Neurocomputing 135 (2014) 229–239

Contents lists available at ScienceDirect

Neurocomputing

journal homepage: www.elsevier.com/locate/neucom

Spectral clustering of high-dimensional data exploiting sparse representation vectors

Sen Wu [a], Xiaodong Feng [a,c,*], Wenjun Zhou [b]

[a] Dongling School of Economics and Management, University of Science and Technology Beijing, Beijing 100083, China
[b] Statistics, Operations, and Management Science, University of Tennessee, Knoxville, TN 37996, USA
[c] Department of Computer Science and Engineering, University of Minnesota Twin Cities, Minneapolis, MN 55455, USA

ARTICLE INFO

Article history:
Received 26 June 2013
Received in revised form
12 December 2013
Accepted 19 December 2013
Communicated by D. Tao
Available online 10 January 2014

Keywords:
Spectral clustering
High-dimensional data
Weight matrix
Sparse representation

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 stepin spectral clustering is to effectively construct a weight matrix by assessing the proximity between each pair of objects. While sparse representation has proved its effectiveness for compressing high-dimensional signals, existing spectral clustering algorithms based on sparse representation use individual sparse coefficients directly. However, exploiting complete sparse representation vectors is expected to reflect more truthful similarity among data objects, since more contextual information is being considered. The intuition is that sparse representation vectors corresponding to two similar objects are expected to be similar, while those of two dissimilar objects are dissimilar. In particular, we propose two weight matrix constructions for spectral clustering based on the similarity of the sparse representation vectors. Experimental results on several real-world, high-dimensional datasets demonstrate that spectral clustering based on the proposed weight matrices outperforms existing spectral clustering algorithms, which use sparse coefficients directly.

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, datasets to be analyzed are getting bigger-with many more data records and attributes. Examples of high-dimensional datasets 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 attention in data mining and related research domains [7].

Spectral clustering with sparse representation has been found to be effective for clustering high-dimensional data. Spectral clustering [8] is based on the spectral graph model, which is equivalent to graph min-cut problem based on a graph structure constructed from the object space. It is powerful and stable for high-dimensional data clustering [9], and is considered superior to traditional clustering algorithms for high-dimensional data clustering due to its deterministic and polynomial-time solution [8]. Nonetheless, the effectiveness of spectral clustering mainly depends on the input 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 [10,11] and Gaussian RBF [9], is based on the Euclidean distance in the original data space, thus not suitable for high-dimensional data due to the "curse of dimensionality" in the original object space. However, sparse representation, coming from compressed sensing [12], proves to be an extremely powerful tool for acquiring, representing, and compressing high-dimensional data by representing each object approximately as a sparse linear combination of other objects. Finding sparse representations transforms the object space into a new sparse space.

Since sparse coefficients represent the contribution of each object to the reconstruction of other objects, existing spectral clustering methods based on sparse representation [13] use these sparse coefficients directly to build the weight matrix. Using the isolated coefficients individually warrants that only local information is utilized. However, we assert that exploiting more contextual information from the whole coefficient vectors promises

* Corresponding author at: Dongling School of Economics and Management, University of Science and Technology Beijing, Beijing 100083, China.
E-mail address: fxdong88@gmail.com (X. Feng).

0925-2312/$ - see front matter
http://dx.doi.org/10.1016/j.neucom.2013.12.027

better assessment of similarity among data objects. Intuitively, our assumption is 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.

Therefore, in this paper, we present a study of exploiting contextual information from sparse representation vectors to construct weight matrices for spectral clustering of high-dimensional data. More specifically, we firstly convert each high-dimensional data object into a vector of sparse coefficients, according to sparse representation theories. Then, the proximity of any two data objects is assessed according to the similarity between their sparse representation vectors. We propose two different weight matrix construction approaches: one is based on the consistency of directions, and the other is based on the consistency of magnitude. We show differences and connections between these two different approaches. Finally, spectral clustering is run on the weight matrices constructed. Extensive experiments on several real-world, high-dimensional datasets show that weights exploiting the contextual information from the sparse representation vectors work better than existing solutions, which only utilize individual sparse coefficients, by a variety of clustering performance metrics.

The main contributions of this paper can be summarized as follows. First of all, we recognize the importance of utilizing contextual information for assessing the similarity between data objects. More specifically, in the context of weight matrix construction for spectral clustering, we find that the sparse representation vectors, compared with individual sparse coefficients, contain more details and stronger evidence of similarity between data objects. In addition, we propose two concrete ways to implement the weight matrix construction utilizing sparse representation vectors. Considering the direction of weight contribution, we examine the consistency of the signs for coefficients in the sparse representation vectors. Considering the magnitude of weight contribution, we evaluate the similarity of the sparse representation vectors using the cosine measure. Finally, we validate the usefulness of the proposed approaches with real-world, high-dimensional datasets, showing that they are both better than existing methods, and work better than each other in different scenarios.

The rest of this paper is organized as follows. Related work and some preliminary details are presented in Section 2. In Section 3, we present the two different approaches to construct the weight matrix using sparse representation vectors, and describe the algorithms in detail. In Section 4, we present experiments on several real-world, high-dimensional datasets and evaluate the clustering performances. Finally, we conclude the work in Section 5.

2. Related work and preliminaries

In this section, we first review a few typical techniques for analyzing high-dimensional data. Then we focus on a brief review of the formulation and derivation of sparse representations. Finally, we focus on clustering methods to utilize sparse representations for high-dimensional data.

2.1. Techniques for high-dimensional data

There are many techniques to deal with high-dimensional signals in the literature. Popular techniques include nonnegative matrix factorization, manifold learning, compressed sensing, and combination in between.

Nonnegative matrix factorization (NMF) is a powerful dimensionality reduction technique. It has been widely applied to image processing and pattern recognition [14]. The basic idea is to approximate a non-negative matrix by the product of two non-negative, low-rank factor matrices. It was first proposed by Paatero and Tapper [15], and has attracted much attention in the research community since then. Research on NMF can be generally categorized into three groups. The first group focuses on assessing the consistency between the original matrix and the approximate matrix, using Kullback–Leibler divergence [14], Euclidean distance [16], earth mover's distance [17], Manhattan distance [18], and so on. The second group of research tries to find the optimal solution efficiently and developing scalable NMF algorithms for large-scale datasets. For example, fast Newton-type methods [19], online NMF with robust stochastic approximation [20], robust near-separable NMF using linear optimization [21], and large scale graph regularized NMF [22]. Finally, the third group of research is to improve the performance of NMF under constrains, or exploiting more information from data. These techniques include sparseness constrained NMF [23], convex model for NMF using $l_{1,\infty}$ regularization [24], discriminant NMF [25], graph regularized NMF [26], manifold regularized discriminative NMF [27], and constrained NMF incorporating the label information [28]. In particular, it has been proved that an extended version of NMF is equivalent to kernel K-means and Laplacian-based spectral clustering [29].

Manifold learning is another popular technique to process high-dimensional data, assuming that the data distribution is supported on a low-dimensional sub-manifold [30]. The key idea is that the locality structure of a high-dimensional dataset should be preserved in a low-dimensional space after dimension reduction, which is exploited as a regularization term [31–33] or constraint [34,35] to be added to the original problem formulation. It has been widely used in computer vision applications, such as image classification [36,37], semi-supervised multiview distance metric learning [38], human action recognition [39], and complex object correspondence construction [40].

Besides the two approach reviewed above, sparse representation, originated from compressed sensing, has also attracted a great deal of attention. It proves to be an extremely powerful tool for acquiring, representing, and compressing high-dimensional data. Due to its high relevance to the discussions in this paper, we provide a more detailed overview of spares representation theories in the following subsection.

2.2. A brief review of sparse representation

Given a sufficient high-dimensional training dataset $X=(\boldsymbol{x}_1,\boldsymbol{x}_2,\ldots,\boldsymbol{x}_n)\in\mathbb{R}^{m\times n}$, where $\boldsymbol{x}_i=(x_{i1},x_{i2},\ldots,x_{im})^T\in\mathbb{R}^m$ is a column vector representing the ith object. Research on manifold learning [30] has shown that any test data point $\boldsymbol{y}$ lies on a lower-dimensional manifold, which can be approximately represented by a linear combination of the training data:

$$\boldsymbol{y}=\alpha_1\boldsymbol{x}_1+\cdots+\alpha_i\boldsymbol{x}_i+\cdots+\alpha_n\boldsymbol{x}_n=X\boldsymbol{\alpha}\in\mathbb{R}^m, \tag{1}$$

where $\boldsymbol{\alpha}=(\alpha_1,\alpha_2,\ldots,\alpha_n)^T$ represents the vector of coefficients that need to be determined.

