A hierarchy weighted low-rank representation for image clustering and classification

Zhiqiang Fu[a], Yao Zhao[a,*], Dongxia Chang[a]

[a]*Institute of Information Science, School of Computer and Information Technology*
Beijing Key Laboratory of Advanced Information Science and Network Technology
Beijing Jiaotong University, Beijing,100044, China

Abstract

Low-rank representation(LRR), which is a powerful method to find the low-dimensional subspace structures embedded in high-dimensional data space, has been used in both unsupervised learning and semi-supervised classification. LRR aims to represent each data point with the lowest rank linearly. In fact, this method doesn't consider the geometrical structures of the data. Thus the similarity and local structure might be lost in the process of learning. Motivated by this, a novel hierarchy weighted low-rank representation (HWLRR) is proposed in this paper. In the new algorithm, a hierarchy weighted matrix is defined to find more samples that may belong to the same subspace by using the affinity propagation. By taking advantage of the affinity propagation, our proposed method can preserve both the local structure and the global structure of the whole dataset. The experimental results on both unsupervised learning and semi-supervised classification demonstrate the superiority of our proposed method.

Keywords: Low-Rank Representation, Clustering, Semi-supervised Learning, Similarity Graph Construction

*Corresponding author
Email address: yzhao@bjtu.edu.cn (Yao Zhao)

1. Introduction

Low-rank representation (LRR) [1, 2] is a promising low-rank learning method and it aims to find representation with the lowest rank for any given dataset. Because of its excellent performance in capturing the low-dimensional structures embedded in the data space, it has been extensively used in image classification [3], image clustering [4], face recognition [5], feature extraction [6], subspace segmentation [7] and face sketch synthesis[8].

As mentioned in matrix completion and low-rank matrix recovery, it always assumes that the data points usually linearly correlate with each other in the subspace if they are drawn from the same pattern. Therefore, data points sampled from different clusters can be regarded as samples nearly drawn from a union of multiple low-rank subspaces. Consequently, it is desirable to develop a technique to find the intrinsic geometrical subspace structure of the given data. In fact, numerous low-rank learning methods have been developed in the past. Among them, LRR and robust principal component analysis (RPCA) [9] are two typical methods. The RPCA implicitly assumes that the embedded data structure lies in just one low-rank subspace. This neglects the individual subspace's specific property. By contrast, the LRR method explicitly considers the data structures approximately embed in several low-dimensional subspaces.

The LRR seeks for the lowest-rank representation based on the assumption that the data approximately spans several low-dimensional subspaces. To this end, plenty of methods basing on LRR have been proposed to solve clustering and classification problems. In Ref.[1], the LRR is developed to cluster the samples into their respective subspaces and remove possible outliers. In this method, they choose the observed data itself as the dictionary. However, the performance may depress if the observations are insufficient or corrupted. To overcome this shortcoming, the Latent LRR [2] was proposed to construct the dictionary using both unobserved and observed data. In fact, noiseless Latent LRR cannot obtain a unique solution. To address this issue, the robust Latent LRR was put forward, which choose the sparsest matrix from the solution set of

the Latent LRR [10]. To improve the computational speed, the LRR adopts a fixed-rank strategy that can reveal the structure of multiple subspaces in closed-form when the data is noiseless [11]. Moreover, Fang et al. [12, 13] combined LRR with the semi-supervised clustering as a solution to the classification.

Conventional LRR algorithms usually capture the subspace by learning the global structure of subspace but miss the local structure. However, to find the geometric structures embedded in the data, LRR should capture not only global relevance but also the local geometric structures. Trying to learn the local geometrical structures embedded in data space, many methods to learn manifold have been proposed, such as ISOMAP [14], Laplacian Eigenmap [15] and Neighborhood Preserving Embedding (NPE) [16].

Motivated by these methods, some modified LRR methods have been proposed. In Ref.[17], the LRR with an adaptive distance penalty was proposed, which uses the distance between two samples as penalty coefficients. This method learns not only the global subspace structure embedded in the dataset but also the local neighbor relationship among samples. Zheng et al. [18] proposed a graph regularized sparse coding aiming to take into account the local manifold of the data and learn the sparse representations. To remove striping noise in hyperspectral images, Lu et al. [19] proposed a novel graph-regularized LRR (GLRR) and improved their approach by adopting the graph technique. Similarly, a general Laplacian regularized low-rank representation framework for data representation was given in Ref.[20]. By making use of the graph regularizer, this method not only captures the global low-dimensional structures but also represents the intrinsic geometric information in data. Moreover, some recent evidence shows that the graph is a better choice than the pairwise for capturing the structure embedded in high-dimensional space [20].

Most of the modified LRR methods mentioned above use k-nearest neighbors to improve the performance. However, the number of neighbors k affects the performance of the algorithm obviously and k nearest neighbors may bring bad effect for representation[21]. Motivated by this, a hierarchy weighted low-rank representation (HWLRR) method is proposed in this paper. In this new

algorithm, a hierarchy weighted matrix is defined to capture the structure of the high hierarchy nearest neighbor relationship. Therefore, more structure information can be obtained. Experiments show that the algorithm we proposed has good results in image clustering and classification.

The remainder of our paper is organized as follows. In Section 2, we briefly review some related works of the LRR. Section 3 provides some definitions necessary for our approach. Then, we introduce the proposed hierarchy weighted low-rank representation in Section 4. Furthermore, the experimental results for image clustering and classification are given in Section 5. Finally, Section 6 concludes our paper.

2. Related Works

The LRR method has attracted much attention due to its overwhelming advantages in a wide range of real-world applications. It aims at capturing the global structure by computing the affinity of the undirected graph[1, 22]. It is based on the assumption that each data in a given dataset can be regarded as sampling from some independent low-dimensional subspaces. Therefore, it can be an effective way to find data points that belong to the same subspace.

For a given dataset Y, Y is sampled from a combination of low dimensional subspace $U_{j=1}^{k} S_j$, where S_1, S_2, S_3,...,S_k are low-dimensional subspaces. We assume that each subspace is independent and has no noisy data, then the low-dimensional subspace can be represented by the block distribution of the matrix. And the low-rank representation can be expressed as follows

$$\min_{Z} \ rank(Z), \ s.t. Y = AZ \tag{1}$$

where $rank(\cdot)$ is the rank of the matrix, A is the dictionary used to represent Y, Z is the coefficient matrix in which each z_i is the coefficient vector of y_i based on the dictionary matrix A. The low-rank constraints can ensure a strong correlation between the data obtained in the same subspace. But in practical application, noise is unavoidable. To improve the robustness, the low-rank

representation model is modified as

$$\min_{Z} \ rank(Z) + \lambda \|E\|_l, \ s.t. Y = AZ + E \tag{2}$$

where λ is a penalty parameter to balance the low-rank term and the reconstruction error, E is the error fitting matrix, and $\|\cdot\|_l$ is regularization of noise. The LRR theory shows that the minimum data rank can be solved by using the kernel norm of the minimization data, and the low-rank representation problem can be expressed as the following minimization problem

$$\min_{Z,E} \ \|Z\|_* + \lambda \|E\|_l, \ s.t. Y = AZ + E \tag{3}$$

where $\|\cdot\|_*$ is the nuclear norm defined as the sum of all singular values of the matrix. By using an appropriately dictionary A, the intrinsic structures of the data can be captured. Normally the data matrix Y is used as the dictionary [22], then the general LRR model can be expressed as

$$\min_{Z,E} \ \|Z\|_* + \lambda \|E\|_l, \ s.t. Y = YZ + E \tag{4}$$

In order to improve the performance of the LRR, plenty of efforts have been adopted by imposing penalty on Z and E. And different kinds of LRR-based methods were developed. To better handle the modified LRR method, a general rank minimization problem can be formulated as

$$\min_{Z} \ \|Z\|_* + Q(Z, E), \ s.t. Y = YZ + E \tag{5}$$

where Q is a penalty function about Z and E. In fact, the penalty function $Q(Z, E)$ shown in Eq.(5) is the most important part of the LRR model. A number of methods have been proposed to construct it. For example, Yin et al.[20] defined $Q(E, Z) = \lambda \|Z\|_1 + \beta tr\left(ZLZ^T\right) + \gamma \|E\|_1$ to learn the manifold structures within data.

3. Hierarchy Weighted Matrix

In this section, some definitions needed in the next section are given. For a given dataset $Y = [y_1, y_2, \cdots, y_n]$, it is usually assumed that the relationship

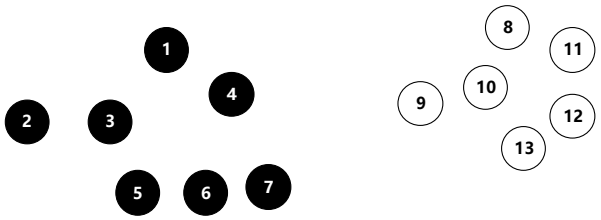

Fig. 1. Dataset Y. Here the points with the same color belong to the same subspace.

between a data point and its neighbors is linear [23]. Thus, the local geometry of all the data points can be effectively characterized by the linear combination of its neighbors. The k-nearest neighbors [24] are usually used. Intuitively, the data point and its k-nearest neighbors belong to the same subspace if k is small. This assumption has been widely accepted and has been used in many algorithms [25, 26]. But it is difficult to give an appropriate k. If k is small, some samples in the same subspace can be lost. In fact, not only the nearest neighbors but also the neighbors of the nearest neighbors probably belong to the same subspace. In order to make full use of the neighbors, a hierarchy weighted matrix is proposed in this section. To illustrate this further, some key definitions are defined as follows.

