Information Sciences 501 (2019) 708–723

Contents lists available at ScienceDirect

Information Sciences

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

Robust ℓ_2–Hypergraph and its applications

Taisong Jin[a], Zhengtao Yu[b,*], Yue Gao[c], Shengxiang Gao[b], Xiaoshuai Sun[a], Cuihua Li[a]

[a] Fujian Key Laboratory of Sensing and Computing for Smart City, School of Information Science and Engineering, Xiamen University, Xiamen 361005, China
[b] School of Information Engineering and Automation, Kunming University of Science and Technology, Kunming 650500, China
[c] School of Software, Tsinghua University, Beijing 100084, China

ARTICLE INFO

Article history:
Received 26 May 2018
Revised 12 January 2019
Accepted 3 March 2019
Available online 13 March 2019

Keywords:
Hypergraph
Hyperedge
Representation coefficients
Ridge regression

ABSTRACT

Hypergraph, an important learning tool to modulate high-order data correlations, has a wide range of applications in machine learning and computer vision. The key issue of the hypergraph-based applications is to construct an informative hypergraph, in which the hyperedges effectively represent the high-order data correlations. In practice, the real-world data is usually sampled from a union of non-linear manifolds. Due to the issues of noise and data corruptions, many data samples deviate from the underlying data manifolds. To construct an informative hypergraph that represents real-world data distribution well, we propose a hypergraph model (ℓ_2-Hypergraph). Our model generates each hyperedge by solving an affine subspace ridge regression problem, where the samples with non-zero representation coefficients are used for hyperege generation. Specifically, to be robust to sparse noise and corruptions, a sparse constraint is imposed on data errors. We have conducted image clustering and classification experiments on real-world datasets. The experimental results demonstrate that our hypergraph model is superior to the existing hypergraph construction methods in both accuracy and robustness to sparse noise.

1. Introduction

Hypergraph learning has received considerable attention in computer vision and pattern recognition [1,46]. Different from a simple graph, each hyperedge in a hypergraph can link more than two vertices. Thus, the high-order relationships of the data are effectively modulated. The hypergraph-based learning methods have been applied to various learning problems [14,21,22,32,41,42].

A key issue in hypergraph learning is to construct an informative hypergraph that effectively modulates the data correlations. In the past decade, most hypergraph learning methods [1,14,21,22,32,41,42,46] adopt the neighborhood-based strategy to construct a hypergraph. In detail, to generate a hyperedge, this strategy takes each data sample as centroid vertex and links it to its K-Nearest-Neighbors (KNN). The neighborhood-based strategy has the following two main drawbacks that remain unsolved: (1) Learning performance is sensitive to the neighborhood size in the k-nearest neighbor selection. A small neighborhood size separates data samples from the same cluster; on the contrary, a large neighborhood size combines da

* Corresponding author.
E-mail address: ztyu@hotmail.com (Z. Yu).

https://doi.org/10.1016/j.ins.2019.03.012
0020-0255/

samples from different clusters [14]. (2) Various types of noises may easily contaminate the real-world data, which degrades the hypergraph learning performance dramatically.

Sparse representation has been recently used for hypergraph learning, where a hyperedge set is generated by solving a sparse representation problem [24,35,44]. For instance, Wang et al. [35] proposed ℓ_1-Hypergraph, where each sample is sparsely reconstructed by its K-nearest-neighbors and the samples with non-zero sparse codes are used to generate the related hyperedges. Zhang et al. [44] further extended ℓ_1-Hypergraph by removing the redundant hyperedges according to the feature learning task. Liu et al. [24] proposed Elastic-net Hypergraph by solving a linear elastic-net problem, where a quadratic component is added to the ℓ_1-norm sparse regularization. Because sparse representation separates the noise components from the original data, sparse representation-based strategy achieves promising learning performance, specifically, on noisy datasets.

Although the sparse representation-based strategy achieves the state-of-the-art learning performance in noisy data, such strategy still has the following two drawbacks, which limit its applications for real-world data:(1) The existing methods are robust to Gaussian noise and sample-specific corruption, the common sparse noise or the mixed noise is ignored. For instance, ℓ_1-Hypergraph adopts the ℓ_2-norm based metric to measure the reconstruction errors, which is only insensitive to Gaussian noise; Elastic-net Hypergraph adopts the ℓ_{21}-norm based metric to measure the reconstruction errors, which is only insensitive to sample-specific corruptions and outliers. (2) Sparse representation is a linear learning model. Thus, the existing sparse representation-based methods are not suited to the data sampled from a union of dependent non-linear manifolds effectively.

For many real-world problems, the sampled high-dimensional data, instead of being distributed uniformly in the ambient space, lie on or close to a union of low-dimensional manifolds. Besides, the sampled data is often contaminated by sparse noise and corruptions, causing the sampled data to deviate from the underlying manifolds. Therefore, it is crucial to develop a novel hypergraph model, which is not only robust to sparse noise or corruptions in real application, but also modulates the underlying manifold structures of the data.

Inspired by the recent advances of hypergraph learning, we propose a hypergraph model (ℓ_2-Hypergraph), which lies in the scope of the representation-based strategy.

The recent proposed regression-based hypergraph and its two instances (ℓ_1-H and ℓ_2-H) [11] are closely relevant to our hypergraph model. However, the regression-based hypergraph has the following two key differences from our model. (1) The regression-based hypergraph adopts the ℓ_2-norm based metric to measure the reconstruction errors, which is still sensitive to sparse noise. (2) The regression-based hypergraph generates a set of hyperedges by solving a linear regression problem, which isn't suitable to cope with the non-linear data in real applications. Our hypergraph model separates the sparse noise component from the original data to make the learning insensitive to sparse noise and corruption, which addresses the first drawback of the regression-based hypergraph. Besides, our hypergraph model incorporates the locality preserving constraint into the linear regression framework, which is used to preserve the local manifold structures of the data. Thus, the second drawback of the regression-based hypergraph is addressed.

The contributions of our work are as follows:

1. Our hypergraph model adopts affine subspace ridge regression, instead of sparse representation, to generate a set of hyperedges. Specifically, a locality preserving constraint is imposed on the representation coefficients to effectively modulate the high-order correlations of the non-linear data.
2. To be robust to sparse noise and corruptions, the components of sparse noise are separated for data reconstruction, where a sparse constraint is imposed on the separated noise component. Then, the derived data representation is used for generating a set of the noise-resistant hyperedges.

Fig. 1 is the flowchart of our proposed hypergraph model. We adopt our model to the image clustering and classification experiments on the real-word datasets. The experimental results demonstrate that our model achieves promising learning performance.

The rest of this article is arranged as follows. Section 2 introduces the related work. Section 3 presents the proposed hypergraph model. Section 4 gives the optimization approach. Section 5 reports the experimental settings, the results and the discussions. Finally, we concludes the article in Section 6.

2. Related work

As an extension of a simple graph, each hyperedge in a hypergraph links more than two data samples that model high-order relationships of the data. Hypergraph $HG = \{V, E, \mathbf{W}\}$ is composed of a vertex set V, a hyperedge set E, and a weight matrix of hyperedges $\mathbf{W}$. The weight of a hyperedge e is denoted as $w(e)$, and the incidence matrix is denoted as $\mathbf{H}$, which indicates whether a vertex belongs to a hyperedge. Based on the incidence matrix and the weights of hyperedges, the vertex degree of each vertex is denoted as $d(v) = \sum_{e\in E} w(e)\mathbf{H}(v,e)$; the edge degree of a hyperedge, e, is denoted as $\delta(e) = \sum_{v\in V}\mathbf{H}(v,e)$. Let $\mathbf{D}_v$, $\mathbf{D}_e$ and $\mathbf{W}_e$ denote the diagonal matrices of vertex degrees, hyperedge degrees, and the edge weights, respectively. The normalized hypergraph Laplacian can be written as $\mathbf{L} = \mathbf{I} - \Theta$, where $\Theta = \mathbf{D}_v^{-\frac{1}{2}}\mathbf{H}\mathbf{W}_e\mathbf{D}_e^{-1}\mathbf{H}^T\mathbf{D}_v^{-\frac{1}{2}}$.

Hypergraph has been applied to various classification, clustering, retrieving, and embedding tasks.

For the classification applications, Yu et al. [41] proposed a hypergraph-based semi-supervised image classification method, where the hyperedge weights are automatically learned. Sun et al. [32] exploited a hypergraph to model high-order

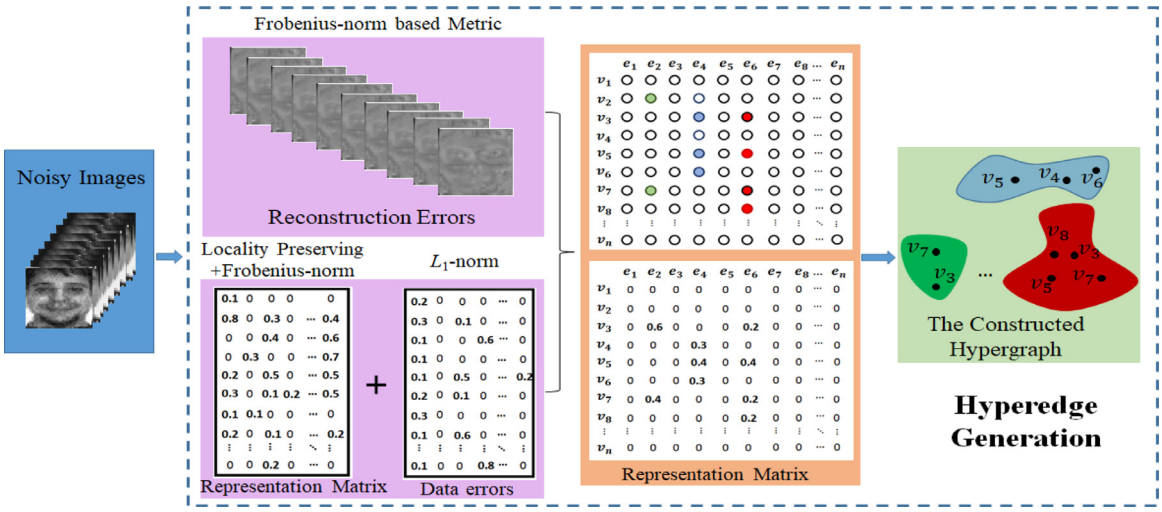

Fig. 1. The flowchart of our proposed hypergraph model.

information among different labels of multi-label classification. Wang et al. [36] proposed a hypergraph-based multi-label learning method, where the label relations are captured by the normalized hypergraph Laplacian. Gao et al. [8] proposed a hypergraph-based hyperspectral image classification method, where both simple graph and hypergraph are combined to represent the data correlations for hyperspectral image-based applications.

