Chapter 9

Robust Spectral Clustering via Sparse Representation

Xiaodong Feng

Additional information is available at the end of the chapter

http://dx.doi.org/10.5772/intechopen.76586

Abstract

Clustering high-dimensional data has been a challenging problem in data mining and machining learning. Spectral clustering via sparse representation has been proposed for clustering high-dimensional data. A critical step in spectral clustering is to effectively construct a weight matrix by assessing the proximity between each pair of objects. While sparse representation proves its effectiveness for compressing high-dimensional signals, existing spectral clustering algorithms based on sparse representation use those sparse coefficients directly. We believe that the similarity measure exploiting more global information from the coefficient vectors will provide more truthful similarity among data objects. The intuition is that the sparse coefficient vectors corresponding to two similar objects are similar and those of two dissimilar objects are also dissimilar. In particular, we propose two approaches of weight matrix construction according to the similarity of the sparse coefficient vectors. Experimental results on several real-world high-dimensional data sets demonstrate that spectral clustering based on the proposed similarity matrices outperforms existing spectral clustering algorithms via sparse representation.

Keywords: spectral clustering, high-dimensional data, weight matrix, sparse representation

1. Introduction

As an important task in data mining cluster analysis aims at partitioning data objects into several meaningful subsets, called clusters, such that data objects are similar to those in the same cluster and dissimilar to those in different clusters. With advances in database technology and real-world need of informed decisions, data sets to be analyzed are getting bigger—with many more records and variables. Examples of high-dimensional data sets include document data [1], user ratings data [2], multimedia data [3], financial time series data [4], gene expression data [5] and so on. Due to the "curse of dimensionality" [6], clustering

high-dimensional data has been a challenging task and therefore, attracts much research in data mining and related research domains [7].

Many of the existing high-dimensional clustering approaches can be categorized into the following three types: dimension reduction [8], subspace clustering [9] and spectral clustering [10–12]. The first two types transform the original feature space to a lower-dimensional space and then apply an ordinary clustering algorithm (such as K-means). The focus is on how to extract more important features of data objects and avoid noises from the less important dimensions. Spectral clustering is based on the spectral graph model, which searches for clusters in the full feature space and is equivalent to graph min-cut problem based on a graph structure constructed from the original objects in vector space [12]. Spectral clustering is also considered superior to traditional clustering algorithms due to its deterministic and polynomial time solution. All these characteristics make spectral clustering suitable for high-dimensional data clustering [10].

The effectiveness of spectral clustering depends on the weights between each pair of data objects. Thus, it is vital to construct a weight matrix that faithfully reflects the similarity information among objects. Traditional simple weight construction, such as ε-ball neighborhood, k-nearest neighbors, inverse Euclidean distance [13, 14] and Gaussian radial basis function (RBF) [12], is based on the Euclidean distance in the original space, thus not suitable for high-dimensional data.

In this chapter, we focus on effective weight construction for spectral clustering, based on sparse representation theory. Sparse representation aims for representing each object approximately by a sparse linear combination of other objects, which comes from the theory of compressed sensing [15]. Coefficients of the sparse linear combination represent the closeness of each object to other objects. Traditional spectral clustering methods based on sparse representation [16] use these sparse coefficients directly to build the weight matrix, thus only local information is utilized. However, exploiting more global information of the whole coefficient vectors promises better performance, followed by an assumption that the sparse representation vectors corresponding to two similar objects should be similar, since they can be reconstructed in a similar fashion using other data objects. If two objects are contributing in a similar manner to the reconstruction of all other objects in the same data set, they are considered similar.

Therefore, this chapter presents a spectral clustering approach of high-dimensional data exploiting global information from sparse representation solution. More specifically, using sparse representation, we firstly convert each high-dimensional data object into a vector of sparse coefficients. Then, the proximity of two data objects is assessed according to the similarity between their sparse coefficient vectors. This construction considers the complete information of the solution coefficient vectors of two objects to analyze the similarity between these two objects rather than directly using a particular single sparse coefficient, which only considers local information. In particular, we propose two different weight matrix construction approaches: one of which is based on consistent sign set (CSS) and the other is based on the cosine similarity (COS) between the two vectors. Extensive experimental results on several image data sets show that similarly exploiting the global information from the solutions of sparse representation works better than using local information of the solutions under a variety of clustering performance metrics.

2. Related work

2.1. Techniques for high-dimensional data

There are many techniques for dealing with high-dimensional signals (or data), popular of which include non-negative matrix factorization (NMF), manifold learning, compressed sensing and some combinations between them.

Non-negative matrix factorization (NMF) is a powerful dimensionality reduction technique and has been widely applied to image processing and pattern recognition applications [17], by approximating a non-negative matrix X by the product of two non-negative low-rank factor matrices W and H. It has attracted much attention since it was first proposed by Paatero and Tapper [18] and has already been proven to be equivalent in terms of optimization process with K-means and spectral clustering under some constraints [19]. The research about NMF can be generally categorized into the following groups. The first group is focused on the distance measures between the original matrix and the approximate matrix, including Kullback–Leibler divergence (KLNMF) [17], Euclidean distance (EucNMF) [20], earth mover's distance metric [21] and Manhattan distance-based NMF (MahNMF) [22]. Besides, there are researches about how to solve the optimization of NMF efficiently and the scalability of NMF algorithms for large-scale data sets, for example, fast Netwon-type methods (FNMA) [23], online NMF with robust stochastic approximation (OR-NMF) [24] and large-scale graph-regularized NMF [25]. Moreover, how to improve the performance of NMF using some constrains or exploiting more information of data is also popular, such as sparseness constrained NMF (NMFsc) [26], convex model for NMF using $l_{1,\infty}$ regularization [27], discriminant NMF (DNMF) [28], graph-regularized NMF (GNMF) [29], manifold regularized discriminative NMF (MD-NMF) [30] and constrained NMF (CNMF) [31] incorporating the label information.

Manifold learning is another theory to process high-dimensional data, assuming that the data distribution is supported on a low-dimensional sub-manifold [32]. The key idea of manifold learning is that the locality structure of high-dimensional data should be preserved in low-dimensional space after dimension reduction, which is exploited as a regularization term [33–35] or constraint [36, 37] to be added to the original problem. It has been widely used to machine learning and computer vision, such as image classification [38], semi-supervised multiview distance metric learning [39], human action recognition [40], complex object correspondence construction [41] and so on.

Besides the abovementioned two approaches for high-dimensional data, in recent years, sparse representation coming from compressed sensing has also attracted a great deal of attention and proves to be an extremely powerful tool for acquiring, representing and compressing high-dimensional data. The following section will briefly review of sparse representation.