Definition 1 (First Order Adjacency Matrix). Given a dataset $Y = [y_1, y_2, \cdots, y_n]$, the first order adjacency matrix W can be constructed by the k-nearest neighbor based on Euclidean distance. Here $W_{i,j}$ shows whether there is a neighborhood relationship between y_i and y_j and it is defined as follows:

$$W_{i,j} = \begin{cases} 1, \ y_j \in N_k(y_i) \\ 0, \ otherwise \end{cases} \tag{6}$$

where $N_k(y_i)$ is the k-nearest neighbors set of data y_i.

Definition 2 (First Hierarchy Neighbor). Given a dataset $Y = [y_1, y_2, \cdots, y_n]$, W is the first order adjacency matrix obtained by Definition 1. If $W_{i,j} = 1$, then y_j is defined as the first hierarchy neighbor of y_i.

Here, for the given dataset showed in Fig.1, each point can find its two nearest neighbors as shown in Fig.2. And the first hierarchy neighbors of point

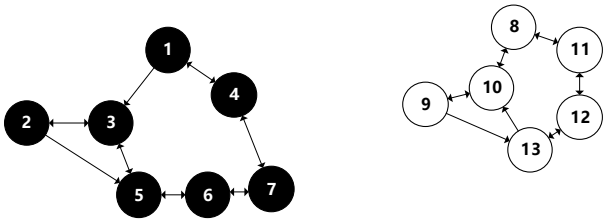

Fig. 2. The k-nearest neighbors of the dataset Y with $k = 2$.

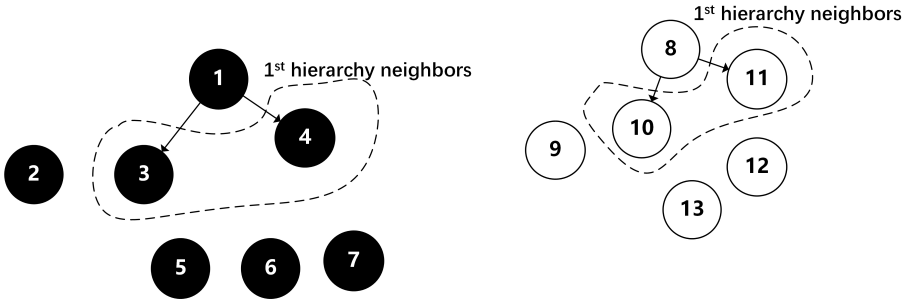

Fig. 3. The first hierarchy neighbors of point 1 and point 8 ($k = 2$). Here point 3 and 4 are the 2-nearest neighbors of point 1, so they are the first hierarchy neighbors of point 1. And point 10 and point 11 are the first hierarchy neighbors of point 8. However, many points in the same subspace are missed in the first hierarchy neighbors.

1 and point 8 are given in Fig.3. It is clearly that many data points that belong to the same subspace of point 1 and point 8 are lost by their first hierarchy neighbors. To address this issue, the higher hierarchy neighbor is defined based on the first hierarchy neighbor as following.

Definition 3 (Higher Hierarchy Neighbor). For a data point y_i, y_j is defined as y_i's $(l+1)$th hierarchy neighbor if y_j is its lth hierarchy neighbor's neighbor.

According to this, the second hierarchy neighbors of point 1 and point 8 can be found easily and are shown in Fig.4. It is obvious that the higher hierarchy neighbors are more than the first hierarchy neighbors and more neighbors can save more data structure. As Fig.5 shows, we can find all the neighbors on different hierarchies for point 1 and point 8.

Definition 2 and Definition 3 give the definitions of the hierarchy neighbors based on the k-nearest neighbors. For a given data point, we define that its

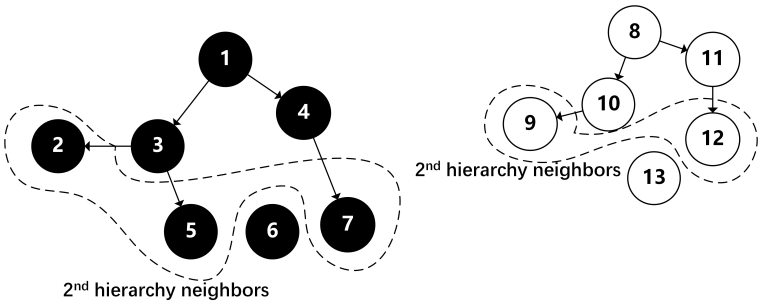

Fig. 4. The second hierarchy neighbors of point 1 and point 8 ($k = 2$). Here, the first hierarchy neighbors' neighbors are the second hierarchy neighbors.

neighbors in the same hierarchy should have equal importance and have different importance with others in different hierarchy. Here we assume that each point can move to another, and the transferring probability is defined.

Definition 4 (First Order Transfer Probability). For a data point y_i, the first order transfer probability from it to its first hierarchy neighbors can be defined as:

$$p = \frac{1}{k} \tag{7}$$

The first order transfer probability given in Definition 4 measures the probability of a point to its first hierarchy neighbors. Beside the first hierarchy neighbors, there are more higher hierarchy neighbor. So the higher order transfer probability is defined in the following.

Definition 5 (Higher Order Transfer Probability). For a data point y_i, y_j is the higher hierarchy neighbor of y_i. Then the higher order transfer probability from y_i to y_j is defined as follow:

$$p_{i,j} = \begin{cases} \left(\frac{1}{k}\right)^{l_j - l_i}, l_j > l_i \; and \; y_i \; , y_j \; \in H_i \\ 0, else \end{cases} \tag{8}$$

where l_i is the ordinal of the hierarchy which y_i lies on and H_i is the set containing the hierarchy neighbors of y_i.

Using the definition of the first order and higher order transfer probability, a completely reachable matrix can be defined by Definition 6.

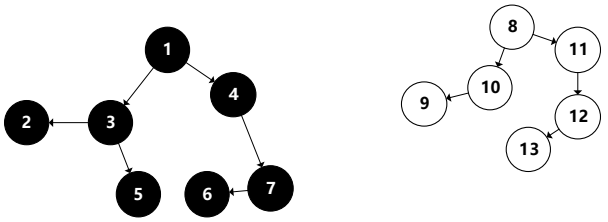

Fig. 5. All the hierarchy neighbors of point 1 and point 8. All the points can be found including first to third hierarchy neighbors.

Definition 6 (Completely Reachable Matrix). The completely reachable matrix P_s is a matrix composed of $p_{i,j}$ definded in Definition 5 and it can be obtained from a recursive formula as:

$$P_{i+1} = d\left(\frac{1}{k^{i+1}} \cdot f\left(f\left(P_i P_1\right) - f\left(P_i\right)\right)\right) + P_i \tag{9}$$

where $P_1 = W$, $d\left(\cdot\right)$ is a function defined as

$$d\left(X\right) = \begin{cases} 0, i = j \\ x_{i,j}, i \neq j \end{cases}$$

where X is a matrix and $x_{i,j}$ is the element on the ith row and jth column. $f(\cdot)$ is an approximate symbolic function which is defined as:

$$f\left(x\right) = \begin{cases} 1, x > 0 \\ 0, x \leq 0 \end{cases}$$

the iteration stops when $P_{i+1} = P_i$ and a steady completely reachable matrix P_s is obtained. For convenience, a penalty matrix, hierarchy weighted matrix T, is defined in Definition 7 basing the completely reachable array P_s.

Definition 7 (Hierarchy Weighted Matrix). The hierarchy weighted matrix T represents the cost of transfering from y_i to y_j and it is defined as:

$$T_{i,j} = \begin{cases} \frac{1}{P_s(i,j)}, P_s(i,j) \neq 0 \\ \infty, P_s(i,j) = 0 \end{cases} \tag{10}$$

and when $P_s(i,j) = 0$, we use 10^6 to represent infinite. Here T is a hierarchy weighted in which the samples' weight depends on the hierarchy they lying on like Fig.6 and the value on the arraw is the weight of each point.

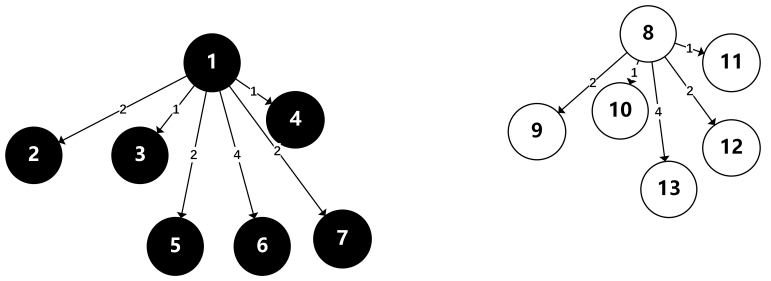

Fig. 6. The hierarchy weighted matrix T ($k = 2$). The value on the arrow is the weight of each point.

The hierarchy weighted matrix T not only takes into account the close and similar data points in the Euclidean space but also takes into the long distance and less similar data points by affinity propagation.