Typically, the number of training objects is much larger than the number of attributes (i.e., $n\gg m$), then Eq. (1) is undetermined, and its solution is not unique.

If we add the constraint that the best solution of $\boldsymbol{\alpha}$ in Eq. (1) should be as sparse as possible, which means that the number of non-zero elements is minimized, then the solution becomes unique. Such a sparse representation can be obtained by solving the following optimization problem:

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

where $\|\cdot\|_0$ denotes the l_0-norm of a vector, counting the number of non-zero entries in the vector. Donoho [41] proves that if matrix

X satisfies the restricted isometry property [42], then Eq. (2) has a unique solution.

However, it is NP-hard to find the sparsest solution of an underdetermined equation. In other words, there is no known approach to find the sparsest solution that is significantly more efficient than exhausting all subsets of the entries of $\boldsymbol{\alpha}$. Researchers in emerging theory of compressed sensing [43] reveal that the non-convex optimization in (2) is equivalent to the following convex l_1 optimization problem if the solution $\boldsymbol{\alpha}$ is sparse enough:

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

where $\|\cdot\|_1$ 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 with standard linear programming methods [44].

Since real-world data contain noise, it may not be possible to express the test sample exactly as a sparse representation of the training data. The sparse solution $\boldsymbol{\alpha}$ can still be approximately obtained by solving the following stable l_1 optimization problem:

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

where ε represents the maximum residual error, and $\|\cdot\|_2$ 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 algorithm [45] to recover the sparse solution from the following l_1 optimization:

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

where λ is a scalar regularization parameter of the Lasso penalty, which directly determines how sparse $\boldsymbol{\alpha}$ will be and balances the tradeoff between reconstruction error and sparsity.

In addition to Lasso, other sparse learning models have also been developed in the literature. For example, the elastic net model [46] adds the l_2-norm of $\boldsymbol{\alpha}$ to Eq. (5) as another penalty term. The double shrinking algorithm [47] compresses image data on both dimensionality and cardinality by building either sparse low-dimensional representations or a sparse projection matrix for dimension reduction. Go decomposition [48] tries to efficiently and robustly decompose a matrix into a low-rank part and a sparse part. Locality structure of manifold can also be combined with sparse representation, such as manifold elastic net [49] and graph regularized sparse coding [50], Laplacian sparse coding [33] and hypergraph Laplacian sparse coding [33].

Learning tasks such as classification and clustering usually perform better and cost less (in terms of time and space) on compressed representations than on the original data [47]. Therefore, supervised learning and pattern recognition based on the sparse representation using these sparse learning models are proposed in the literature, such as sparse representation based classification (SRC) [51], Local SRC [52], and Kernel SRC [53]. These methods are found to outperform a traditional classifier, such as support vector machines, nearest neighbor classifiers, and nearest subspace classifiers. Inspired by the successful application of sparse representation in the above supervised learning approaches, researchers have also exploited sparse representation in unsupervised [54–56] and semi-supervised learning [57,58].

2.3. Spares representation for clustering

Given a high-dimensional dataset $X = (\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_n) \in \mathbb{R}^{m\times n}$, where $\boldsymbol{x}_i = (x_{i1}, x_{i2}, \ldots, x_{im})^T \in \mathbb{R}^m$ represents the ith data object, we can use the solution from Eq. (5) to represent each object $\boldsymbol{x}_i$ as a linear combination of other objects. The coefficient vector of $\boldsymbol{x}_i$, $\boldsymbol{\alpha}_i$, can be calculated by solving the following Lasso optimization:

$$\boldsymbol{\alpha}_i^* = \arg \min_{\boldsymbol{\alpha}_i} \lambda\|\boldsymbol{\alpha}_i\|_1 + \|\boldsymbol{x}_i - X_i\boldsymbol{\alpha}_i\|_2, \tag{6}$$

where $X_i = X\backslash\boldsymbol{x}_i = (\boldsymbol{x}_1, \ldots, \boldsymbol{x}_{i-1}, \boldsymbol{x}_{i+1}, \ldots, \boldsymbol{x}_n)$ consists of all data objects except for $\boldsymbol{x}_i$, and the optimal solution $\boldsymbol{\alpha}_i^* = (\alpha_{i1}, \alpha_{i2}, \ldots, \alpha_{i,i-1}, \alpha_{i,i+1}, \ldots, \alpha_{in})^T$ consists of sparse coefficients corresponding to each data object in X_i, $\forall i = 1, 2, \ldots, n$. Formally, we provide the following definitions of terminology to be used throughout this paper.

Definition 1 (*Sparse representation vector*). Suppose the optimal solution for Eq. (6) is $\boldsymbol{\alpha}_i^* = (\alpha_{i1}, \ldots, \alpha_{i,i-1}, \alpha_{i,i+1}, \ldots, \alpha_{in})^T$, then the vector

$$\boldsymbol{\alpha}_i = (\alpha_{i1}, \ldots, \alpha_{i,i-1}, 0, \alpha_{i,i+1}, \ldots, \alpha_{in})^T, \tag{7}$$

is called the sparse representation vector of data object $\boldsymbol{x}_i$, $\forall i = 1, 2, \ldots, n$.

Definition 2 (*Sparse coefficient*). The jth element in the sparse representation vector of data object $\boldsymbol{x}_i$, α_{ij}, is the sparse coefficient of data object $\boldsymbol{x}_j$ for data object $\boldsymbol{x}_i$, $\forall i, j = 1, 2, \ldots, n$.

An interpretation for the sparse coefficient α_{ij} is data object $\boldsymbol{x}_j$'s contribution to the reconstruction of data object $\boldsymbol{x}_i$. So the sparse representation vector of $\boldsymbol{x}_i$ is a vector of contribution weights from all data objects to the reconstruction of $\boldsymbol{x}_i$. Note that by definition, since $\alpha_{ii} = 0$ ($\forall i = 1, 2, \ldots, n$), there is no contribution from a data object to itself. Also, the sparse coefficient does not have the reciprocity property. In other words, α_{ij} and α_{ji} are typically not equal, implying different levels of reconstruction contribution between a pair of data objects.

Existing weight matrix construction methods via sparse representation are based on the assumption that the sparse coefficients reflect the closeness or similarity between data objects. A few measures, including the sparsity induced similarity (SIS) measure [58], the l_1 directed graph construction (DGC) measure [57], and the nonnegative (NN) sparsity induced similarity measure [56] have been proposed in the literature. Since these measures will serve as baselines for our experiments, we describe their computation formulas as follows.

The sparsity induced similarity (SIS) measure [58] is computed as follows:

$$SIS_{ij} = \frac{\tilde{\alpha}_{ij} + \tilde{\alpha}_{ji}}{2}, \tag{8}$$

where

$$\tilde{\alpha}_{ij} = \frac{\max\{\alpha_{ij}, 0\}}{\sum_{k=1}^{n} \max\{\alpha_{ik}, 0\}}. \tag{9}$$

The main idea is to ignore negative contributions and symmetrize sparse coefficients for each pair of data objects. It is found to significantly improve label propagation performance in semi-supervised learning [58].