2.2. Brief review of sparse representation

Given a sufficient high-dimensional training data set, $\mathbf{X} = [\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_n] \in \mathrm{R}^{\mathrm{m} \times \mathrm{n}}$, where $\boldsymbol{x}_i = [\boldsymbol{x}_{i1}, \boldsymbol{x}_{i2}, \ldots, \boldsymbol{x}_{im}]\mathrm{T} \in \mathrm{R}^{\mathrm{m}}$ is the column vector of the i-th object. Research on manifold learning [32] has

proved that any new test data y lie on a lower dimensional manifold, which can be approximately represented by a linear combination of the training objects:

$$\mathbf{y} = \alpha_1 x_1 + \cdots + \alpha_i x_i + \cdots + \alpha_n x_n = \mathbf{X}\alpha \in \mathbf{R}^m. \tag{1}$$

Obviously, if m ≫ n, Eq. (1) is overdetermined, and α can usually be found as its unique solution. Typically, the number of attributes is much less than that of training objects (i.e. m ≪ n) and Eq. (1) is undetermined, so its solution is not unique.

However, if we add the constraint that the best solution of Eq. (1) should be as sparse as possible, which means that the number of non-zero elements is minimized, the solution becomes unique. Such a sparse representation can be obtained by solving the optimization problem:

$$\alpha^* = \arg\min_{\alpha} \ \|\alpha\|_0 \ \ subject \ \ to \ \ \mathbf{y} = \mathbf{X}\alpha, \tag{2}$$

where $||.||_0$ denotes the l_0-norm of a vector, counting the number of non-zero entries in the vector. Donoho [42] proves that if matrix $\mathbf{X}$ satisfies restricted isometry property (RIP) [43], Eq. (2) has a unique solution of $\boldsymbol{\alpha}$.

However, it is NP-hard to find the sparsest solution of an underdetermined equation: that is, there is no known approach to find the sparsest solution that is significantly more efficient than exhausting all subsets of the entries for α. Researchers in emerging theory of compressed sensing [44] reveal that the non-convex optimization in (2) is equal to the following convex l_1 optimization problem if the solution α is sparse enough:

$$\alpha^* = \arg\min_{\alpha} \ \|\alpha\|_1 \ \ subject \ \ to \ \ \mathbf{y} = \mathbf{X}\alpha, \tag{3}$$

where $||.||_1$ denotes the l_1-norm of a vector, summing the absolute value of each entry in the vector. This problem can be solved in polynomial time by standard linear programming methods [45].

Since the real data contains noise, it may not be possible to express the test sample exactly as a sparse representation of the training data. The sparse solution α can still be approximately obtained by solving the following stable l_1 optimization problem:

$$\alpha^* = \arg\min_{\alpha} \ \|\alpha\|_1 \ \ subject \ \ to \ \ \|\mathbf{y} - \mathbf{X}\alpha\|_2 \le \varepsilon, \tag{4}$$

where ε is the maximum residual error; $||.||_2$ denotes the l_2-norm of a vector.

In many situations, we do not know the noise level ε beforehand. Then we can use the Lasso (least absolute shrinkage and selection operator) [46] optimization algorithm to recover the sparse solution from the following l_1 optimization:

$$\alpha^* = \arg\min_{\alpha} \ \lambda\|\alpha\|_1 + \|\mathbf{y} - \mathbf{X}\alpha\|_2, \tag{5}$$

where λ is a scalar regularization parameter of the Lasso penalty, which directly determines how sparse α will be and balances the trade-off between reconstruction error and sparsity.

In addition to Lasso, other sparse learning models are also developed. It will be the elastic net model [47] if the l_2-norm of α is also added to Eq. (5) as another penalty term. Double shrinking algorithm (DSA) [48] compresses image data on both dimensionality and cardinality via building either sparse low-dimensional representations or a sparse projection matrix for dimension reduction. Go decomposition (GoDec) [49] tried to efficiently and robustly decompose a matrix with the low-rank part L and the sparse part S. Locality structure of manifold can also be combined with sparse representation, such as manifold elastic net (MEN) [50] and graph-regularized sparse coding (GraphSC) [51], laplacian sparse coding (LSc) [35] and Hypergraph laplacian sparse coding (HLSc) [35].

Learning tasks such as classification and clustering usually perform better and cost less (time and space) on compressed representations than on the original data [48]. Therefore, supervised learning and pattern recognition based on the sparse representation coefficients using these sparse learning models are proposed, such as sparse representation-based classification (SRC) [52], Local_SRC [53], Kernel_SRC [54] and the methods outperform traditional classifier, such as SVM, nearest neighbor (NN) and nearest subspace (NS).

2.3. Sparse representation for clustering

Inspired by the successful application of sparse representation in the above-supervised learning approaches, researchers have also exploited sparse representation in unsupervised [55–57] and semi-supervised clustering [58, 59]. The main idea of clustering via sparse representation is to build weight matrix directly from normalized and symmetrized coefficients of sparse representation coefficients, called sparsity-induced similarity (SIS) measure [59]. To a certain extent, weight measure approaches derived from sparse representation can reveal the neighborhood structure without calculating Euclidean distance, which means a great potential to clustering high-dimensional data.

Some significant work applying SIS to spectral clustering is reviewed as follows. Sparse subspace clustering [55] directly uses the sparse representation of vectors lying in a single low-dimensional linear subspace to cluster the data into separate subspaces, followed by applying spectral clustering. It is also extended to clustering data contaminated by noise, missing entries or outliers. Experiments show that its performance for clustering motion trajectories outperforms state-of-the-art methods, such as power factorization and principal component analysis. Image clustering via sparse representation [56] characterizes the graph adjacency structure and graph weights by sparse linear coefficients, which is more effective than Gaussian RBF [12] to cluster an image data set. In semi-supervised learning by sparse representation [18], the graph adjacency structure as well as the graph weights of the directed graph construction is derived simultaneously and in a parameter-free manner to utilize both labeled and unlabeled data. Experiments on semi-supervised face recognition and image classification demonstrate the superiority over the counterparts based on traditional graphs (e.g. ε-ball neighborhood, k-nearest neighbors). Compared to approaches using SIS of real

numbers, non-negative SIS measure [57] exploits the symmetric coefficients of non-negative sparse representation as weight matrix, which outperforms similarity measures, such as SIS and Euclidean (with Gaussian RBF baseline [12]), in cluster analysis of spam images.

However, all the above-existing approaches based on sparse representation treat directly the coefficients or just normalized coefficients of sparse representation as the weight matrix. These cannot exactly reflect the similarity between objects because the coefficients of sparse representation are somehow local similarity and sensitive to outliers. Our approach is expected to provide more effective weight matrix construction using more global content from the solution coefficients of sparse representation.

