Visual Classification by ℓ_1-Hypergraph Modeling

Meng Wang, *Member, IEEE*, Xueliang Liu, and Xindong Wu, *Fellow, IEEE*

Abstract—Visual classification has attracted considerable research interests in the past decades. In this paper, a novel ℓ_1-hypergraph model for visual classification is proposed. Hypergraph learning, as a natural extension of graph model, has been widely used in many machine learning tasks. In previous work, hypergraph is usually constructed by attribute-based or neighborhood-based methods. That is, a hyperedge is generated by connecting a set of samples sharing a same feature attribute or in a neighborhood. However, these methods are unable to explore feature space globally or sensitive to noises. To address these problems, we propose a novel hypergraph construction approach that leverages sparse representation to generate hyperedges and learns the relationship among hyperedges and their vertices. First, for each sample, a hyperedge is generated by regarding it as the centroid and linking it as well as its nearest neighbors. Then, the sparse representation method is applied to represent the centroid vertex by other vertices within the same hyperedge. The vertices with zero coefficients are removed from the hyperedge. Finally, the representation coefficients are used to define the incidence relation between the hyperedge and the vertices. In our approach, we also optimize the hyperedge weights to modulate the effects of different hyperedges. We leverage the prior knowledge on the hyperedges so that the hyperedges sharing more vertices can have closer weights, where a graph Laplacian is used to regularize the optimization of the weights. Our approach is named ℓ_1-hypergraph since the ℓ_1 sparse representation is employed in the hypergraph construction process. The method is evaluated on various visual classification tasks, and it demonstrates promising performance.

Index Terms—Visual classification, hypergraph, regularization, hyperedge

1 Introduction

Visual classification, as a fundamental problem in computer vision and data mining, has drawn intensive attention in past decades. In the literature, a lot of classification approaches are proposed, such as k-nearest neighbors method (KNN), support vector machine (SVM), decision tree, Bayes model, linear discriminant analysis (LDA), and graph-based methods [17], [35], [42], [50].

Among these methods, hypergraph learning [22], [24], as a natural extension of the graph model, has been widely investigated and achieved promising performance in many tasks. This approach is built on a hypergraph structure which contains two parts, a vertex set and a hyperedge set. The same as graph model, vertices are used to represent data samples. However, different from a graph edge that only links two vertices, a hyperedge can link to any number of vertices. In other words, a hyperedge can be viewed as a set of vertices, and hence hypergraph is able to capture the high-order relationship of samples naturally. In particular, hypergraph model can be an effective tool to comprehensively mine the complex sample relationships in visual data and thus achieved encouraging performance in many visual classification tasks [16], [23], [58].

In previous work, there are two types of strategies for building a hypergraph, i.e., attribute-based and neighborhood-based approaches. Early works mainly employ the attribute-based strategy, in which a hyperedge connects the samples that share a feature attribute [16]. Since a hyperedge only considers a single feature dimension, this approach cannot exploit feature space globally in the hypergraph construction step. The neighborhood-based approach addresses this weakness by connecting samples in a neighborhood. More specifically, for each sample, a hyperedge is generated by connecting this centroid sample and its k nearest neighbors [23], [58]. However, as nearest neighbors methods are sensitive to noises and there are usually many noisy data in visual classification tasks, the performance of this approach is usually degraded. In addition, it is not easy to tune the parameter k in practical implementations.

To solve these problems, we present a novel method that leverages sparse representation to construct a hypergraph. In detail, we first generate a hyperedge by connecting a vertex and its k nearest neighbors. (1) This step is the same with neighborhood-based hypergraph construction [23], [58]. We then employ sparse representation to represent the centroid vertex by its neighbors. (2) The vertices with zero coefficients are removed from the hyperedge. We also use the representation coefficients to define the incidence relationship of the hyperedge and its vertices. The idea is inspired by the latest research that suggests locality is very essential for effectively reconstructing the original signal by nonlinear function learning method [51], [59]. By employing the sparse representation step, we can remove noisy vertices linked by a hyperedge, as most of them are expected to have a zero representation coefficient. It also facilitates the setting of k, as the sparse representation step actually has a sample selection effect and thus we can simply set a large value to k. In our scheme, we also optimize the weights of

- M. Wang and X. Liu are with the School of Computer Science and Information Engineering, Hefei University of Technology, Hefei 230009, China. E-mail: {eric.mengwang@gmail.com, liuxueliang1982}@gmail.com.
- X. Wu is with the School of Computer Science and Information Engineering, Hefei University of Technology, Hefei 230009, China, and the Department of Computer Science, University of Vermont, Burlington, VT 05405. E-mail: xwu@uvm.edu.

*Manuscript received 14 Oct. 2014; revised 12 Mar. 2015; accepted 16 Mar. 2015. Date of publication 19 Mar. 2015; date of current version 3 Aug. 2015. Recommended for acceptance by L. Khan.
For information on obtaining reprints of this article, please send e-mail to: reprints@ieee.org, and reference the Digital Object Identifier below.
Digital Object Identifier no. 10.1109/TKDE.2015.2415497*

hyperedges to further investigate the impact of different hyperedges. Instead of a simple ℓ_1 or ℓ_2 regularizer, we introduce the prior knowledge on hyperedges which suggests that the hyperedges sharing more vertices should have closer weights. Based on such assumption, a graph Laplacian regularizer is employed to optimize the hyperedge weights. We name the hypergraph constructed in our way ℓ_1-hypergraph to differentiate it with the conventional hypergraphs, as the incidence matrix of the proposed hypergraph model is generated with the ℓ_1 sparse representation on data samples. We use the proposed method in visual classification tasks and promising results are achieved.

The proposed approach has multiple advantages.

- In comparison with the conventional neighborhood-based hypergraph learning methods, the proposed method can refine the samples selected by KNN and thus noisy data can be removed.
- The approach is not sensitive to the parameter k as the sparse representation approach has a sample selection effect.
- We introduce a novel method to optimize the weights of hyperedges, which further improves the performance of our approach.

The remainder of this paper is structured as follows. First, we review some related works in Section 2. We then explain our hypergraph construction and learning approach in Section 3. We discuss experimental results in Section 4. Finally, our conclusions are presented in Section 5.

2 Related Work

Visual classification solves the problem of classifying an image by its content, which is an essential task in multimedia and data mining communities and has been extensively studied for many years [36]. In the past decades, lots of researches have been done concerning sophisticated supervised model to classify visual documents into different categories. In early work, popular techniques like k-nearest neighbors classifier [4], Naive Bayes [34] and SVM [33] were widely used. In [5], Bosch et al. provided a method of inhibiting background clutter and adding invariance to object instance's position and employed random forests to classify images. In [43], Perronnin et al. applied Fisher Kernels to image categorization where an image is characterized as a gradient distribution on GMM model learned from image features. In [21], a novel multiple rank regression model for matrix data classification is proposed. Multiple-rank left projecting vectors and right projecting vectors are employed to regress each matrix data set to its label for each category. In [56], Yang et al. proposed a local regression and global alignment method to learn a robust Laplacian matrix for semi-supervised ranking.

Besides these supervised learning methods that train a model from labeled samples, semi-supervised learning [27], which incorporates not only labeled but also unlabeled samples for improving classification performance, has also been exploited in visual classification problem. Many methods have been proposed, such as transductive SVM [7], Co-Training [19], and graph-based model [39], [45]. In machine learning, graph is a fundamental tool for modeling pairwise relationship among objects and solving many tasks like classification [44], [45], ranking [20], [2], and clustering [26], [40], [46]. For example, Yang et al. [57] proposed an algorithm for image annotation by integrating label correlation and visual similarity mining into a joint framework. In the proposed method, a graph model is first constructed according to image visual features, and a multi-label classifier is then trained by simultaneously uncovering the shared structure common to different labels and the visual graph for image annotation. In [31], Li et al. addressed the binary coding problem and proposed a method to directly optimize the graph Laplacian in spectral hashing framework. In [10], in order to solve the web image annotation problem, Chen et al. proposed a sparse-graph-based multi-label propagation on large-scale image dataset. In [52], Wang et al. introduced a web image search reranking approach that explores multiple modalities in a graph-based learning scheme. The approach can effectively integrate the learning of relevance scores, weights of modalities, and the distance metric and its scaling of each modality into a unified scheme. However, in graph-based learning, an edge is only able to model the pairwise relationship of two samples as it connects two vertices, and thus it is unable to model high-order relationship.