For the clustering applications, Docournau et al. [3] proposed a formulation of a random walk in a directed hypergraph to solve the semi-supervised image segmentation problem. Kim et al. [18] proposed a hypergraph-based correlation clustering framework for image segmentation, where a hypergraph is used to improve the basic correlation clustering formulation. Huang et al. [12] proposed hypergraph-based unsupervised image categorization, where image categorization is formulated as a problem of hypergraph partition. Huang et al. [13] applied a hypergraph to solve the video object segmentation problem.

For the retrieval applications, Huang et al. [14] proposed a hypergraph-based image retrieval method, which assigned each vertex to the hyperedge in a probabilistic way.

For the embedding applications, Gao et al. [7] incorporated hypergraph learning into the sparse coding framework, allowing sparse codes to distribute over a hypergraph manifold. Hong et al. [9] proposed a multi-view hypergraph-based dimensionality reduction method, where multi-view data is exploited to enhance learning performance. Tian et al. [33] proposed a hypergraph-based semi-supervised learning method to classify gene expression and genomic hybridization data. Yu et al. [40] proposed a multi-view sparse coding method for image click prediction, where a hypergraph is used to exploit the complementary information of different view features. For the other applications, Zass et al. [42] modeled the feature set as a hypergraph, which formulates a graph matching of the feature sets as a hypergraph matching problem. Fang et al. [6] determined the influence estimation using a hypergraph to find topical influential users and images. Jin et al. [16] proposed a hypergraph regularized low-rank matrix factorization method, which shows promising clustering results on real-world image datasets. Jin et al. [17] proposed a multi-hypergraph regularized sparse coding method, which effectively discovers the underlying data manifold.

To illustrate the drawbacks of the existing methods, we give the illustrative examples (See Fig. 2).

As shown in Fig. 2(a), the data points sampled from two non-linear manifolds (ψ_1 and ψ_2) lie within the circle (defined by a dotted line) with center at e_0. The data points a, b, $c \in \psi_1$ and e_0, f_0, $g \in \psi_2$, where the sampled points a, b, and c are nearer to e_0 than d_0, f_0, g. Note that because of the effects of noise, data point c sampled from ψ_1 is near to ψ_2. The neighborhood-based strategy selects a, b and c to link e_0, leading to an un-informative hyperedge.

Sparse representation-based strategy cannot connect related data in one hyperedge as completely as possible. It is indicated by one example in Fig. 2(b), where data point e_0 is the to-be-reconstructed data, and the calculated coefficients for d_0, f_0 and g are 0.3, 0.4 and 0, respectively. Data points d_0 and f_0 are selected for hyperedge generation; whereas, data point g, which has the same feature as e_0, is discarded in this example. On the other hand, sparse representation is a linear learning model, which cannot guarantee good learning performance on the non-linear manifold data.

3. The proposed hypergraph model

In this section, we propose a novel hypergraph model to make an attempt robustly represent the high-order correlations of the non-linear data.

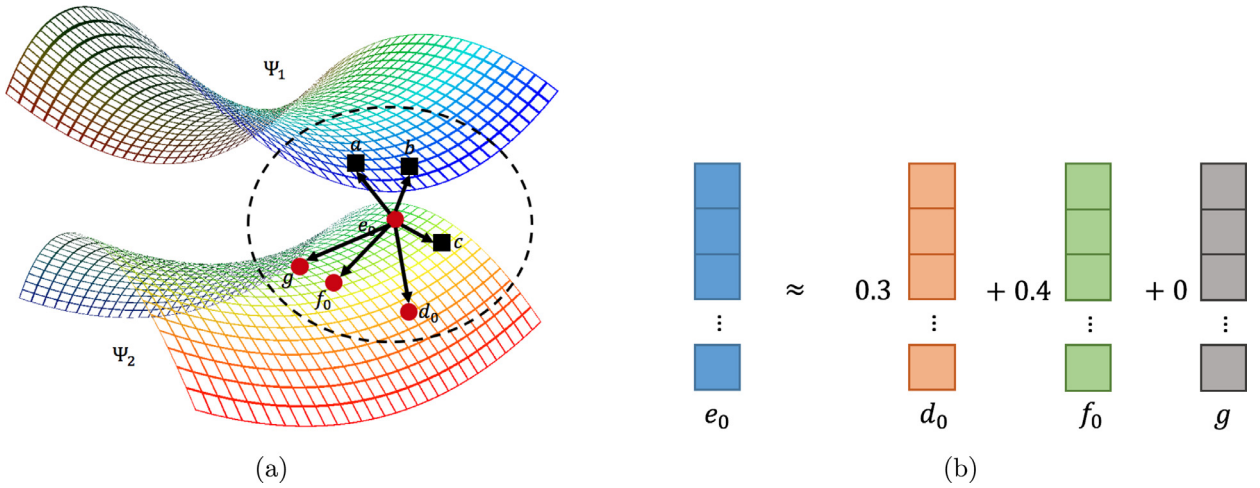

Fig. 2. The examples to illustrate the drawbacks of the existing hypergraph construction methods. (a) The data points sampled from two close manifolds. (b) Sparse representation of a data point.

Table 1
List of important notations.

	Notation and description
$\mathbf{X}$	The feature matrix of data samples
$\mathbf{x}_i$	The ith data sample
$\mathbf{Q}$	The locality adapter matrix
$\mathbf{q}_i$	The locality adapter vector of the ith sample
$\mathbf{C}$	The coefficients matrix of data samples
$\mathbf{c}_i$	The coefficients vector of the ith sample
$\mathbf{E}$	The data errors matrix
$\mathbf{e}_i$	Data errors vector of the ith sample
$\mathbf{W}$	The weight matrix of hyperedges
$\mathbf{L}$	The hypergraph Laplacian
e_i	Hyperedge corresponding to $\mathbf{x}_i$
v_i	Centroid vertex
$\mathbf{H}$	The incidence matrix
$\mathbf{H}_{ij}$	The entry indicating if the ith vertex belongs to the jth hyperedge

3.1. Data representation via ridge regression

Suppose there is a collection of N data samples drawn from a union of nonlinear manifolds in close proximity. For convenience, a collection of data samples are represented as $\mathbf{X} = [\mathbf{x_1}, \ldots, \mathbf{x}_i, \ldots, \mathbf{x}_N] \in \mathbb{R}^{M\times N}$, where $\mathbf{x}_i \in \mathbb{R}^M$ is the ith sample. Various linear regression methods [38,39] are proposed to data reconstruction. Our proposed hypergraph model links highly correlated data samples close to the same nonlinear manifold for hyperedge generation. (For important notations, see Table 1)

We try to reconstruct the data by itself, which is formulated as

$$\begin{aligned} &\min_{\mathbf{C},\mathbf{E}} \|\mathbf{X} - \mathbf{XC} - \mathbf{E}\|_F^2 + \frac{\lambda_1}{2}\|\mathbf{C}\|_F^2 + \frac{\lambda_2}{2}\|\mathbf{Q}\odot\mathbf{C}\|_F^2 + \mu\|\mathbf{E}\|_1 \\ &\quad s.t.\ \mathbf{C}^T\mathbf{1} = \mathbf{1}, \mathbf{diag}(\mathbf{C}) = \mathbf{0}, \end{aligned} \tag{1}$$

where $\mathbf{C} = [\mathbf{c}_1\mathbf{c}_2\ldots\mathbf{c}_N] \in \mathbb{R}^{N\times N}$ is the coefficients matrix of data samples, and $\mathbf{c}_i$ is the coefficient vector of the ith sample; $\mathbf{E} = [\mathbf{e}_1\mathbf{e}_2\ldots\mathbf{e}_N] \in \mathbb{R}^{M\times N}$ is the data errors matrix and $\mathbf{e}_i \in \mathbb{R}^M$ is the data errors vector of the ith sample; $\mathbf{Q} = [\mathbf{q}_1\mathbf{q}_2\ldots\mathbf{q}_N] \in \mathbb{R}^{N\times N}$ is the locality adapter matrix, aiming to preserve the local manifold structures; $\mathbf{q}_i \in \mathbb{R}^N$ is the locality adapter vector of the ith sample; $\odot$ denotes element-wise multiplication, and $\lambda_1 > 0$, $\lambda_2 > 0$ and $\mu > 0$ are the trade-off parameters.

The objective function contains four terms:

(1) **Data reconstruction term**. The term separates the sparse noise component and measures the reconstruction errors using the *Frobenius*-norm based metric. Thus, the data representation is robust to not only sparse noise, but also the mixed noise of Gaussian noise plus sparse corruption.

(2) **Frobenius-norm regularization term of the Coefficients matrix**. *Frobenius*-norm of the coefficients matrix makes a group of correlated data samples approximately equal, which tends to choose the high-correlated samples together for hyperedge generation [25].

(3) **Locality preserving term of the Coefficients matrix**. Because the local manifold structure of the data is considered as piece-wise affine subspace [4], the weighted least squares is taken as the locality preserving term [34], where the weight vector, $\mathbf{q}_i = [\mathbf{q}_{i1}\mathbf{q}_{i2}\dots\mathbf{q}_{iN}]^T \in \mathbb{R}^N$, is the locality adapter vector of the ith sample. Specifically, the jth entry ($j = 1, 2, \dots, N$) is the affinity between $\mathbf{x}_i$ and the jth sample [4,34]:

$$\mathbf{q}_{ij} = \begin{cases} \frac{\exp(\|\mathbf{x}_i - \mathbf{x}_j\|_2)}{\sum_{t\neq i}\exp(\|\mathbf{x}_t - \mathbf{x}_j\|_2)} & \text{if } i \neq j \\ 0 & \text{if } i = j \end{cases} \tag{2}$$

(4) **Sparse regularization term of the data errors matrix**. This is used to handle the sparse components of the data. By minimizing the sparse noise and corruptions, sparse noise component of the data is effectively modulated.