2.4. Graph construction with sparse representation

In clustering analysis, given a high-dimensional object data set $\mathbf{X} = [\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_n] \in \mathrm{R}^{\mathrm{m} \times \mathrm{n}}$, $\boldsymbol{x}_i = [\boldsymbol{x}_{i1}, \boldsymbol{x}_{i2}, \ldots, \boldsymbol{x}_{im}]^{\mathrm{T}} \in \mathrm{R}^{\mathrm{m}}$, we can use Eq. (5) to represent each objects $\boldsymbol{x}_{\mathrm{i}}$ as a linear combination of other objects. The coefficients vector α_i of $\mathbf{x}_{\mathrm{i}}$ can be calculated by solving the following Lasso optimization:

$$\alpha_i^* = \arg\min_{\alpha_i} \ \lambda\|\alpha_i\|_1 + \|x_i - \mathbf{X}_i\alpha_i\|_2, \tag{6}$$

where $\mathbf{X}_i = \mathbf{X} \setminus \boldsymbol{x}_{\mathrm{i}} = [\boldsymbol{x}_1, \ldots, \boldsymbol{x}_{i-1}, \boldsymbol{x}_{i+1}, \ldots, \boldsymbol{x}_{\mathrm{n}}]$; $\boldsymbol{\alpha}_i = [\boldsymbol{\alpha}_{i,1}, \ldots, \boldsymbol{\alpha}_{i,(i-1)}, \boldsymbol{\alpha}_{i,(i+1)}, \ldots, \boldsymbol{\alpha}_{i,n}]^{\mathrm{T}}$.

Once we get the coefficient vector $\boldsymbol{\alpha}_i$ for each object $\boldsymbol{x}_i$ (i = 1, 2, …, n) as a sparse representation of all other data objects by solving the l_1 optimization Eq. (6),we can construct the weight matrix using different approaches.

Existing weight matrix constructions via sparse representation are based on the assumption that coefficients in the sparse representation reflect the closeness or similarity between two data objects. For example, the SIS measure [20] is computed as:

$$w_{ij} = \frac{\max\{\alpha_{i,j}, 0\}}{\sum_{k=1,k\neq i}^{n} \max\{\alpha_{i,k}, 0\}}; \quad SIS_{ij} = \frac{w_{ij} + w_{ji}}{2}. \tag{7}$$

The l_1 Directed Graph Construction (DGC) measure [19] is computed as:

$$DGC_{ij} = \frac{|\alpha_{i,j}| + |\alpha_{j,i}|}{2}. \tag{8}$$

Obviously, the similarity calculation using the absolute coefficients in Eq. (8) will mistake the big negative coefficient as high similarity, resulting in a cluster of two objects with apparent opposite attributes value.

The non-negative SIS measure [22] adds a non-negative constraint in l_1 optimization Eq. (6):

$$\alpha_i^* = \arg\min_{\alpha_i} \ \lambda\|\alpha_i\|_1 + \|x_i - \mathbf{X}_i\alpha_i\|_2 \ \text{s.t.}\alpha_{i,j} > 0. \tag{9}$$

Then the non-negative SIS measure is computed as:

$$NN_{ij} = \frac{\alpha_{i,j}}{\sum_{k=1,k\neq i}^{n} \alpha_{i,k}}. \tag{10}$$

3. Sparse representation for spectral clustering

Our proposed clustering algorithm consists of three steps: (1) solving l_1 optimization of sparse representation to obtain the coefficients of each object; (2) constructing weight matrix between objects on the basis of coefficients using more global content forms the solution coefficients of sparse representation; and (3) exploiting the spectral clustering algorithm with the weight matrix to get partition of the graph.

Compared to the direct construction methods using the independent solutions of Eq. (6), we have the assumption that for any two objects $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$; the more similar they are, the more similar the corresponding coefficient vectors (e.g., α_i and α_j) are not only a particular coefficient ($\alpha_{i,j}$ or $\alpha_{j,i}$). According to this assumption, we propose the following two graph adjacency structure and weight matrix constructions, which are expected to use the global information of the solution coefficients.

3.1. Proximity based on a consistent sign set

To get the similarity of two objects clearly and logically, we firstly find an object set for each of the two different objects $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ from the object data set $\mathbf{X}$, called CSS. This definition is based on the assumption that the more objects of which a pair of objects both positively contribute to the reconstruction, the more similar the pair of objects are. In particular, the sparse reconstruction coefficients corresponding to $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ for every object in this set are both positive, defined as follows:

$$CSS(x_i, x_j) = \{x_k | (\alpha_{k,i} > 0 \wedge \alpha_{k,j} > 0), k \neq i, k \neq j\} \ \ \forall i \neq j. \tag{11}$$

Furthermore, we can construct graph adjacency structure and weight matrix as follows. A directed edge is placed between objects $\boldsymbol{x}_i$ and $\boldsymbol{x}_{jj}$ if $CSS(\boldsymbol{x}_i, \boldsymbol{x}_j) \neq \Phi$ and the weight between object $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ is defined as the ratio of the $CSS(\boldsymbol{x}_i, \boldsymbol{x}_j)$'s cardinal to the total number of objects:

$$w_{ij} = \begin{cases} \dfrac{|CSS(\mathbf{x}_i, \mathbf{x}_j)|}{n} & i \neq j \\ 0 & i = j \end{cases}, \tag{12}$$

where n is the total number of objects in $\mathbf{X}$. Obviously, the weight is between 0 and 1.

3.2. Proximity based on cosine similarity of coefficient vector

We can construct coefficient matrix $\mathbf{A}$ of data set $\mathbf{X}$, to which transforming solution coefficients of Eq. (6) are:

$$\mathrm{A}(i,j) = \alpha'_{i,j} = \begin{cases} \alpha_{i,j} & i \neq j \\ 0 & i = j \end{cases}. \tag{13}$$

A directed edge is placed from object $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ if angle cosine of the two corresponding vectors is greater than 0, that is:

$$\frac{\alpha'_i \cdot \alpha'_j}{\left\|\alpha'_i\right\|_2 \times \left\|\alpha'_j\right\|_2} > 0, \tag{14}$$

where $\boldsymbol{\alpha}_i'$ denotes the i-th row vector of A.

The weight between object $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ is defined as the cosine similarity of $\boldsymbol{\alpha}_i'$ and $\boldsymbol{\alpha}_j'$:

$$w_{ij} = \begin{cases} \max\left(0, \dfrac{\alpha'_i \cdot \alpha'_j}{\left\|\alpha'_i\right\|_2 \times \left\|\alpha'_j\right\|_2}\right) & i \neq j \\ 0 & i = j \end{cases}. \tag{15}$$

From the above similarity calculation formula, two objects have large similarity in condition that the corresponding solution coefficients of Eq. (6) are much similar, which is expected to use the whole solution coefficients.