As a natural extension of the graph model, hypergraph has been employed as a more useful tool for relationship modeling [30], [49]. A hypergraph generalizes the traditional graph by allowing a hyperedge to connect more than two vertices. So, along with its weight, a hyperedge is able to measure the closeness of a group of samples, and it can be considered as a high-order modeling. In [23], a probabilistic hypergraph learning framework for image retrieval is presented. In the approach, images are taken as vertices in a weighted hypergraph and the task of searching for images is formulated as a hypergraph ranking problem. In [53], Wong et al. proposed a hypergraph-based 3D object description method, in which the vertices denote the surface patches of an object in a computer-aided design system and the hyperedges represent connections between pairs of boundary segments. A class-specific hypergraph is proposed in [55] to explore both scale-invariant-feature-transform (SIFT) feature and global geometric constraints for object recognition, in which the vertices of the constructed hypergraph represent images belonging to an object category. In [8], Bu et al. modeled social media information and music content with a hypergraph and build a music recommendation system upon it. As previously mentioned, the existing hypergraph learning methods mainly use attribute-based or neighborhood-based strategies to build a hypergraph, and these methods cannot explore feature space globally or suffer from noisy data. To overcome these problems, we improve the neighborhood-based hypergraph construction approach by employing sparse representation to select vertices linked by a hyperedge. We also define incidence relationship between the hyperedge and its vertices and optimize the weights of hyperedges.

Recently, motivated by the success of sparse coding in signal processing, an effective model, sparse representation, has attracted a lot of attentions in computer vision and data mining. Many research efforts tried to use sparse representation in visual content analysis and promising results are reported [25], [32], [54], [60]. For example, in [60], Yuan et al. addressed the problem of visual classification with multiple features and proposed a multitask joint sparse representation model to combine the strengths of multiple features for recognition. In the proposed method, a joint sparsity-inducing norm is utilized to enforce class-level joint sparsity pattern among the multiple representation vectors. Motivated by the fact that kernel trick can capture the nonlinear similarity of features, Gao et al. [15] proposed a kernel sparse representation method to represent high-dimensional features by mapping them with an implicit kernel function. In [25], to learn both optimal intrinsic manifold and sparse code jointly, Jin et al. represented images by sparse coding with a graph regularizer. In [14], to address the problem of automatically uncovering the underlying group structure from images, Feng et al. proposed a novel auto-grouped sparse representation method which could group semantically correlated feature elements together by optimally fusing their multiple sparse representations. Cheng et al. [3] employed sparse representation to model the relationship among samples in graph construction, whereby a so-called ℓ_1-graph method is developed. However, to our knowledge, our work is the first attempt that employs sparse representation to model the relationship among samples in hypergraph construction.

3 ℓ_1 Hypergraph Learning

3.1 Hypergraph Model

In machine learning, graph model is a fundamental tool for modeling pairwise relationships among samples, with the vertices denoting the samples and the relationships between two samples are measured by edges. The graph can be undirected or directed, depending on whether the relationships among objects are symmetric.

A learning algorithm can be performed on such a graph to classify unlabeled samples. For example, in a social network, the relations of different people could be formulated as a graph model, with the vertices and edges representing people and their relationships, respectively. However, in visual content analysis, images are represented by high-dimensional features, and the relationship among multiple images needs to be modeled to further analyze their content. But the graph model cannot handle the relation among multiple objects well. A natural way of formulating complex relational objects is the hypergraph model, where a hyperedge can link more than two vertices.

Mathematically, let $\mathcal{V}$ denote a finite set of objects, and let $\mathcal{E}$ be the family of subsets e of $\mathcal{V}$ such that $\bigcup_{e\in\mathcal{E}} = \mathcal{V}$. Then, we call $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathbf{w})$ a hypergraph with vertex set $\mathcal{V}$, hyperedge set $\mathcal{E}$, and hyperedge weight vector $\mathbf{w}$. A $|\mathcal{V}| \times |\mathcal{E}|$ incidence matrix $\mathbf{H}$ represents $\mathcal{G}$ with the following elements:

$$\mathbf{H}(v,e) = \begin{cases} 1, & \text{if } v \in e \\ 0, & \text{otherwise.} \end{cases} \tag{1}$$

We list the notations and their definitions used throughout the paper in Table 1. Also note that we may use "sample" and "vertex" interchangeably.

TABLE 1
Notations and Definitions

Notation	Definition
$\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathbf{w})$	$\mathcal{G}$ denotes a hypergraph, where $\mathcal{V}$, $\mathcal{E}$, and $\mathbf{w}$ are the set of vertices, set of hyperedges, and weights of the hyperedges, respectively.
v	a vertex in the hypergraph.
e	a hyperedge in the hypergraph.
$\mathbf{H}$	Incidence matrix of the hypergraph.
n	The number of vertices, that is, the size of $\mathcal{V}$.
$\mathbf{W}$	The diagonal matrix of $\mathbf{w}$.
$\mathbf{D}_v$	The diagonal matrix of vertex degrees.
$\mathbf{D}_e$	The diagonal matrix of hyperedge degrees.
$\mathbf{Y}$	The label matrix of samples, in which $Y_{ij} = 1$ indicates the ith sample belongs to the jth class, and otherwise $Y_{ij} = 0$.
$\mathbf{L}$	The laplacian matrix of hypergraph.
$\mathbf{F}$	Predicted classification scores.

Based on equation (1), the degree of each vertex $v \in \mathcal{V}$ is defined as

$$d(v) = \sum_{e\in\mathcal{E}} \mathbf{w}(e)\mathbf{H}(v,e), \tag{2}$$

while the degree of hyperedge $e \in \mathcal{E}$ is given by

$$\delta(e) = \sum_{v\in\mathcal{V}} \mathbf{H}(v,e). \tag{3}$$

We use $\mathbf{D}_v$, $\mathbf{D}_e$, and $\mathbf{W}$ to denote the diagonal matrices of vertex degrees, hyperedge degrees, and hyperedge weights, respectively.

In previous work, one common way to construct a hypergraph is the KNN method [23], [58]. In the constructed hypergraph structure, each data sample is denoted by a vertex and its k-nearest neighbors are linked via a hyperedge. In this literature, the neighboring samples are determined based on euclidean distance. As we know, KNN is very sensitive to noisy data, and the hypergraph is not so robust when noisy data are involved. In addition, how to choose an optimized parameter k is also an open issue in constructing the hypergraph. If k is small, a very sparse incidence matrix would be obtained. On the contrary, if k is too large, many irrelevant items would be connected and thus degrade classification performance.

3.2 ℓ_1-Hypergraph Construction

To tackle the problem in neighborhood-based hypergraph construction, we propose a sparse representation method. The latest research suggests that locality is very essential for effectively representing the original signal by a linear/nonlinear function learning method [48], [51], [59]. In this work, we use the sparse representation to formulate the relation between a hyperedge and its vertices, and we then use the representation coefficients to generate the incidence matrix. The aim of sparse representation is to find a set of basic samples to reconstruct an input vector as the linear combination of these basis vectors. In our problem, we would like to reconstruct a centroid sample by other data samples in the same hyperedge. The vertices with zero coefficients are removed from the hyperedge. We use the representation coefficients to indicate the incidence relations between the hyperedge and its vertices. Since the incidence relations cannot be negative, we add a non-negative constraint on the representation coefficients in the sparse representation model.

Mathematically, let $\mathbf{v}$ and $\mathbf{P}$ denote the centroid sample and its k nearest neighbors respectively. These data samples are connected by a hyperedge e. The sparse representation of $\mathbf{v}$ by $\mathbf{P}$ is to solve the following optimization function.

$$\begin{aligned}&\arg\min_{\mathbf{x}} \|\mathbf{Px}-\mathbf{v}\|^2+\gamma\|\mathbf{x}\|_1\\ &\text{s.t. } \forall i,\ x_i \geq 0.\end{aligned} \tag{4}$$

Here, we can interpret the first term in equation (4) as the reconstruction term which tries to force the algorithm to provide a good representation of the original signal $\mathbf{v}$ and the second term as the sparsity penalty which forces our representation of $\mathbf{v}$ to be sparse. The constant γ is a parameter that modulates the effects of the two terms. The non-negative constraint $x_i \geq 0$ forces the representation coefficients non-negative.