In addition, two constraints are imposed on the coefficients matrix, which simultaneously ensure that each sample is reconstructed as an affine combination of the remaining samples.

(1) **The affine constraint $\mathbf{C}^T\mathbf{1} = \mathbf{1}$**, which ensures that the sum of representation coefficients of each sample is equal to 1.

(2) **The constraint diag($\mathbf{C}$) = $\mathbf{0}$**, which are used to eliminate the trivial solution of reconstructing each sample as itself.

The coefficients matrix characterizes how data samples contribute to the data reconstruction, which is useful to discover the underlying structures of the data. For our model, (1) the modeling of data errors makes the coefficient derivation procedure robust to sparse noise or mixed noise of sparse corruption plus Gaussian noise. (2) The locality preserving term makes the derived representation coefficients of each sample respect the local manifold structures. (3) The *Frobenius*-norm regularization term of the coefficients matrix ensures that representation coefficients of high-correlated samples are similar.

It is worth emphasizing that Sparse Subspace Clustering (SSC) in [5] is also robust to sparse noise and corruptions. However, (1) SSC is a simple graph model, which can modulate only the pairwise relationships between two samples; whereas our model is a hypergraph model, which can modulate the high-order data correlations. (2) SSC considers each variable independently, which cannot guarantee the group effects of connecting as complete as possible related data; whereas our model uses the *Frobenius*-norm, which is suitable to discover the grouping structures of the data. (3) SSC is a linear learning model, which isn't suitable to handle the non-linear manifold data; our model uses the locality preserving term to discover the underlying local manifold structures.

3.2. Incidence relationship definition of a hyperedge

The incidence relationship between a hyperedge and its vertices is crucial for hypergraph learning. The traditional methods usually adopt a 0-1 strategy to define the incidence matrix of hyperedges [1,46], i.e., if the vertex belongs to a hyperedge, the entry of incidence matrix is assigned to 1; otherwise it is set to 0. Thus, all the vertices within a hyperedge are treated equally, which ignores the relative affinity relationship among the vertices.

To modulate the affinity relationship among the vertices within a hyperedge, the probability hypergraph [14] defines the incidence matrix in a probabilistic way for image ranking; ℓ_1-Hypergraph [35] defines the incidence relationship between a hyperedge and its vertices according to the representation coefficients of a sparse representation problem, showing the impressive image classification results. We adopt two different strategies to define the incidence relation between a hyperedge and its vertices.

• **(0-1) Incidence Relationship**

(0-1) strategy is commonly used incidence relation definition, which has been used for many applications. Considering that representation coefficients are often dense, we retain only those coefficients over a given threshold. Then, the incidence relation between a hyperedge and its vertices is defined as [24]

$$h(v_j, e_i) = \begin{cases} 1, & \text{if } |c_{ij}| \geqslant \theta_1 \\ 0, & \text{otherwise} \end{cases}, \tag{3}$$

where e_i is a hyperedge associated with data sample $\mathbf{x}_i$, v_j represents the jth data sample, c_{ij} is the jth representation coefficient of $\mathbf{x}_i$, and θ_1 is a given threshold parameter. As indicated in Eq. (3), a sample $\mathbf{x}_j$ is assigned to e_i based on whether the coefficient c_{ij} of the sample is larger than the threshold θ_1.

• **Probability Incidence Relationship**

Inspired by the probability hypergraph [14] and ℓ_1-Hypergraph [35], we define the incidence relation between a hyperedge and its vertices using representation coefficients. The incidence matrix of the hyperedges is defined as

$$h(v_j, e_i) = \begin{cases} |c_{ij}|, & \text{if } |c_{ij}| \geqslant \theta_2 \\ 0, & \text{otherwise} \end{cases}, \tag{4}$$

where e_i is the hyperedge associated with the centroid vertex $\mathbf{x}_i$, c_{ij} is the jth representation coefficient of $\mathbf{x}_i$ and θ_2 is a given threshold parameter. Thus, not only the incidence relationship but also the affinity among the vertices within the hyperedge is effectively modulated based on the representation coefficients.

3.3. Hyperedge weighting

Hyperedge weighting schemes, most of which are associated with a specific task, are crucial for hypergraph-based applications. For instance, the hyperedge weight is simply set equal to 1 [46] or, for image ranking, is calculated by summing up the pairwise affinities of vertices within the hyperedge [14]. In other works, the hyperedge weight is automatically learned for image classification [41].

To handle the noisy data, it is not suitable to directly measure the similarity between two samples. Thus, we take coefficients vector of each sample as the feature, and measure their similarity via the dot product of these two feature vectors [24]. In other words, the similarity between two samples is defined as

$$S(v_i, v_j) = \left|\left\langle \mathbf{c}_i, \mathbf{c}_j \right\rangle\right|. \tag{5}$$

Finally, we sum up all the similarities of the vertices within a hyperedge as the weight of hyperedge e_i is calculated as [14]

$$w(e_i) = \sum_{v_j \in e_i, j \neq i} S(v_i, v_j). \tag{6}$$

Our proposed hypergraph model has the following two key advantages: (1) Each hyperedge links only the vertices lying on or close to the same non-linear manifold. (2) Our hyperedge generation procedure is robust to sparse noise or mixed noise of Gaussian noise plus sparse corruption. Since the constructed hypergraph is based on ridge regression, we refer to our model as ℓ_2-Hypergraph.

3.4. The proposed hypergraph construction algorithm

The main procedure of our hypergraph model is provided in Algorithm 1.

Algorithm 1: Robust ℓ_2-Hypergraph learning.

Input: Data matrix $\mathbf{X}$
Output: ℓ_2-Hypergraph
1 Reconstruct $\mathbf{X}$ according to Eq. (1);
2 **for** $i = 1, \ldots, N$ **do**
  (a) Define the incidence vector between hyperedge, e_i and its vertices according to Eq. (3) or Eq. (6) ;
  (b) Compute the weight of hyperedge, e_i according to Eq. (6);
end

As illustrated in Algorithm 1, our hypergraph model adopts two approaches to define the incidence relationship between a hyperedge and its vertices, resulting in two different hypergraph construction methods. We refer to the method according to Eq. (3) as ℓ_2-HG1, and the one according to Eq. (4) as ℓ_2-HG2.

4. Optimization

In this section, we present an optimization approach for optimizing Eq. (1).

4.1. The ADM-based optimization

Many optimization methods, such as Accelerated Proximal Gradient (APG) [27] and Alternating Direction Multiplier (ADM) [43], have been proposed. Among them, the ADM approach has the advantage of being suitable to handle the large datasets. Thus, we adopt the ADM approach to optimize the objective function of our model.

By introducing an auxiliary matrix, $\mathbf{P} \in \mathbb{R}^{N \times N}$, we rewrite Eq. (1) as

$$\begin{aligned} &\min_{\mathbf{P},\mathbf{C},\mathbf{E}} \|\mathbf{X} - \mathbf{X}\mathbf{P} - \mathbf{E}\|_F^2 + \tfrac{\lambda_1}{2}\|\mathbf{C}\|_F^2 + \tfrac{\lambda_2}{2}\|\mathbf{Q} \odot \mathbf{C}\|_F^2 + \mu\|\mathbf{E}\|_1 \\ &s.t. \quad \mathbf{P}^T\mathbf{1} = \mathbf{1}, \mathbf{P} = \mathbf{C} - \mathrm{diag}(\mathbf{C}), \end{aligned} \tag{7}$$

To make the objective function strictly convex with respect to the optimization variables $(\mathbf{P}, \mathbf{C}, \mathbf{E})$, two penalty terms of the constraints are incorporated into the objective function of Eq. (7), resulting in

$$\begin{aligned} \min_{\mathbf{P},\mathbf{C},\mathbf{E}} &\|\mathbf{X} - \mathbf{X}\mathbf{P} - \mathbf{E}\|_F^2 + \tfrac{\lambda_1}{2}\|\mathbf{C}\|_F^2 + \tfrac{\lambda_2}{2}\|\mathbf{Q} \odot \mathbf{C}\|_F^2 + \mu\|\mathbf{E}\|_1 \\ &+ \tfrac{\rho}{2}\left\|\mathbf{P}^T\mathbf{1} - \mathbf{1}\right\|_2^2 + \tfrac{\rho}{2}\|\mathbf{P} - (\mathbf{C} - \mathrm{diag}(\mathbf{C}))\|_F^2 \\ &s.t. \quad \mathbf{P}^T\mathbf{1} = \mathbf{1}, \mathbf{P} = \mathbf{C} - \mathrm{diag}(\mathbf{C}) \end{aligned} \tag{8}$$

where $\rho > 0$ is the augmented Lagrange parameter.

By adding two the Lagrange multipliers ($\mathbf{M_1}$,$\mathbf{M_2}$) for two equality constraints in Eq. (8), the Lagrange function of Eq. (8) is formulated as

$$\begin{aligned} L(\mathbf{P},\mathbf{C},\mathbf{E},\mathbf{M_1},\mathbf{M_2}) &= \|\mathbf{X}-\mathbf{XP}-\mathbf{E}\|_F^2+\tfrac{\lambda_1}{2}\|\mathbf{C}\|_F^2+\tfrac{\lambda_2}{2}\|\mathbf{Q}\odot\mathbf{C}\|_F^2 \\ &+\mu\|\mathbf{E}\|_1+\tfrac{\rho}{2}\left\|\mathbf{P}^T\mathbf{1}-\mathbf{1}\right\|_2^2+\tfrac{\rho}{2}\|\mathbf{P}-(\mathbf{C}-\mathrm{diag}(\mathbf{C}))\|_F^2 \\ &+tr(\mathbf{M}_2^T(\mathbf{P}-(\mathbf{C}-\mathrm{diag}(\mathbf{C}))+\mathbf{M}_1^T(\mathbf{P}^T\mathbf{1}-\mathbf{1}), \end{aligned} \tag{9}$$

where $tr(\cdot)$ is the trace operator of a given matrix.