3.3. The relationship between consistent sign set (CSS) and cosine similarity of coefficient vector (COS)

Since proposed proximity based on both CSS and COS are trying to exploit more information from the solution coefficient of sparse representation, the relationship between each other is following:

1. Both of them asses the weight between two objects according to the similarity between the corresponding coefficient vectors of the two objects. However, the difference is that proximity based on CSS uses the column vectors of the coefficient matrix **A** while proximity based on COS calculates the similarity between row vectors, which means two understandings of the coefficient matrix. The reason for defining these two approaches like this is just experimental.
2. Proximity based on COS is to calculate the similarity of the original coefficient vector, while CSS can be considered as the discretization of the original coefficient vector with threshold zero. Therefore, proximity based on COS can be seen as the generalization of that based on CSS.
3. Specifically, another equivalent way to understand proximity based on CSS is as follows:
   - Transform the coefficients matrix A to DA: $DA(i,j) = \begin{cases} 1 & A(i,j) > 0 \\ 0 & else \end{cases}$;
   - The weight between $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ is:

- $w_{ij} = \begin{cases} \frac{DA^i \cdot DA^j}{n} & i \neq j \\ 0 & i = j \end{cases}$, where DA^i denotes the i-th column vector of DA.

Obviously, the inner product (DA^i DA^j) between DA^i and DA^j is equal to CSS ($\boldsymbol{x}_i$, $\boldsymbol{x}_j$)'s cardinal $|\mathrm{CSS}(\boldsymbol{x}_i, \boldsymbol{x}_j)|$.

To illustrate the differences between our approaches for weight construction and others also using sparse representation, an example is given as follows. Assume that the coefficient matrix **A** of a data set with five objects obtained from solution coefficients of Eq. (6) is as the following 5×5 matrix:

$$\mathbf{A} = \alpha'_{i,j} = \begin{bmatrix} 0 & 0.3 & 0.6 & 0.6 & -0.7 \\ 0.4 & 0 & 0.5 & 0.6 & -0.6 \\ 0.4 & 0.4 & 0 & -0.1 & -0.2 \\ -0.6 & -0.3 & 0.2 & 0 & 0.7 \\ -0.5 & 0.3 & 0.2 & 0.4 & 0 \end{bmatrix}$$

According to the above introduction of different weight constructions:

1. SIS_{13} = 0.4, SIS_{12} = 0.2, DGC_{13} = 0.5 and DGC_{12} = 0.35, and these numbers show that the similarity between $\boldsymbol{x}_1$ and $\boldsymbol{x}_3$ is larger than that between $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$. However, in our approaches using more entries in **A**, CSS_{13} = 1/5 = 0.2, CSS_{12} = 2/5 = 0.4, COS_{13} = 0.24 and COS_{12} = 0.98 and these numbers show the different weights compared to the first group, where *CSS* and *COS* are the abbreviation of the above two proximity approaches, respectively.
2. DGC_{25} = 0.45, CSS_{25} = 0, COS_{25} = 0.16, thus *DGC* mistakes the big negative coefficient (α_{25}') as high similarity while *CSS* and *COS* both give lower similarity.

3.4. Algorithm description

Algorithm 1 describes the general procedure for spectral clustering of high-dimensional data, using sparse representation. The basic idea is to extract coefficients of sparse representation (Lines 1–4); construct a weight matrix using the coefficients (Line 5); and feed the weight matrix into a spectral clustering algorithm (Line 6) to find the best partitioning efficiently.

Algorithm 1. General procedure for spectral clustering of high-dimensional data.

Input: high-dimensional training data set $\mathbf{X} = [\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_n] \in \mathrm{R}^{m \times n}$, where $\boldsymbol{x}_i = [\boldsymbol{x}_{i1}, \boldsymbol{x}_{i2}, \ldots, \boldsymbol{x}_{im}]\mathrm{T} \in \mathrm{R}^m$ represents the i-th data object; the number of clusters K.

Parameter: penalty coefficient λ for Lasso optimization

Output: cluster labels corresponding to each data object: $\mathbf{c} = [c_1, c_2, \ldots, c_n]$

//standardize the input data for Lasso optimization

1 for each *data object* $\boldsymbol{x}_i \in$ X do

// Solve Eq. (6) with Lasso optimization

2 Set $\mathbf{X}_i = \mathbf{X} \backslash \boldsymbol{x}_i = [\boldsymbol{x}_1, \ldots, \boldsymbol{x}_{i-1}, \boldsymbol{x}_{i+1}, \ldots, \boldsymbol{x}_n]$;

3 $\boldsymbol{\alpha}_i^* \leftarrow \arg\min \lambda \| \boldsymbol{\alpha}_i \|_1 + \| \boldsymbol{x}_i - \mathbf{X}\boldsymbol{\alpha}_i \|_2$;

4 end

5 $\mathbf{W} \leftarrow$ ConstructWeightMatrix($\boldsymbol{\alpha}$);

6 $\mathbf{c} \leftarrow$ SpectralClustering($\mathbf{W}$);

7 return **c.**

The construct weight matrix () sub-routine can exploit any weight matrix construction method, such as those mentioned in Section 4. In particular, we describe the algorithm for computing the two newly proposed weight matrices, one based on the CSS (see Section 4.1) and the other based on COS of sparse coefficient vectors (see Section 4.2).

Algorithm 2 describes the procedure to construct the weight matrix according to the concept of CSS. To find the *CSS* of each pair of data objects (the two outermost loops), there is the need of checking the sparse coefficients of each remaining object to these two objects, so the time complexity of weight matrix constructions based on *CSS* is *O* (n^3).

Algorithm 2. Construct weight matrix based on consistent sign set.

Input: Coefficients for sparse representation $\boldsymbol{\alpha}$

Output: Weight matrix $\mathbf{W}$

1 for $i \leftarrow 1$ to n do

2 for $j \leftarrow 1$ to$n$ do

3 if $j = i$ then $w_{ij} \leftarrow 0$;

4 else

5 $n_{css} \leftarrow 0$;

6 for $k \leftarrow 1$ to n do

7 if $k \neq i$ and $k \neq$ j and $\boldsymbol{\alpha}_{k,i} > 0$ and$\boldsymbol{\alpha}_{k,j} > 0$ then

8 $n_{css} \leftarrow n_{css} + 1$;

9 end

10 end

11 $w_{ij} \leftarrow n_{css}/n$;

12 end

13 end

14 end

Algorithm 3 describes the procedure to construct the weight matrix according to the COS of the sparse coefficients between each pair of items. The computation complexity for calculating the

COS of two vectors of length n is $O(n)$, and there are $O(n^2)$ pairs of data objects whose COS needs to be computed. Thus the complexity for COS-based weight matrix construction is $O(n^3)$.

Algorithm 3. Construct weight matrix based on similarity of coefficient vector.

Input: Coefficients for sparse representation $\boldsymbol{\alpha}$

Output: Weight matrix $\mathbf{W}$

1 for $i \leftarrow 1$ to n do

2 for $j \leftarrow 1$ to$n$ do

3 if $j = i$ then $w_{ij} \leftarrow 0$;

4 else

5 $cosine \leftarrow \frac{\alpha_i' \cdot \alpha_j'}{\|\alpha_i'\|_2 \times \left\|\alpha_j'\right\|_2}$

6 if $cosine > 0$ then $w_{ij} \leftarrow cosine$;

7 else $w_{ij} \leftarrow 0$;

8 end

9 end

10 end

As shown in line 6 of Algorithm 1, after constructing the weight matrix **W**, we can use the classical spectral clustering algorithm [10] to discover the cluster structure of high-dimensional data.