4. Hierarchy Weighted Low-Rank Representation

As mentioned in [27], a good penalty function should reveal the intrinsic dimensionality [28] or complexity of the data by capturing the global structures embedded in the data. Many methods construct this penalty function based on the k-nearest neighbors. Therefore, they are often sensitive to local data noise and error. Moreover, these methods generally use fixed parameter k and may fail to capture datum-adaptive neighborhoods. To improve the performance of the methods based on the k-nearest neighbors, a hierarchy weighted low-rank representation algorithm (HWLRR) is proposed in this section.

4.1. The HWLRR

In general, the LRR method usually consists of the following two steps. First, the samples which may belong to the same subspace are found for each data point. Then, each data point is linearly represented using the least number of these samples.

In our HWLRR, the hierarchy weighted matrix proposed in Section 3 is used to find the samples which may belong to the same subspace. The hierarchy

weighted matrix T not only learns the local structure based on k-nearest neighbors but also takes into the global structure by affinity propagation. Thus, the following function is defined to determine the coefficients between corresponding pairs of samples:

$$\sum_{i,j} T_{i,j} Z_{i,j} \tag{11}$$

where $T_{i,j}$ is the cost of transferring from y_i to y_j and Z is the LRR matrix as defined in Eq.5. $Z_{i,j}$ is the related coefficients between y_i and y_i. Minimizing Eq.11 can assign the large coefficient to the edge between points with the low cost and attach the small coefficient to the edge between points with the high cost. Combining Eq.11 with the LRR, the proposed hierarchy weighted low-rank representation (HWLRR) method is defined as follows

$$\min_{Z} \ \|Z\|_* + \beta \sum_{i,j} T_{i,j} Z_{i,j}, s.t.\ Y = YZ, Z \geq 0 \tag{12}$$

where Z is the low-rank representation matrix of the dataset. $\beta > 0$ is a balance parameter. Using matrix operation to replace the element operation, the HWLRR can be reformulated as

$$\min_{Z} \ \|Z\|_* + \beta tr\left(\Theta\left(T \otimes Z\right)\right), s.t.\ Y = YZ, Z \geq 0 \tag{13}$$

where $tr\left(*\right)$ is the trace of matrix, Θ is a matrix whose elements are all 1 and $\otimes$ is Hadamar operator.

But in the real world, noise always exist in the data, and it will affect the data processing. Therefore, we should consider the presence of noise in the data Y. Here, we introduce the noise matrix E to fit the noise, and the model can be changed into:

$$\min_{Z,E} \ \|Z\|_* + \lambda \|E\|_1 + \beta tr\left(\Theta\left(T \otimes Z\right)\right), s.t.\ Y = YZ + E, Z \geq 0 \tag{14}$$

where $\|\cdot\|_1$ is the l_1-norm.

To solve the HWLRR problems, we first introduce a variable matrix J instead of Z then the HWLRR can be rewritten as:

$$\min_{Z,E} \ \|Z\|_* + \lambda \|E\|_1 + \beta tr\left(\Theta\left(T \otimes J\right)\right), s.t.\ Y = YZ + E, Z \geq 0, Z = J \tag{15}$$

The enhanced Lagrange function [29] can be used to solve this problem. The Lagrange function of the problem is expressed as follows:

$$\begin{aligned} & L\left(Z, E, J, M_1, M_2, \mu\right) \\ & =\|Z\|_*+\lambda\|E\|_1+\beta tr\left(\Theta\left(T \otimes J\right)\right) \\ & +\left\langle M_1, Y-YZ-E\right\rangle+\left\langle M_2, Z-J\right\rangle \\ & +\frac{\mu}{2}\left(\|Y-YZ-E\|_F^2+\|Z-J\|_F^2\right) \end{aligned} \tag{16}$$

where M_1, M_2 are two Lagrange multiplier, $\mu > 0$ is penalty coefficient. $\langle \cdot, \cdot \rangle$ means inner product. Simplify formula Eq.(16), we can get:

$$\begin{aligned} & L\left(Z, E, J, M_1, M_2, \mu\right) \\ & =\|Z\|_*+\lambda\|E\|_1+\beta tr\left(\Theta\left(T \otimes J\right)\right)+ \\ & \frac{\mu}{2}\left(\|Y-YZ-E+M_1/\mu\|_F^2+\|Z-J+M_2/\mu\|_F^2\right) \\ & -\frac{1}{2\mu}\left(\|M_1\|_F^2+\|M_2\|_F^2\right) \end{aligned} \tag{17}$$

The problem Eq.(17) is unconstrained, so we can fix other variables, and update Z, E, J as follow.

$$Z_{k+1}=\arg\ \min_Z\|Z\|_*+\frac{\mu}{2}\left(\left\|Y-YZ-E_k+\frac{M_{1,k}}{\mu}\right\|_F^2+\left\|Z-J_k+\frac{M_{2,k}}{\mu}\right\|_F^2\right) \tag{18}$$

$$E_{k+1}=\arg\ \min_E\ \|E\|_1+\frac{\mu}{2}\left\|Y-YZ_k-E+\frac{M_{1,k}}{\mu}\right\|_F^2 \tag{19}$$

$$J_{k+1}=\arg\ \min_J\ \beta tr\left(\Theta\left(T \otimes J\right)\right)+\frac{\mu}{2}\left\|Z_k-J+\frac{M_{2,k}}{\mu}\right\|_F^2 \tag{20}$$

The premise of formula Eq.(18) is :

$$q\left(Z\right)=\frac{\mu}{2}\left(\left\|Y-YZ-E_k+\frac{M_{1,k}}{\mu}\right\|_F^2+\left\|Z-J_k+\frac{M_{2,k}}{\mu}\right\|_F^2\right)$$

By linearizing the quadratic term and adding the proximal term, the following approximate formula can be obtained:

$$\begin{aligned} & Z_{k+1}=\arg\ \min_Z\|Z\|_*+q\left(Z_k\right)+\left\langle\nabla_Z q, Z-Z_k\right\rangle+\frac{\mu\eta_Z}{2}\left(\|Z-Z_k\|_F^2\right) \\ & =\arg\ \min_Z\|Z\|_*+\frac{\beta\eta_Z}{2}\|Z-Z_k+\frac{1}{\eta_Z}\left(-Y^T\left(Y-YZ-E+M_1/\mu\right)\right) \\ & +\left(Z-J+M_2/\mu\right)\|_F^2 \end{aligned} \tag{21}$$

$\nabla_Z q$ is the partial derivative of q relative to Z. This problem can be obtained by using singular value threshold operator, and we can get:

$$\begin{aligned} Z_{k+1} = \Phi_{\frac{1}{\eta_Z \beta}}(Z_k + \frac{1}{\eta_Z}(Y^T(Y - YZ_k - E_k + M_{1,k}/\mu) \\ -(Z_k - J_k + M_{2,k}/\beta))) \end{aligned} \tag{22}$$

where Φ is SVT operator [30] and $\frac{1}{\eta_Z \beta}$ is the threshold of Φ. Eq.(19) can be solved directly by the expansion operator:

$$E_{k+1} = \Psi_{\frac{\lambda}{\mu}}(Y - YZ + M_1/\mu) \tag{23}$$

where Ψ is expansion operator, and $\frac{\lambda}{\mu}$ is the threshold of expansion operator. Eq.(20) has independent solutions for different samples, so it can be decomposed into n independent subproblems to obtain the closed form solutions:

$$\begin{aligned} &[J_{k+1}]_{*,i} = \arg\min_{[H]_{*,i}} \beta [T]_{*,i}^T [J]_{*,i} + \frac{\mu}{2} \left\| [Z_k]_{*,i} - [J]_{i,*} + [M_{2,k}]_{*,i}/\mu \right\|_F^2 \\ &= [Z_k]_{*,i} + [M_{2,k}]_{*,i} \Big/ \mu - \beta [T]_{*,i}/\mu \ \ (i = 1, ..., n) \end{aligned} \tag{24}$$

Other parameters are iterated by using the formula:

$$M_{1\ k+1} = M_{1\ k} + \mu_k (Y_{k+1} - Y_{k+1} Z_{k+1} - E_{k+1}) \tag{25}$$

$$M_{2\ k+1} = M_{2\ k} + \mu_k (Z_{k+1} - J_{k+1}) \tag{26}$$

$$\mu_{k+1} = \min(\mu_{\max}, \rho \mu_k) \tag{27}$$

4.2. Description of the HWLRR

In our HWLRR algorithm, a hierarchy weighted matrix is defined to find data points that may belong to the same subspace. And the hierarchy weighted matrix is constructed by affinity propagation. Then add this matrix to the original LRR and the new model can be solved using the enhanced Lagrange function.