Eq. (9) is a typical ADM formulation, which can be solved by two separate steps: primal variable updating and dual ascending, i.e., the ADM approach iteratively updates the primal variables ($\mathbf{P}$, $\mathbf{C}$, $\mathbf{E}$) and the Lagrange multipliers ($\mathbf{M_1}$,$\mathbf{M_2}$) to obtain the optimal solution. For convenience, we denote the optimization variables at iteration k as ($\mathbf{P}^{(k)}$, $\mathbf{C}^{(k)}$, $\mathbf{E}^{(k)}$) and denote Lagrange multipliers at iteration k as ($\mathbf{M}_1^{(k)}$,$\mathbf{M}_2^{(k)}$). These variables are updated as follows.

- ($\mathbf{P}-$**Update**): Updating $\mathbf{P}$ relies on the following problem by fixing the other variables and removing irrelevant terms:

$$\begin{aligned} \mathbf{P}^{(k+1)} &= \arg\min_{\mathbf{P}} \left\|\mathbf{X}-\mathbf{XP}-\mathbf{E}^{(k)}\right\|_F^2+\tfrac{\rho}{2}\left\|\mathbf{P}^T\mathbf{1}-\mathbf{1}\right\|_2^2 \\ &+tr\left(\mathbf{M}_2^{(k)T}\left(\mathbf{P}-\left(\mathbf{C}^{(k)}-\mathrm{diag}\left(\mathbf{C}^{(k)}\right)\right)\right)\right) \\ &+\tfrac{\rho}{2}\left\|\mathbf{P}-\left(\mathbf{C}^{(k)}-\mathrm{diag}\left(\mathbf{C}^{(k)}\right)\right)\right\|_F^2 \\ &+\mathbf{M}_1^{(k)T}\left(\mathbf{P}^T\mathbf{1}-\mathbf{1}\right). \end{aligned} \tag{10}$$

Differentiating the objective function with respect to $\mathbf{P}$ and setting it to zero, we obtain

$$\left(\mathbf{X}^T\mathbf{X}+\rho\mathbf{I}+\rho\mathbf{1}\mathbf{1}^T\right)\mathbf{P}^{(k+1)}=\mathbf{X}^T\left(\mathbf{X}-\mathbf{E}^{(k)}\right)+\rho\left(\mathbf{1}\mathbf{1}^T+\mathbf{C}^{(k)}\right)-\mathbf{1}\mathbf{M}_1^{(k)T}-\mathbf{M}_2^{(k)}. \tag{11}$$

- ($\mathbf{C}-$**Update**): Updating $\mathbf{C}$ relies on the following problem by fixing the other variables and removing irrelevant terms:

$$\begin{aligned} \mathbf{C}^* &= \arg\min_{\mathbf{C}} \tfrac{\lambda_1}{2}\|\mathbf{C}\|_F^2+\tfrac{\lambda_2}{2}\|\mathbf{Q}\odot\mathbf{C}\|_F^2 \\ &+\tfrac{\rho}{2}\left\|\mathbf{P}^{(k+1)}-(\mathbf{C}-\mathrm{diag}(\mathbf{C}))\right\|_F^2 \\ &+tr(\mathbf{M}_2^{(k)T}(\mathbf{P}^{(k+1)}-(\mathbf{C}-\mathrm{diag}(\mathbf{C})))) \end{aligned} \tag{12}$$

Likewise, we obtain

$$\left((\rho+\lambda_1)\mathbf{I}+\lambda_2\mathbf{diag}\left(\mathbf{Q}^2\right)\right)\mathbf{C}^*=\rho\mathbf{P}^{(k+1)}+\mathbf{M}_2^{(k)}. \tag{13}$$

Eqs. (11) and (13) are the $N\times N$ systems of the linear equations; $\mathbf{P}$ and $\mathbf{C}$ are updated by solving the problem of $N\times N$ system of linear equations. After obtaining the optimal solution, $\mathbf{C}^*$, of Eq. (12), $\mathbf{C}^{(k+1)}=\mathbf{C}^*-\mathrm{diag}(\mathbf{C}^*)$.

- ($\mathbf{E}-$**Update**): Updating $\mathbf{E}$ relies on the following problem by fixing the other variables and removing irrelevant terms [43] :

$$\mathbf{E}^{(k+1)}=\arg\min_{\mathbf{E}}\left\|\mathbf{X}-\mathbf{XP}^{(k+1)}-\mathbf{E}^{(k)}\right\|_F^2+\mu\left\|\mathbf{E}^{(k)}\right\|_1. \tag{14}$$

This optimization problem has the closed-form solution

$$\mathbf{E}^{(k+1)}=\tau_\mu\left(\mathbf{XP}^{(k+1)}-\mathbf{X}\right), \tag{15}$$

where $\tau_\beta(\cdot)$ is the shrinkage-thresholding operator [30].

- (**Gradient Ascent Update**)

We perform a gradient ascent update with the step size of ρ on the Lagrange multipliers ($\mathbf{M}_1^{(k)},\mathbf{M}_2^{(k)}$) by fixing ($\mathbf{P}^{(k+1)},\mathbf{C}^{(k+1)},\mathbf{E}^{(k+1)}$):

$$\mathbf{M_1^{(k+1)}}=\mathbf{M_1^{(k+1)}}+\rho(\mathbf{P^{(k+1)^T}1}-\mathbf{1}) \tag{16}$$

$$\mathbf{M_2^{(k+1)}}=\mathbf{M_2^{(k)}}+\rho(\mathbf{P^{(k+1)}}-\mathbf{C^{(k+1)}}) \tag{17}$$

These five steps are repeated until convergence condition is satisfied or the maximal number of iterations is reached. In our article, the convergence condition is defined as $\left\|\mathbf{P}^{(k)T}\mathbf{1}-1\right\|_\infty\le\varepsilon$, $\left\|\mathbf{P}^{(k)}-\mathbf{C}^{(k)}\right\|_\infty\le\varepsilon$, $\left\|\mathbf{P}^{(k)}-\mathbf{P}^{(k-1)}\right\|_\infty\le\varepsilon$ and $\left\|\mathbf{E}^{(k)}-\mathbf{E}^{(k-1)}\right\|_\infty\le\varepsilon$, where ε is error tolerance for the primal and dual residuals. For our clustering and classification experiments, when ε is set to 0.001, our proposed two methods achieve promising learning performance.

4.2. Convergence properties

ADM converges under the condition that its objective function contains, at most, two block variables. Since the Lagrange function, contains three blocks (**P**, **C**, **E**), it is difficult to give a strict mathematical proof to prove its convergence. However, recent studies demonstrate that when the optimal gap of each iteration monotonically decreases, ADM has good convergence properties [23]. The convexity of the Lagrange function with respect to the single variable guarantees that ADM has good convergence properties in practice.

4.3. Computational complexity

The computational complexity of the ADM approach scales to $O(TMN^2)$, where N is the number of samples, M is data dimensionality and T is the number of the iterations.

5. Experiments and discussions

5.1. Experimental settings and the compared methods

To evaluate the performance of our proposed hypergraph model, we conducted the image clustering and classification tasks on the real-world datasets. In our experiments, we compared the following graph-based methods:

*K***NN**−**HG** [46]. *K*NN-HG adopts the neighborhood-based approach to construct a hypergraph, where the Gaussian kernel function metric is used to find the neighbors.

ℓ_1−**HG** [35]. ℓ_1-HG adopts sparse representation to construct a hypergraph, where each sample is represented by its k-nearest-neighbors and the ℓ_2-norm based metric is used to measure the reconstruction errors.

EN−**HG** [24]. EN-HG adopts the elastic-net model to construct a hypergraph, where the ℓ_{21}-norm based metric is used to measure the reconstruction errors.

ℓ−**H** [11]. ℓ−**H** is the regression-based hypergraph model, which has two instances: ℓ_1−**H** and ℓ_2−**H**. ℓ_1−**H** adopts the traditional sparse representation to construct a hypergraph, whereas ℓ_2−**H** adopts the ridge regression to construct a hypergraph. Both instances adopt the ℓ_2-norm based metric to measure the reconstruction errors.

SSC [5]. SSC adopts sparse representation to construct a simple graph, where the sparse noise is separated for data reconstruction and the ℓ_2-norm based metric is used to measure the reconstruction errors.

ℓ_2−**HG**. ℓ_2-HG is our model, which has two variants according to the incidence definition of a hyperedge. We refer to two methods as ℓ_2-HG1 (0–1 incidence) and ℓ_2-HG2 (Probability incidence).

Among these methods, *K*NN-HG is constructed by the neighborhood-based approach; whereas, the others are the representation-based methods. The optimal parameter values of all the methods are chosen by the cross-validation tests.

5.2. Experimental results of image clustering

Graph-based clustering, which first computes the eigenvector matrix, consisting of the first K eigenvectors of graph Laplacian or hypergraph Laplacian, is a spectral clustering-based model [28,30]. Furthermore, K-means clustering is employed on the rows of the eigenvector matrix to obtain the final clustering results.

In our clustering experiments, two popular benchmarks – Accuracy of Clustering (AC) and Normalized Mutual Information (NMI) [45] – are used to measure clustering performance. We have independently repeated clustering experiments five times and reported the average clustering results for comparison. For fair comparison, we adopted the same hyperedge weighting scheme as our hypergraph model to define the hyperedge weights of EN-HG and ℓ_1-HG.

For the experiments on the CMU PIE [31] and the Coil20 [26] datasets, all the data points from each dataset were scaled to be unitary in the Euclidean norm. For computational efficiency, Eigen-faces with dimensionality 80 are used for each image on the CMU PIE and the Coil20 datasets.

The CMU PIE face dataset contains 68 subjects. The face images were captured under varying poses, illumination, and expression. We considered a total of 6800 images by randomly selecting 100 images for each subject. All the images were manually aligned and cropped. The cropped images are 32×32 pixels, with 256 gray levels per pixel.