The main characteristics of our proposed algorithm include the following: (1) compared to traditional graph construction induced from the Euclidean distance or other measures in the original high-dimensional space, the weight matrix is constructed after transforming the high dimensional data space into another space via sparse representation, which is expected to have better performance resulting from the superiority of compressed sensing [58] for high-dimensional data; (2) our graph construction based on consistent sign set or similarity of coefficient vector can simultaneously complete both the graph adjacency and weight matrix, while traditional graph constructions (such as ε-ball neighborhood or k-nearest neighbors) complete the two tasks separately, which are interrelated and should not be separated [19]; (3) rather than existing graph constructions via sparse representation directly and independently applying the solution of l_1 optimization for each object in Eq. (6) to determine a row of the weight matrix, our approach considers the global information from the coefficients of the whole object set to calculate one element in the weight matrix.

4. Experimental results

In this section, we use experimental results to demonstrate the performance of our proposed approaches on real-world data sets using several effectiveness evaluations.

We select three data sets from the UCI machine-learning repository [60] and three face recognition data sets [61–63], which are well known in the machine learning and data mining research community. **Table 1** lists a summary of these data sets.

In Yale and ORL face data sets, each image is transformed into a 32×32 pixel configuration using Matlab Image Processing Toolbox. In Yale B data set, which has 10 clusters, and each cluster has 585 image data, since the original size (5850) is too big, which leads to too much time consumption in clustering, we randomly select 60 images from the totally 585 images in each cluster. In all the data sets, each image is normalized to have unit norm.

We use interior-point method-based l1_ls_matlab tool [64] to solve Eq. (6) for each data object and then implement different weight matrices for algorithm 1 in Matlab to cluster each data set. **Table 2** shows a summary of the proposed and baseline algorithms.

Since the true class labels of each data set are known, five commonly used external cluster validation metrics [66–68] are employed to evaluate the clustering results, namely clustering accuracy (CA) and normalized mutual information (NMI)[1].

Data set	# Instance	# Attributes	# Classes	Source
Heart	270	13	2	UCI
Image (Image segmentation)	2310	18	7	UCI
Yale	165	1024	15	[61]
Yale B	600	1200	10	[62]
ORL Face	400	1024	40	[63]
Movement	360	90	15	UCI

Table 1. Summary of data sets.

Name	Description	Source	Role
CSS	Spectral clustering with weight matrix from consistent sign set	Section 3.1	Solution proposed
COS	Spectral clustering with weight matrix from cosine similarity of sparse coefficients	Section 3.2	Solution proposed
RBF	Spectral clustering with weight matrix from Gaussian RBF	[12]	Baseline
SIS	Spectral clustering with weight matrix from sparsity induced similarity measure	[59]	Baseline
DGC	Spectral clustering with weight matrix from l_1 Directed Graph Construction	[58]	Baseline
NN	Spectral clustering with weight matrix from non-negative sparsity induced similarity measure	[57]	Baseline
KM	k-means clustering	[65]	Baseline

Table 2. Summary of algorithms to be compared.

[1]We use the matlab toolbox from: http://www.cad.zju.edu.cn/home/dengcai/Data/data.html

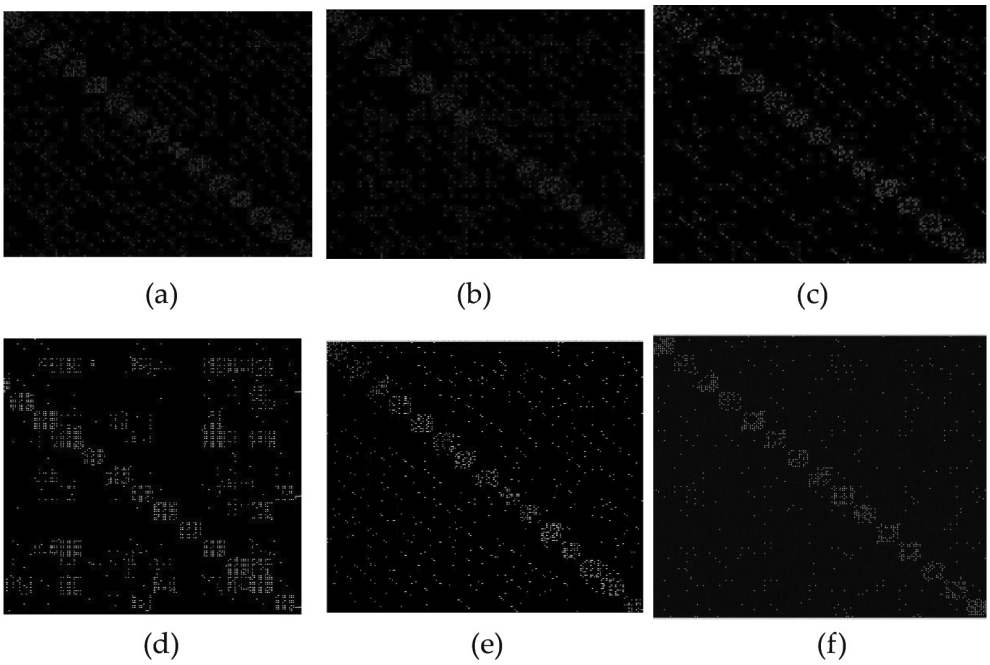

Figure 1. Visualization of the graph weight matrices of the Yale data set, where images from the same subject are arranged together. (a) SIS, (b) DGC, (c) NN, (d) RBF, (e) CSS, (f) COS.

To illustrate the weight matrices from different approaches, we demonstrate the visual property of the proposed graph weight matrices in comparison with traditional ones in **Figure 1**, taking the Yale data set as an example. In **Figure 1**, each subfigure is a weight matrix with N*N (entries larger than the threshold is shown in white, otherwise black) and images from the same cluster are arranged together. These sparse representation-based graphs include consistent sign set (CSS), cosine similarity of coefficient vectors (COS), induced similarity measure (SIS), l_1 directed graph construction (DGC), nonnegative sparsity induced similarity measure (NN) and Gaussian RBF (RBF).

Since none of the original weight matrices constructed by the five approaches is sparse, we set threshold values (0.2 for COS; 0.388 for CSS; 0 for RBF (σ is set 4 in RBF); 0.02 for the other three matrices) to get the best sparse matrices of different threshold values in **Figure 1**. A value larger than the threshold is shown in white, otherwise black. Normally, the clustering performance will be good if the weights between two objects from different clusters are little while weights from the same cluster are large. This comment can be equalized in the matrix with above arrangement that good matrix should be compact in diagonal position and sparse in other positions.