In addition, the l_1 directed graph construction (DGC) measure [57] is computed as follows:

$$DGC_{ij} = \frac{|\alpha_{ij}| + |\alpha_{ji}|}{2}. \tag{10}$$

Obviously, using the absolute coefficients in Eq. (10) will make the mistake of considering negative coefficients as high similarity, resulting in putting two objects with opposite attribute values into the same cluster. Unsupervised or semi-supervised learning using DGC to construct weight matrix is constantly emerging in the literature. Sparse subspace clustering [54] 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 with the weights constructed using

DGC. 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 [55] characterizes the graph adjacency structure and graph weights by sparse linear coefficients and weight construction of DGC, which is more effective than Gaussian RBF [9] to cluster an image dataset. In semi-supervised learning, the graph adjacency structure as well as the graph weights of the directed graph construction is derived simultaneously using DGC and in a parameter-free manner to utilize both labeled and unlabeled data [57]. 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 SIS and DGC, which uses real numbers from sparse representations, the nonnegative sparsity induced similarity measure (NN) [56] adds a nonnegative constraint in l_1 optimization equation (6):

$$\boldsymbol{\alpha}_i^* = \arg\min_{\boldsymbol{\alpha}_i} \lambda \|\boldsymbol{\alpha}_i\|_1 + \|\boldsymbol{x}_i - X_i\boldsymbol{\alpha}_i\|_2 \quad \text{subject to } \boldsymbol{\alpha}_i \geq \boldsymbol{0} \tag{11}$$

Then, NN exploits the symmetric coefficients of nonnegative sparse representation as a weight matrix:

$$NN_{ij} = \frac{\alpha_{ij}}{\sum_{k=1}^{n} \alpha_{ik}}. \tag{12}$$

Earlier study [56] has shown that NN outperforms SIS and Euclidean (with Gaussian RBF baseline [9]) in cluster analysis of spam images.

To a certain extent, weight measures derived from sparse representation can reveal the proximity structure among data objects without calculating the Euclidean distance, which means a great potential to clustering high-dimensional data. However, all existing approaches using spares representation reviewed above treat the sparse coefficients individually. This treatment cannot reflect the true similarity between objects, because individual sparse coefficients represent local similarity, and are sensitive to outliers. Our approach based on the overall sparse representation vectors is expected to provide more effective weight matrix construction, since more contextual information is being considered.

3. Sparse representation vectors 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 using the complete solution coefficients of sparse representation; (3) exploiting the spectral clustering algorithm with the weight matrix to find the partitioning result.

Compared to the direct weight construction methods reviewed in Section 2.3, our assumption is that for any two objects $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$, the more similar they are, the more similar the corresponding coefficient vectors (i.e., $\boldsymbol{\alpha}_i$ and $\boldsymbol{\alpha}_j$) are, but not only the particular coefficients between them (e.g., α_{ij} and α_{ji}). According to this assumption, we propose the following two weight matrix construction approaches.

3.1. Proximity based on consistent sign set

Considering two data objects, $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$, if they contribute in the same direction to the reconstruction of all other objects in the same dataset, they are considered similar. Thus our first idea to assess the similarity between $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ is based on the number of consistent signs in their sparse contribution vectors. In this section, we first provide a formal definition of the sparse contribution vector and the consistent sign set, and then provide the computation formula for the similarity based on consistent sign sets.

Definition 3 (*Sparse contribution vector*). Given matrix

$$A = (\boldsymbol{\alpha}_1, \boldsymbol{\alpha}_2, \ldots, \boldsymbol{\alpha}_n) = \begin{pmatrix} 0 & \alpha_{21} & \cdots & \alpha_{n1} \\ \alpha_{12} & 0 & \cdots & \alpha_{n2} \\ \vdots & \vdots & \ddots & \vdots \\ \alpha_{1n} & \alpha_{2n} & \cdots & 0 \end{pmatrix}, \tag{13}$$

where each column is a sparse representation vector of a data object, each row vector

$$\boldsymbol{\alpha}_i' = (\alpha_{1,i}, \ldots, \alpha_{i-1,i}, 0, \alpha_{i+1,i}, \ldots, \alpha_{n,i}), \tag{14}$$

is called the sparse contribution vector of data object $\boldsymbol{x}_i$, $\forall i = 1, 2, \ldots, n$.

Definition 4 (*Consistent sign set*). Given their sparse contribution vectors, the consistent sign set between $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ is the set of other objects to which the contribution from $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ is both positive. Formally,

$$S(\boldsymbol{x}_i, \boldsymbol{x}_j) = \{\boldsymbol{x}_k | \alpha_{ki} > 0 \wedge \alpha_{kj} > 0\} \tag{15}$$

The proximity between objects $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ base on the consistent sign set (CSS) is then defined as follows:

$$CSS_{ij} = \frac{|S(\boldsymbol{x}_i, \boldsymbol{x}_j)|}{n} \tag{16}$$

where $|\cdot|$ stands for the cardinality of a set, and n is the total number of objects in X. Obviously, the weight takes a value between 0 and 1.

3.2. Proximity based on cosine similarity of coefficient vector

Consider $\boldsymbol{\alpha}_i$ and $\boldsymbol{\alpha}_j$, the sparse representation vectors of data objects $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$. If $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ are similar, then we expect their sparse representation vectors $\boldsymbol{\alpha}_i$ and $\boldsymbol{\alpha}_j$ to be similar. Since cosine measure is a commonly used as a similarity measure between two vectors, the second approach we consider is to construct the weight matrix based on the cosine similarity between the sparse representation vectors. In particular, the weight between object $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ is defined as follows.

$$COS_{ij} = \max\left\{0, \frac{\boldsymbol{\alpha}_i \cdot \boldsymbol{\alpha}_j}{\|\boldsymbol{\alpha}_i\|_2 \times \|\boldsymbol{\alpha}_j\|_2}\right\} \tag{17}$$

3.3. Numeric examples

To illustrate the differences between our approaches for weight construction and others (e.g., those reviewed in Section 2.3), two numeric examples are given as follows.

Assume that there are $n=5$ data objects in a dataset X, and we find the following sparse contribution vectors for $\boldsymbol{x}_4$ and $\boldsymbol{x}_5$:

$\boldsymbol{\alpha}_4' = (.3, .4, .4, .0, -.1)$;
$\boldsymbol{\alpha}_5' = (.3, .4, .4, -.1, .0)$.

Then we can that $CSS_{45} = .6$, which shows a strong proximity between $\boldsymbol{x}_4$ and $\boldsymbol{x}_5$ because they both have positive contribution to $\boldsymbol{x}_1$, $\boldsymbol{x}_2$, and $\boldsymbol{x}_3$. However, $SIS_{45} = 0$ and $DGC_{45} = .1$ both show a low proximity between $\boldsymbol{x}_4$ and $\boldsymbol{x}_5$, since they only consider the contribution to each other, utilizing local information only.

Assume further that for the $n=5$ data objects in a dataset X, and we find the following sparse representation vectors for $\boldsymbol{x}_4$ and $\boldsymbol{x}_5$:

$\boldsymbol{\alpha}_4 = (.3, .4, .4, .0, -.1)^T$;

$\boldsymbol{\alpha}_5 = (.3, .4, .4, -.1, .0)^T$.

Then we can assume that $COS_{45} = .97$, which shows a strong proximity between $\boldsymbol{x}_4$ and $\boldsymbol{x}_5$ because the reconstruction weights from $\boldsymbol{x}_1$, $\boldsymbol{x}_2$, and $\boldsymbol{x}_3$ are quite consistent to both of them. However, $SIS_{45} = 0$ and $DGC_{45} = .1$ still show a low proximity between $\boldsymbol{x}_4$ and $\boldsymbol{x}_5$, since they only consider the contribution to each other.

3.4. The relationship between CSS and COS

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

First, both of them assess 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 sparse contribution vector, while proximity based on COS calculates the similarity between sparse representation vectors, which mean two different understanding of the coefficient matrix. The reason for defining these two approaches like this is just experimental.

Moreover, 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.

Finally, specifically another equivalent way to understand proximity based on CSS is as follows. First, transform the coefficient matrix A to DA, such that

$$DA(i,j) = \begin{cases} 1, & A(i,j) > 0 \\ 0 & \text{otherwise} \end{cases} \tag{18}$$

Then the weight between $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ is:

$$w_{ij} = \begin{cases} \dfrac{DA^i \cdot DA^j}{n}, & i \neq j \\ 0, & i = j \end{cases} \tag{19}$$

where DA^i denotes the ith column vector of DA.

Obviously, the inner product between DA^i and DA^j (i.e., $DA^i \cdot DA^j$) is equal to $CSS(\boldsymbol{x}_i, \boldsymbol{x}_j)$'s cardinal $|CSS(\boldsymbol{x}_i, \boldsymbol{x}_j)|$. Therefore the similarity measure in CSS can also be seen as the cosine similarity of two binary vectors regardless of the augmentation of denominator in the fraction which demonstrates the consistency between CSS and COS.