The Coil20 object dataset contains 1440 32×32 gray images of twenty objects. The objects were placed on a motorized turntable against a black background. To vary the pose of the object with respect to a fixed camera, the turntable was rotated 360 degrees. Images of the objects were taken at pose intervals of five-degrees. The objects have a wide variety of complex geometric and reflectance characteristics. We conducted the clustering experiments on corrupted data, where the following three types of data corruption were considered [29]. (1) *Gaussian Noise*. Gaussian noise from normal distribution is added to each image, x; that is, $\widetilde{\mathbf{x}} = \mathbf{x} + \alpha n$, where α is the corruption ratio ranging from 15% to 60% with an interval of 15%, and n is the noise following a standard normal distribution. (2) *Random Pixels Corruption*. The values of a percentage of pixels randomly chosen from each image are replaced with those following a uniform distribution over [0, pmax], where pmax is the current image largest pixel value. (3) *Mixed Noise*. Mixed noise is obtained by adding the same percentages of Gaussian noise and random pixels corruption to each image.

(a) (b) (c)

Fig. 3. The examples of corrupted images on Coil20 dataset. (a) Gaussian noise. (b) Random pixels corruptions. (c) Mixed noise of Gaussian noise plus Random pixels corruption. Compared with Gaussian noise and random pixels corruption, mixed noise is more difficult to handle.

Table 2
Clustering on PIE with Gaussian noise.

Gaussian Noise	AC(%)				NMI(%)			
Corrupted ratio	15%	30%	45%	60%	15%	30%	45%	60%
*K*NN-HG [46]	80.6	77.2	69.5	60.5	84.2	80.3	74.8	65.4
ℓ_1-HG [35]	80.4	78.3	77.6	75.4	85.3	81.9	78.2	76.8
EN-HG [24]	82.1	80.3	76.3	76.3	86.4	82.1	80.1	78.4
SSC [5]	78.3	76.2	75.4	74.3	83.6	80.1	76.3	73.8
ℓ_1-H [11]	81.2	79.8	77.4	75.2	85.6	84.3	79.8	77.7
ℓ_2-H [11]	81.9	80.5	78.1	76.2	86.3	84.9	80.2	78.6
ℓ_2-HG1	83.1	82.8	81.2	80.3	89.3	85.3	83.2	82.7
ℓ_2-HG2	**86.3**	**84.9**	**83.3**	**82.4**	**89.8**	**87.3**	**86.4**	**83.3**

Table 3
Clustering on PIE with corruption.

Random pixels	AC(%)				NMI(%)			
Corrupted ratio	15%	30%	45%	60%	15%	30%	45%	60%
*K*NN-HG [46]	77.3	74.0	63.8	58.3	80.3	77.7	72.8	63.3
ℓ_1-HG [35]	77.9	76.4	76.3	75.2	81.7	80.5	78.4	77.2
EN-HG [24]	78.6	78.6	77.2	76.2	81.9	81.6	80.3	78.9
SSC [5]	76.9	75.3	73.4	72.1	80.7	78.1	77.6	75.3
ℓ_1-H [11]	78.6	76.3	75.4	73.2	81.1	80.6	78.7	76.3
ℓ_2-H [11]	79.2	77.3	75.9	74.1	81.9	80.9	79.1	77.3
ℓ_2-HG1	80.1	**79.7**	78.2	77.3	**84.7**	**83.2**	81.4	79.6
ℓ_2-HG2	**80.3**	79.6	**80.9**	**79.1**	83.6	82.2	**82.9**	**80.3**

Table 4
Clustering on PIE with mixed noise.

Mixed Noise	AC(%)				NMI(%)			
Corrupted ratio	15%	30%	45%	60%	15%	30%	45%	60%
*K*NN-HG [46]	75.9	69.1	59.5	48.1	76.2	71.3	69.3	56.1
ℓ_1-HG [35]	74.1	73.1	70.1	70.6	79.3	75.5	73.2	72.1
EN-HG [24]	75.9	74.5	72.2	71.6	80.1	76.1	76.1	74.9
SSC [5]	72.1	70.6	68.6	66.2	75.6	74.1	72.6	71.3
ℓ_1-H [11]	75.8	74.5	73.1	70.6	78.6	75.8	74.2	73.6
ℓ_2-H [11]	76.4	74.8	74.1	72.4	79.8	76.5	75.1	74.2
ℓ_2-HG1	77.1	**76.9**	**74.3**	72.2	**81.1**	**80.3**	77.2	75.5
ℓ_2-HG2	**78.6**	75.3	73.5	**73.1**	79.1	80.2	**78.4**	**76.5**

For the CMU Pie and Coil20 datasets, we randomly selected 50% of the images from each dataset and added Gaussian noise, random pixels corruption and mixed noise to each image, respectively, resulting in six corrupted datasets. Fig. 3 shows the examples of the corrupted images of the Coil20 dataset. Tables 2–7 list the experimental results and comparisons for the corrupted datasets. From Tables 2–7, we have the following observations:

Table 5
Clustering on Coil20 with Gaussian Noise.

Gaussian Noise	AC(%)				NMI(%)			
Corrupted ratio	15%	30%	45%	60%	15%	30%	45%	60%
*K*NN-HG [46]	74.3	70.4	64.2	57.4	83.3	79.3	72.8	66.4
ℓ_1-HG [35]	75.2	74.4	72.1	70.3	85.1	83.6	81.2	80.7
EN-HG [24]	74.3	73.2	71.2	69.5	84.2	82.3	80.3	79.9
SSC [5]	73.2	71.2	70.3	68.1	83.1	80.4	78.6	76.1
ℓ_1-H [11]	74.9	73.8	70.4	70.1	83.9	82.7	80.8	79.2
ℓ_2-H [11]	75.2	74.6	71.5	69.8	84.9	82.6	81.7	80.9
ℓ_2-HG1	76.7	75.3	73.2	71.4	86.8	84.3	83.2	82.7
ℓ_2-HG2	**78.6**	**78.3**	**77.3**	**73.2**	**88.6**	**85.2**	**85.3**	**83.2**

Table 6
Clustering on Coil20 with corruption.

Random pixels	AC(%)				NMI(%)			
Corrupted ratio	15%	30%	45%	60%	15%	30%	45%	60%
*K*NN-HG [46]	75.3	70.6	62.8	54.6	84.3	75.4	71.2	61.3
ℓ_1-HG [35]	74.1	72.3	71.1	70.6	84.3	82.4	80.6	78.3
EN-HG [24]	73.1	71.5	70.8	68.8	84.3	82.2	80.3	77.1
SSC [5]	70.6	68.5	67.2	64.3	82.1	80.1	78.2	76.1
ℓ_1-H [11]	75.8	72.6	70.9	69.2	83.9	83.0	79.8	78.9
ℓ_2-H [11]	76.2	73.1	71.4	70.3	84.1	83.2	81.9	79.3
ℓ_2-HG1	75.8	**74.3**	**72.3**	69.5	84.5	85.3	83.1	81.2
ℓ_2-HG2	**77.6**	73.2	**72.3**	**71.4**	**86.6**	**84.2**	**83.3**	**82.3**

Table 7
Clustering on Coil20 with mixed noise.

Mixed noise	AC(%)				NMI(%)			
Corrupted ratio	15%	30%	45%	60%	15%	30%	45%	60%
*K*NN-HG [46]	73.3	67.2	59.3	50.1	82.2	77.3	71.1	63.3
ℓ_1-HG [35]	72.1	70.1	67.2	65.3	83.1	80.3	78.3	75.1
EN-HG [24]	71.3	69.1	64.2	63.2	81.3	78.1	76.3	74.2
SSC [5]	71.1	68.4	66.1	61.1	80.2	76.3	74.1	71.2
ℓ_1-H [11]	72.8	70.9	67.2	64.9	82.7	80.8	77.7	76.2
ℓ_2-H [11]	73.3	71.4	68.5	65.6	83.3	81.5	78.5	77.2
ℓ_2-HG1	74.3	72.1	**70.2**	66.3	83.3	83.3	81.2	77.4
ℓ_2-HG2	**75.2**	**73.1**	**70.2**	**67.2**	**84.1**	**84.3**	**82.1**	**79.2**

1. In most cases, the representation-based methods (ℓ_1-HG, EN-HG, SSC, ℓ-H and our methods) obtain the better clustering results than *K*NN-HG. As data corruption ratios increase, the performance gap between the representation-based methods and *K*NN-HG increases, further demonstrating that the representation-based approach is robust to data noise and corruptions.
2. In most cases, ℓ_1-HG is superior to EN-HG on Coil20 dataset. On the contrary, EN-HG outperforms ℓ_1-HG on CMU PIE dataset. ℓ-H achieves promising learning performance on both two datasets. The main reason is attributed that Coil20 dataset has the obvious non-linear manifold structures. ℓ_1-HG reconstructs each sample by its *K*-neareset-neighbors, which can represent the local manifold structures. However, EN-HG and SSC adopt the linear learning model to generate a set of the edges or hyperedges, which isn't suitable to handle the non-linear data. ℓ-H uses linear regression of the entire data to construct a hypergraph, which is robust to noise.
3. Our methods consistently and significantly outperform the compared methods on the sparse noisy datasets and mixed noisy datasets, and obtain the best clustering results. Specifically, our methods achieve promising performance on the Coil20 dataset, which has the obvious manifold structures. The experimental results further demonstrate that our methods are not only robust to noise and random pixel corruption, but also represent the local manifold structures.

5.3. Experimental results of image classification

The graph-based classification model integrates the classification loss function and the graph regularizer to allow class labels to smooth over the graph. One widely used graph-based classification framework [35,41] is defined as

$$\mathbf{F}^* = \arg\min_{\mathbf{F}} \left\{ \sum_{i=1}^{c} \mathbf{F}_i^T \Phi \mathbf{F}_i + \gamma \|\mathbf{F} - \mathbf{Y}\|^2 \right\}, \tag{18}$$

Table 8
Classification on Coil20 with Gaussian noise.

Gaussian Noise	ACC(%)			
Corrupted ratio	15%	30%	45%	60%
*K*NN-HG [46]	92.9	86.6	82.2	78.2
ℓ_1-HG [35]	**95.7**	93.8	92.4	90.7
EN-HG [24]	92.3	90.7	90.4	88.3
SSC [5]	91.7	89.7	88.6	86.4
ℓ_1-H [11]	93.6	92.2	91.3	88.2
ℓ_2-H [11]	94.4	93.1	91.8	89.3
SVM [2]	93.1	91.4	89.8	87.6
CNN [20]	93.6	92.1	90.2	86.3
ℓ_2-HG1	95.3	93.9	92.2	91.4
ℓ_2-HG2	95.6	**94.6**	**93.6**	**92.4**

Table 9
Classification on Coil20 with corruption.