From **Figure 1**, we have the following observations: (1) matrices in all subfigures are compact in diagonal position; (2) the matrix of COS is sparser than others in lower left or upper right

parts. This means that there are less inter-cluster adjacency connections in the COS than other graphs, so COS can encode more discriminating information and hence is more effective in spectral clustering than other traditional graphs; (3) CSS has a similar performance to SIS, DGC and NN Graph in Yale data set.

The clustering results obtained from the seven clustering algorithms with different evaluation metrics are reported in **Tables 3** and **4**, each of which corresponds to one evaluation metric. For each data set, the best results are in bold. All the numbers, except the last two rows in each table, represent the best clustering results using different lasso parameters (λ). The last two rows in each table present the average performance of each algorithm over all six data sets. Since the k-means clustering within spectral clustering is sensitive to initial centroids, we run spectral clustering 50 times for each case and report the mean and standard deviation (std).

From **Tables 3** and **4**, we can clearly see that, generally, CSS or COS algorithm gets the best clustering performance with all the two evaluation metrics. However, there are also some particular cases where CSS or COS does not get best result. For example, though NN gets the best CA on Yale B data sets, COS gets almost the same CA result as NN, that is, from 0.8937 to 0.8940; though NN also gets the best NMI for movement data set, COS gets best result in other metric on this data set. In particular, COS performs better than CSS with mean value of evaluation metrics, and the average standard deviation between 50 random tests of CSS is lowest for all metrics except CA.

Overall, for most data sets, CSS and COS show better performance than those baselines, which are robust across various external validation metrics. However, it is noticed that COS outperforms CSS in terms of all average mean metrics except CA, and CSS outperforms COS in

Data set		CSS	COS	DGC	SIS	NN	BRF	KM
Heart	Mean	0.7704	**0.8174**	0.5852	0.7889	0.7519	0.7963	0.7320
	(Std)	**(0.0000)**	**(0.0000)**	**(0.0000)**	**(0.0000)**	**(0.0000)**	**(0.0000)**	(0.0882)
Image	Mean	0.7631	**0.7921**	0.7020	0.7820	0.7360	0.5335	0.6215
	(Std)	**(0.0148)**	(0.0323)	(0.0339)	(0.0341)	(0.0379)	(0.0305)	(0.0355)
Yale	Mean	0.6823	**0.7408**	0.7178	0.7023	0.6417	0.6635	0.5482
	(Std)	(0.0432)	(0.0345)	(0.0414)	**(0.0314)**	(0.0412)	(0.0395)	(0.0529)
Yale B	Mean	0.8572	0.8937	0.8320	0.8620	**0.8940**	0.6918	0.6862
	(Std)	(0.0791)	(0.0713)	(0.0768)	(0.0635)	(0.0767)	**(0.0226)**	(0.0721)
ORL face	Mean	0.7315	**0.7570**	0.7225	0.7243	0.6903	0.7314	0.7196
	(Std)	(0.0247)	(0.0218)	(0.0222)	(0.0244)	**(0.0207)**	(0.0252)	(0.0311)
Movement	mean	0.5241	**0.5472**	0.5009	0.5183	0.5304	0.4874	0.4653
	(Std)	(0.0222)	**(0.0187)**	(0.0248)	(0.0193)	(0.0271)	(0.0232)	(0.0203)
Average	Mean	0.7214	**0.7580**	0.6767	0.7296	0.7074	0.6506	0.6288
	(Std)	(0.0307)	(0.0298)	(0.0332)	(0.0288)	(0.0339)	**(0.0235)**	(0.0500)

Table 3. Evaluation of all algorithms with CA as metric.

Data set		CSS	COS	DGC	SIS	NN	BRF	KM
Heart	Mean	0.2208	**0.3149**	0.0511	0.1791	0.0331	0.2712	0.2028
	(Std)	**(0.0000)**	**(0.0000)**	**(0.0000)**	**(0.0000)**	(0.0312)	**(0.0000)**	(0.1013)
Image	Mean	0.7088	**0.7451**	0.5921	0.7319	0.6637	0.7451	0.6122
	(Std)	**(0.0071)**	(0.0184)	(0.0171)	(0.0176)	(0.0357)	(0.0284)	(0.0437)
Yale	Mean	0.7137	**0.7815**	0.7513	0.7641	0.6926	0.6989	0.6484
	(Std)	(0.0211)	**(0.0183)**	(0.0203)	(0.0188)	(0.0198)	(0.0271)	(0.0342)
Yale B	Mean	0.9008	**0.9526**	0.9202	0.9379	0.9510	0.7768	0.7858
	(Std)	(0.0302)	(0.0360)	(0.0422)	(0.0326)	(0.0398)	**(0.0224)**	(0.0621)
ORL face	Mean	0.8477	**0.8688**	0.8492	0.8547	0.8426	0.8512	0.8620
	(Std)	(0.0116)	(0.0108)	**(0.0105)**	(0.0118)	(0.0109)	(0.0140)	(0.0157)
Movement	Mean	0.5891	0.5914	0.5933	0.6000	**0.6306**	0.5741	0.5818
	(Std)	(0.0128)	**(0.0124)**	(0.0140)	(0.0130)	(0.0173)	(0.0169)	(0.0180)
Average	Mean	0.6635	**0.7090**	0.6262	0.6779	0.6356	0.6529	0.6155
	(Std)	**(0.0138)**	(0.0160)	(0.0174)	(0.0156)	(0.0258)	(0.0181)	(0.0458)

Table 4. Evaluation of all algorithms with NMI as metric.

terms of all average standard metrics. It can be explained that CSS is more stable because its discretization may lower the variance of the pairwise of similarity, while COS get more generalized information of the pairwise of similarity leading to better average metrics but higher variance. Therefore, the choice between stability and quality should be taken into account when it is facing the clustering problem, in practice, using this kind of approach.

Finally, we plot the averages of the mean value and standard deviation (from the last two rows of the five tables), for comparing clustering algorithms, as shown in **Figure 2**.

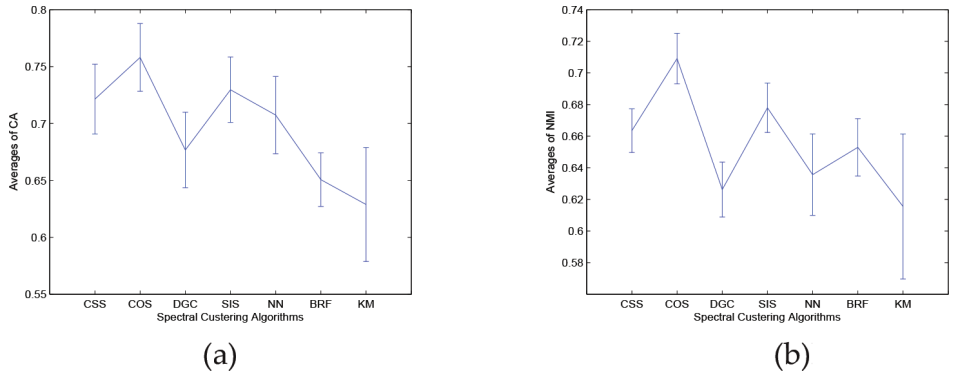

Figure 2. Error bar of different algorithms (a) CA, (b) NMI.