Each data sample v_i could be used as the centroid vertex to generate a hyperedge e_i and the hypergraph is built on the hyperedge set, while the incidence matrix $\mathbf{H}$ of the hypergraph is defined by the representation coefficients as

$$\mathbf{H}(v_j,e_i)=\begin{cases}x_i^j, & \text{if } v_j\in e_i\\ 0, & \text{otherwise,}\end{cases} \tag{5}$$

where e_i is generated with the centroid vertex v_i, and x_i^j is jth element of representation coefficients $\mathbf{x}_i$.

3.3 ℓ_1-Hypergraph Modeling

According to [61], hypergraph can be employed to solve a lot of machine learning problems, including classification, clustering and ranking. A multiple class classification problem can be formulated as the following regularization framework:

$$\arg\min_{\mathbf{F},\mathbf{w}} \mathbf{\Omega}(\mathbf{F})+\lambda\mathbf{R}_{emp}(\mathbf{F})+\mu\mathbf{\Phi}(\mathbf{w}). \tag{6}$$

Here $\lambda > 0$ and $\mu > 0$ are trade-off parameters that balance empirical loss and regularizers. In this objective function, there are two parameters to optimize, i.e., the classification relevance scores $\mathbf{F}$ and hyperedge weights $\mathbf{w}$. We can also find that there are three parts in this objective function. The first part $\Omega(\mathbf{F})$ is the empirical loss, defined as

$$\begin{aligned}\mathbf{\Omega}(\mathbf{F})=&\frac{1}{2}\sum_{e\in\mathcal{E}}\sum_{u,v\in\mathcal{V}}\frac{\mathbf{w}(e)\mathbf{H}(u,e)\mathbf{H}(v,e)}{\delta(e)}\\ &\times\left(\frac{\mathbf{F}(u)}{\sqrt{d(u)}}-\frac{\mathbf{F}(v)}{\sqrt{d(v)}}\right)^2.\end{aligned} \tag{7}$$

Letting $\mathbf{\Theta}=\mathbf{D}_v^{-\frac{1}{2}}\mathbf{HWD}_e^{-1}\mathbf{H}^T\mathbf{D}_v^{-\frac{1}{2}}$ and $\mathbf{\Delta}=\mathbf{I}-\mathbf{\Theta}$, the normalized cost function can be rewritten as

$$\mathbf{\Omega}(\mathbf{F})=\sum_i^c \mathbf{F}_i^T\mathbf{\Delta}\mathbf{F}_i, \tag{8}$$

where $\mathbf{\Delta}$ is a positive semi-definite matrix, known as the Laplacian of the hypergraph, and c is the number of classes in the problem.

The second part in equation (6) is the loss function $\mathbf{R}_{emp}(\mathbf{F})$, defined as

$$\mathbf{R}_{emp}(\mathbf{F})=\|\mathbf{F}-\mathbf{Y}\|^2=\sum_i^c\|\mathbf{F}_i-\mathbf{Y}_i\|^2, \tag{9}$$

where $\mathbf{Y}$ is the initial label matrix, in which $Y_{ij}=1$ indicates ith sample belongs to the jth class, and otherwise $Y_{ij}=0$.

The last part of equation (6) is the regularizer of weights $\mathbf{w}$. As we know, hyperedges have different effects and performing a weighting on the hyperedges is useful in exploiting the relevance between the vertices. Here, we integrate the learning of hyperedge weights into the formulation, with the hypergraph modeling formulated as a regularization framework. As shown in previous work [11], an effective weighting is helpful for improving system performance.

In machine learning, a q-norm regularization is frequently used, i.e.,

$$\mathbf{\Phi}(\mathbf{w})=\|\mathbf{w}\|_q, \tag{10}$$

where $q=2$ and $q=1$ denote ℓ_2 and ℓ_1 regularizer, respectively. However, these regularizers neglect the relationships of hyperedges. In addition, simply applying these regularizers without other constraints will lead to trivial solutions, i.e., the optimization of equation (6) tends to let $\mathbf{w}=0$.

In our approach, we assume that hyperedges sharing a lot of vertices in hypergraph should receive close weights and thus pose a graph Laplacian regularizer on hyperedge weights. Specifically, let θ_{ij} denote the similarity between hyperedge e_i and e_j. That is, $\theta_{ij}=1$ if the two hyperedges e_i and e_j could be reconstructed by at least N_k shared vertices, which provide a way to model the interaction between two vertices from indirect interactions through neighbors. To assign weights to hyperedges consistent with the prior knowledge, we define the following Laplacian cost function over the hyperedge weights:

$$\Phi(\mathbf{w})=\frac{1}{2}\sum_{i,j=1}^n\theta_{ij}\left(\frac{\mathbf{w}(e_i)}{\sqrt{\tau(e_i)}}-\frac{\mathbf{w}(e_j)}{\sqrt{\tau(e_j)}}\right)^2, \tag{11}$$

where θ_{ij} is the number of shared vertices of hyperedges e_i and e_j, and $\tau(e_i)=\sum_j\theta_{ij}$. We add a constraint $\sum_{i=1}^n w_i=1$ to fix the sum of all hyperedge weights and avoid trivial solution.

Until now, the optimization function in equation (6) can be written as

$$\arg\min_{\mathbf{F},\mathbf{w}} \mathbf{F}^T\Delta\mathbf{F} + \lambda\|\mathbf{F}-\mathbf{Y}\|^2 + \mu\sum_{i,j=1}^{n}\theta_{ij}\left(\frac{\mathbf{w}(e_i)}{\sqrt{\tau(e_i)}} - \frac{\mathbf{w}(e_j)}{\sqrt{\tau(e_j)}}\right)^2$$
$$s.t. \sum_{i=1}^{n} w_i = 1. \tag{12}$$

3.4 Optimization Solving

The objective function (equtation (12)) is not jointly convex with respect to **F** and **w**. However, it is convex with respect to **F** if **w** is fixed, and vice versa. Thus, the problem can be solved by an alternating optimization method.

First, if we fix **w**, the learning task is to minimize the sum of the two terms:

$$\arg\min_{\mathbf{F}} \left\{\mathbf{F}^T\Delta\mathbf{F} + \lambda\|\mathbf{F}-\mathbf{Y}\|^2\right\}. \tag{13}$$

The partial derivative of the objective function in (13) with respect to **F** is

$$\mathbf{F}(\mathbf{I}-\mathbf{\Theta}) + \lambda(\mathbf{F}-\mathbf{Y}) = 0$$
$$\Longrightarrow \mathbf{F} = \frac{\lambda}{1+\lambda}\left(\mathbf{I} - \frac{\mathbf{\Theta}}{1+\lambda}\right)^{-1}\mathbf{Y}, \tag{14}$$

where $\mathbf{\Theta} = \mathbf{D}_v^{-1/2}\mathbf{H}\mathbf{W}\mathbf{D}_e^{-1}\mathbf{H}^T\mathbf{D}_v^{-1/2}$, as defined previously.

Then, with a fixed **F**, we can optimize the parameters **w**. Now the optimization becomes

$$\arg\min_{\mathbf{w}} \mathbf{F}^T\Delta\mathbf{F} + \frac{\mu}{2}\sum_{i,j=1}^{n}\theta_{ij}\left(\frac{\mathbf{w}(e_i)}{\sqrt{\tau(e_i)}} - \frac{\mathbf{w}(e_j)}{\sqrt{\tau(e_j)}}\right)^2$$
$$s.t. \sum_{i=1}^{Ne} w_i = 1. \tag{15}$$

Letting

$$\gamma_i(e) = \frac{1}{2}\sum_{u,v\in\mathcal{V}}\frac{\mathbf{H}(u,e)\mathbf{H}(v,e)}{\delta(e)}$$
$$\times\left(\frac{f_i(u)}{\sqrt{d(u)}} - \frac{f_i(v)}{\sqrt{d(v)}}\right)^2, \tag{16}$$

equation (15) can be rewritten as

$$\arg\min_{\mathbf{w}}\left\{\sum_{i}^{c}\mathbf{w}^T\gamma_i + \frac{\mu}{2}\sum_{i,j=1}^{n}\theta_{ij}\left(\frac{\mathbf{w}(e_i)}{\sqrt{\tau(e_i)}} - \frac{\mathbf{w}(e_j)}{\sqrt{\tau(e_j)}}\right)^2\right\}$$
$$s.t. \sum_{i=1}^{n} w_i = 1. \tag{17}$$

Obviously, equation (17) can be cast as a quadratic problem with linear constraints and solved by the interior-point algorithm [6].