3.5. 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: A high-dimensional dataset $X = (\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_n) \in \mathbb{R}^{m \times n}$, where $\boldsymbol{x}_i = (x_{i1}, x_{i2}, \ldots, x_{im})^T \in \mathbb{R}^m$ represents the ith data object; the number of clusters K
Param: penalty coefficient λ for Lasso optimization
Output: cluster labels corresponding to each data object $\boldsymbol{c} = (c_1, c_2, \ldots, c_n)$

1 **foreach** *data object* $\boldsymbol{x}_i \in X$ **do**
2 Set $X_i = X \backslash \boldsymbol{x}_i = (\boldsymbol{x}_1, \ldots, \boldsymbol{x}_{i-1}, \boldsymbol{x}_{i+1}, \ldots, \boldsymbol{x}_n)$;
3 $\boldsymbol{\alpha}_i^* \leftarrow \arg\min_{\boldsymbol{\alpha}_i} \lambda \|\boldsymbol{\alpha}_i\|_1 + \|\boldsymbol{x}_i - X_i \boldsymbol{\alpha}_i\|_2$;
4 **end**
5 $W \leftarrow$ `ConstructWeightMatrix` $(\boldsymbol{\alpha}_1^*, \boldsymbol{\alpha}_2^*, \ldots, \boldsymbol{\alpha}_n^*)$;
6 $\boldsymbol{c} \leftarrow$ `SpectralClustering`(W);
7 **return** $\boldsymbol{c}$

The `ConstructWeightMatrix()` sub-routine can exploit any weight matrix construction method, such as those mentioned in Section 3. In particular, we describe the algorithm for computing the two newly proposed weight matrices, one is based on the consistent sign set (see Section 3.1) and the other based on cosine similarity of sparse representation vectors (see Section 3.2).

Algorithm 2 describes the procedure to construct the weight matrix according to the concept of consistent sign set. 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 W

1 **for** $i \leftarrow 1$ to n **do**
2 **for** $j \leftarrow 1$ to n **do**
3 **if** $j = i$ **then** $w_{ij} \leftarrow 0$ **else**
4 $n_{CSS} \leftarrow 0$;
5 **for** $k \leftarrow 1$ to n **do**
6 **if** $k \neq i$ and $k \neq j$ and $\alpha_{k,i} > 0$ and $\alpha_{k,j} > 0$ **then**
7 $n_{CSS} \leftarrow n_{CSS} + 1$;
8 **end**
9 **end**
10 $w_{ij} \leftarrow \frac{n_{CSS}}{n}$
11 **end**
12 **end**
13 **end**

Algorithm 3 describes the procedure to construct the weight matrix according to the cosine similarity of the sparse coefficients between each pair of items. The computation complexity for calculating the cosine similarity of two vectors of length n is $O(n)$, and there are $O(n^2)$ pairs of data objects whose cosine similarity needs to be computed. Thus the complexity for cosine similarity based weight matrix construction is $O(n^3)$.

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

Input: Coefficients for sparse representation $\boldsymbol{\alpha}$
Output: Weight matrix W

1 **for** $i \leftarrow 1$ to n **do**
2 **for** $j \leftarrow 1$ to n **do**
3 **if** $j = i$ **then** $w_{ij} \leftarrow 0$ **else**
4 $\cos \leftarrow \frac{\boldsymbol{\alpha}_i' \cdot \boldsymbol{\alpha}_j'}{\|\boldsymbol{\alpha}_i'\|_2 \times \|\boldsymbol{\alpha}_j'\|_2}$;
5 **if** $\cos > 0$ **then** $w_{ij} \leftarrow \cos$ **else** $w_{ij} \leftarrow 0$
6 **end**
7 **end**
8 **end**

As shown in Line 6 of Algorithm 1, after constructing the weight matrix W, we use can the classical spectral clustering

algorithm [9] to discover the cluster structure of high-dimensional data.

The main characteristics of our proposed algorithm are the following:

1. Compared to traditional graph construction induced from the Euclidean distance or other measures in the original high-dimensional space, proposed weight matrix is constructed by transforming the high-dimensional data space into another space via sparse representation, which is expected to have better performance attributed to the superiority of compressed sensing [12] 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 [57].
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 complete 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 on real-world datasets to demonstrate the effectiveness of our proposed approaches.

4.1. Datasets

We select three datasets from the UCI machine learning repository [59] and three face recognition datasets [60–62] as the benchmark for experiments. These are typical high-dimensional datasets, and are widely used in the machine learning and data mining research community. Table 1 lists a summary of these datasets.

The raw datasets have been preprocessed as follows. In the `Yale` and `ORL Face` datasets, each image is transformed into the 32×32 pixel configuration using the MATLAB Image Processing Toolbox. A stratified sample is taken from the `Yale_B` dataset to alleviate the computation cost for clustering. Specifically, we randomly select 60 images from each of the 10 classes in `Yale_B`. Datasets from UCI have been re-scaled so that each attribute ranges from 0 to 1. Face recognition datasets are converted to unit norm.

Table 1
Summary of datasets.

Name	# Instances	# Attributes	# Classes	Source
`Heart`	270	13	2	UCI [59]
`Image`	2310	18	7	UCI [59]
`Movement`	360	90	15	UCI [59]
`Yale`	165	1024	15	Georghiades [60]
`Yale B`	600	1200	10	Georghiades et al. [61]
`ORL Face`	400	1024	40	ORL [62]

4.2. Algorithm implementation

We use a MATLAB implementation [63] based on the interior-point method to solve sparse coefficients for each data object. Then, we implement different weight matrices for in MATLAB to cluster each dataset. Table 2 shows a summary of the proposed and baseline algorithms.

4.3. Evaluation metrics

Since the true class labels of each dataset are known, five commonly used external cluster validation metrics [65–67] are employed to evaluate the clustering results. Namely, clustering accuracy (CA), entropy (E), F-measure (F), normalized mutual information (NMI), and rand index (RI), as listed in Table 3. These metrics are commonly used to measure the consistency between the clusters identified by a clustering algorithm and the true groups (i.e., by class labels) of the data objects.

Specifically, assume that a dataset with n objects from K classes was clustered into K clusters. In Table 3, n_{kj} represents the number of objects from class k and attributed to cluster j, $n_k = \sum_{j=1}^{K} n_{kj}$, and $n_j = \sum_{k=1}^{K} n_{kj}$. f_{00} is the number of pairs of objects in different clusters and also in different true classes; f_{11} denotes the number of pairs of objects in the same clusters and also in the same true classes. The last column in Table 3 summarizes the direction of these measures. Namely, the smaller E is, or the larger the CA/F/NMI/RI is, the better the clustering performance.

4.4. Performance comparison

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 Fig. 1, taking the `Yale` dataset as an example. In Fig. 1, each subfigure is a weight matrix with N-by-N entries (entries larger than the threshold are shown in white, otherwise black). 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 are sparse, we set a threshold to get the best sparse matrices in Fig. 1. The best sparse matrix is defined as maximize $r = r\_within - r\_between$ for each cutoff c of threshold $\theta \in [0, 1]$ with interval .01. r_within is defined as the ratio of the

Table 2
Summary of algorithms to be compared.

Name	Description	Source	Role
CSS	Spectral clustering with consistent sign set as the weight matrix	Section 3.1	Solution Proposed
COS	Spectral clustering with cosine similarity of sparse coefficients as the weight matrix	Section 3.2	Solution Proposed
RBF	Spectral clustering with weight matrix from Gaussian RBF	Ng et al. [9]	Baseline
SIS	Spectral clustering with weight matrix from sparsity induced similarity measure	Cheng et al. [58]	Baseline
DGC	Spectral clustering with weight matrix from l_1 directed graph construction	Yan and Wang [57]	Baseline
NN	Spectral clustering with weight matrix from nonnegative sparsity induced similarity measure	Gao et al. [56]	Baseline
KM	K-means clustering with raw sparse representation vectors	Hartigan and Wong [64]	Baseline

Table 3
A few commonly used external validation metrics.