The HWLRR algorithm can be described as follows:

1. Construct the first order adjacency matrix W by the k-nearest neighbors of each sample.

Algorithm 1 Solving the HWLRR

Input: dataset Y, parameter $\lambda > 0$, $\beta > 0$

Output: optimal solutions Z and E

Initialize: $Z = J = M_2 = 0$, $E = M_1 = 0$, $W = T = 0$, $\mu_0 = 1$, $u_{\max} = 10000$, $\eta_Z = 2\|Y\|^2$, $\xi = 0.00001$, $\rho = 1.01$, $t = 0$

Pretreatment: Finding k nearest neighbors of data points and the k-nearest neighbors are used to obtain the nearest neighbor matrix W, and $T_{i,j} = \begin{cases} \frac{1}{P_{s(i,j)}}, P_{s(i,j)} \neq 0 \\ \infty, P_{s(i,j)} = 0 \end{cases}$ is used to obtain the hierarchy weighted T

While not converged **do**

1: Update Z by $Z_{k+1} = \Phi_{\frac{1}{n_z\beta}}(Z_k + \frac{1}{\eta_Z}(Y^T(Y - YZ_k - E_k + M_{1,k}/\mu) - (Z_k - J_k + M_{2k}/\beta))$

2: Update E by $E_{k+1} = \Psi_{\frac{\lambda}{n}}(Y - YZ + M_1/\mu)$

3: Update J by $[J_{k+1}]_{*,i} = [Z_k]_{*,i} + [M_{2,k}]_{*,i}/\mu - \beta[T]_{*,i}/\mu (i = 1, \ldots, n)$

4: Update M_1 by $M_{1\ k+1} = M_{1\ k} + \mu_k(Y_{k+1} - Y_{k+1}Z_{k+1} - E_{k+1})$

5: Update M_2 by $M_{2\ k+1} = M_{2\ k} + \mu_k(Z_{k+1} - J_{k+1})$

6: Update μ by $\mu_{k+1} = \min(\mu_{\max}, \rho\mu_k)$

7: t = t + 1

8: check the convergence conditions $\|Z_{k+1} - Z_k\| / \|Z_k\| \geq \xi$

End while

2. Figure out the completely reachable matrix P_s.
3. Obtain the hierarchy weighted matrix T to represent the penalty of each point.
4. Add matrix T to the original LRR model as Eq.(13) and solve the model using the Algorithm 1.

4.3. Complexity Analysis

In our HWLRR, the computational cost includes two parts, affinity propagation and solving the objective function. For a given dataset $Y = [y_1, y_2, \cdots, y_n]$, each data y_i has d dimensions and has k-nearest neighbors. The computational complexity of affinity propagation can be figured out by Eq.(9). Because Eq.(9)

is a recurrence formula, its computational complexity is $O(t_1(n^2+5n))$, where t_1 is the number of iterations and $t_1 \leq log_k n$. If r is the lowest rank which the algorithm can learn and t_2 is the number of total iterations, the computational cost of Eq.(22) is $O(rn^2)$. Meanwhile, in Eq.(23) and Eq.(24), we can update the matrix whose complexity is $O(n^2)+O(n)$. Then, the computational complexity of our algorithm is $O(t_1(n^2+5n)+t_2(rn^2+n^2+n))$.

5. Experiments

In this section, the effectiveness of the HWLRR algorithm is verified through the experiments. Here, it is tested in clustering and semi-supervised classification respectively. In the experiments, six representative datasets, including the COIL20, UMIST, ORL, MNIST, C-Cube and PenDigits datasets are used. All algorithms are run on MATLAB 2019b on a PC with AMD(R) Ryzen R7-3700X CPU at 4.2 GHz, RAM 16.00GB, and Windows 10 operating system.

5.1. Datasets

- The COIL20[1] [31] is a dataset proposed by Columbia University. There are 20 objects in this dataset, and each of them was photographed 72 images with a different angle. For computational efficiency, we turn the images from 128×128 pixels to 32×32 pixels and transform them into gray-scale so that they can be converted as 1024 dimensional vectors.

- The ORL dataset[2][32] contains ten human's frontal face images, and each person was photographed with 40 images. For each person, the photos are taken varying the facial expressions, lighting, and facial details.

- UMIST[3] [33] contains 564 images of 20 subjects. For each subject, it consists of images varying in race, sex or appearance and contains a range

[1] http://www.cs.columbia.edu/CAVE/software/softlib/coil-20.php
[2] http://www.cl.cam.ac.uk/research/dtg/attarchive/facedatabase.html
[3] https://www.sheffield.ac.uk/eee/research/iel/research/face

of poses from profile to frontal views. Each person has its label as 1a, 1b, ... 1t etc. and the images belong to the same person are numbered chronologically. All the images are 220 × 220 pixels and saved in PGM format.

- MNIST[4] [34] is one of most famous datasets in machine learning. It contains 70000 handwritten images of 10 numbers and each image is 28×28. In the experiments, we randomly selected 200 images from each number and made up a new dataset with 2000 digits.

- The C-Cube dataset[5] [35] is a cursive word dataset with 50000 samples. It includes both upper and lower case of 26 letters. We randomly selected 40 samples of each class and made up a new dataset with 2000 digits.

- PenDigits dataset[6] [36] is a handwritten images for 10 numbers. It was created as a digit database by collecting 250 samples from 44 writers. In the experiments, we randomly selected 100 images from each number and made up a new dataset with 1000 digits.

5.2. Image Clustering

To extensively assess the clustering performance of our approach, we compare it with the following methods:

- Kmeans++ clustering algorithm (Kmeans++) [37]
- Spectral clustering (SC) [38]
- Low-rank representation (LRR) [39]
- Latent low-rank representation (LatLRR) [6]
- Non-negative sparse laplacian regularized low-rank representation (NSLLRR) [20]

[4] http://yann.lecun.com/exdb/mnist/
[5] http://ccc.idiap.ch
[6] http://archive.ics.uci.edu/ml/datasets/Pen-Based+Recognition+of+Handwritten+Digits

- Low-rank representation with adaptive distance penalty (LRRADP) [17]
- Adaptive weighted nonnegative low-rank representation (AWNLRR) [21]

K-means++ is an advanced K-means clustering algorithm which can be regarded as a bassline for clustering. Spectral clustering is a classical clustering algorithm that uses the feature of data to obtain an affinity matrix and cluster data by spectral decomposition. LRR, LatLRR, NSLLRR, LRRADP and AWNLRR are five LRR algorithms that can be benchmarked as the LRR. The SC is used to clustering the data by using the affinity matrix produced by the five LRR algorithms. To compare the qualities of the final clustering results, the adjusted rand index (ARI), normalized mutual information (NMI), accuracy (ACC), precision and F-score are employed. For all the experiments, the run has been repeated ten times to eliminate the effects of randomness embedded in the algorithms. The mean, standard deviation of metrics, and the time cost are given.

Table 1 - 6 show the clustering performance metrics and running time of different methods on the datasets. From these tables, it may be realized that the proposed method obtains the best performance in almost all cases. It means that the hierarchy weighted matrix can capture the structure embedded in the data space better. From the comparison of Kmeans++, SC and LRR methods, it is obvious that in most cases, low-rank representation feature is effective in obtaining a better clustering performance than the original pixel feature. But in some datasets as COIL20, the pixel feature can also obtain pretty good clustering performance. From the comparison of LRR, LatLRR and NSLLRR, one can conclude that introducing manifold information (i.e., Laplacian regularization) can obtain a better performance. As the Laplacian regularization is constructed by nearest neighbor relationships, this also proves the benefit of keeping the local structure of data. From all the tables, we can find that AWNLRR and LRRADP obtain better performance than NSLLRR. NSLLRR mainly introduces the similar Laplacian regularization to LRR to capture the local relationship of data. AWNLRR and LRRADP use the Euclidean distance

as a constraint to keep the geometric relationship among data points. One can conclude that the distance constraint is more effective than the Laplacian term in learning the intrinsic relationships of data. Both NSLLRR and HWLRR use the nearest neighbor relationship to learn the local structure in the data. However, from the comparison of the two methods, it is obvious that the proposed method performs much better than NSLLRR. This is due to that high hierarchy neighbors preserve more global structure embedded in the data. However, the computation complexity of the proposed algorithm is relatively high where the bottleneck is the number of iterations.

Table 1
Clustering performance metrics (mean ± standard deviation) and running times of different methods on the COIL20 database. Bold numbers denote the best results.

Methods	ARI	NMI	ACC	Precision	F-score	Time(s)
Kmeans++	0.497±0.036	0.718±0.017	0.545±0.047	0.614±0.023	0.525±0.034	0.12
SC	0.572±0.019	0.759±0.008	0.635±0.018	0.618±0.019	0.594±0.018	12.93
LRR	0.414±0.024	0.663±0.012	0.514±0.026	0.485±0.017	0.445±0.022	74.96
LatLRR	0.184±0.012	0.452±0.012	0.277±0.014	0.307±0.024	0.234±0.011	221.82
NSLLRR	0.497±0.018	0.707±0.011	0.607±0.017	0.542±0.015	0.522±0.017	3221.95
LRRADP	0.621±0.031	0.792±0.013	0.678±0.039	0.673±0.019	0.640±0.029	1258.43
AWNLRR	0.693±0.017	0.838±0.006	0.740±0.016	0.730±0.011	0.708±0.016	68.91
HWLRR	**0.778±0.031**	**0.908±0.011**	**0.809±0.034**	**0.845±0.016**	**0.790±0.029**	1014.70

Table 2
Clustering performance metrics (mean ± standard deviation) and running times of different methods on the ORL database. Bold numbers denote the best results.