Since the iteration consistently degrades the objective function in equation (12) and the objective is lower bounded by zero, the process is guaranteed to converge.

The alternating optimization implementation is summarized in Algorithm 1.

Algorithm 1. Proposed ℓ_1-Hypergraph Learning Approach

```
1: Take each data sample as the centroid vertex and find its $k$ nearest neighbors to form a hyperedge.
2: Perform sparse representation to reconstruct the centroid vertex with the vertices in the hyperedge, i.e., its neighbors. Remove the vertices with zero coefficients from the hyperedge.
3: Use the representation coefficients to generate the incidence matrix, as indicated in equation (5).
4: Compute the hypergraph Laplacian $\mathbf{\Delta}$ and the hyperedge similarities $\theta_{ij}$.
5: Optimize $\mathbf{F}$ according to equation (14).
6: Optimize $\mathbf{w}$ according to equation (17).
7: Update hypergraph Laplacian $\mathbf{\Delta}$ according to the new $\mathbf{w}$.
8: Go to step 4, and iterate until convergence.
```

After obtaining optimized **F**, the ith sample can be assigned to the kth class that satisfies

$$k = \arg\min_{j} F_{ij} \tag{18}$$

It is worth mentioning that we can also adopt other strategies to infer labels from the the predicted scores, such as incorporating the prior distribution of class labels [62] or learning the thresholds for final decision, and better results may be achieved in the situation when multiple classes are unbalanced [47].

3.5 Computational Cost

As we need to perform a sparse representation for each sample, the computational cost of hypergraph construction process scales as $O(dk^2n)$, where d is the dimensionality of features, k is neighborhood size, and n is the number of samples. After that, we implement the alternating optimization process. We can find that there are two steps in the alternating optimization. In the first step, the classification scores **F** could be computed directly from equation (14) and the computational cost scales as $O(n^3)$. For the second step that optimizes the hypergraph weights **w** in equation (17), the optimization could be solved with computational complexity $O(n^3)$ by an interior-point algorithm [6], where n is the number of vertices. Hence, the computational complexity of our proposal is $O(dk^2n + K_l n^3)$, where K_l is the number of iterations in the alternating optimization. This complexity is on par with many existing graph-based and hypergraph-based methods, such as [52] and [58].

4 Experiments

To demonstrate the effectiveness of the proposed approach, we conduct experiments on six datasets which are popularly used in classification experiments, i.e., face dataset ORL32 [1], Columbia Object Image Library COIL20 [41], Caltech20 (a subset of image object dataset Caltech 256 [18]), MNIST2000 (a subset of handwritten digital image dataset MNIST [29]), 20NewsGroup [28], and spoken letter dataset Isolet [13]. We compare the performance of the proposed approach with six methods, i.e., the KNN hypergraph classification, Adaptive hypergraph classification,

Fig. 1. Some samples in the six dataset. From top to bottom: ORL32, COIL20, Caltech2000, MNIST 2000.

k-nearest neighbors, graph-based classification [2], SVM [9], and Laplacian SVM [38].

4.1 Datasets

Our proposal is validated on six datasets which are popularly used in classification experiments. The details of the dataset are as follows.

The ORL32 [1] is a face dataset which is collected by Cambridge University Computer Laboratory. There are 40 different people in this dataset, while for each object, ten images are taken at different times, varying the lighting, facial expressions and facial details. Each of the images is in size of 32 $\times$ 32 and concatenated as a 1,024D feature vector.

The MNIST dataset [29] is a large-scale handwritten digital images dataset. For each digit, we randomly select 200 images and build a subset. All the digital images have been size-normalized and centered in a fixed-size image(28$\times$28), and thus each of the images could be represented by a 728D feature vector.

The COIL20 dataset [41] contains 1,440 images in 20 object categories. For each object, 72 images are captured in a wide variety of complex geometric, appearance and reflectance characteristics. All the images are also normalized in 32$\times$32 size and represented by a 1,024D feature vector.

The Caltech20 dataset is a subset of Caltech 256 dataset [18] which contains 20 objects (i.e., ak47, american-flag, backpack, baseball-bat, baseball-glove). There are 2,548 images in total. We use the locality-constrained linear coding method [51] to represent each of them as a 2,1504D feature vector.

The 20 Newsgroups dataset [28] is a textual collection of approximately 20,000 newsgroup documents, partitioned evenly across 20 different newsgroups. In this work, 2,000 documents randomly sampled from the original subset are exploited. In this dataset, all the documents are formatted by TF-IDF model [37] with the dictionary size 61,188.

ISOLET [13] is a database of letters of the English alphabet spoken in isolation. The whole database consists of 7,800 spoken letters, two productions of each letter by 150 people. The speakers are grouped into sets of 30 speakers each, and they are referred to as isolet1 through isolet5. The first series isolet1 is used in our experiments. In this subset, each record is represented by spectral coefficients, contour and sonorant features [12].

Several examples from the above four images datasets are demonstrated in Fig. 1 for illustration. It is worth noting that 20 Newsgroups and ISOLET are not visual datasets and they are exploited in our work to further evaluate the effectiveness of the proposed method in different domains. Detailed statistical characteristics of different datasets are listed in Table 2.

TABLE 2
Details of the Six Datasets

Name	Feature Dimension	#classes	#samples
ORL32	1,024	40	400
COIL20	1,024	20	1,440
Caltech20	21,504	20	2,548
MNIST	784	10	2,000
20NewsGroup	61,188	20	2,000
ISOLET	617	30	1,560

4.2 Experimental Configuration

For the visual classification tasks, we compare the performance of the following methods.

1) ℓ_1-hypergraph classification (denoted as "ℓ_1-HG"). The parameters λ and μ are tuned by 5-fold cross validation. The neighborhood size k varies in {5, 10, 15, 20} in the hyperedge generation step and the iteration number of the alternating optimization process is set to 10.
2) Adaptive hypergraph classification [58] (denoted as "Adaptive-HG"). The KNN method is employed to build the hypergraph, and regular 0-1 representation is used to indicate whether a vertex belongs to a hyperedge, and hyperedge weights are optimized by l_2 regularization. The parameter settings are the same as the ℓ_1-hypergraph classification. More details can be found in [58].
3) KNN-hypergraph classification [23] (denoted as "KNN-HG"). As detailed in Algorithm 2, the method is similar to adaptive hypergraph classification, but the relationship of each vertex and a hyperedge is defined in a probabilistic way. The weight of a hyperedge is computed from the incidence matrix directly $\mathbf{w}_e = \sum_{v\in e}\mathbf{H}(v,e)$. The parameters settings are the same as the ℓ_1-hypergraph classification.
4) Pair-wise graph-based classification (denoted as "Graph"). As detailed in Algorithm 3, a normal graph is constructed according to the similarity matrix of the objects. Graph Laplacian is computed based on the affinity and vertex degree matrices, and classification scores are predicted accordingly.
5) Support Vector Machine(denoted as "SVM"). SVM [9] is one of the most effective supervised classification methods. Given a set of training examples, SVM builds a model that assigns new examples to one category, based on the "margin maximization" strategy. We adopt radius kernel and the involved parameters are also tuned by five-fold validation.
6) Laplacian SVM(denoted as "LapSVM"). LapSVM is a semi-supervised learning method, which is extended from the standard SVM by adding a smoothness penalty term to represent both labeled and unlabeled data with the graph Laplacian. The involved parameters are tuned by five-fold validation.