Measure	Description	Formula	Direction
CA	Clustering accuracy	$\frac{1}{n}\sum_{k=1}^{K} n_{kk}$	↑
E	Entropy	$\sum_{k=1}^{K}\frac{n_k}{n}\left(-\frac{1}{\log_2 K}\sum_{j=1}^{K}\frac{n_{kj}}{n_k}\log_2\frac{n_{kj}}{n_k}\right)$	↓
F	F-measure	$\sum_{k=1}^{K}\frac{n_k}{n}\cdot\max_{1\le j\le K}\left\{\frac{2\cdot\frac{n_{kj}}{n_k}\cdot\frac{n_{kj}}{n_j}}{\frac{n_{kj}}{n_k}+\frac{n_{kj}}{n_j}}\right\}$	↑
NMI	Normalized mutual information	$\frac{\sum_{k=1}^{K}\sum_{j=1}^{K} n_{kj}\log_2\frac{nn_{kj}}{n_k n_j}}{\sqrt{\sum_{k=1}^{K} n_k\log_2\frac{n_k}{n}}\sqrt{\sum_{j=1}^{K} n_j\log_2\frac{n_j}{n}}}$	↑
RI	Rand index	$\frac{f_{00}+f_{11}}{n(n-1)/2}$	↑

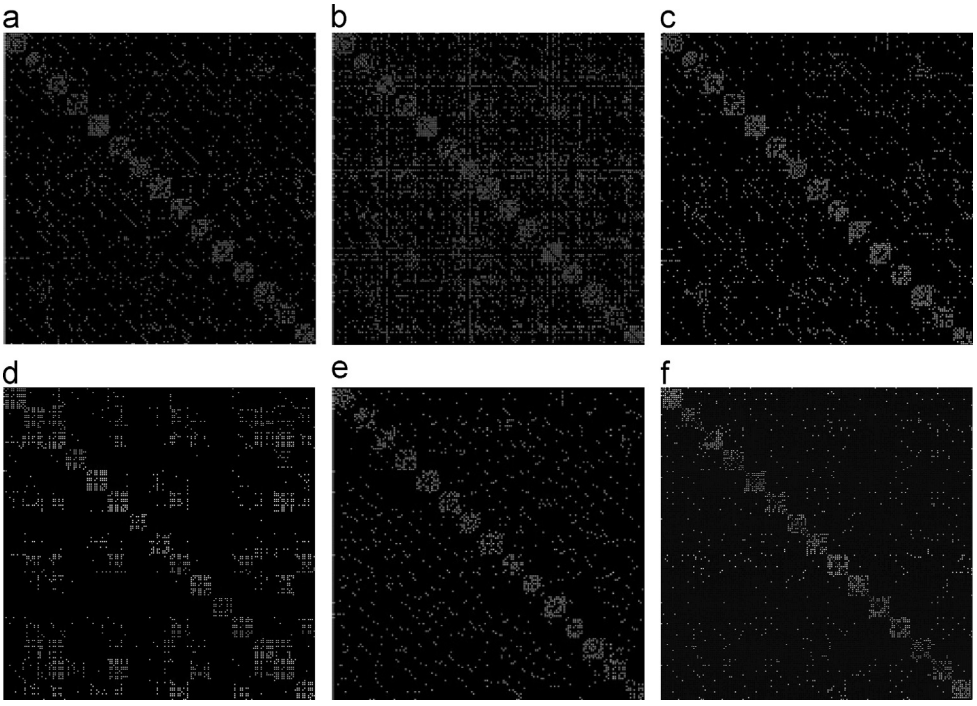


Fig. 1. Visualization of the graph weight matrices of the `Yale` dataset, where images from the same subject are arranged together. (a) SIS ($\theta=.01, r=.52$), (b) DGC ($\theta=.01, r=.50$), (c) NN ($\theta=.01, r=.42$), (d) RBF ($\theta=.97$, $\sigma=4, r=.38$), (e) CSS ($\theta=.35, r=.45$), (f) COS ($\theta=.15, r=.70$).

Table 4
Evaluation of all algorithms with CA as metric.

Dataset		CSS	COS	DGC	SIS	NN	RBF	KM
`Heart`	Mean	.7704	**.8174**	.5852	.7889	.7519	.7963	.7320
	(std)	(**.0000**)	(**.0000**)	(**.0000**)	(**.0000**)	(**.0000**)	(**.0000**)	(.0882)
`Image`	Mean	.7631	**.7921**	.7020	.7820	.7360	.5335	.6215
	(std)	(**.0148**)	(.0323)	(.0339)	(.0341)	(.0379)	(.0305)	(.0355)
`Movement`	Mean	.5241	**.5472**	.5009	.5183	.5304	.4874	.4653
	(std)	(.0222)	(**.0187**)	(.0248)	(.0193)	(.0271)	(.0232)	(.0203)
`Yale`	Mean	.6823	**.7408**	.7178	.7023	.6417	.6635	.5482
	(std)	(.0432)	(.0345)	(.0414)	(**.0314**)	(.0412)	(.0395)	(.0529)
`Yale B`	Mean	.8572	.8937	.8320	.8620	**.8940**	.6918	.6862
	(std)	(.0791)	(.0713)	(.0768)	(.0635)	(.0767)	(**.0226**)	(.0721)
`ORL Face`	Mean	.7315	**.7570**	.7225	.7243	.6903	.7314	.7196
	(std)	(.0247)	(.0218)	(.0222)	(.0244)	(**.0207**)	(.0252)	(.0311)
Average	Mean	.7214	**.7580**	.6767	.7296	.7074	.6506	.6288
	(std)	(.0397)	(.0368)	(.0405)	(.0345)	(.0412)	**(.0264)**	(.0554)

number of the white cells within clusters to the number of all possible cells within clusters; *r_between* is defined as the ratio of the number of the white cells between clusters to the number of all possible cells between clusters.

Normally, the clustering performance will be good if the weights between two objects from different clusters are small (black cells in Fig. 1), while weights from the same cluster are large (white cells in Fig. 1). This comment can shown that in the matrix

Table 5
Evaluation of all algorithms with E as metric.

Dataset		CSS	COS	DGC	SIS	NN	RBF	KM
Heart	Mean	.7773	**.6865**	.9379	.8156	.8051	.7288	.7718
	(std)	(**.0000**)	(**.0000**)	(**.0000**)	(**.0000**)	(.2096)	(**.0000**)	(.0893)
Image	Mean	.2939	.2852	.4137	**.2774**	.3468	.5639	.4175
	(std)	(**.0100**)	(.0273)	(.0259)	(.0290)	(.0346)	(.0393)	(.0300)
Movement	Mean	.4253	**.4084**	.4263	.4159	.4193	.4309	.4376
	(std)	(.0200)	(**.0161**)	(.0165)	(.0161)	(.0219)	(.0170)	(.0186)
Yale	Mean	.2898	**.2273**	.2726	.2558	.3227	.3074	.3747
	(std	(**.0199**)	(.0212)	(.0251)	(.0208)	(.0235)	(.0274)	(.0416)
Yale B	Mean	.0472	**.0243**	.0444	.0347	.0272	.2027	.1517
	(std)	(.0214)	(.0204)	(.0192)	(**.0157**)	(.0198)	(.0278)	(.0514)
ORL Face	Mean	.1591	**.1399**	.1644	.1577	.1774	.1551	.1540
	(std)	(.0132)	(**.0120**)	(.0123)	(.0139)	(.0133)	(.0148)	(.0197)
Average	Mean	.3321	**.2953**	.3766	.3262	.3497	.3981	.3846
	(std)	**(.0159)**	(.0183)	(.0187)	(.0181)	(.0883)	(.0244)	(.0483)

Table 6
Evaluation of all algorithms with F as metric.

Dataset		CSS	COS	DGC	SIS	NN	RBF	KM
Heart	Mean	.7699	**.8142**	.6374	.7331	.5714	.7958	.7414
	(std)	(**.0000**)	(**.0000**)	(**.0000**)	(**.0000**)	(.0327)	(**.0000**)	(.0837)
Image	Mean	.7496	**.7757**	.6921	.7737	.7353	.5570	.6458
	(std)	(**.0192**)	(.0354)	(.0299)	(.0374)	(.0330)	(.0345)	(.0359)
Movement	Mean	.5342	**.5601**	.5206	.5306	.5483	.4870	.4670
	(std)	(.0220)	(**.0190**)	(.0238)	(.0205)	(.0251)	(.0232)	(.0215)
Yale	Mean	.6763	**.7153**	.6820	.6540	.6108	.6629	.5896
	(std)	(**.0286**)	(.0304)	(.0346)	(.0327)	(.0411)	(.0367)	(.0364)
Yale B	Mean	.8627	**.9103**	.8473	.8807	.9053	.6869	.6868
	(std)	(.0719)	(.0661)	(.0741)	(.0565)	(.0718)	(**.0520**)	(.0754)
ORL Face	Mean	.7304	**.7584**	.7181	.7210	.6786	.7305	.7102
	(std)	(.0252)	(.0257)	(.0229)	(.0252)	(**.0224**)	(.0264)	(.0319)
Average	Mean	.7205	**.7557**	.6829	.7155	.6750	.6533	.6401
	(std)	(.0353)	(.0355)	(.0380)	(.0334)	(.0411)	**(.0328)**	(.0529)

Table 7
Evaluation of all algorithms with NMI as metric.