5. Conclusion

In this chapter, we present a study of spectral clustering based on sparse representation, using two novel weight matrix construction approaches to assess the consistency of two sparse vectors. This construction considers the global information of the solution coefficient vectors of two objects to analyze the similarity between these two objects rather than directly using the sparse coefficients, which only considers local information. Evaluation experiments on real-world data sets show that spectral clustering for high-dimensional data using our novel weight matrix construction exploiting global information outperforms direct k-means and spectral clustering approaches using Gaussian RBF, SIS, l_1-directed graph construction and non-negative SIS in five evaluation metrics (CA and NMI).

These results demonstrate a reliable performance of our algorithm and therefore promise wide applicability in practice. The findings also shed light on developing global solutions theories in the future work.

Figure 2 clearly demonstrates that COS and CSS algorithms outperform other algorithms, and COS is better than CSS on average. CSS obtains the least average value of standard deviation among all seven algorithms. The KM and DGC algorithms have comparable performance, which is usually worse than the other algorithms.

Author details

Xiaodong Feng

Address all correspondence to: fengxd1988@hotmail.com

School of Public Administration, University of Electronic Science and Technology of China, Chengdu, Sichuan, China

References

[1] Liu Y, Wang X, Wu C. ConSOM: A conceptional self-organizing map model for text clustering. Neurocomputing. 2008;**71**:857-862

[2] Koren Y, Bell R, Volinsky C. Matrix factorization techniques for recommender systems. Computer. 2009;**42**:30-37

[3] Bhatt CA, Kankanhalli MS. Multimedia data mining: State of the art and challenges. Multimedia Tools Applications. 2011;**51**:35-76

[4] Zhang X, Liu J, Du Y, Lv T. A novel clustering method on time series data. Expert Systems with Applications. 2011;**38**:11891-11900

[5] Sun J, Chen W, Fang W, Wun X, Xu W. Gene expression data analysis with the clustering method based on an improved quantum-behaved particle swarm optimization. Engineering Applications of Artificial Intelligence. 2012;**25**:376-391

[6] Steinbach M, Ertoz L, Kumar V. The challenges of clustering high dimensional data. In: New Directions in Statistical Physics. Berlin, Germany: Springer; 2004. pp. 273-309

[7] Chen X, Ye Y, Xu X, Huang JZ. A feature group weighting method for subspace clustering of high-dimensional data. Pattern Recognition. 2012;**45**:434-446

[8] Song Q, Ni J, Wang G. A fast clustering-based feature subset selection algorithm for high dimensional data. IEEE Transactions on Knowledge and Data Engineering. 2011;**9**:1-14

[9] Parsons L, Haque E, Liu H. Subspace clustering for high dimensional data: A review. ACM SIGKDD Explorations Newsletter. 2004;**6**:90-105

[10] Von Luxburg U. A tutorial on spectral clustering. Statistics and Computing. 2007;**17**:395-416

[11] Shi J, Malik J. Normalized cuts and image segmentation. IEEE Transactions on Pattern Analysis and Machine Intelligence. 2000;**22**:888-905

[12] Ng AY, Jordan MI, Weiss Y. On spectral clustering: Analysis and an algorithm. Advances in Neural Information Processing Systems. 2002;**2**:849-856

[13] Belkin M, Niyogi P. Laplacian eigenmaps for dimensionality reduction and data representation. Neural Computation. 2003;**15**:1373-1396

[14] Tenenbaum JB, De Silva V, Langford JC. A global geometric framework for nonlinear dimensionality reduction. Science. 2000;**290**:2319-2323

[15] Donoho DL. Compressed sensing. IEEE Transactions on Information Theory. 2006;**52**: 1289-1306

[16] Wright J, Ma Y, Mairal J, Sapiro G, Huang TS, Yan S. Sparse representation for computer vision and pattern recognition. Proceedings of the IEEE. 2010;**98**:1031-1044

[17] Lee DD, Seung HS. Learning the parts of objects by non-negative matrix factorization. Nature. 1999;**401**:788-791

[18] Paatero P, Tapper U. Positive matrix factorization: A non-negative factor model with optimal utilization of error estimates of data values. Environmetrics. 1994;**5**:111-126

[19] Ding CH, He X, Simon HD. On the equivalence of nonnegative matrix factorization and spectral clustering. In: SIAM International Conference on Data Mining; 2005. pp. 606-610

[20] Lee DD, Seung HS. Algorithms for non-negative matrix factorization. Advances in Neural Information Processing Systems. 2001;**13**:556-562

[21] Sandler R, Lindenbaum M. Nonnegative matrix factorization with Earth mover's distance metric for image analysis. IEEE Transactions on Pattern Analysis and Machine Intelligence. 2011;**33**:1590-1602

[22] Guan N, Tao D, Luo Z, Shawe-Taylor J, MahNMF. Manhattan Non-Negative Matrix Factorization, arXiv preprint arXiv:1207.3438;2012

[23] Kim D, Sra S, Dhillon IS. Fast Newton-type methods for the least squares nonnegative matrix approximation problem. In: SIAM International Conference on Data Mining; 2007

[24] Guan N, Tao D, Luo Z, Yuan B. Online nonnegative matrix factorization with robust stochastic approximation. IEEE Transactions on Neural Networks and Learning Systems. 2012;**23**:1087-1099

[25] Sun M, Hamme HV. Large scale graph regularized non-negative matrix factorization with l1 normalization based on Kullback–Leibler divergence. IEEE Transaction on Signal Processing. 2012;**60**:3876-3880

[26] Hoyer PO. Non-negative matrix factorization with sparseness constraints. The Journal of Machine Learning Research. 2004;**5**:1457-1469

[27] Esser E, Moller M, Osher S, Sapiro G, Xin J. A convex model for nonnegative matrix factorization and dimensionality reduction on physical space. IEEE Transactions on Image Processing. 2012;**21**:3239-3252

[28] Zafeiriou S, Tefas A, Buciu I, Pitas I. Exploiting discriminant information in nonnegative matrix factorization with application to frontal face verification. IEEE Transactions on Neural Networks. 2006;**17**:683-695

[29] Cai D, He X, Han WJ, Huang TS. Graph regularized nonnegative matrix factorization for data representation. IEEE Transactions on Pattern Analysis and Machine Intelligence. 2011;**33**:1548-1560

[30] Guan N, Tao D, Luo Z, Yuan B. Manifold regularized discriminative nonnegative matrix factorization with fast gradient descent. IEEE Transactions on Image Processing. 2011;**20**: 2030-2048

