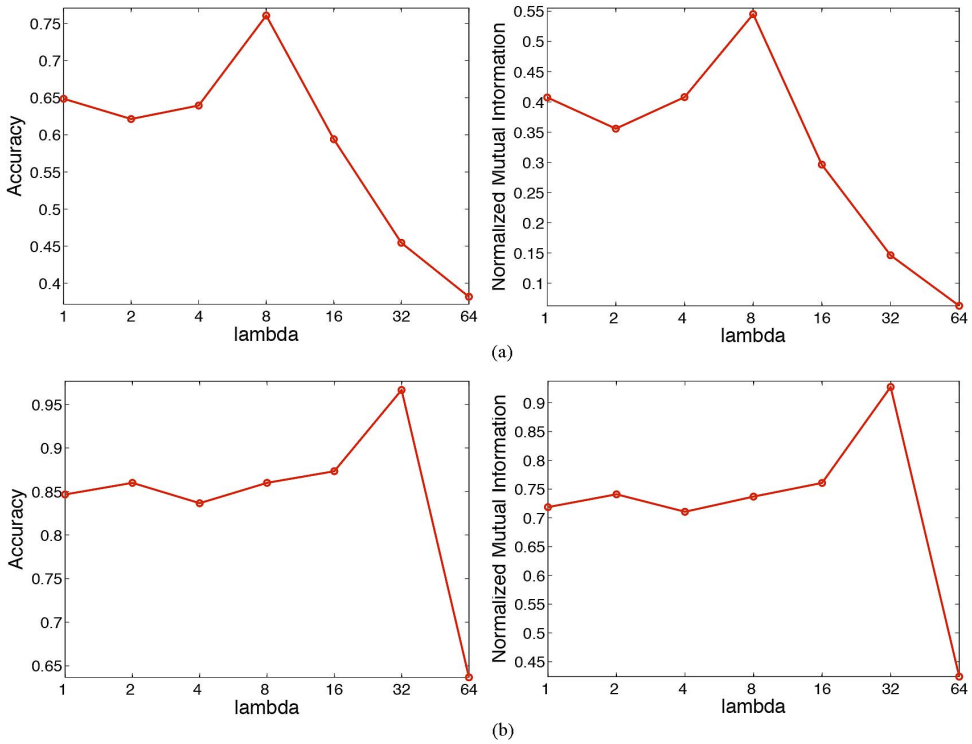


Fig. 4. Experiments on lamda. (a) Yale. (b) ORL.