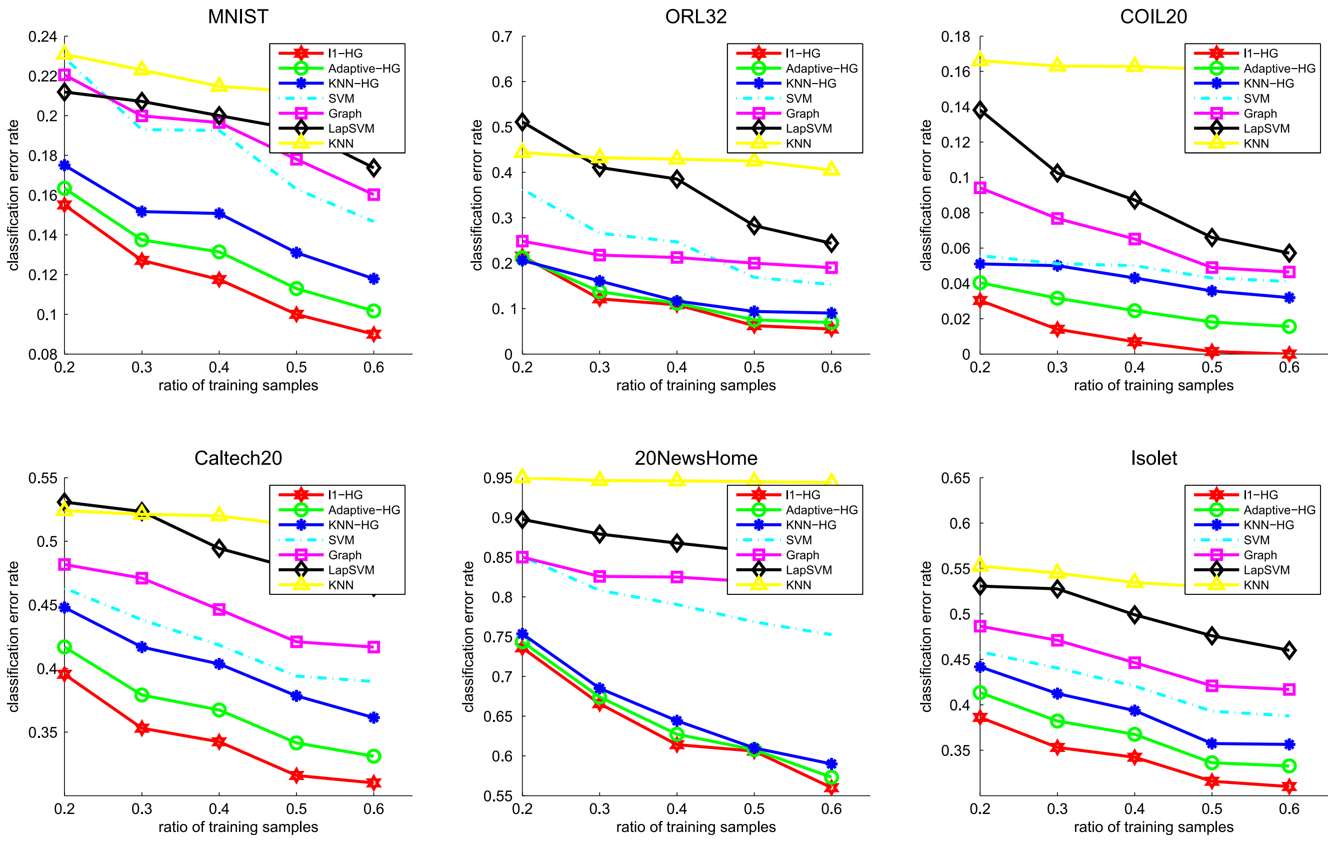

Fig. 2. Classification performance comparison of the seven methods. ℓ_1-**HG**: the proposed ℓ_1-hypergraph classification; **Adaptive-HG**: adaptive hypergraph classification [58]; **KNN-HG**: KNN Hypergraph classification; **SVM**: SVM classification. **Graph**: pair-wise graph-based classification; **LapSVM**: Laplacian SVM classification. **KNN**: k-nearest neighbors classification.

7) k-nearest neighbors classification (denoted as "KNN"), which simply classifies a sample by a majority voting over its neighbors' classes.

Among the above approaches, the hypergraph-based and graph-based methods are all transductive as they only infer the labels of a given dataset. The rest three methods are inductive as they result in a discrimination rule that is able to predict the labels of out-of-sample data.

Algorithm 2. KNN Hypergraph Classification

```
1: For each vertex, find its $k$ nearest neighbors to build a hyperedge.
2: Compute the hypergraph incidence matrix $\mathbf{H}$ and hypergraph Laplacian $\mathbf{L}$.
3: Compute $\mathbf{w}$ by $w(e_i) = \sum_{v_j \in e_i} A(i,j)$.
4: Optimize $\mathbf{F}$ according to eq. (14).
5: Label the testing data according to eq. (18).
```

In our experiments, each dataset is randomly split into two non-overlapping subsets, one used for training and the other for testing. Specifically, for each dataset, we randomly select different percentages of samples as the training data and the left ones are used for testing purpose. For all the classification methods, we independently repeat the experiments five times with randomly selected training samples and report the average results. Classification error rate is used as the performance evaluation metric.

Algorithm 3. Pair-Wise Graph-Based Classification

```
1: Compute the similarity matrix $Sim$ of samples.
2: Construct a simple graph $G_s$ according to the similarity matrix. For each vertex, connect it with its $k$ nearest neighbors.
3: Compute the simple graph affinity matrix $A_g$: $A_g(i,j) = Sim(i,j)$ if the $i$-th and $j$-th vertices are connected, otherwise $A_g(i,j) = 0$.
4: Compute the vertex degree matrix $D = \sum_j A_g(i,j)$.
5: Compute the graph Laplacian $\Delta_g = I - \Theta_g = I - D^{-1/2} A_g D^{-1/2}$.
6: Optimize $\mathbf{F}$ according to eq. (14).
7: Label the testing data according to eq. (18).
```

4.3 Experimental Results

4.3.1 Classification Accuracy Comparison

First, we compare the classification accuracies of different methods. We vary the percentages of training samples and demonstrate the classification results in Fig. 2. We can see that the proposed ℓ_1-hypergraph method outperforms the other methods in all six datasets. For example, in MNIST data, when half data are used for training, the error rate of our classification approach is 0.1 while the error rates of the other methods are 0.14 (Adaptive Hypergraph), 0.18 (SVM), 0.20 (LapSVM), and 0.23 (KNN), respectively. The conclusion can be drawn from other datasets as well. Our method, which not only takes

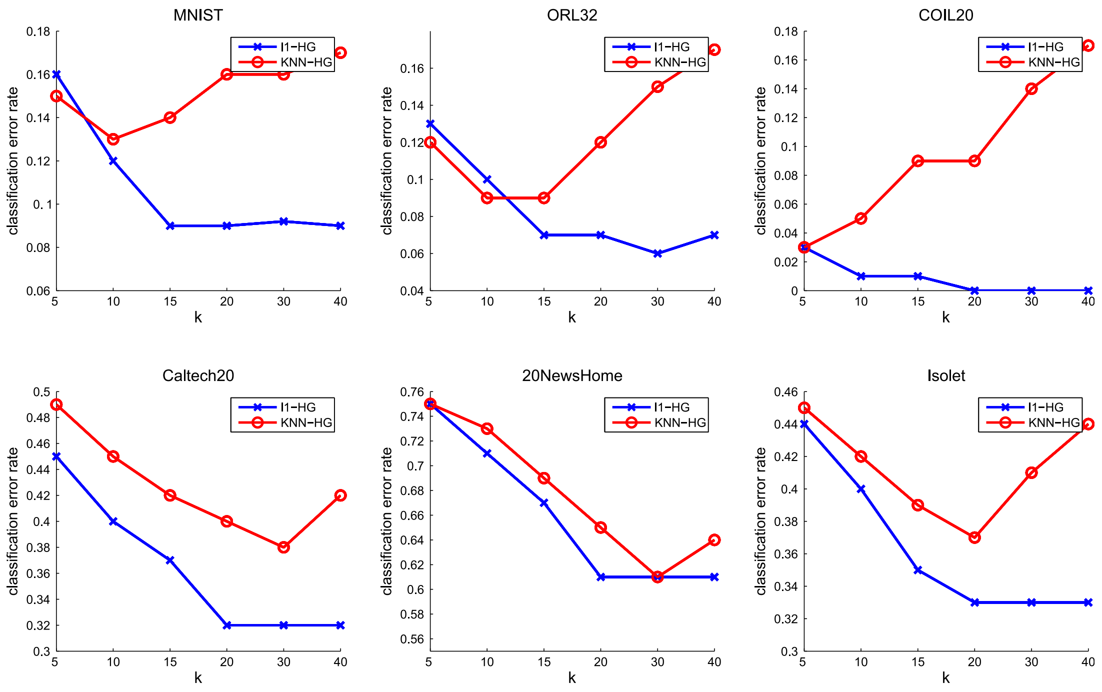

Fig. 3. Performance of the proposed method and normal hypergraph classification under different neighborhood. l_1**-HG**: the proposed l_1 hypergraph classification; **KNN-HG**: the KNN Hypergraph classification.

advantage of probabilistic hypergraph constructed by sparse representation but also optimizes hyperedge weights by Laplacian regularization, achieves the best performance among the compared methods. It demonstrates the effectiveness and feasibility of our approach.