[31] Liu H, Wu Z, Li X, Cai D, Huang TS. Constrained nonnegative matrix factorization for image representation. IEEE Transactions on Pattern Analysis and Machine Intelligence. 2012;**34**:1299-1311

[32] Roweis ST, Saul LK. Nonlinear dimensionality reduction by locally linear embedding. Science. 2000;**290**:2323-2326

[33] Luo Y, Tao D, Geng B, Xu C, Maybank S. Manifold regularized multi-task learning for semi-supervised multi-label image classification. 2013;**22**:523-536

[34] Belkin M, Niyogi P, Sindhwani V. Manifold regularization: A geometric framework for learning from labeled and unlabeled examples. The Journal of Machine Learning Research. 2006;**7**:2399-2434

[35] Gao S, Tsang I, Chia L. Laplacian sparse coding, hypergraph laplacian sparse coding, and applications. IEEE Transactions on Pattern Analysis and Machine Intelligence. 2013;**35**: 92-104

[36] Zhou Y, Barner K. Locality constrained dictionary learning for nonlinear dimensionality reduction. IEEE Signal Processing Letters. 2012;**20**:335-338

[37] Wang J, Yang J, Yu K, Lv F, Huang T, Gong Y. Locality-constrained linear coding for image classification, computer vision and pattern recognition (CVPR). In: 2010 IEEE Conference on, (IEEE, 2010); pp. 3360-3367

[38] Yu J, Tao D, Wang M. Adaptive hypergraph learning and its application in image classification. IEEE Transactions on Image Processing. 2012;**21**:3262-3272

[39] Yu J, Wang M, Tao D. Semi-supervised multiview distance metric learning for cartoon synthesis. IEEE Transactions on Image Processing. 2012;**21**:4636-4648

[40] Deng X, Liu X, Song M, Cheng J, Bu J, Chen C. LF-EME: Local features with elastic manifold embedding for human action recognition. Neurocomputing. 2013;**99**:144-153

[41] Yu J, Liu D, Tao D, Seah HS. Complex object correspondence construction in two-dimensional animation. IEEE Transactions on Image Processing. 2011;**20**:3257-3269

[42] Donoho DL. For most large underdetermined systems of equations, the minimal l1-norm near-solution approximates the sparsest near-solution. Communications on Pure and Applied Mathematics. 2006;**59**:907-934

[43] Candès EJ. The restricted isometry property and its implications for compressed sensing. Comptes Rendus Mathematique. 2008;**346**:589-592

[44] Candès EJ, Compressive sampling. In: Proceedings of the International Congress of Mathematicians; 22-30 August 2006: Invited Lectures; Madrid. 2006. pp. 1433-1452

[45] Chen SS, Donoho DL, Saunders MA. Atomic decomposition by basis pursuit. SIAM Journal on Scientific Computing. 1998;**20**:33-61

[46] Tibshirani R. Regression shrinkage and selection via the lasso. Journal of the Royal Statistical Society. Series B (Methodological). 1996:267-288

[47] Zou H, Hastie T. Regularization and variable selection via the elastic net. Journal of the Royal Statistical Society: Series B (Statistical Methodology). 2005;**67**:301-320

[48] Zhou T, Tao D. Double shrinking for sparse dimension reduction. IEEE Transactions on Pattern Analysis and Machine Intelligence. 2013;**35**:92-104

[49] Zhou T, Tao D, Godec: Randomized low-rank & sparse matrix decomposition in noisy case. In: Proceedings of the 28th International Conference on Machine Learning (ICML-11); 2011. pp. 33-40

[50] Zhou T, Tao D, Wu X. Manifold elastic net: A unified framework for sparse dimension reduction. Data Mining and Knowledge Discovery. 2011;**22**:340-371

[51] Zheng M, Bu J, Chen C, Wang C, Zhang L, Qiu G, Cai D. Graph regularized sparse coding for image representation. IEEE Transactions on Image Processing. 2011;**20**:1327-1336

[52] Wright J, Yang AY, Ganesh A, Sastry SS, Ma Y. Robust face recognition via sparse representation. IEEE Transactions on Pattern Analysis and Machine Intelligence. 2009;**31**: 210-227

[53] Li C, Guo J, Zhang H. Local sparse representation based classification. In: Pattern Recognition (ICPR), 2010 20th International Conference on, (IEEE, 2010); pp. 649-652

[54] Gao S, Tsang IW, Chia L. Kernel sparse representation for image classification and face recognition. In: Computer Vision–ECCV 2010. Berlin, Germany: Springer; 2010. pp. 1-14

[55] Elhamifar E, Vidal R. Sparse subspace clustering. In: Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on, (IEEE, 2009); pp. 2790-2797

[56] Jiao J, Mo X, Shen C. Image clustering via sparse representation. In: Advances in Multimedia Modeling. Springer; 2010. pp. 761-766

[57] Gao Y, Choudhary A, Hua G. A nonnegative sparsity induced similarity measure with application to cluster analysis of spam images. In: Acoustics Speech and Signal Processing (ICASSP), 2010 IEEE International Conference on, (IEEE, 2010); pp. 5594-5597

[58] Yan S, Wang H. Semi-supervised learning by sparse representation. In: SIAM International Conference on Data Mining; 2009. pp. 792-801

[59] Cheng H, Liu Z, Yang J. Sparsity induced similarity measure for label propagation. In: Computer Vision, 2009 IEEE 12th International Conference on, (IEEE, 2009); pp. 317-324

[60] UCI Data Sets. http://archive.ics.uci.edu/ml/datasets/ [Accessed: November 10, 2012]

[61] Georghiades A. Yale Face. 2013. http://cvc.yale.edu/projects/yalefaces/yalefaces.html

[62] Georghiades A, Belhumeur P, Kriegman D, Yale Face_B. 2013. http://cvc.yale.edu/projects/yalefacesB/yalefacesB.html

[63] ORL Face. AT&T Lab Cambridge. 2013. http://www.face-rec.org/databases/

[64] Koh K, Kim SJ, Boyd S, l1_ls_matlab. 2013. http://www.stanford.edu/~boyd/l1_ls/l1_ls_matlab.zip

[65] Hartigan JA, Wong MA. Algorithm AS 136: A k-means clustering algorithm. Journal of the Royal Statistical Society. Series C (Applied Statistics). 1979;**28**:100-108

[66] Huang Z. Extensions to the k-means algorithm for clustering large data sets with categorical values. Data Mining and Knowledge Discovery. 1998;**2**:283-304

[67] Jing L, Ng MK, Huang JZ. An entropy weighting k-means algorithm for subspace clustering of high-dimensional sparse data. IEEE Transactions on Knowledge and Data Engineering. 2007;**19**:1026-1041

[68] Deng Z, Choi K, Chung F, Wang S. EEW-SC, enhanced entropy-weighting subspace clustering for high dimensional gene expression data clustering analysis. Applied Soft Computing. 2011;**11**:4798-4806