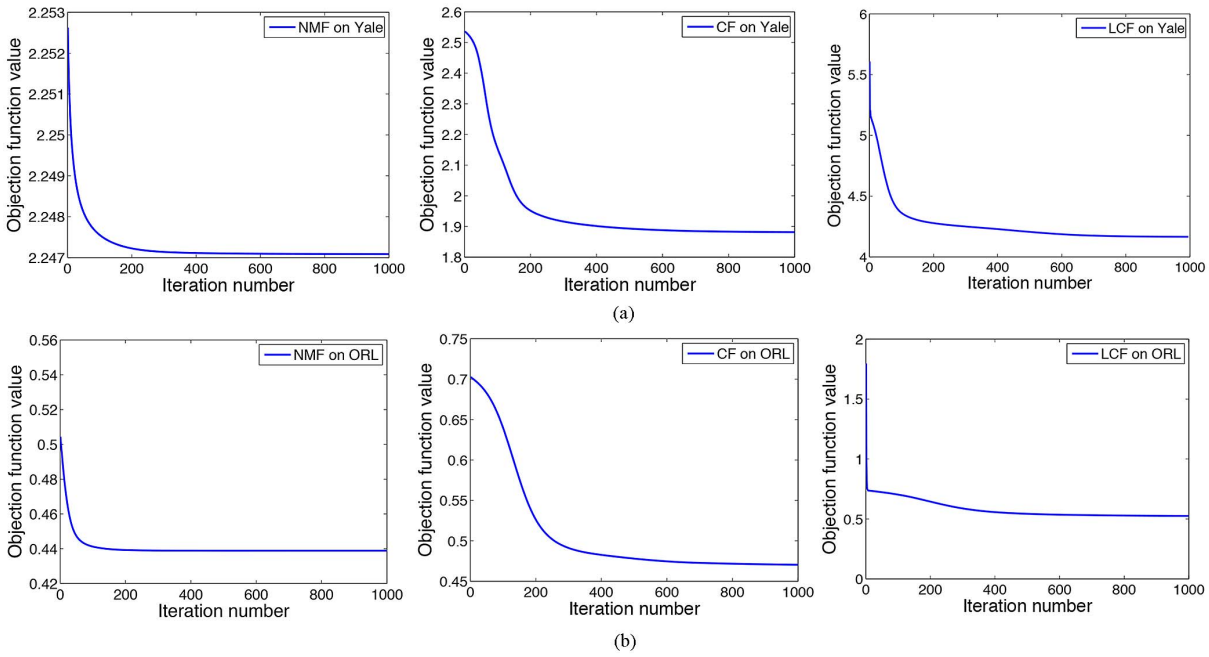


Fig. 5. Convergence curve of NMF, CF, and LCF. (a) Yale. (b) ORL.

from mixture of three Gaussians in a 2-D space were randomly generated. NMF and LCF were conducted to obtain three basis vectors and cluster these data points into three clusters. Fig. 3 shows that our LCF obtains better result than NMF. The bases obtained by NMF are far away from the original points. However, since we add a locality constraint, the three bases obtained by LCF exactly reside in the cluster centers, which leads to a better data representation. Note that the two matrix factors W and V were randomly initialized.

E. Parameter Selection

Our LCF algorithm has an essential parameter: the regularization parameter λ, which is used to control the importance of the locality constraint. In the following, we examine the impacts of this parameter on the performance of LCF.

We show the performance of LCF under different settings of λ on the two data sets in Fig. 4 from accuracy and normalized mutual information, respectively. For brevity, we just tested the impact of λ on the performance of three clusters. To demonstrate that our algorithm is not so sensitive to the parameter, we investigated the impacts of our parameter from a much wider range. For each data set, we varied λ as different number of power of two and found the best result. It is easy to see that LCF can achieve good performance with the λ varying from 2^0 to 2^6 on both Yale and ORL data sets. For the Yale data set, the best result is achieved when $\lambda = 8$ and for the ORL data set, the best result is achieved when $\lambda = 32$.

F. Convergence Study

The updating rules for minimizing the objective function of LCF are essentially iterative. We have proved that these rules are convergent. Here, we investigate the convergence rate of our iterative update rules.

Fig. 5 shows the convergence curves of NMF, CF, and LCF on both data sets. For each figure, the y-axis is the value of objective function and the x-axis denotes the number of iterations. We can see that the multiplicative updating rules for all the three algorithms converge very fast, usually within 200 iterations. Especially, our LCF algorithm has a faster convergence rate than the original CF algorithm.

VII. Conclusion

In this paper, we proposed a novel matrix factorization method, called LCF. This method enforces a locality constraint into the traditional concept factorization. By requiring the concepts (basis vectors) to be as close to the original data points as possible, each datum can be represented by a linear combination of only a few nearby basis concepts, thus achieving sparsity and locality simultaneously. The experimental results on two standard face databases have demonstrated the effectiveness of our approach over other matrix factorization techniques, especially for the data clustering applications.

References

[1] J. Fan, Y. Gao, and H. Luo, "Integrating concept ontology and multitask learning to achieve more effective classifier training for multilevel image annotation," *IEEE Trans. Image Process.*, vol. 17, no. 3, pp. 407–426, Mar. 2008.

[2] J. Fan, Y. Shen, N. Zhou, and Y. Gao, "Harvesting large-scale weakly-tagged image databases from the web," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2010, pp. 802–809.

[3] J. Fan, Y. Shen, C. Yang, and N. Zhou, "Structured max-margin learning for inter-related classifier training and multilabel image annotation," *IEEE Trans. Image Process.*, vol. 20, no. 3, pp. 837–854, Mar. 2011.

[4] D. Tao, X. Li, X. Wu, and S. J. Maybank, "General averaged divergence analysis," in *Proc. IEEE Int. Conf. Data Mining*, Oct. 2007, pp. 302–311.

[5] D. Tao, X. Li, X. Wu, and S. J. Maybank, "Geometric mean for subspace selection," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 31, no. 2, pp. 260–274, Feb. 2009.

[6] W. Liu, D. Tao, and J. Liu, "Transductive component analysis," in *Proc. IEEE Int. Conf. Data Mining*, Dec. 2008, pp. 433–442.