4.3.2 *On Parameter k*

In the proposed approach, a hyperedge is constructed in two steps. First, k nearest samples are found for each sample, and then the sparse representation algorithm is performed on the neighbor set to further refine the samples. For each dataset, we use half of the data for training and the results are reported in Fig. 3. We also demonstrate the results of KNN hypergraph model which does not employ sparse representation in hyperedge construction. From the figure, we can see that in most cases the error rate curves of the KNN hypergraph method exhibit a "$\vee$" shape. As mentioned before, this is due to the fact that, a small k results in a too sparse incidence matrix and a large k will lead to many irrelevant items connected by a hyperedge. We can see that the best value of k varies across datasets for the KNN hypergraph method. For example, $k = 10$ fits the MNIST dataset well, while $k = 30$ is the best for the Caltech20 and 20NewsHome datasets. On the contrary, we can see that the classification error rate of the proposed ℓ_1 hypergraph method degrades with the increasing of the k and then it quickly becomes fairly stable. This is because the sparse representation step has a sample selection effect, which thus makes the parameter k insensitive.

4.3.3 *On Parameter γ*

In our proposed approach, sparse representation is employed to construct the hypergraph (see Section 3.2). Clearly, the parameter γ in equation (4) plays an important role in the computation of representation coefficient. For example, if γ is too small, the equation (4) degrades as the normal ℓ_2 constraint brings lots of noises in constructing the hyperedge. On the contrary, a large γ will lead to a strong sparse constraint and few vertices will be connected by a hyperedge. Therefore, here we test the sensitivity of this parameter. In our experiments, we vary the value of γ in $\{2^{-2}, 2^{-1}, 1, 2, 2^2, 2^3\}$, and we report the results in Fig. 4. From the figure, we can see that, for most datasets, the best performance is obtained when $\gamma = 1$ or $\gamma = 2$. The classification performance of the proposed method will not severely degrade when its value varies in a fairly wide range.

4.3.4 *On the Effect of Hyperedge Weighting*

Here we investigate the impact of the hyperedge weighting approach and compare the classification results of ℓ_1 hypergraph learning with and without hyperedge weighting. In the experiment, we follow the same experimental settings of Section 4.3.1. The results are demonstrated in Fig. 5. From the results we can observe that, for all datasets, the approach with hyperedge weighting yields better performance than that without hyperedge weighting. So, the results demonstrates the effectiveness of our hyperedge weighting method.

5 Conclusion

In this work, we address the hypergraph construction problem and propose a method to build a hypergraph with a sparse representation strategy. Different from the previous work where the hypergraph is built by attribute-based or neighborhood-based methods, our approach employs a learning-based method to build the hypergraph.

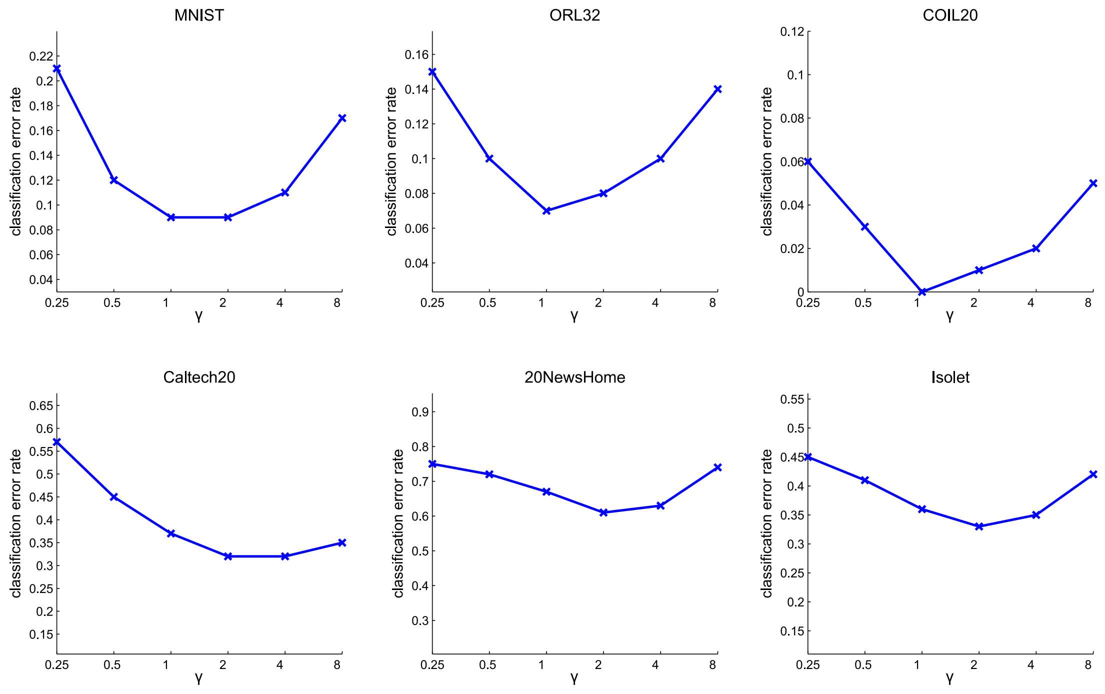

Fig. 4. The variation of classification performance with respect to the parameter γ.

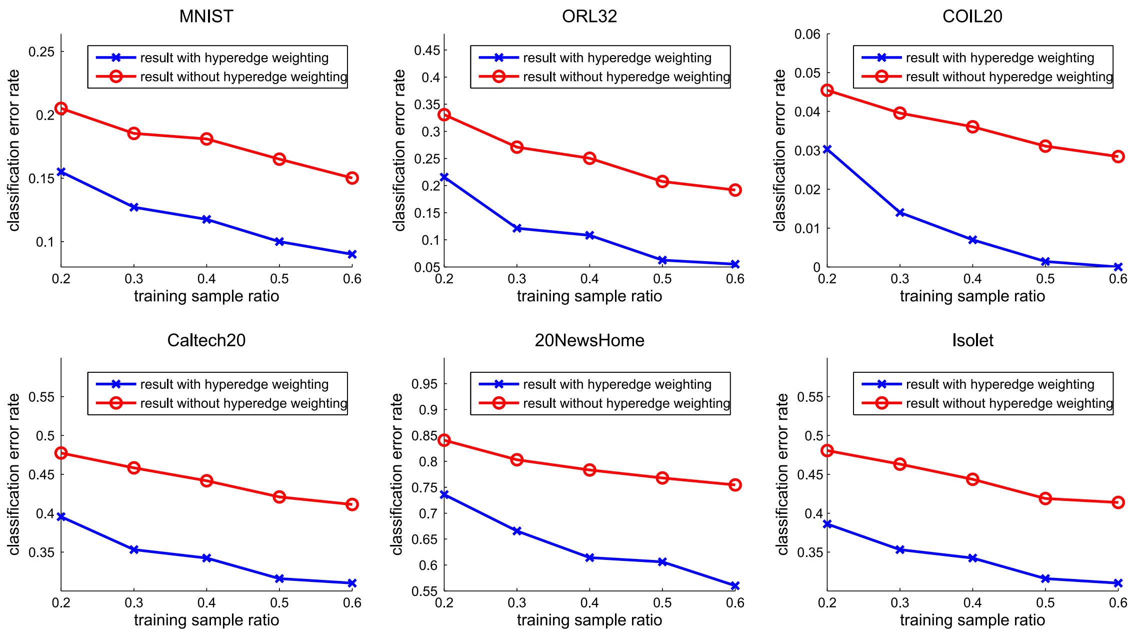

Fig. 5. The performance comparison of ℓ_1 hypergraph learning with and without hyperedge weighting.

Specifically, we first reconstruct each sample by its nearest neighbors via sparse representation. Then, the vertices with zero coefficients are removed from the corresponding hyperedge. We use the representation coefficients to determine the incidence matrix of the hypergraph. In addition, we also introduce a novel method to learn the weights of hyperedges, based on the assumption that the weights of two hyperedges sharing more vertices should be close.