Methods	ARI	NMI	ACC	Precision	F-score	Time(s)
Kmeans++	0.309±0.034	0.713±0.018	0.480±0.036	0.438±0.028	0.328±0.033	0.03
SC	0.509±0.029	0.805±0.013	0.631±0.026	0.562±0.027	0.520±0.028	1.20
LRR	0.535±0.025	0.818±0.010	0.650±0.020	0.594±0.022	0.546±0.024	8.94
LatLRR	0.529±0.031	0.817±0.016	0.656±0.026	0.584±0.028	0.541±0.030	29.10
NSLLRR	0.538±0.021	0.822±0.007	0.653±0.022	0.611±0.017	0.550±0.020	126.70
LRRADP	0.460±0.022	0.781±0.008	0.590±0.022	0.519±0.018	0.473±0.021	87.67
AWNLRR	0.490±0.022	0.811±0.006	0.637±0.023	0.601±0.016	0.504±0.021	4.95
HWLRR	**0.571±0.012**	**0.834±0.004**	**0.679±0.025**	**0.632±0.008**	**0.581±0.012**	97.05

Table 3

Clustering performance metrics (mean ± standard deviation) and running times of different methods on the UMIST database. Bold numbers denote the best results.

Methods	ARI	NMI	ACC	Precision	F-score	Time(s)
Kmeans++	0.284±0.017	0.595±0.012	0.403±0.021	0.341±0.013	0.323±0.016	0.02
SC	0.341±0.020	0.635±0.016	0.449±0.015	0.380±0.019	0.375±0.019	1.80
LRR	0.361±0.035	0.651±0.021	0.473±0.025	0.402±0.031	0.394±0.033	14.32
LatLRR	0.145±0.014	0.424±0.018	0.320±0.017	0.237±0.022	0.196±0.012	43.04
NSLLRR	0.426±0.032	0.698±0.016	0.536±0.026	0.463±0.035	0.456±0.030	268.65
LRRADP	0.332±0.032	0.626±0.017	0.433±0.029	0.378±0.032	0.367±0.031	149.84
AWNLRR	0.360±0.026	0.646±0.019	0.462±0.024	0.399±0.025	0.393±0.024	9.06
HWLRR	**0.521±0.030**	**0.756±0.012**	**0.604±0.028**	**0.579±0.022**	**0.547±0.028**	157.58

Table 4

Clustering performance metrics (mean ± standard deviation) and running times of different methods on the MNIST database. Bold numbers denote the best results.

Methods	ARI	NMI	ACC	Precision	F-score	Time(s)
Kmeans++	0.353±0.022	0.493±0.014	0.518±0.029	0.442±0.019	0.421±0.019	0.30
SC	0.379±0.020	0.493±0.016	0.566±0.029	0.451±0.017	0.442±0.018	36.32
LRR	0.022±0.002	0.055±0.001	0.166±0.006	0.129±0.002	0.123±0.001	59.13
LatLRR	0.024±0.002	0.048±0.002	0.181±0.006	0.138±0.008	0.128±0.004	132.79
NSLLRR	0.328±0.019	0.452±0.012	0.517±0.022	0.404±0.014	0.396±0.016	4732.76
LRRADP	0.427±0.018	0.573±0.013	0.591±0.034	0.536±0.011	0.489±0.015	3725.37
AWNLRR	0.493±0.031	0.624±0.017	0.629±0.045	0.580±0.034	0.547±0.028	171.46
HWLRR	**0.577±0.028**	**0.694±0.017**	**0.735±0.042**	**0.669±0.027**	**0.622±0.025**	2271.42

Table 5

Clustering performance metrics (mean ± standard deviation) and running times of different methods on the C-Cube database. Bold numbers denote the best results.

Methods	ARI	NMI	ACC	Precision	F-score	Time(s)
Kmeans++	0.218±0.006	0.544±0.005	0.346±0.015	0.263±0.008	0.235±0.006	0.04
SC	0.233±0.011	0.556±0.007	0.383±0.014	0.255±0.012	0.248±0.011	15.70
LRR	0.201±0.012	0.515±0.011	0.349±0.018	0.221±0.012	0.217±0.012	1.61
LatLRR	0.200±0.007	0.517±0.007	0.348±0.013	0.222±0.007	0.217±0.007	0.69
NSLLRR	0.230±0.005	0.562±0.003	0.381±0.010	0.257±0.006	0.246±0.005	3967.60
LRRADP	0.229±0.005	0.553±0.006	0.375±0.011	0.250±0.005	0.245±0.005	3224.28
AWNLRR	0.230±0.008	0.553±0.006	0.385±0.009	0.257±0.007	0.246±0.007	149.83
HWLRR	**0.283±0.008**	**0.600±0.005**	**0.437±0.008**	**0.317±0.006**	**0.298±0.008**	2223.59

Table 6

Clustering performance metrics (mean ± standard deviation) and running times of different methods on the PenDigits database. Bold numbers denote the best results.

Methods	ARI	NMI	ACC	Precision	F-score	Time(s)
Kmeans++	0.488±0.039	0.661±0.018	0.633±0.053	0.596±0.022	0.543±0.033	0.01
SC	0.516±0.020	0.667±0.006	0.651±0.037	0.591±0.017	0.566±0.017	7.87
LRR	0.515±0.020	0.656±0.015	0.671±0.043	0.579±0.015	0.564±0.017	0.46
LatLRR	0.378±0.019	0.530±0.018	0.567±0.022	0.454±0.020	0.441±0.018	0.30
NSLLRR	0.459±0.029	0.606±0.016	0.620±0.035	0.526±0.021	0.514±0.025	377.76
LRRADP	0.528±0.023	0.664±0.013	0.675±0.035	0.610±0.007	0.574±0.020	239.80
AWNLRR	0.525±0.025	0.659±0.016	0.679±0.023	0.588±0.020	0.573±0.023	26.70
HWLRR	**0.532±0.020**	**0.713±0.012**	**0.680±0.027**	**0.629±0.012**	**0.582±0.017**	319.12

5.3. Semi-Supervised Classification

In this section, the performances of our HWLRR, LRR, LatLRR, NSLLRR, LRRADP and AWNLRR are compared through semi-supervised classification. All the algorithms compute the affinity matrix firstly and then use the GFHF [40] algorithm to predict the labels for the unlabeled points using the affinity matrix. In the experiments, the number of labeled points per cluster is varied from 1 to 7 and the labeled samples are randomly selected. For all the experiments, the run has been repeated ten times to eliminate the effects of randomness embedded in the algorithms. To compare the qualities of the final classification results obtained, the ACC, recall and F-score are employed. The mean and standard deviation of the metrics are given in Table 7-12.

From these tables, we can see that the ACC obtained by all the six algorithms increases as the number of labels increases. From comparision of NSLLRR, LRRADP, AWNLRR, one can conclude that distance constraint is more effective than Laplacian constraint for capture the relationship among samples. Because Laplacian term is constructed by nearest neighbor relationship, it should be pointed out that Laplacian constraint is also a distance relationship of sample and its nearest neighbors. This can prove the valuable of introducing more sample relationships. Furthermore, it can be realized that our HWLRR algorithm performs superior to LRR, LatLRR, NSLLRR, LRRADP and AWNLRR algorithms in most experiments. Compared with other LRR methods, it is obvious

that adding the hierarchy weighted matrix can obtain a better low-rank representation matrix. For the COIL20 dataset, its data structure is suitable for the hierarchy weighted matrix so that HWLRR can obtain the high classification accuracy with a few labeled data.

Table 7

Classification performance metrics (mean ± standard deviation) on the COIL20 database. #Tr denotes the number of labeled samples of a subject. Bold numbers denote the best results.

#Tr	Metrics	LRR	LatLRR	NSLLRR	LRRADP	AWNLRR	HWLRR
1	ACC	0.362±0.130	0.119±0.024	0.226±0.045	0.662±0.032	0.774±0.019	**0.861±0.013**
	Recall	0.173±0.088	0.057±0.115	0.094±0.063	0.514±0.023	0.655±0.023	**0.699±0.015**
	F-score	0.246±0.074	0.094±0.004	0.150±0.024	0.563±0.031	0.711±0.019	**0.794±0.029**
2	ACC	0.420±0.100	0.126±0.022	0.222±0.064	0.748±0.019	0.805±0.012	**0.865±0.008**
	Recall	0.186±0.081	0.058±0.085	0.103±0.099	0.629±0.018	0.710±0.019	**0.691±0.015**
	F-score	0.264±0.093	0.096±0.005	0.158±0.031	0.660±0.019	0.747±0.015	**0.794±0.012**
3	ACC	0.606±0.077	0.134±0.027	0.222±0.046	0.781±0.014	0.829±0.018	**0.885±0.013**
	Recall	0.360±0.047	0.061±0.062	0.102±0.090	0.674±0.016	0.725±0.010	**0.741±0.023**
	F-score	0.428±0.112	0.099±0.007	0.156±0.029	0.701±0.015	0.767±0.026	**0.830±0.031**
4	ACC	0.587±0.223	0.124±0.033	0.250±0.057	0.806±0.015	0.840±0.016	**0.890±0.011**
	Recall	0.412±0.060	0.059±0.127	0.114±0.077	0.707±0.017	0.740±0.021	**0.756±0.012**
	F-score	0.461±0.194	0.097±0.005	0.169±0.035	0.728±0.019	0.779±0.020	**0.842±0.016**
5	ACC	0.758±0.062	0.139±0.033	0.270±0.070	0.826±0.015	0.848±0.008	**0.887±0.012**
	Recall	0.615±0.058	0.063±0.126	0.137±0.087	0.731±0.019	0.748±0.026	**0.744±0.014**
	F-score	0.641±0.089	0.099±0.008	0.189±0.046	0.753±0.019	0.791±0.010	**0.832±0.023**
6	ACC	0.793±0.021	0.161±0.034	0.260±0.037	0.829±0.014	0.849±0.006	**0.903±0.007**
	Recall	0.675±0.026	0.067±0.070	0.120±0.046	0.737±0.016	0.753±0.020	**0.783±0.015**
	F-score	0.693±0.028	0.103±0.010	0.175±0.030	0.757±0.020	0.791±0.011	**0.861±0.012**
7	ACC	0.821±0.022	0.148±0.039	0.247±0.064	0.839±0.018	0.859±0.006	**0.904±0.007**
	Recall	0.707±0.024	0.062±0.091	0.117±0.074	0.757±0.021	0.774±0.016	**0.785±0.010**
	F-score	0.724±0.025	0.101±0.014	0.173±0.043	0.772±0.021	0.803±0.011	**0.864±0.010**