Random Pixels	ACC(%)			
Corrupted ratio	15%	30%	45%	60%
*K*NN-HG [46]	88.6	85.0	81.2	76.3
ℓ_1-HG [35]	90.2	88.2	85.8	84.7
EN-HG [24]	88.7	87.2	86.3	85.4
SSC [5]	86.4	85.3	85.2	84.4
ℓ_1-H [11]	89.6	87.6	84.2	83.9
ℓ_2-H [11]	89.5	87.9	84.6	83.8
SVM [2]	88.6	86.4	82.1	79.9
CNN [20]	87.3	85.6	83.2	81.6
ℓ_2-HG1	91.1	89.9	87.9	86.4
ℓ_2-HG2	**92.6**	**91.6**	**89.3**	**88.4**

where Φ is a graph Laplacian or a hypergraph Laplacian, c is the number of data classes; $\mathbf{F}$ signifies the classification relevant scores; $\mathbf{Y}$ is the initial label matrix, in which $\mathbf{Y}_{ij} = 1$ if the ith data sample is labeled as the jth class; otherwise, it is $\mathbf{Y}_{ij} = 0$ and γ is tradeoff parameter and it is simply set to 1. When obtaining the closed-form solution of $\mathbf{F}$, the label of the ith sample is recognized as $j^* = \arg\max_j \mathbf{F}_{ij}$, $(j = 1, 2, \ldots c)$ (See [41] and [35] for greater detail.).

We conducted the classification experiments on both the Coil20 object dataset and USPS digit dataset [15] and reported the classification results. The classification performance is measured by the classification accuracy for classification (ACC), which is defined as the ratio between the number of the correct classification and the size of test dataset.

The USPS digit dataset contains 9298 16×16 gray images; the digits '0' through '9' scanned from U.S. Postal Service envelopes. Each image contains a single digit, handwritten by different people. The USPS image dataset is commonly applied to a series of data analysis tasks. All the data points for USPS were first normalized within the range, 0–255, and then scaled to be unitary and Eigen-faces with dimensionality 80 are used for each image of the USPS dataset.

To better evaluate the classification performance of our model, we also compared our methods to two important and successful non-graph based classification methods, which includes Support Vector Machine (SVM) [2] and Convolutional Neural Networks (CNN) [20]. The CNN architecture we used is based on LeNet [20] with the simple modifications, where the Rectified Linear Unit is used as the nonlinear unit.

For the SVM classifier, we employ multi-class SVM with a one-against-one configuration using LIBSVM toolbox. Since the RBF kernel produces better accuracy in practice [10], we train each dataset with the RBF kernel $K(\mathbf{x}_i, \mathbf{x}_j) \equiv e^{-\gamma \|\mathbf{x}_i - \mathbf{x}_j\|^2}$, where γ is set to 1.

For the CNN classifier, we employ the revised LeNet-5 CNN architecture, which consists of two sets of convolutional and average pooling layers, followed by a flattening convolutional layer, then two fully-connected layers and finally a softmax classifier. Specifically, the Rectified Linear Unit is used as the nonlinear unit.

To ascertain how data errors affect learning performance of the hypergraph-based applications, we added Gaussian noise, random pixel corruption and mixed noise to Coil20 and USPS datasets in the same way as to the CMU PIE corrupted dataset. We selected 20% of the labeled data samples from each subject to form the training set, and then conducted the classification experiments on the resultant corrupted datasets.Tables 8–13 list the classification results on the corrupted datasets. As shown in Tables 8–13, we have the following observations:

1. In most cases, the representation-based semi-supervised classification methods (ℓ_1-HG, EN-HG, SSC, ℓ-H and our methods) obtain the better classification results than *K*NN-HG. As the corruption ratios increase, the representation-based methods outperform the corresponding neighborhood-based methods by a larger performance margin. The reason is that

Table 10
Classification on Coil20 with mixed noise.

Mixed noise	ACC(%)			
Corrupted ratio	15%	30%	45%	60%
*K*NN-HG [46]	87.6	82.3	70.2	55.2
ℓ_1-HG [35]	88.6	87.5	83.2	81.2
EN-HG [24]	86.2	85.6	81.3	80.3
SSC [5]	85.3	84.1	81.8	82.1
ℓ_1-H [11]	87.8	86.5	83.4	82.1
ℓ_2-H [11]	88.2	86.8	84.4	82.3
SVM [2]	87.6	84.2	80.3	78.1
CNN [20]	88.3	85.2	81.6	79.8
ℓ_2-HG1	90.1	88.8	86.2	81.3
ℓ_2-HG2	**91.3**	**89.9**	**87.3**	**83.2**

Table 11
Classification on USPS with Gaussian noise.

Gaussian noise	ACC(%)			
Corrupted ratio	15%	30%	45%	60%
*K*NN-HG [46]	95.3	90.3	82.7	73.4
ℓ_1-HG [35]	**97.2**	94.2	88.1	80.4
EN-HG [24]	92.3	91.2	90.2	88.3
SSC [5]	90.4	88.7	86.4	85.2
ℓ_1-H [11]	93.4	92.3	87.6	84.1
ℓ_2-H [11]	94.2	92.4	89.6	84.5
SVM [2]	94.6	92.2	88.1	83.2
CNN [20]	92.7	91.2	87.3	82.6
ℓ_2-HG1	94.6	93.1	92.2	90.9
ℓ_2-HG2	96.6	**95.3**	**93.9**	**92.9**

Table 12
Classification on USPS with corruption .

Random pixels	ACC(%)			
Corrupted ratio	15%	30%	45%	60%
*K*NN-HG [46]	91.3	88.3	81.8	70.3
ℓ_1-HG [35]	90.2	88.2	87.8	85.4
EN-HG [24]	91.4	88.9	88.5	86.3
SSC [5]	88.3	87.4	86.8	84.3
ℓ_1-H [11]	91.5	89.6	88.4	86.2
ℓ_2-H [11]	91.6	90.3	88.7	87.0
SVM [2]	91.3	89.6	87.1	83.2
CNN [20]	89.8	87.6	84.3	79.2
ℓ_2-HG1	92.6	91.2	89.2	87.2
ℓ_2-HG2	**94.6**	**93.2**	**91.9**	**89.9**

Table 13
Classification on USPS with mixed noise.

Mixed Noise	ACC(%)			
Corrupted ratio	15%	30%	45%	60%
*K*NN-HG [46]	88.1	82.1	73.1	60.1
ℓ_1-HG [35]	88.2	86.3	83.1	80.1
EN-HG [24]	90.1	88.6	86.1	83.1
SSC [5]	86.1	86.1	85.1	81.3
ℓ_1-H [11]	90.2	88.1	87.3	84.3
ℓ_2-H [11]	90.1	88.9	88.0	85.1
SVM [2]	91.3	90.1	87.6	84.2
CNN [20]	90.4	88.8	85.3	83.2
ℓ_2-HG1	91.6	90.3	88.1	85.2
ℓ_2-HG2	**93.1**	**91.8**	**90.6**	**86.6**

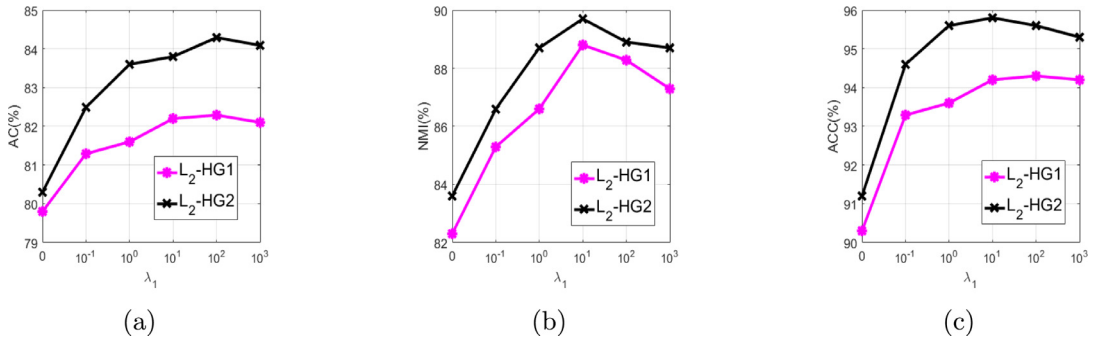

Fig. 4. The plot of clustering and classification performance versus tradeoff parameter λ_1. (a) accuracy of clustering versus λ_1, (b) normalized mutual information of clustering versus λ_1 and (c) accuracy of classification versus λ_1.

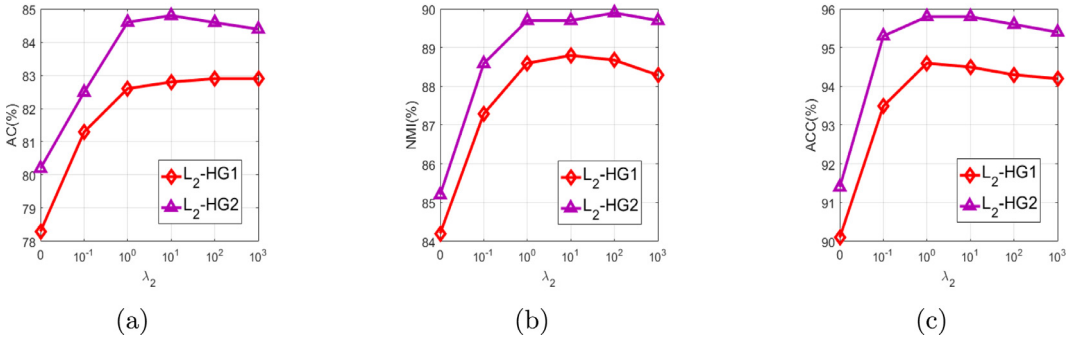

Fig. 5. The plot of clustering and classification performance versus tradeoff parameter λ_2. (a) accuracy of clustering versus λ_2, (b) normalized mutual information of clustering versus λ_2 and (c) accuracy of classification versus λ_2.

the representation-based approach effectively eliminates data errors from the original data, which is suitable to employ the classification task.