[7] X. He, "Laplacian regularized d-optimal design for active learning and its application to image retrieval," *IEEE Trans. Image Process.*, vol. 19, no. 1, pp. 254–263, Jan. 2010.

[8] X. He, M. Ji, and H. Bao, "Graph embedding with constraints," in *Proc. Int. Joint Conf. Artif. Intell.*, 2009, pp. 1065–1070.

[9] D. Cai, C. Zhang, and X. He, "Feature selection for multi-cluster data," in *Proc. 16th ACM SIGKDD Conf. Knowl. Discovery Data Mining*, 2010, pp. 333–342.

[10] B. A. Olshausen and D. J. Field, "Emergence of simple-cell receptive field properties by learning a sparse code for natural images," *Nature*, vol. 381, no. 6583, pp. 607–609, 1996.

[11] M. Zheng, J. Bu, C. Chen, C. Wang, L. Zhang, G. Qiu, *et al.*, "Graph regularized sparse coding for image representation," *IEEE Trans. Image Process.*, vol. 20, no. 5, pp. 1327–1336, May 2011.

[12] D. D. Lee and H. S. Seung, "Learning the parts of objects by non-negative matrix factorization," *Nature*, vol. 401, pp. 788–791, Aug. 1999.

[13] D. D. Lee and H. S. Seung, "Algorithms for non-negative matrix factorization," in *Proc. Adv. NIPS*, 2001, pp. 1–7.

[14] W. Xu and Y. Gong, "Document clustering by concept factorization," in *Proc. ACM Int. Conf. Res. Develop. Inf. Retr.*, Jul. 2004, pp. 202–209.

[15] D. Cai, X. He, X. Wu, and J. Han, "Non-negative matrix factorization on manifold," in *Proc. 8th IEEE Int. Conf. Data Mining*, Dec. 2008, pp. 63–72.

[16] D. Cai, X. He, J. Han, and T. Huang, "Graph regularized non-negative matrix factorization for data representation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 33, no. 8, pp. 1548–1560, Jun. 2011.

[17] D. Cai, X. He, and J. Han, "Locally consistent concept factorization for document clustering," *IEEE Trans. Knowl. Data Eng.*, vol. 23, no. 6, pp. 902–913, Jun. 2011.

[18] P. O. Hoyer, "Non-negative matrix factorization with sparseness constraints," *J. Mach. Learn. Res.*, vol. 5, pp. 1457–1469, Dec. 2004.

[19] K. Yu, T. Zhang, and Y. Gong, "Nonlinear learning using local coordinate coding," in *Proc. Adv. Neural Inf. Process. Syst.*, 2009, pp. 2223–2231.

[20] S. Li, X. Hou, H. Zhang, and Q. Cheng, "Learning spatially localized, parts-based representation," in *Proc. IEEE Int. Conf. Comput. Vis. Pattern Recognit.*, Feb. 2001, pp. 207–212.

[21] W. Xu, X. Liu, and Y. Gong, "Document clustering based on non-negative matrix factorization," in *Proc. Int. Conf. Res. Develop. Inf. Retr.*, Aug. 2003, pp. 267–273.

[22] D. Cai, X. He, X. Wang, H. Bao, and J. Han, "Locality preserving nonnegative matrix factorization," in *Proc. Int. Joint Conf. Artif. Intell.*, 2009, pp. 1010–1015.

[23] J. P. Brunet, P. Tamayo, T. R. Golub, and J. P. Mesirov, "Metagenes and molecular pattern discovery using matrix factorization," *Proc. Nat. Acad. Sci.*, vol. 101, no. 12, pp. 4164–4169, Mar. 2004.

[24] H. Kim and H. Park, "Sparse non-negative matrix factorization via alternating non-negativity-constrained least squares for microarray data analysis," *Bioinformatics*, vol. 23, no. 12, pp. 1495–1502, 2007.

[25] D. Greene, G. Cagney, N. Krogan, and P. Cunningham, "Ensemble non-negative matrix factorization methods for clustering protein-protein interactions," *Bioinformatics*, vol. 24, no. 15, pp. 1722–1728, 2008.

[26] G. Zhou, Z. Yang, S. Xie, and J. Yang, "Online blind source separation using incremental nonnegative matrix factorization with volume constraint," *IEEE Trans. Neural Netw.*, vol. 22, no. 4, pp. 550–560, Apr. 2011.