To evaluate the performance of the proposed approach, we conduct experiments on six datasets, including a face dataset, two visual object datasets, two digital handwriting datasets, a text dataset, and an audio dataset. Experimental results demonstrate that the proposed method achieves better results in comparison with a lot of existing methods, including KNN, SVM, Laplacian SVM, graph-based classification, and two existing hypergraph learning methods. Our experimental results also demonstrate that the approach is not sensitive to the parameter k, which can be an advantage in practical classification tasks.

Acknowledgments

This work was partially supported by the National 973 Program of China under grants 2013CB329604 and 2014CB347600, the Program for Changjiang Scholars and Innovative Research Team in University (PCSIRT) of the

Ministry of Education, China, under grant IRT13059, and the National Nature Science Foundation of China under grants 61272393, 61322201, and 61432019. Meng Wang is the corresponding author.

References

[1] [Online]. Available: http://www.cl.cam.ac.uk/research/dtg/attarchive/facedatabase.html

[2] S. Agarwal, "Ranking on graph data," in *Proc. Int. Conf. Mach. Learn.*, 2006, pp. 25–32.

[3] S. Yan, Y. Fu, T. S. Huang, B. Cheng, and J. Yang, "Learning with l1-graph analysis," *IEEE Trans. Image Process.*, vol. 19, no. 4, pp. 858–866, Apr. 2010.

[4] O. Boiman, E. Shechtman, and M. Irani, "In defense of nearest-neighbor based image classification," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, Jun. 2008, pp. 1–8.

[5] A. Bosch, A. Zisserman, and X. Muoz, "Image classification using random forests and ferns," in *Proc. IEEE 11th Int. Conf. Comput. Vis.*, Oct. 2007, pp. 1–8.

[6] S. Boyd and L. Vandenberghe, *Convex Optimization*, Cambridge, U.K.: Cambridge Univ. Press, Mar. 2004.

[7] L. Bruzzone, M. Chi, and M. Marconcini, "A novel transductive SVM for semisupervised classification of remote-sensing images," *IEEE Trans. Geosci. Remote Sensing*, vol. 44, no. 11, pp. 3363–3373, Nov. 2006.

[8] J. Bu, S. Tan, C. Chen, C. Wang, H. Wu, L. Zhang, and X. He, "Music recommendation by unified hypergraph: Combining social media information and music content," in *Proc. Int. Conf. Multimedia*, 2010, pp. 391–400.

[9] C. Burges, "A tutorial on support vector machines for pattern recognition," *Data Mining Knowl. Discovery*, vol. 2, no. 2, pp. 121–167, 1998.

[10] X. Chen, Y. Mu, H. Liu, S. Yan, Y. Rui, and T.-S. Chua, "Large-scale multilabel propagation based on efficient sparse graph construction," *ACM Trans. Multimedia Comput. Commun. Appl.*, vol. 10, no. 1, pp. 6:1–6:20, Dec. 2013.

[11] J. Derrac, I. Triguero, S. Garcia, and F. Herrera, "Integrating instance selection, instance weighting, and feature weighting for nearest neighbor classifiers by coevolutionary algorithms," *IEEE Trans. Syst., Man, Cyber., Part B: Cybern.*, vol. 42, no. 5, pp. 1383–1397, Oct. 2012.

[12] M. A. Fanty and R. Cole, "Spoken letter recognition," in *Proc. Adv. Neural Inf. Process. Syst.*, 1990, p. 220.

[13] M. A. Fanty and R. A. Cole, "Spoken letter recognition," in *Proc. Adv. Neural Inf. Process. Syst.*, 1990, pp. 385–390.

[14] J. Feng, X. Yuan, Z. Wang, H. Xu, and S. Yan, "Auto-grouped sparse representation for visual analysis," *IEEE Trans. Image Process.*, vol. 23, no. 12, pp. 5390–5399, Dec. 2014.

[15] S. Gao, I. W.-H. Tsang, and L.-T. Chia, "Sparse representation with kernels," *IEEE Trans. Image Process.*, vol. 22, no. 2, pp. 423–434, Feb. 2013.

[16] Y. Gao, M. Wang, H. Luan, J. Shen, S. Yan, and D. Tao, "Tag-based social image search with visual-text joint hypergraph learning," in *Proc. 19th ACM Int. Conf. Multimedia*, 2011, pp. 1517–1520.

[17] Geetika, "Article: A survey of classification methods and its applications," *Int. J. Comput. Appl.*, vol. 53, no. 17, pp. 14–16, Sep. 2012.

[18] G. Griffin, A. Holub, and P. Perona, "Caltech-256 object category dataset," California Inst. Technol., Pasadena, CA, USA, Tech. Rep. CNS-TR-2007-001, 2007.

[19] S. Gupta, J. Kim, K. Grauman, and R. J. Mooney, "Watch, listen & learn: Co-training on captioned images and videos," in *Proc. Eur. Conf. Mach. Learn. Principles Practice Knowl. Discovery Databases*, Sep. 2008, pp. 457–472.

[20] C. Hernandez and G. Navarro, "Compressed representations for web and social graphs," *Knowl. Inf. Syst.*, vol. 40, no. 2, pp. 279–313, 2014.

[21] C. Hou, F. Nie, D. Yi, and Y. Wu, "Efficient image classification via multiple rank regression," *IEEE Trans. Image Process.*, vol. 22, no. 1, pp. 340–352, Jan. 2013.

[22] T. Hu, C. Liu, Y. Tang, J. Sun, H. Xiong, and S. Y. Sung, "High-dimensional clustering: A clique-based hypergraph partitioning framework," *Knowl. Inf. Syst.*, vol. 39, no. 1, pp. 61–88, 2014.

[23] Y. Huang, Q. Liu, S. Zhang, and D. N. Metaxas, "Image retrieval via probabilistic hypergraph ranking," in *Proc. Internal Conf. Comput. Vis. Pattern Recog.*, 2010, pp. 3376–3383.

[24] T. Hwango, Z. Tian, R. Kuangy, and J. P. Kocher, "Learning on weighted hypergraphs to integrate protein interactions and gene expressions for cancer outcome prediction," in *Proc. 8th Int. Conf. Data Mining*, 2008, pp. 293–302.

[25] T. Jin, Z. Yu, L. Li, and C. Li, "Multiple graph regularized sparse coding and multiple hypergraph regularized sparse coding for image representation," *Neurocomputing*, vol. 154, pp. 245–256, 2014.

[26] P. Kazienko, K. Musial, and T. Kajdanowicz, "Multidimensional social network in the social recommender system," *IEEE Trans. Syst., Man Cybern., Part A: Syst. Humans*, vol. 41, no. 4, pp. 746–759, Jul. 2011.

[27] X. Kong, M. K. Ng, and Z.-H. Zhou, "Transductive multilabel learning via label set propagation," *IEEE Trans. Knowl. Data Eng.*, vol. 25, no. 3, pp. 704–719, Mar. 2013.

[28] K. Lang, "Newsweeder: Learning to filter netnews," in *Proc. 12th Int. Conf. Mach. Learn.*, 1995, pp. 331–339.

[29] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proc. IEEE*, vol. 86, no. 11, pp. 2278–2324, Nov. 1998.

[30] H. Lee-Kwang and C.-H. Cho, "Hierarchical reduction and partition of hypergraph," *IEEE Trans. Syst., Man, Cybern., Part B: Cybern.*, vol. 26, no. 2, pp. 340–344, Apr. 1996.

[31] P. Li, M. Wang, J. Cheng, C. Xu, and H. Lu, "Spectral hashing with semantically consistent graph for image indexing," *IEEE Trans. Multimedia*, vol. 15, no. 1, pp. 141–152, Jan. 2013.

[32] P. Li, Q. Wang, W. Zuo, and L. Zhang, "Log-euclidean kernels for sparse representation and dictionary learning," in *Proc. IEEE Int. Conf. Comput. Vis.*, Dec. 2013, pp. 1601–1608.

[33] S. Liu, H. Yi, L.-T. Chia, and D. Rajan, "Adaptive hierarchical multi-class SVM classifier for texture-based image classification," in *Proc. IEEE Int. Conf. Multimedia Expo*, Jul. 2005, pp. 1–4.