5.4. Effect of the parameters

There are two parameters affecting the performance of our HWLRR. In the following, we study the influence of λ and β on the accuracy by varying the value of λ and β on the UMIST dataset. Here, we vary one parameter and keep the other parameter fixed. The parameters increase from 10^{-5} to 10^{5}. The results are showed in Fig.7. In the figure, the Z-axis shows the ACC results obtained

Table 8

Classification performance metrics (mean ± standard deviation) on the ORL database. #Tr denotes the number of labeled samples of a subject. Bold numbers denote the best results.

#Tr	Metrics	LRR	LatLRR	NSLLRR	LRRADP	AWNLRR	HWLRR
1	ACC	0.559±0.019	0.158±0.048	0.566±0.078	0.582±0.015	0.581±0.028	**0.662±0.028**
	Recall	0.287±0.022	0.038±0.110	0.305±0.028	0.359±0.013	0.222±0.037	**0.429±0.016**
	F-score	0.357±0.041	0.065±0.009	0.377±0.084	0.398±0.019	0.317±0.081	**0.506±0.033**
2	ACC	0.692±0.026	0.155±0.038	0.715±0.039	0.707±0.019	0.669±0.019	**0.745±0.023**
	Recall	0.450±0.037	0.036±0.105	0.491±0.047	0.478±0.027	0.357±0.030	**0.544±0.019**
	F-score	0.515±0.045	0.062±0.011	0.556±0.069	0.528±0.029	0.459±0.045	**0.614±0.023**
3	ACC	0.732±0.014	0.163±0.045	0.759±0.024	0.785±0.016	0.704±0.024	**0.817±0.018**
	Recall	0.497±0.018	0.035±0.044	0.559±0.022	0.574±0.031	0.406±0.028	**0.652±0.021**
	F-score	0.561±0.022	0.063±0.015	0.615±0.030	0.620±0.033	0.503±0.047	**0.699±0.023**
4	ACC	0.771±0.027	0.273±0.086	0.800±0.020	0.795±0.032	0.762±0.024	**0.847±0.024**
	Recall	0.554±0.033	0.049±0.082	0.622±0.027	0.587±0.036	0.504±0.026	**0.720±0.019**
	F-score	0.616±0.039	0.083±0.024	0.674±0.027	0.634±0.039	0.586±0.045	**0.756±0.035**
5	ACC	0.793±0.021	0.453±0.256	0.810±0.025	0.834±0.013	0.770±0.030	**0.881±0.018**
	Recall	0.568±0.019	0.159±0.120	0.626±0.027	0.658±0.020	0.503±0.032	**0.760±0.029**
	F-score	0.629±0.031	0.214±0.198	0.686±0.028	0.697±0.020	0.591±0.060	**0.788±0.027**
6	ACC	0.802±0.026	0.857±0.095	0.818±0.013	0.843±0.024	0.792±0.026	**0.895±0.016**
	Recall	0.579±0.040	0.628±0.110	0.609±0.019	0.653±0.030	0.529±0.032	**0.761±0.022**
	F-score	0.643±0.046	0.682±0.214	0.671±0.020	0.700±0.046	0.615±0.038	**0.795±0.025**
7	ACC	0.811±0.020	0.846±0.129	0.836±0.027	0.860±0.025	0.816±0.021	**0.910±0.023**
	Recall	0.567±0.029	0.558±0.154	0.621±0.046	0.659±0.029	0.549±0.029	**0.777±0.039**
	F-score	0.636±0.032	0.618±0.314	0.687±0.047	0.711±0.042	0.630±0.035	**0.811±0.044**

Table 9

Classification performance metrics (mean ± standard deviation) on the UMIST database. #Tr denotes the number of labeled samples of a subject. Bold numbers denote the best results.

#Tr	Metrics	LRR	LatLRR	NSLLRR	LRRADP	AWNLRR	HWLRR
1	ACC	0.444±0.030	0.133±0.017	0.234±0.060	0.462±0.020	0.103±0.025	**0.794±0.017**
	Recall	0.266±0.031	0.056±0.097	0.101±0.115	0.284±0.043	0.056±0.123	**0.677±0.022**
	F-score	0.295±0.028	0.087±0.005	0.160±0.024	0.319±0.031	0.106±0.014	**0.727±0.021**
2	ACC	0.620±0.049	0.136±0.013	0.300±0.072	0.577±0.020	0.094±0.014	**0.853±0.017**
	Recall	0.448±0.052	0.056±0.102	0.135±0.081	0.422±0.038	0.053±0.133	**0.790±0.017**
	F-score	0.472±0.057	0.087±0.005	0.195±0.044	0.446±0.037	0.100±0.004	**0.813±0.022**
3	ACC	0.684±0.048	0.132±0.014	0.333±0.094	0.651±0.048	0.098±0.019	**0.870±0.033**
	Recall	0.527±0.065	0.057±0.093	0.149±0.086	0.498±0.053	0.053±0.087	**0.801±0.026**
	F-score	0.549±0.066	0.092±0.004	0.216±0.058	0.524±0.054	0.101±0.005	**0.825±0.033**
4	ACC	0.761±0.034	0.147±0.019	0.457±0.102	0.718±0.043	0.113±0.024	**0.909±0.014**
	Recall	0.631±0.046	0.061±0.098	0.245±0.047	0.578±0.048	0.056±0.105	**0.843±0.014**
	F-score	0.645±0.045	0.093±0.005	0.313±0.082	0.600±0.049	0.106±0.012	**0.863±0.023**
5	ACC	0.794±0.017	0.148±0.021	0.483±0.212	0.778±0.021	0.119±0.039	**0.909±0.027**
	Recall	0.669±0.020	0.061±0.084	0.274±0.134	0.674±0.037	0.057±0.095	**0.839±0.029**
	F-score	0.685±0.024	0.094±0.006	0.343±0.202	0.691±0.039	0.107±0.019	**0.861±0.040**
6	ACC	0.842±0.029	0.152±0.022	0.770±0.120	0.782±0.041	0.116±0.018	**0.928±0.015**
	Recall	0.739±0.033	0.063±0.086	0.619±0.125	0.680±0.059	0.054±0.028	**0.863±0.014**
	F-score	0.751±0.039	0.094±0.006	0.659±0.177	0.696±0.057	0.103±0.003	**0.885±0.016**
7	ACC	0.857±0.031	0.169±0.021	0.877±0.033	0.790±0.026	0.124±0.014	**0.935±0.014**
	Recall	0.768±0.047	0.066±0.079	0.806±0.045	0.686±0.034	0.056±0.139	**0.879±0.017**
	F-score	0.775±0.044	0.099±0.004	0.820±0.045	0.706±0.036	0.106±0.006	**0.896±0.016**

Table 10

Classification performance metrics (mean ± standard deviation) on the MNIST database. #Tr denotes the number of labeled samples of a subject. Bold numbers denote the best results.