2. Our proposed ℓ_2-HG2 consistently and significantly outperforms the other methods and achieves the best classification results on the sparse noisy datasets; our proposed ℓ_2-HG1 is comparable to ℓ_1-HG in some cases. Specifically, our methods also outperform SVM and CNN, which are the typical non-graph based methods. The reasons are that our model constructs an informative hypergraph, which is not only discriminative for the data sampled from a union of manifolds, but also robust to noise and outliers; The non-graph based methods are relatively sensitive to the sparse noise. For the CNN model, the size of training dataset is too small to learn a more discriminative classifier.

5.4. Parameter setting

Our proposed two methods have five essential parameters: λ_1, λ_2, μ, θ_1 and θ_2. (1) λ_1 is the *Frobenius*-norm regularization parameter of the coefficients matrix. (2) λ_2 is the locality preserving regularization parameter of the coefficients matrix. (3) μ is the sparse regularization parameter of data errors. (4) θ_1 is the threshold parameter of our proposed ℓ_2-HG1. (5) θ_2 is the threshold parameter of our proposed ℓ_2-HG2. For the threshold parameters θ_1 and θ_2, we set the threshold parameter as the function of the average of the largest representation coefficient of each sample, i.e., $\theta_1 = \theta_1^t\, mean_i(c_i^{\max})$ and $\theta_2 = \theta_2^t\, mean_i(c_i^{\max})$, where $c_i^{\max}$ is the largest representation coefficient for sample i, $(i = 1, \ldots, N)$.

To analyze how five parameters affect learning performance, we conducted the parameter tuning experiments on the Coil20 dataset by varying the values of one parameter while fixing the other parameters. Figs. 4–8 give the experimental results with different parameter values.

From Figs. 4–8, we can have the following observations:

1. As the values of parameter λ_1 increase, the learning performance of our methods increases accordingly. When the parameter values are larger than 100, the learning performance is slightly degraded. When λ_1 is set to zero, the learning performance of our methods is degraded by a large performance margin. The experimental results demonstrate that *Frobenius*-norm (ℓ_2-norm) based regularization is crucial to enhance the learning performance of the hypergraph-based clustering and classification.
2. As the values of parameter λ_2 increase, the learning performance of our methods increases; when the parameter values are larger than 10, the learning performance of our methods is fairly stable. When λ_2 is set to zero, the learning perfor-

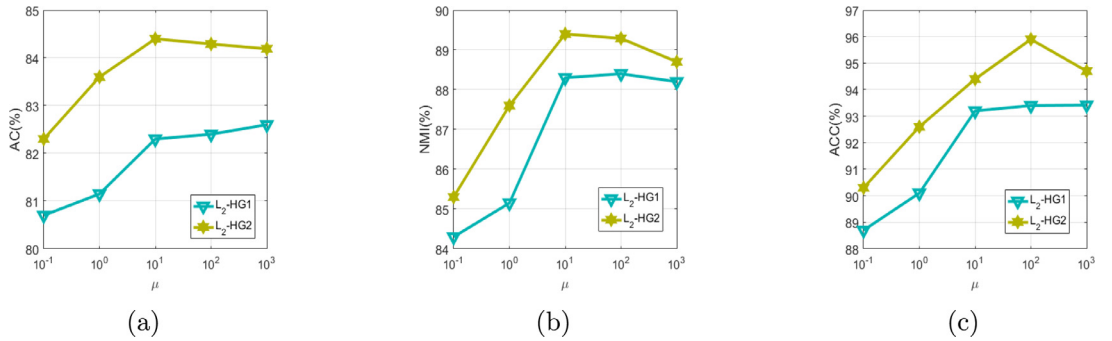

Fig. 6. The plot of clustering and classification performance versus tradeoff parameter μ. (a) Accuracy of clustering versus μ. (b) Normalized mutual information of clustering versus μ. (c) Accuracy of classification versus μ.

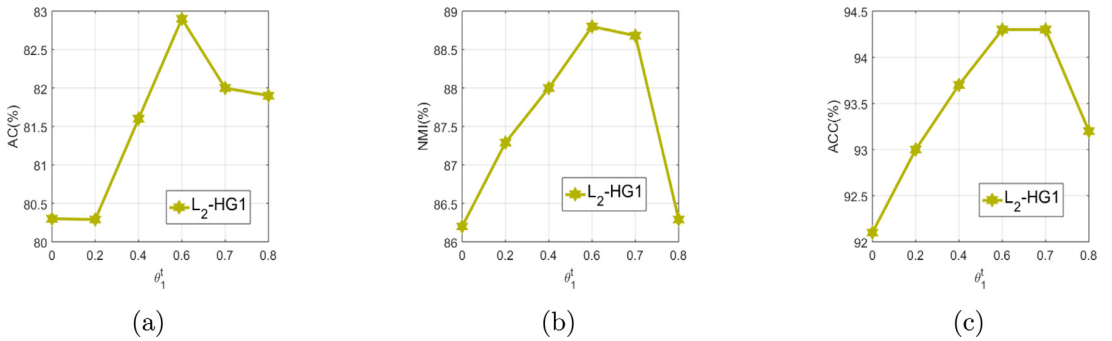

Fig. 7. The plot of clustering and classification performance versus parameter θ_1. (a) Accuracy of clustering versus θ_1. (b) Normalized mutual information of clustering versus θ_1. (c) Accuracy of classification versus θ_1.

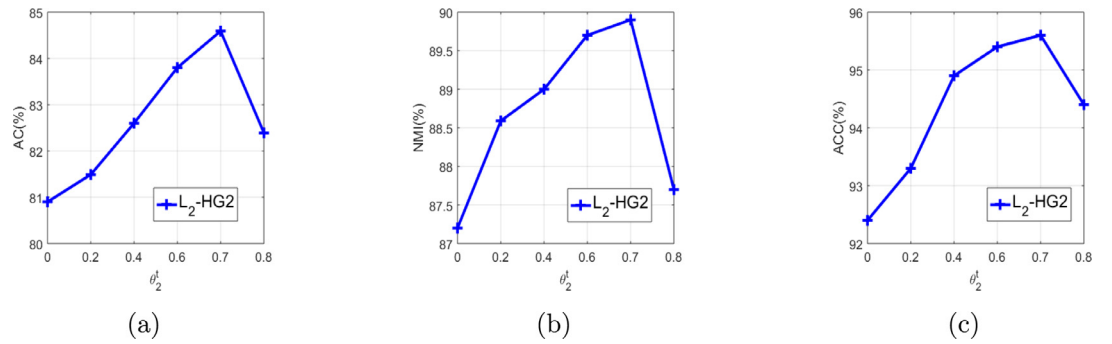

Fig. 8. The plot of clustering and classification performance versus parameter θ_2. (a) Accuracy of clustering versus θ_2. (b) Normalized mutual information of clustering versus θ_2. (c) Accuracy of classification versus θ_2.

mance of our methods is degraded by a large performance margin. These experimental results demonstrate that manifold respecting is crucial for enhancing the learning performance of the hypergraph-based clustering and classification.

3. As the values of parameter μ increase, the learning performance of our methods increases accordingly and then begins to be stable. The experimental results indicate that separating the sparse noise components from the original data is the key issue to achieve the robust learning performance.
4. As the values of parameter θ_1 increase, the learning performance of our proposed ℓ_2-HG1 increases accordingly. When the parameter value is greater than a large value, the learning performance is degraded slightly. The experimental results indicate that the hypergraph learning performance is insensitive to the parameter θ_1.
5. As the values of parameter θ_2 increase, the learning performance of our proposed ℓ_2-HG2 increases accordingly. Until the parameter is greater than a large value, the learning performance degenerates slightly. The experimental results indicate that the hypergraph learning performance is insensitive to the parameter θ_2.

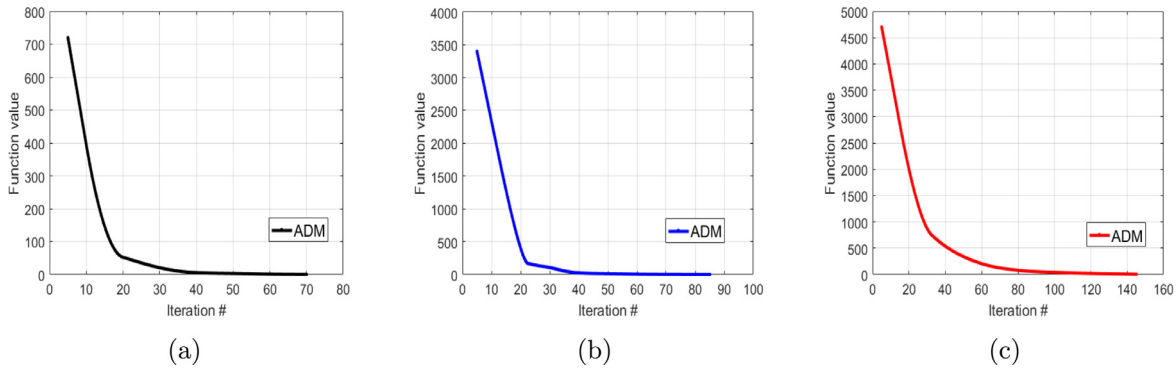

Fig. 9. Convergence curves of the ADM approach. (a) PIE dataset. (b) Coil20 dataset. (c) USPS dataset.

5.5. Convergence of the ADM-based optimization

Our proposed hypergraph model adopts the ADM approach to optimize the objective function. Thus, the convergence properties of the ADM approach is crucial to construct an informative hypergraph. For our convergence experiments, the primal variables (**P**, **C, E**),and the Lagrange multipliers ($\mathbf{M}_1$,$\mathbf{M}_2$) are set to zeros. In the following, we present the convergence curves of the ADM approach.

As shown in Fig. 9, the objective function value decreases rapidly at the outset, and then, decreases slowly. After less than 100 iterations, the objective function values are stable.

6. Further discussions

Recently, the deep neural network models have achieved promising learning performance for various learning tasks. Specifically, the convolutional neural networks achieve impressive classification results on many benchmark datasets. In our article, we compare our method with one important CNN model-LeNet [20]. For training a deep learning model, one key issue is to prepare a large number of training samples for obtaining the suitable weights of the neural networks. Since the size of the training dataset used in our article is very small, the deep learning model does not achieve impressive results in our experiments.