[27] C. Ding, T. Li, and M. I. Jordan, "Convex and semi-nonnegative matrix factorizations," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 32, no. 1, pp. 44–55, Jan. 2010.

[28] D. Donoho and V. Stodden, "When does non-negative matrix factorization give a correct decomposition into parts?" in *Proc. Adv. Nural Inf. Process. Syst.*, 2004, pp. 1–5.

[29] Y. Chen, J. Zhang, D. Cai, W. Liu, and X. He, "Nonnegative local coordinate factorization for image representation," *IEEE Trans. Image Process.*, vol. 22, no. 3, pp. 969–979, Mar. 2013.

[30] B. Gao, W. L. Woo, and S. S. Dlay, "Variational regularized 2-D nonnegative matrix factorization," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 23, no. 5, pp. 703–716, May 2012.

[31] Y. Wang and Y. Jia, "Fisher non-negative matrix factorization for learning local features," in *Proc. Asian Conf. Comput. Vision*, 2004, pp. 1–6.

[32] O. Zoidi, A. Tefas, and I. Pitas, "Multiplicative update rules for concurrent nonnegative matrix factorization and maximum margin classification," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 24, no. 3, pp. 422–434, Mar. 2013.

[33] N. Guan, D. Tao, Z. Luo, and B. Yuan, "Online nonnegative matrix factorization with robust stochastic approximation," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 23, no. 7, pp. 1087–1099, Jul. 2012.

[34] H. Kuhn and A. Tucker, "Nonlinear programming," in *Proc. 2nd Berkeley Symp. Math. Stat. Probab.*, 1951, pp. 481–492.

[35] J. Kivinen and M. Warmuth, "Additive versus exponentiated gradient updates for linear prediction," *J. Inf. Comput.*, vol. 132, no. 1, pp. 1–64, 1997.
[36] C. J. Lin, "Projected gradient methods for nonnegative matrix factorization," *Neural Comput.*, vol. 19, no. 10, pp. 2756–2779, 2007.
[37] C. J. Lin, "On the convergence of multiplicative update algorithms for nonnegative matrix factorization," *IEEE Trans. Neural Netw.*, vol. 18, no. 6, pp. 1589–1596, Nov. 2007.
[38] M. W. Berry, M. Browne, A. N. Langville, V. P. Pauca, and R. J. Plemmons, "Algorithms and applications for approximate nonnegative matrix factorization," *Comput. Stat. Data Anal.*, vol. 52, no. 1, pp. 155–173, 2007.
[39] F. Sha, Y. Lin, L. Saul, and D. Lee, "Multiplicative updates for nonnegative quadratic programming," *Neural Comput.*, vol. 19, no. 8, pp. 2004–2031, 2007.
[40] L. Lovasz and M. Plummer, *Matching Theory*. North Holland, Budapest: Akadémiai Kiadó, 1986.
[41] H. Lee, A. Battle, R. Raina, and A. Ng, "Efficient sparse coding algorithms," in *Proc. Adv. Neural Inf. Process. Syst.*, 2007, pp. 1–8.

Haifeng Liu (M'11) is an Associate Professor with the College of Computer Science, Zhejiang University, Hangzhou, China. Her current research interests include machine learning, pattern recognition, web mining, and information dissemination.

Zheng Yang received the B.S. degree in network engineering from Sun Yat-Sen University, Guangzhou, China, in 2010. He is currently pursuing the Ph.D. degree in computer science with Zhejiang University, Hangzhou, China.

His current research interests include machine learning and data mining.

Ji Yang is currently pursuing the bachelor's degree with the College of Computer Science, Zhejiang University, Hangzhou, China.

His current research interests include machine learning, data mining, and information retrieval.

Zhaohui Wu (M'00–SM'05) received the Ph.D. degree in computer science from Zhejiang University, Hangzhou, China, and Kaiserslautern University, Kaiserslautern, Germany, in 1993.

He is currently a Professor with the College of Computer Science and the Vice Principal with Zhejiang University. His current research interests include distributed artificial intelligence, grid computing and systems, and embedded ubiquitous computing.

Dr. Wu is a Senior Member of the IEEE Computer Society.

Xuelong Li (M'02–SM'07–F'12) is a Full Professor with the Center for OPTical IMagery Analysis and Learning (OPTIMAL), State Key Laboratory of Transient Optics and Photonics, Xi'an Institute of Optics and Precision Mechanics, Chinese Academy of Sciences, Xi'an, China.