#Tr	Metrics	LRR	LatLRR	NSLLRR	LRRADP	AWNLRR	HWLRR
1	ACC	0.100±1.462	0.132±0.004	0.202±0.018	0.322±0.097	0.308±0.075	**0.311±0.130**
	Recall	0.099±0.005	0.100±0.090	0.121±0.064	0.157±0.072	0.155±0.092	**0.177±0.119**
	F-score	0.181±2.925	0.140±0.013	0.179±0.009	0.247±0.078	0.245±0.062	**0.276±0.084**
2	ACC	0.100±1.462	0.128±0.004	0.226±0.038	0.565±0.072	**0.538±0.093**	0.510±0.097
	Recall	0.099±0.005	0.100±0.118	0.138±0.093	0.316±0.041	**0.291±0.040**	0.266±0.047
	F-score	0.181±2.925	0.150±0.016	0.184±0.010	0.391±0.092	**0.383±0.096**	0.371±0.085
3	ACC	0.100±1.462	0.135±0.003	0.230±0.035	0.629±0.055	**0.683±0.059**	0.667±0.058
	Recall	0.099±0.005	0.101±0.041	0.138±0.119	0.395±0.026	**0.470±0.032**	0.429±0.045
	F-score	0.181±2.925	0.138±0.007	0.180±0.013	0.465±0.056	**0.530±0.068**	0.521±0.069
4	ACC	0.100±1.462	0.133±0.005	0.214±0.019	0.692±0.029	0.721±0.058	**0.733±0.053**
	Recall	0.099±0.005	0.100±0.108	0.128±0.078	0.497±0.020	0.527±0.030	**0.534±0.023**
	F-score	0.181±2.925	0.142±0.015	0.174±0.007	0.540±0.036	0.585±0.052	**0.600±0.057**
5	ACC	0.100±1.462	0.131±0.008	0.224±0.037	0.743±0.030	**0.774±0.027**	0.767±0.051
	Recall	0.099±0.005	0.100±0.106	0.134±0.094	0.563±0.027	**0.612±0.031**	0.600±0.036
	F-score	0.181±2.925	0.143±0.014	0.192±0.009	0.595±0.038	0.640±0.034	**0.642±0.054**
6	ACC	0.100±1.462	0.131±0.008	0.216±0.025	0.756±0.020	0.797±0.023	**0.812±0.019**
	Recall	0.099±0.005	0.100±0.104	0.128±0.068	0.583±0.021	0.640±0.024	**0.656±0.018**
	F-score	0.181±2.925	0.139±0.014	0.186±0.005	0.611±0.025	0.662±0.031	**0.685±0.027**
7	ACC	0.100±1.462	0.131±0.007	0.229±0.035	0.786±0.014	0.808±0.020	**0.813±0.037**
	Recall	0.099±0.005	0.100±0.073	0.135±0.111	0.629±0.016	0.660±0.021	**0.663±0.023**
	F-score	0.181±2.925	0.140±0.012	0.196±0.010	0.648±0.016	0.678±0.028	**0.688±0.038**

Table 11

Classification performance metrics (mean ± standard deviation) on the C-Cube database. #Tr denotes the number of labeled samples of a subject. Bold numbers denote the best results.

#Tr	Metrics	LRR	LatLRR	NSLLRR	LRRADP	AWNLRR	HWLRR
1	ACC	0.033±0.013	0.235±0.014	0.203±0.016	0.283±0.011	0.165±0.027	**0.369±0.019**
	Recall	0.020±0.243	0.096±0.007	0.078±0.017	0.137±0.010	0.035±0.121	**0.195±0.020**
	F-score	0.039±0.000	0.109±0.009	0.112±0.014	0.163±0.009	0.064±0.011	**0.233±0.021**
2	ACC	0.058±0.021	0.305±0.018	0.207±0.008	0.324±0.015	0.202±0.027	**0.434±0.011**
	Recall	0.020±0.203	0.137±0.015	0.080±0.023	0.152±0.011	0.051±0.076	**0.254±0.012**
	F-score	0.040±0.001	0.152±0.013	0.117±0.006	0.188±0.010	0.088±0.020	**0.286±0.014**
3	ACC	0.066±0.036	0.346±0.014	0.175±0.012	0.338±0.015	0.218±0.019	**0.465±0.011**
	Recall	0.022±0.272	0.164±0.007	0.055±0.054	0.155±0.011	0.057±0.049	**0.282±0.010**
	F-score	0.042±0.004	0.182±0.007	0.091±0.011	0.193±0.010	0.097±0.024	**0.304±0.008**
4	ACC	0.092±0.043	0.375±0.017	0.095±0.042	0.370±0.010	0.237±0.024	**0.494±0.021**
	Recall	0.023±0.268	0.178±0.011	0.034±0.136	0.170±0.008	0.061±0.039	**0.307±0.018**
	F-score	0.044±0.005	0.199±0.011	0.063±0.012	0.211±0.010	0.103±0.017	**0.327±0.020**
5	ACC	0.147±0.044	0.400±0.007	0.115±0.033	0.373±0.013	0.273±0.031	**0.509±0.011**
	Recall	0.029±0.194	0.198±0.007	0.040±0.096	0.168±0.009	0.078±0.034	**0.319±0.009**
	F-score	0.053±0.011	0.220±0.006	0.072±0.011	0.211±0.011	0.124±0.026	**0.336±0.009**
6	ACC	0.200±0.030	0.426±0.012	0.159±0.042	0.391±0.014	0.273±0.030	**0.531±0.010**
	Recall	0.033±0.126	0.214±0.009	0.053±0.082	0.180±0.010	0.080±0.040	**0.340±0.013**
	F-score	0.060±0.012	0.236±0.009	0.090±0.019	0.221±0.010	0.125±0.030	**0.357±0.012**
7	ACC	0.216±0.043	0.430±0.010	0.149±0.023	0.404±0.015	0.302±0.023	**0.542±0.005**
	Recall	0.034±0.107	0.218±0.012	0.048±0.049	0.186±0.018	0.096±0.020	**0.349±0.012**
	F-score	0.062±0.011	0.243±0.013	0.084±0.011	0.230±0.016	0.147±0.018	**0.364±0.008**

Table 12

Classification performance metrics (mean ± standard deviation) on the PenDigits database. #Tr denotes the number of labeled samples of a subject. Bold numbers denote the best results.

#Tr	Metrics	LRR	LatLRR	NSLLRR	LRRADP	AWNLRR	HWLRR
1	ACC	0.371±0.021	0.276±0.044	0.334±0.039	0.629±0.048	0.275±0.107	**0.594±0.061**
	Recall	0.226±0.035	0.153±0.092	0.181±0.035	0.446±0.041	0.147±0.073	**0.417±0.038**
	F-score	0.284±0.022	0.215±0.017	0.247±0.024	0.503±0.046	0.247±0.056	**0.513±0.038**
2	ACC	0.297±0.051	0.357±0.041	0.329±0.052	**0.696±0.045**	0.418±0.147	0.667±0.026
	Recall	0.170±0.106	0.201±0.044	0.194±0.048	**0.512±0.038**	0.230±0.115	0.465±0.016
	F-score	0.252±0.018	0.254±0.026	0.259±0.036	**0.570±0.039**	0.334±0.119	0.563±0.028
3	ACC	0.352±0.054	0.360±0.082	0.298±0.069	0.715±0.036	0.412±0.134	**0.720±0.054**
	Recall	0.198±0.052	0.211±0.074	0.163±0.060	0.504±0.029	0.218±0.087	**0.508±0.030**
	F-score	0.277±0.026	0.265±0.053	0.244±0.030	0.592±0.032	0.319±0.118	**0.595±0.061**
4	ACC	0.373±0.087	0.358±0.061	0.348±0.090	0.745±0.034	0.587±0.100	**0.752±0.061**
	Recall	0.215±0.096	0.186±0.084	0.202±0.069	0.553±0.034	0.382±0.076	**0.563±0.034**
	F-score	0.287±0.037	0.252±0.032	0.276±0.044	0.627±0.038	0.472±0.103	**0.631±0.072**
5	ACC	0.496±0.076	0.364±0.058	0.428±0.052	0.771±0.018	0.593±0.072	**0.785±0.049**
	Recall	0.293±0.044	0.179±0.076	0.263±0.021	0.610±0.017	0.348±0.044	**0.620±0.026**
	F-score	0.358±0.051	0.254±0.028	0.326±0.038	0.648±0.016	0.458±0.082	**0.674±0.055**
6	ACC	0.515±0.055	0.393±0.118	0.466±0.027	0.764±0.019	0.645±0.062	**0.794±0.049**
	Recall	0.293±0.051	0.193±0.033	0.304±0.035	0.607±0.018	0.420±0.023	**0.620±0.021**
	F-score	0.367±0.057	0.271±0.064	0.358±0.021	0.647±0.021	0.512±0.081	**0.683±0.057**
7	ACC	0.323±0.099	0.524±0.130	0.450±0.062	0.785±0.018	0.665±0.069	**0.833±0.050**
	Recall	0.164±0.063	0.281±0.037	0.306±0.034	0.627±0.020	0.452±0.027	**0.686±0.031**
	F-score	0.249±0.045	0.353±0.105	0.363±0.047	0.660±0.026	0.536±0.082	**0.730±0.058**

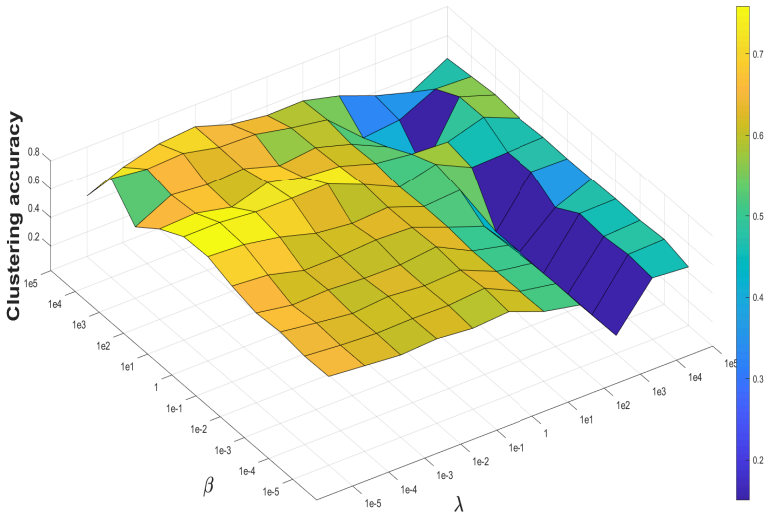

Fig. 7. The influence of λ and β on our HWLRR algorithm

with different parameter settings. From the results, we know that the effect of λ and β is not remarkable in $[10^{-5}, 1]$. Therefore, we set $\lambda = \beta = 0.1$ in our experiments.