Dataset		CSS	COS	DGC	SIS	NN	RBF	KM
Heart	Mean	.2208	**.3149**	.0511	.1791	.0331	.2712	.2028
	(std	(**.0000**)	(**.0000**)	(**.0000**)	(**.0000**)	(.0312)	(**.0000**)	(.1013)
Image	Mean	.7088	**.7451**	.5921	.7319	.6637	.7451	.6122
	(std)	(**.0071**)	(.0184)	(.0171)	(.0176)	(.0357)	(.0284)	(.0437)
Movement	Mean	.5891	.5914	.5933	.6000	**.6306**	.5741	.5818
	(std)	(.0128)	(**.0124**)	(.0140)	(.0130)	(.0173)	(.0169)	(.0180)
Yale	Mean	.7137	**.7815**	.7513	.7641	.6926	.6989	.6484
	(std)	(.0211)	(**.0183**)	(.0203)	(.0188)	(.0198)	(.0271)	(.0342)
Yale B	Mean	.9008	**.9526**	.9202	.9379	.9510	.7768	.7858
	(std)	(.0302)	(.0360)	(.0422)	(.0326)	(.0398)	(**.0224**)	(.0621)
ORL Face	Mean	.8477	**.8688**	.8492	.8547	.8426	.8512	.8620
	(std)	(.0116)	(.0108)	(**.0105**)	(.0118)	(.0109)	(.0140)	(.0157)
Average	Mean	.6635	**.7090**	.6262	.6779	.6356	.6529	.6155
	(std)	**(.0169)**	(.0193)	(.0216)	(.0184)	(.0278)	(.0205)	(.0544)

Table 8
Evaluation of all algorithms with RI as metric.

Dataset		CSS	COS	DGC	SIS	NN	RBF	KM
Heart	Mean	.6449	**.6971**	.5358	.6074	.5026	.6744	.6277
	(std)	(**.0000**)	(**.0000**)	(**.0000**)	(**.0000**)	(.0009)	(**.0000**)	(.0684)
Image	Mean	.9106	**.9122**	.8816	.9031	.8964	.7899	.8534
	(std)	(**.0065**)	(.0130)	(.0117)	(.0139)	(.0124)	(.0151)	(.0156)
Movement	Mean	.9142	**.9167**	.9069	.9105	.9130	.9128	.9003
	(std)	(**.0032**)	(.0048)	(.0066)	(.0068)	(.0082)	(.0087)	(.0052)
Yale	Mean	.9246	**.9499**	.9222	.9291	.9278	.9387	.9172
	(std)	(**.0052**)	(.0063)	(.0109)	(.0093)	(.0077)	(.0083)	(.0138)
Yale B	Mean	.9336	**.9719**	.9519	.9642	.9705	.9223	.9019
	(std)	(.0258)	(.0246)	(.0330)	(.0198)	(.0285)	(**.0097**)	(.0303)
ORL Face	Mean	.9808	**.9825**	.9783	.9794	.9748	.9811	.9785
	(std)	(**.0017**)	(.0018)	(.0021)	(.0023)	(.0029)	(.0022)	(.0036)
Average	Mean	.8848	**.9050**	.8628	.8823	.8642	.8698	.8632
	(std)	(.0112)	(.0118)	(.0152)	(.0110)	(.0136)	**(.0089)**	(.0318)

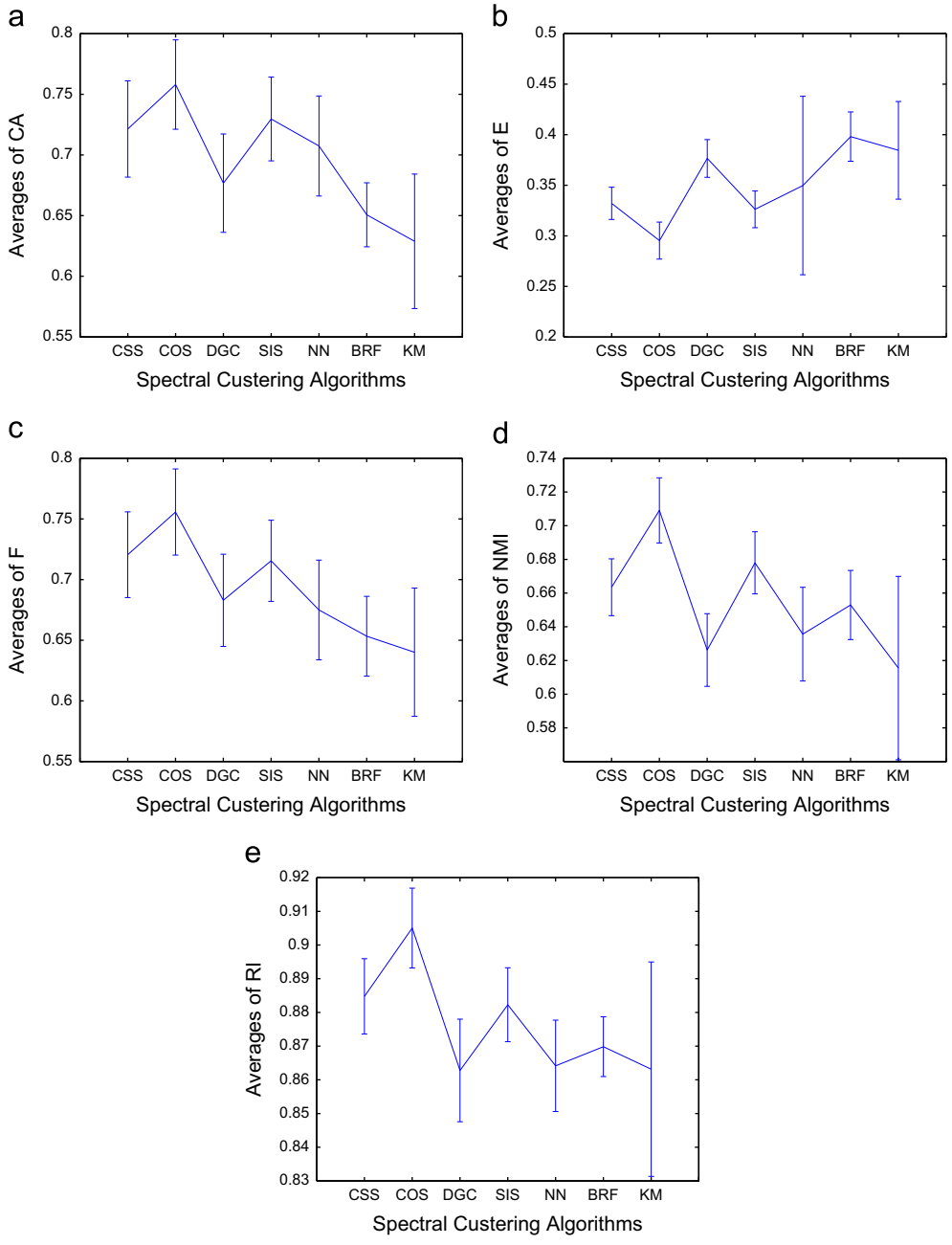


Fig. 2. A comparison of average clustering results from each method. (a) CA, (b) Entropy, (c) F, (d) NMI, and (e) RI.

with above arrangement effective matrix should be compact in diagonal position and sparse in other position.

From Fig. 1, we have the following observations: (1) matrixes 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 approximate performance compared with SIS and NN graph; DGC and RBF are worse than other weight matrices in the Yale dataset.