To address this drawback, one possible approach is to fine-tune the pre-trained deep learning models such as [37] based on our new data, where a set of pre-trained weights are used to initialize the parameters for model training. Besides, note that the existing CNN learning methods can hardly model the data correlations, whereas the recent proposed graph convolutional networks (GCN) [19] can mitigate this problem effectively. Thus, we can model the high-order data correlations via a hypergraph. Then, both the original data and high-order correlation are used as the inputs of the network for classification, where the classification results are derived by considering the high-order correlation.

7. Conclusion

In this paper, we proposed a novel hypergraph model to formulate the high-order data correlations. Compared with the existing methods, our model has the following two key advantages: (1) Our hypergraph model adopts affine subspace ridge regression to choose the vertices of a hyperedge, which is insensitive to sparse noise and corruption. (2) Our hypergraph model is specifically applicable for data sampled from a union of multiple non-linear manifolds. We have applied our model to image clustering and classification. Experimental results on the real-world image datasets demonstrate that our proposed two hypergraph construction methods are superior to the-state-of-the-art methods. Currently, hypergraph learning usually adopts the neighborhood-based approach to generate a set of the hyperedges, which is sensitive to noise and data corruptions. Of further interest is the adoption of other representation-based strategy to generate an informative hyperedge set.

Acknowledgements

The work was supported by National key research and development plan project (nos. 2018YFC0830105, 2018YFC0830100), in part by the National Natural Science Foundation of China (nos. 61732005, 61672271, 61761026, 61762056, 61702136, 61773130, 61876042, 61876161), in part by Yunnan high and new technology industry project (no. 201606), and in part by the Natural Science Fundation of Yunnan Province (no. 2018FB104).

References

[1] S. Agarwal, J. Lim, L. Zelnik, P. Perona, D. Kriegman, S. Belongie, Beyond pairwise clustering, in: Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on, 2, IEEE, 2005, pp. 838–845.
[2] C. Chang, C. Lin, LIBSVM: a library for support vector machines, ACM Trans. Intell. Syst. Technol. 2 (3) (2011) 1–27.
[3] A. Ducournau, A. Bretto, Random walks in directed hypergraphs and application to semi-supervised image segmentation, Comput. Vis. Image Understand. 120 (2014) 91–102.
[4] E. Elhamifar, R. Vidal, Sparse manifold clustering and embedding, in: Advances in Neural Information Processing Systems, 2011, pp. 55–63.
[5] E. Elhamifar, R. Vidal, Sparse subspace clustering: algorithm, theory, and applications, IEEE Trans. Pattern Anal. Mach. Intell. 35 (11) (2013) 2765–2781.
[6] Q. Fang, J. Sang, C. Xu, Y. Rui, Topic-sensitive influencer mining in interest-based social media networks via hypergraph learning, IEEE Trans. Multimedia 16 (3) (2014) 796–812.
[7] 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 (1) (2013) 92–104.
[8] Y. Gao, R. Ji, P. Cui, Q. Dai, G. Hua, Hyperspectral image classification through bilayer graph-based learning, IEEE Trans. Image Process. 23 (7) (2014) 2769–2778.
[9] C. Hong, J. Yu, J. Li, X. Chen, Multi-view hypergraph learning by patch alignment framework, Neurocomputing 118 (2013) 79–86.
[10] C. Hsu, C. Lin, A comparison of methods for multi-class support vector machines, IEEE Trans. Neural Netw. 13 (2) (2002) 415–425.
[11] S. Huang, D. Yang, B. Liu, X. Zhang, Regression-based hypergraph learning for image clustering and classification arXiv, Comput. Vis. Pattern Recognit. (2016).
[12] Y. Huang, Q. Liu, F. Lv, Y. Gong, D.N. Metaxas, Unsupervised image categorization by hypergraph partition, IEEE Trans. Pattern Anal. Mach. Intell. 33 (6) (2011) 1266–1273.
[13] Y. Huang, Q. Liu, D. Metaxas, Video object segmentation by hypergraph cut, in: Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on, IEEE, 2009, pp. 1738–1745.
[14] Y. Huang, Q. Liu, S. Zhang, D.N. Metaxas, Image retrieval via probabilistic hypergraph ranking, in: Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on, IEEE, 2010, pp. 3376–3383.
[15] J.J. Hull, A database for handwritten text recognition research, IEEE Trans. Pattern Anal. Mach. Intell. 16 (5) (1994) 550–554.
[16] T. Jin, J. Yu, J. You, K. Zeng, C. Li, Z. Yu, Low-rank matrix factorization with multiple hypergraph regularizer, Pattern Recognit. 48 (3) (2015) 1011–1022.
[17] T. Jin, Z. Yu, L. Li, C. Li, Multiple graph regularized sparse coding and multiple hypergraph regularized sparse coding for image representation, Neurocomputing 154 (2015) 245–256.
[18] S. Kim, C.D. Yoo, S. Nowozin, P. Kohli, Image segmentation using higher-order correlation clustering, IEEE Trans. Pattern Anal. Mach. Intell. 36 (9) (2014) 1761–1774.
[19] T.N. Kipf, M. Welling, Semi-supervised classification with graph convolutional networks, in: international Conference on Learning Representations, 2017.
[20] Y. Lecun, L. Bottou, Y. Bengio, P. Haffner, Gradient-based learning applied to document recognition, Proc. IEEE 86 (11) (1998) 2278–2324.
[21] H. Lee Kwang, C.H. Cho, Hierarchical reduction and partition of hypergraph, IEEE Trans. Syst. Man Cybern.Part B (Cybern.) 26 (2) (1996) 340–344.
[22] L. Li, T. Li, News recommendation via hypergraph learning: encapsulation of user behavior and news content, in: Proceedings of the Sixth ACM International Conference on Web Search and Data MMining, ACM, 2013, pp. 305–314.
[23] G. Liu, Z. Lin, S. Yan, J. Sun, Y. Yu, Y. Ma, Robust recovery of subspace structures by low-rank representation, IEEE Trans. Pattern Anal. Mach. Intell. 35 (1) (2013) 171–184.
[24] Q. Liu, Y. Sun, C. Wang, T. Liu, D. Tao, Elastic net hypergraph learning for image clustering and semi-supervised classification, IEEE Trans. Image Process. 26 (1) (2017) 452–463.
[25] C.Y. Lu, H. Min, Z.Q. Zhao, L. Zhu, D.S. Huang, S. Yan, Robust and efficient subspace segmentation via least squares regression, in: Computer Vision–ECCV 2012, 2012, pp. 347–360.
[26] S.A. Nene, S.K. Nayar, H. Murase, et al., Columbia Object Image Library (Coil-20), 1996.
[27] Y. Nesterov, Introductory lectures on convex optimization: a basic course, 87, Springer Science & Business Media, 2013.
[28] A.Y. Ng, M.I. Jordan, Y. Weiss, On spectral clustering: analysis and an algorithm, in: Advances in Neural Information Processing Systems, 2002, pp. 849–856.
[29] X. Peng, L. Zhang, Z. Yi, K.K. Tan, Learning locality-constrained collaborative representation for robust face recognition, Pattern Recognit. 47 (9) (2014) 2794–2806.
[30] J. Shi, J. Malik, Normalized cuts and image segmentation, IEEE Trans. Pattern Anal. Mach. Intell. 22 (8) (2000) 888–905.
[31] T. Sim, S. Baker, M. Bsat, The cmu pose, illumination, and expression (pie) database, in: Automatic Face and Gesture Recognition, 2002. Proceedings. Fifth IEEE International Conference on, IEEE, 2002, pp. 53–58.
[32] L. Sun, S. Ji, J. Ye, Hypergraph spectral learning for multi-label classification, in: Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM, 2008, pp. 668–676.
[33] Z. Tian, T. Hwang, R. Kuang, A hypergraph-based learning algorithm for classifying gene expression and arraycgh data with prior knowledge, Bioinformatics 25 (21) (2009) 2831–2838.
[34] J. Wang, J. Yang, K. Yu, F. Lv, T. Huang, Y. Gong, Locality-constrained linear coding for image classification, in: Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on, IEEE, 2010, pp. 3360–3367.
[35] M. Wang, X. Liu, X. Wu, Visual classification by ℓ_1-hypergraph modeling, IEEE Trans. Knowl. Data Eng. 27 (9) (2015) 2564–2574.
[36] Y. Wang, P. Li, C. Yao, Hypergraph canonical correlation analysis for multi-label classification, Signal Process. 105 (2014) 258–267.
[37] L. Windrim, A. Melkumyan, R.J. Murphy, A. Chlingaryan, R. Ramakrishnan, Pretraining for hyperspectral convolutional neural network classification, IEEE Trans. Geosci. Remote Sens. 56 (5) (2018) 2798–2810, doi:10.1109/TGRS.2017.2783886.
[38] M. Yin, S. Xie, Z. Wu, Y. Zhang, J. Gao, Subspace clustering via learning an adaptive low-rank graph, IEEE Trans. Image Process. 27 (8) (2018) 3716–3728.
[39] M. Yin, D. Zeng, J. Gao, Z. Wu, S. Xie, Robust multinomial logistic regression based on rpca, IEEE J. Sel. Top. Signal Process. 12 (6) (2018) 1144–1154.
[40] J. Yu, Y. Rui, D. Tao, Click prediction for web image reranking using multimodal sparse coding, IEEE Trans. Image Process. 23 (5) (2014) 2019–2032.
[41] J. Yu, D. Tao, M. Wang, Adaptive hypergraph learning and its application in image classification, IEEE Trans. Image Process. 21 (7) (2012) 3262–3272.
[42] R. Zass, A. Shashua, Probabilistic graph and hypergraph matching, in: Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on, IEEE, 2008, pp. 1–8.
[43] Y. Zhang, Recent advances in alternating direction methods: practice and theory, IPAM Workshop on Continuous Optimization, 2010.
[44] Z. Zhang, L. Bai, Y. Liang, E. Hancock, Joint hypergraph learning and sparse regression for feature selection, Pattern Recognit. 63 (2017) 291–309.
[45] 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 (5) (2011) 1327–1336.
[46] D. Zhou, J. Huang, B. Schölkopf, Learning with hypergraphs: clustering, classification, and embedding, in: Advances in neural information processing systems, 2007, pp. 1601–1608.