6. Conclusion

In this paper, a novel hierarchy weighted low-rank representation (HWLRR) is proposed to capture the high hierarchy relationship among data points better by hierarchy weight. In the HWLRR algorithm, a hierarchy weighted matrix is defined firstly. In this matrix, the penalty of samples is depended on the hierarchy that samples lie on. Explained by probability, the samples obtain the same penalty when they lie on the same hierarchy and get the different penalty with others lying on different hierarchies. With the hierarchy weighted matrix, HWLRR can learn local neighbor relationships by k-nearest neighbor and also preserve the global clustering structure by the higher hierarchy neighbors. Therefore, HWLRR is suitable for a semi-supervised label propagation method

and clustering methods (e.g., GFHF and SC). All the experimental results described in this paper show the effectiveness and superiority of our HWLRR. Moreover, our methods obtain higher classification accuracy with fewer samples labeled for classification.

Although the results presented here are extremely encouraging, there is an issue that deserves in-depth study in the future. The affinity propagation used here only stop when no sample can be found. Therefore, some samples in the different subspaces may be connected and this will deteriorate the clustering performance of the algorithm. A mechanism that allows the propagation to stop automatically should be investigated.

Acknowledgements

The authors would like to thank the Editor and the anonymous referees for their helpful comments to improve the quality of the paper. This paper was supported in part by the National Natural Science Foundation of China(61532005).

References

[1] G. Liu, Z. Lin, Y. Yu, Robust subspace segmentation by low-rank representation, in: Proceedings of the International Conference on Machine Learning, 2010, pp. 663–670.

[2] G. Liu, S. Yan, Latent low-rank representation for subspace segmentation and feature extraction, in: Proceedings of the IEEE International Conference on Computer Vision, 2011, pp. 1615–1622.

[3] Z. He, L. Liu, S. Zhou, Y. Shen, Learning group-based sparse and low-rank representation for hyperspectral image classification, Pattern Recognit. 60 (2016) 1041–1056.

[4] Z. Zhang, K. Zhao, Low-rank matrix approximation with manifold regularization, IEEE Trans. Pattern Anal. Machine Intell. 35 (2012) 1717–1729.

[5] H. Du, X. Zhang, Q. Hu, Y. Hou, Sparse representation-based robust face recognition by graph regularized low-rank sparse representation recovery, Neurocomputing 164 (2015) 220–229.

[6] G. Liu, S. Yan, Latent low-rank representation for subspace segmentation and feature extraction, in: Proceedings of the IEEE International Conference on Computer Vision, 2011, pp. 1615–1622.

[7] C. Chen, C. Wei, Y. F. Wang, Low-rank matrix recovery with structural incoherence for robust face recognition, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2012, pp. 2618–2625.

[8] M. Zhang, N. Wang, Y. Li, X. Gao, Deep latent low-rank representation for face sketch synthesis, IEEE Trans. Neural Netw. Learn. Syst. 30 (2019) 3109–3123.

[9] E. J. Candès, X. Li, Y. Ma, J. Wright, Robust principal component analysis?, J. ACM (JACM) 58 (2011) 1–37.

[10] H. Zhang, Z. Lin, C. Zhang, J. Gao, Robust latent low rank representation for subspace clustering, Neurocomputing 145 (2014) 369–373.

[11] R. Liu, Z. Lin, F. De la Torre, Z. Su, Fixed-rank representation for unsupervised visual learning, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2012, pp. 598–605.

[12] X. Fang, Y. Xu, X. Li, Z. Lai, W. K. Wong, Robust semi-supervised subspace clustering via non-negative low-rank representation., IEEE Trans. Cybernet. 46 (2016) 1828–1838.

[13] X. Fang, Y. Xu, X. Li, Z. Lai, W. K. Wong, Learning a nonnegative sparse graph for linear regression, IEEE Trans. Image Process. 24 (2015) 2760–2771.

[14] J. B. Tenenbaum, V. De Silva, J. C. Langford, A global geometric framework for nonlinear dimensionality reduction, Science 290 (2000) 2319–2323.

[15] M. Belkin, P. Niyogi, Laplacian eigenmaps for dimensionality reduction and data representation, Neural Comput. 15 (2003) 1373–1396.

[16] X. He, D. Cai, S. Yan, H. Zhang, Neighborhood preserving embedding, in: Proceedings of the IEEE International Conference on Computer Vision, 2005, pp. 1208–1213.

[17] L. Fei, Y. Xu, X. Fang, J. Yang, Low rank representation with adaptive distance penalty for semi-supervised subspace classification, Pattern Recognit. 67 (2017) 252–262.

[18] M. Zheng, J. Bu, C. Chen, C. Wang, L. Zhang, G. Qiu, D. Cai, Graph regularized sparse coding for image representation, IEEE Trans. Image Process. 20 (2010) 1327–1336.

[19] X. Lu, Y. Wang, Y. Yuan, Graph-regularized low-rank representation for destriping of hyperspectral images, IEEE Trans. Geosci. Remote Sens. 51 (2013) 4009–4018.

[20] M. Yin, J. Gao, Z. Lin, Laplacian regularized low-rank representation and its applications, IEEE Trans. Pattern Anal. Machine Intell. 38 (2016) 504–517.

[21] J. Wen, B. Zhang, Y. Xu, J. Yang, N. Han, Adaptive weighted nonnegative low-rank representation, Pattern Recognit. 81 (2018) 326–340.

[22] G. Liu, Z. Lin, S. Yan, J. Sun, Y. Yu, Y. Ma, Robust recovery of subspace structures by low-rank representation, IEEE Trans. Pattern Anal. Machine Intell. 35 (2012) 171–184.

[23] M. Belkin, P. Niyogi, Laplacian eigenmaps and spectral techniques for embedding and clustering, in: Proceedings of the International Conference on Neural Information Processing Systems, 2002, pp. 585–591.

[24] K. Fukunaga, L. Hostetler, K-nearest-neighbor bayes-risk estimation, IEEE Trans. Inf. Theory 21 (1975) 285–293.

[25] X. Wang, A fast exact k-nearest neighbors algorithm for high dimensional search using k-means clustering and triangle inequality, in: Proceedings of the International Joint Conference on Neural Networks, 2011, pp. 1293–1299.

[26] J. M. Keller, M. R. Gray, J. A. Givens, A fuzzy k-nearest neighbor algorithm, IEEE Trans. Syst. Man Cybern. (1985) 580–585.

[27] L. Zhuang, Z. Zhou, S. Gao, J. Yin, Z. Lin, Y. Ma, Label information guided graph construction for semi-supervised learning, IEEE Trans. Image Process. 26 (2017) 4182–4192.

[28] F. Camastra, A. Staiano, Intrinsic dimension estimation: Advances and open problems, Inf. Sci. 328 (2016) 26–41.

[29] J. Yang, X. Yuan, Linearized augmented lagrangian and alternating direction methods for nuclear norm minimization, Math. Comput. 82 (2013) 301–329.

[30] J. Cai, E. J. Candès, Z. Shen, A singular value thresholding algorithm for matrix completion, SIAM J. Optim. 20 (2010) 1956–1982.

[31] S. A. Nene, S. K. Nayar, H. Murase, et al., Columbia object image library (coil-20), Technical report CUCS-005-96 (1996) 1–6.

[32] F. S. Samaria, A. C. Harter, Parameterisation of a stochastic model for human face identification, in: IEEE workshop on applications of computer vision, 1994, pp. 138–142.

[33] D. B. Graham, N. Allinson, Face recognition: from theory to applications, NATO ASI Ser. F. Comput. Syst. Sci. 163 (1998) 446–456.

[34] Y. LeCun, L. Bottou, Y. Bengio, P. Haffner, Gradient-based learning applied to document recognition, Proceedings of the IEEE 86 (1998) 2278–2324.

[35] F. Camastra, M. Spinetti, A. Vinciarelli, Offline cursive character challenge: a new benchmark for machine learning and pattern recognition algorithms., in: Proceedings of the International Conference on Pattern Recognition, 2006, pp. 913–916.

[36] F. Alimoglu, E. Alpaydin, Y. Denizhan, Combining multiple classifiers for pen-based handwritten digit recognition, Doc. Anal. Recognit. 3 (1996) 637–640.

[37] D. Arthur, S. Vassilvitskii, K-means++: The advantages of careful seeding, in: ACM-SIAM Symposium on Discrete Algorithms (SOD), 2007, pp. 1027–1035.

[38] X. S. Yu, J. Shi, Multiclass spectral clustering, in: Proceedings of the IEEE International Conference on Computer Vision, 2003, pp. 313–319.

[39] G. Liu, Z. Lin, S. Yan, J. Sun, Y. Yu, Y. Ma, Robust recovery of subspace structures by low-rank representation, IEEE Trans. Pattern Anal. Machine Intell. 35 (2012) 171–184.

[40] X. Zhu, Z. Ghahramani, J. D. Lafferty, Semi-supervised learning using gaussian fields and harmonic functions, in: Proceedings of the International Conference on Machine Learning, 2003, pp. 912–919.