The clustering results obtained from the seven clustering algorithms with different evaluation metrics are reported in Tables 4–8, each of which corresponds to one evaluation metric. For each dataset, the best results are in bold. All the numbers, except for 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 datasets. 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 4–8, we can clearly see that, generally, the COS algorithm gets the best mean clustering performance with all five evaluation metrics in almost every dataset. However, there are also some particular cases where COS does not get best result. For example, though NN gets the best CA on `Yale B` datasets, COS gets almost the same CA result as NN, that is, .8937–.8940; though NN also gets the best NMI for the `Movement` dataset and SIS gets best E in the `Image` dataset, COS gets best result in other metric on these two datasets. CSS gets the second best average mean with F and NMI metrics and the approximate performance compared to SIS, which is better than other four approaches. CSS get lowest average standard deviation with NMI and E metrics. However, it is noticed that COS outperforms CSS in terms average mean with all metrics, while CSS is superior to COS in terms of average std with all metrics except for CA. 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 approaches.

Overall, for most datasets, CSS and COS have better performance than those baselines, which is robust across various external validation metrics.

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

Fig. 2 clearly demonstrates that the COS algorithm outperforms other algorithms; CSS has better performance than SIS with F and RI metrics, approximative performance with other metrics. CSS has the lower average value of standard deviation than COS. The KM and DGC algorithms have comparable performance, which is usually worse than the other algorithms.

5. Conclusions

In this paper, 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 representation 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. Evaluation experiments on real-world datasets show that spectral clustering for high dimensional data using our novel weight matrix construction outperforms direct k-means and spectral clustering approaches using Gaussian RBF, SIS, l_1-directed graph construction and nonnegative SIS in five evaluation metrics (CA, Entropy, F-measure, NMI, RI). These results demonstrate a reliable performance of our algorithm, and therefore promise wide applicability in practice. The findings also shed light on developing theories of complete information from solutions in the future work.

Acknowledgments

This work is supported by the National Natural Science Foundation of China under Grant no. 71271027, China scholarship Council and the Fundamental Research Funds for the Central Universities of China under Grant no. FRF-TP-10-006B.

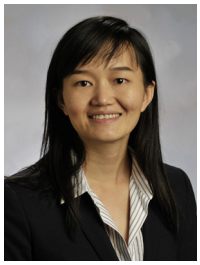
References

[1] Y. Liu, X. Wang, C. Wu, ConSOM: A conceptional self-organizing map model for text clustering, Neurocomputing 71 (2008) 857–862.
[2] Y. Koren, R. Bell, C. Volinsky, Matrix factorization techniques for recommender systems, Computer 42 (2009) 30–37.
[3] C.A. Bhatt, M.S. Kankanhalli, Multimedia data mining: state of the art and challenges, Multimed. Tools Appl. 51 (2011) 35–76.
[4] X. Zhang, L. Jiaqi, D. Yu, L. Tingjie, A novel clustering method on time series data, Expert Syst. Appl. 38 (2011) 11891–11900.
[5] J. Sun, W. Chen, W. Fang, X. Wun, W. Xu, Gene expression data analysis with the clustering method based on an improved quantum-behaved particle swarm optimization, Eng. Appl. Artif. Intell. 25 (2012) 376–391.
[6] M. Steinbach, L. Ertöz, V. Kumar, The challenges of clustering high dimensional data, in: New Directions in Statistical Physics, Springer Berlin Heidelberg, 2004, pp. 273–309.
[7] X. Chen, Y. Ye, X. Xu, Z.J. Huang, A feature group weighting method for subspace clustering of high-dimensional data, Pattern Recognit. 45 (2012) 434–446.
[8] U. von Luxburg, A tutorial on spectral clustering, Stat. Comput. 17 (2007) 395–416.
[9] A.Y. Ng, M.I. Jordan, Y. Weiss, On spectral clustering: analysis and an algorithm, Adv. Neural Inf. Process. Syst. 2 (2002) 849–856.
[10] M. Belkin, P. Niyogi, Laplacian eigenmaps for dimensionality reduction and data representation, Neural Comput. 15 (2003) 1373–1396.
[11] J.B. Tenenbaum, V. de Silva, J.C. Langford, A global geometric framework for nonlinear dimensionality reduction, Science 290 (2000) 2319–2323.
[12] D.L. Donoho, Compressed sensing, IEEE Trans. Inf. Theory 52 (2006) 1289–1306.
[13] J. Wright, Y. Ma, J. Mairal, G. Sapiro, T.S. Huang, S. Yan, Sparse representation for computer vision and pattern recognition, Proc. IEEE 98 (2010) 1031–1044.
[14] D.D. Lee, S.H. Seung, Learning the parts of objects by non-negative matrix factorization, Nature 401 (1999) 788–791.
[15] P. Paatero, U. Tapper, Positive matrix factorization: a non-negative factor model with optimal utilization of error estimates of data values, Environmetrics 5 (1994) 111–126.
[16] D.D. Lee, H.S. Seung, Algorithms for non-negative matrix factorization, Adv. Neural Inf. Process. Syst. 13 (2001) 556–562.
[17] R. Sandler, M. Lindenbaum, Nonnegative matrix factorization with earth mover's distance metric for image analysis, IEEE Trans. Pattern Anal. Mach. Intell. 33 (2011) 1590–1602.
[18] N. Guan, D. Tao, Z. Luo, J.S. Taylor, MahNMF: Manhattan non-negative matrix factorization, arXiv preprint, arXiv:1207.3438, 2012.
[19] D. Kim, S. Sra, I.S. Dhillon, Fast newton-type methods for the least squares nonnegative matrix approximation problem, in: Proceedings of the 2007 SIAM International Conference on Data Mining, 2007.
[20] N. Guan, D. Tao, Z. Luo, B. Yuan, Online nonnegative matrix factorization with robust stochastic approximation, IEEE Trans. Neural Netw. Learn. Syst. 23 (2012) 1087–1099.
[21] N. Gillis, R. Luce, Robust near-separable nonnegative matrix factorization using linear optimization, arXiv preprint, arXiv:1302.4385, 2013.
[22] M. Sun, H.V. Hamme, Large scale graph regularized non-negative matrix factorization with l_1 normalization based on Kullback–Leibler divergence, IEEE Trans. Signal Process. 60 (2012) 3876–3880.
[23] P.O. Hoyer, Non-negative matrix factorization with sparseness constraints, J. Mach. Learn. Res. 5 (2004) 1457–1469.
[24] E. Esser, M. Moller, S. Osher, G. Sapiro, J. Xin, A convex model for nonnegative matrix factorization and dimensionality reduction on physical space, IEEE Trans. Image Process. 21 (2012) 3239–3252.