[34] Y.-F. Liu, J.-M. Guo, and J.-D. Lee, "Halftone image classification using LMS algorithm and naive Bayes," *IEEE Trans. Image Process.*, vol. 20, no. 10, pp. 2837–2847, Oct. 2011.

[35] D. Lu and Q. Weng, "A survey of image classification methods and techniques for improving classification performance," *Int. J. Remote Sensing*, vol. 28, no. 5, pp. 823–870, 2007.

[36] D. Lu and Q. Weng, "A survey of image classification methods and techniques for improving classification performance," *Int. J. Remote Sensing*, vol. 28, no. 5, pp. 823–870, Jan. 2007.

[37] C. D. Manning, P. Raghavan, and H. Schütze, *Introduction Information Retrieval*, 1st ed. Cambridge, U.K.: Cambridge Univ. Press, Jul. 2008.

[38] S. Melacci and M. Belkin, "Laplacian support vector machines trained in the primal," *J. Mach. Learn. Res.*, vol. 12, pp. 1149–1184, Mar. 2011.

[39] A. Morales-Gonzlez, N. Acosta-Mendoza, A. Gago-Alonso, E. B. Garca-Reyes, and J. E. Medina-Pagola, "A new proposal for graph-based image classification using frequent approximate subgraphs," *Pattern Recog.*, vol. 47, no. 1, pp. 169–177, 2014.

[40] R. Moskovitch and Y. Shahar, "Effective data summarization for hierarchical clustering in large datasets," *Knowl. Inf. Syst.*, vol. 42, no. 1, pp. 21–48, 2015.

[41] S. A. Nene, S. K. Nayar, and H. Murase, "Columbia object image library (coil-20)," Columbia Univ., New York, NY, USA, Tech. Rep. CUCS-005-96, Feb. 1996.

[42] B. Ni, S. Yan, and A. Kassim, "Learning a propagable graph for semisupervised learning: Classification and regression," *IEEE Trans. Knowl. Data Eng.*, vol. 24, no. 1, pp. 114–126, Jan. 2012.

[43] F. Perronnin and C. Dance, "Fisher kernels on visual vocabularies for image categorization," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, Jun. 2007, pp. 1–8.

[44] J. Shen, D. Tao, and X. Li, "Modality mixture projections for semantic video event detection," *IEEE Trans. Cybern.*, vol. 18, no. 11, pp. 1587–1596, Nov. 2008.

[45] J. Tang, X.-S. Hua, M. Wang, Z. Gu, G.-J. Qi, and X. Wu, "Correlative linear neighborhood propagation for video annotation," *IEEE Trans. Syst., Man, Cybern., Part B: Cybern.*, vol. 39, no. 2, pp. 409–416, Apr. 2009.

[46] T. Tassa and D. Cohen, "Anonymization of centralized and distributed social networks by sequential clustering," *IEEE Trans. Knowl. Data Eng.*, vol. 25, no. 2, pp. 311–324, Feb. 2013.

[47] B. C. Wallace and I. J. Dahabreh, "Improving class probability estimates for imbalanced data," *Knowl. Inf. Syst.*, vol. 41, no. 1, pp. 33–52, 2014.

[48] F. Wang and C. Zhang, "Label propagation through linear neighborhoods," *IEEE Trans. Knowl. Data Eng.*, vol. 20, no. 1, pp. 55–67, Jan. 2008.

[49] J. Wang, F. Wang, C. Zhang, H. Shen, and L. Quan, "Linear neighborhood propagation and its applications," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 31, no. 9, pp. 1600–1615, Sep. 2009.

[50] J. Wang, J. Wang, G. Zeng, R. Gan, S. Li, and B. Guo, "Fast neighborhood graph search using cartesian concatenation," in *Proc. Int. Conf. Comput. Vis.*, 2013, pp. 2128–2135.

[51] J. Wang, J. Yang, K. Yu, F. Lv, T. Huang, and Y. Gong, "Locality-constrained linear coding for image classification," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2010, pp. 3360–3367.

[52] M. Wang, H. Li, D. Tao, K. Lu, and X. Wu, "Multimodal graph-based reranking for web image search," *IEEE Trans. Image Process.*, vol. 21, no. 11, pp. 4649–4661, Nov. 2012.

[53] A. K. C. Wong, S. Lu, and M. Rioux, "Recognition and shape synthesis of 3-d objects based on attributed hypergraphs," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 11, no. 3, pp. 279–290, Mar. 1989.

[54] J. Wright, A. Yang, A. Ganesh, S. Sastry, and Y. Ma, "Robust face recognition via sparse representation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 31, no. 2, pp. 210–227, Feb. 2009.

[55] S. Xia and E. Hancock, "3d object recognition using hyper-graphs and ranked local invariant features," in *Structural, Syntactic, and Statistical Pattern Recognition*. New York, NY, USA: Springer, 2008, pp. 117–126.

[56] Y. Yang, F. Nie, D. Xu, J. Luo, Y. Zhuang, and Y. Pan, "A multimedia retrieval framework based on semi-supervised ranking and relevance feedback," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 5, no. 34, pp. 723–742, Apr. 2012.

[57] Y. Yang, F. Wu, F. Nie, H. Shen, Y. Zhuang, and A. Hauptmann, "Web and personal image annotation by mining label correlation with relaxed visual graph embedding," *IEEE Trans. Image Process.*, vol. 21, no. 3, pp. 1339–1351, Mar. 2012.

[58] J. Yu, D. Tao, and M. Wang, "Adaptive hypergraph learning and its application in image classification," *IEEE Trans. Image Process.*, vol. 21, no. 7, pp. 3262–3272, Jul. 2012.

[59] K. Yu, T. Zhang, and Y. Gong, "Nonlinear learning using local coordinate coding," in *Proc. Adv. Neural Inf. Process. Syst.*, 2009, pp. 1–9.

[60] X.-T. Yuan, X. Liu, and S. Yan, "Visual classification with multi-task joint sparse representation," *IEEE Trans. Image Process.*, vol. 21, no. 10, pp. 4349–4360, Oct. 2012.

[61] D. Zhou, J. Huang, and B. Schlkopf, "Learning with hypergraphs: Clustering, classification, and embedding," in *Proc. Adv. Neural Inf. Process. Syst.*, 2006, pp. 1601–1608.

[62] X. Zhu, Z. Ghahramani, and J. Lafferty, "Semi-supervised learning using gaussian fields and harmonic functions," in *Proc. 20th Int. Conf. Mach. Learn.*, 2003, pp. 912–919.

Meng Wang received the BE and PhD degrees in the Special Class for the Gifted Young and the Department of Electronic Engineering and Information Science from the University of Science and Technology of China (USTC), Hefei, China, respectively. He is a professor at the Hefei University of Technology, China. His current research interests include multimedia content analysis, search, mining, recommendation, and large-scale computing. He received the best paper awards successively from the 17th and 18th ACM International Conference on Multimedia, the best paper award from the 16th International Multimedia Modeling Conference, the best paper award from the 4th International Conference on Internet Multimedia Computing and Service, and the best demo award from the 20th ACM International Conference on Multimedia. He is a member of the IEEE.

Xueliang Liu received the PhD degree from EURECOM, France. He is an associate professor at the Hefei University of Technology, China. His research interests include multimedia content classification and event detection. He has published more than 20 papers in these areas.

Xindong Wu received the bachelor's and master's degrees in computer science from the Hefei University of Technology, China, and the PhD degree in artificial intelligence from the University of Edinburgh, Britain. He is a Yangtze River Scholar in the School of Computer Science and Information Engineering, Hefei University of Technology, China, a professor of computer science at the University of Vermont. His research interests include data mining, knowledge-based systems, and web information exploration. He is the Steering Committee chair of the IEEE International Conference on Data Mining (ICDM), the editor-in-chief of *Knowledge and Information Systems* (KAIS, by Springer), and a series editor of the Springer Book Series on *Advanced Information and Knowledge Processing* (AI&KP). He was the editor-in-chief of the *IEEE Transactions on Knowledge and Data Engineering* (*TKDE*, by the IEEE Computer Society) between 2005 and 2008. He served as a program committee chair/cochair for the 2003 IEEE International Conference on Data Mining (ICDM '03), the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD-07), and the 19th ACM Conference on Information and Knowledge Management (CIKM 2010). He is a fellow of the IEEE.