[25] S. Zafeiriou, A. Tefas, S. Zafeiriou, A. Tefas, Exploiting discriminant information in nonnegative matrix factorization with application to frontal face verification, IEEE Trans. Neural Netw. 17 (2006) 683–695.
[26] D. Cai, X. He, J. Han, T.S. Huang, Graph regularized nonnegative matrix factorization for data representation, IEEE Trans. Pattern Anal. Mach. Intell. 33 (2011) 1548–1560.
[27] N. Guan, D. Tao, Z. Luo, B. Yuan, Manifold regularized discriminative nonnegative matrix factorization with fast gradient descent, IEEE Trans. Image Process. 20 (2011) 2030–2048.
[28] H. Liu, Z. Wu, X. Li, D. Cai, T.S. Huang, Constrained nonnegative matrix factorization for image representation, IEEE Trans. Pattern Anal. Mach. Intell. 34 (2012) 1299–1311.
[29] C.H. Ding, X. He, H.D. Simon, On the equivalence of nonnegative matrix factorization and spectral clustering, in: Proceedings of the 2005 SIAM International Conference on Data Mining, SIAM, 2005, pp. 606–610.
[30] S.T. Roweis, L.K. Saul, Nonlinear dimensionality reduction by locally linear embedding, Science 290 (2000) 2323–2326.
[31] Y. Luo, D. Tao, B. Geng, C. Xu, S.J. Maybank, Manifold regularized multi-task learning for semi-supervised multilabel image classification, IEEE Trans. Image Process. 22 (2013) 523–536.
[32] M. Belkin, P. Niyogi, V. Sindhwani, Manifold regularization: a geometric framework for learning from labeled and unlabeled examples, J. Mach. Learn. Res. 7 (2006) 2399–2434.
[33] S. Gao, I.W.-H. Tsang, L.-T. Chia, Laplacian sparse coding, hypergraph laplacian sparse coding, and applications, IEEE Trans. Pattern Anal. Mach. Intell. 35 (2013) 92–104.
[34] Y. Zhou, K. Barner, Locality constrained dictionary learning for nonlinear dimensionality reduction, IEEE Signal Process. Lett. 20 (2013) 335–338.
[35] J. Wang, J. Yang, K. Yu, F. Lv, T. Huang, Y. Gong, Locality-constrained linear coding for image classification, in: Proceedings of the 23rd IEEE Conference on Computer Vision and Pattern Recognition (CVPR), IEEE, 2010, pp. 3360–3367.
[36] J. Yu, D. Tao, M. Wang, Adaptive hypergraph learning and its application in image classification, IEEE Trans. Image Process. 21 (2012) 3262–3272.
[37] Y. Luo, D. Tao, S. Member, C. Xu, C. Xu, H. Liu, Y. Wen, Multiview vector-valued manifold regularization for multilabel image classification, IEEE Trans. Neural Netw. Learn. Syst. 24 (2013) 709–722.
[38] J. Yu, M. Wang, D. Tao, Semi-supervised multiview distance metric learning for cartoon synthesis, IEEE Trans. Image Process. 21 (2012) 4636–4648.
[39] X. Deng, X. Liu, M. Song, J. Cheng, J. Bu, C. Chen, LF-EME: Local features with elastic manifold embedding for human action recognition, Neurocomputing 99 (2012) 144–153.
[40] J. Yu, D. Liu, D. Tao, H.S. Seah, Complex object correspondence construction in two-dimensional animation, IEEE Trans. Image Process. 20 (2011) 3257–3269.
[41] D.L. Donoho, For most large underdetermined systems of equations, the minimal l_1-norm near-solution approximates the sparsest near-solution, Commun. Pure Appl. Math. 59 (2006) 797–829.
[42] E.J. Candès, The restricted isometry property and its implications for compressed sensing, Comptes Rendus Math. 346 (2008) 589–592.
[43] E.J. Candès, Compressive sampling, in: Proceedings of the International Congress of Mathematicians, 2006, pp. 1433–1452.
[44] S.S. Chen, D.L. Donoho, M.A. Saunders, Atomic decomposition by basis pursuit, SIAM J. Sci. Comput. 20 (1998) 33–61.
[45] R. Tibshirani, Regression shrinkage and selection via the lasso, J. R. Stat. Soc.: Ser. B (Stat. Methodol.) 58 (1996) 267–288.
[46] H. Zou, T. Hastie, Regularization and variable selection via the elastic net, J. R. Stat. Soc.: Ser. B (Stat. Methodol.) 67 (2005) 301–320.
[47] T. Zhou, D. Tao, Double shrinking sparse dimension reduction, IEEE Trans. Pattern Anal. Mach. Intell. 22 (2013) 244–257.
[48] T. Zhou, D. Tao, GoDec: randomized low-rank & sparse matrix decomposition in noisy case, in: Proceedings of the 28th International Conference on Machine Learning, ACM, 2011, pp. 33–40.
[49] T. Zhou, D. Tao, X. Wu, Manifold elastic net: a unified framework for sparse dimension reduction, Data Min. Knowl. Discov. 22 (2011) 340–371.
[50] 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 (2011) 1327–1336.
[51] J. Wright, A.Y. Yang, A. Ganesh, S.S. Sastry, Y. Ma, Robust face recognition via sparse representation, IEEE Trans. Pattern Anal. Mach. Intell. 31 (2009) 210–227.
[52] C.-G. Li, J. Guo, H.-G. Zhang, Local sparse representation based classification, in: Proceedings of the 20th International Conference on Pattern Recognition, IEEE, 2010, pp. 649–652.
[53] S. Gao, I. W.-H. Tsang, L.-T. Chia, Kernel sparse representation for image classification and face recognition, in: Proceedings of the 11th European Conference on Computer Vision, Part IV, 2010, pp. 1–14.
[54] E. Elhamifar, R. Vidal, Sparse subspace clustering, in: Proceedings of the 2009 IEEE Conference on Computer Vision and Pattern Recognition, IEEE, 2009, pp. 2790–2797.
[55] J. Jiao, X. Mo, C. Shen, Image clustering via sparse representation, in: Advances in Multimedia Modeling, Springer, 2010, pp. 761–766.
[56] Y. Gao, A. Choudhary, G. Hua, A nonnegative sparsity induced similarity measure with application to cluster analysis of spam images, in: Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing, IEEE, 2010, pp. 5594–5597.
[57] S. Yan, H. Wang, Semi-supervised learning by sparse representation, in: Proceedings of the SIAM International Conference on Data Mining, SIAM, 2009, pp. 792–801.
[58] H. Cheng, Z. Liu, J. Yang, Sparsity induced similarity measure for label propagation, in: Proceedings of IEEE 12th International Conference on Computer Vision, IEEE, 2009, pp. 317–324.
[59] K. Bache, M. Lichman, UCI machine learning repository, ⟨http://archive.ics.uci.edu/ml⟩, 2013.
[60] A. Georghiades, Yale face, ⟨http://cvc.yale.edu/projects/yalefaces/yalefaces.html⟩, 2013.
[61] A. Georghiades, P. Belhumeur, D. Kriegman, Yale face_b, ⟨http://cvc.yale.edu/projects/yalefacesB/yalefacesB.html⟩, 2013.
[62] ORL face, AT&T Lab Cambridge, ⟨http://www.face-rec.org/databases/⟩, 2013.
[63] K. Koh, S.-J. Kim, S. Boyd, `l1_ls`: Simple MATLAB solver for l_1-regularized least squares problems, Version Beta, ⟨http://www.stanford.edu/~boyd/l1_ls/⟩, 2008 (accessed on 10.11.2012).
[64] J.A. Hartigan, M.A. Wong, Algorithm as 136: a k-means clustering algorithm, J. R. Stat. Soc.: Ser. C (Appl. Stat.) 28 (1979) 100–108.
[65] Z.J. Huang, Extensions to the k-means algorithm for clustering large data sets with categorical values, Data Min. Knowl. Discov. 2 (1998) 283–304.
[66] L. Jing, M.K. Ng, Z.J. Huang, An entropy weighting k-means algorithm for subspace clustering of high-dimensional sparse data, IEEE Trans. Knowl. Data Eng. 19 (2007) 1026–1041.
[67] Z. Deng, K.-S. Choi, F.-L. Chung, S. Wang, EEW-SC: Enhanced entropy-weighting subspace clustering for high dimensional gene expression data clustering analysis, Appl. Soft Comput. 11 (2011) 4798–4806.

Sen Wu received the Ph.D. degree in control theory and control engineering from the University of Science and Technology Beijing, Beijing, China, in 2002. She is currently a Professor with the Department of Management Science and Engineering, Dongling School of Economics and Management, University of Science and Technology, Beijing. Her general areas of research are data mining, complex networks, and personalized recommendation, with a special interest in solving the problems rose from the emerging data-intensive applications. She is currently the Principal Investigator of a NSFC project and a MOE project. She has published four books and over 70 papers in refereed conference proceedings and journals. She has also been a reviewer for NSFC proposals, leading academic journals, and many international conferences in her area. She is a Board Member of the Asian Association of Management Science and Applications. She was a recipient of "I love my teacher—the most excellent teacher in my heart" Honor at the University of Science and Technology, Beijing.

Xiaodong Feng is a Ph.D. student in the Dongling School of Economics and Management of University of Science and Technology Beijing, China. He received his Master's degree in Management in 2011, also from University of Science and Technology Beijing. His research interest is in the area of data mining, currently concentrating on high-dimensional data clustering.



Wenjun Zhou received the Ph.D. degree from Rutgers, the State University of New Jersey, the M.S. degree from the University of Michigan-Ann Arbor, and the B.S. degree from the University of Science and Technology of China (USTC). She is currently an Assistant Professor in the Department of Statistics, Operations, and Management Science at the University of Tennessee Knoxville (UTK), and a faculty affiliate with The Center for Intelligent Systems and Machine Learning (CISML) at UTK. Her general research areas are data mining and statistical computing. She has published in refereed journals and conference proceedings, such as Machine Learning, INFORMS Journal on Computing (JOC), ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD), and IEEE International Conference on Data Mining (ICDM). She has received the ACM SIGKDD 2008 Best Student Paper Runner-Up Award, and was among the top five finalists for ACM SIGKDD Doctoral Dissertation Award in 2011. She is a member of ACM, IEEE, and INFORMS.