Scalable Constrained Spectral Clustering

Jianyuan Li, *Member, IEEE*, Yingjie Xia, *Member, IEEE*, Zhenyu Shan, and Yuncai Liu, *Senior Member, IEEE*

Abstract—Constrained spectral clustering (CSC) algorithms have shown great promise in significantly improving clustering accuracy by encoding side information into spectral clustering algorithms. However, existing CSC algorithms are inefficient in handling moderate and large datasets. In this paper, we aim to develop a scalable and efficient CSC algorithm by integrating sparse coding based graph construction into a framework called *constrained normalized cuts*. To this end, we formulate a scalable constrained normalized-cuts problem and solve it based on a closed-form mathematical analysis. We demonstrate that this problem can be reduced to a generalized eigenvalue problem that can be solved very efficiently. We also describe a principled k-way CSC algorithm for handling moderate and large datasets. Experimental results over benchmark datasets demonstrate that the proposed algorithm is greatly cost-effective, in the sense that (1) with less side information, it can obtain significant improvements in accuracy compared to the unsupervised baseline; (2) with less computational time, it can achieve high clustering accuracies close to those of the state-of-the-art.

Index Terms—Constrained spectral clustering, sparse coding, efficiency, scalability

1 Introduction

CURRENTLY, data in a wide variety of areas tend to large scales. For many traditional learning based data mining algorithms, it is a big challenge to efficiently mine knowledge from the fast increasing data such as information streams, images and even videos. To overcome the challenge, it is important to develop scalable learning algorithms.

Constrained clustering is an important area in the research communities of machine learning. Researchers proposed many new algorithms [1], [2], [3], [4], [5], [6], [7], [8], [9], [10], [11], [12], [13], [14], [15], [16], which improve clustering accuracy by means of encoding *side information* into unsupervised clustering algorithms. Here, *side information* might be labelled data [15], pairwise constraints [2], relative comparison constraints [17], and so forth. In this paper, we refer to it as labelled data. In practice, labelled data are often costly to obtain and so the typical problem in this area is to improve clustering by using *a little of* side information.

We understand that constrained *spectral* clustering (CSC) algorithms [9], [10], [11], [12], [13], [14], [15], [16] are in general better than other constrained clustering algorithms in terms of accuracy, in part because of the high accuracies of unsupervised spectral clustering [18], [19], [20], [21], [22]. However, the scalability and efficiency of previous CSC algorithms are in general poor. Specifically, the memory complexity of existing CSC algorithms [9], [10], [11], [12], [13], [14], [15], [16] is $O(n^2)$ and the time complexity is $O(n^3)$, where n is the total number of instances. This hampers the applications of CSC algorithms towards moderate and large datasets.

In this paper, we develop an efficient and scalable CSC algorithm that can well handle moderate and large datasets. The SCACS algorithm can be understood as a scalable version of the well-designed but less efficient algorithm known as Flexible Constrained Spectral Clustering (FCSC) [15], [16]. To our best knowledge, our algorithm is the first efficient and scalable version in this area, which is derived by an integration of two recent studies, the constrained normalized cuts [15], [16] and the graph construction method based on sparse coding [23]. However, it is by no means straightforward to integrate the two existing methods. In the rest paper, we mainly answer three questions: how do we achieve an effective integration? how do we derive the SCACS algorithm in a principled way? and how well does the proposed algorithm perform?

The structure of the rest paper is as follows: in Section 2, we revisit the constrained normalized-cuts problem and the method of graph construction based on sparse coding; in Section 3 we formulate the problem to be solved and present a closed-form mathematical analysis; in Section 4, we propose our algorithm based on the results of Section 3; in Section 5, we evaluate the proposed algorithm over benchmark datasets and discuss the results; In Section 6, we draw the conclusion and mention the future work.

For notation, vectors and matrices are denoted by bold lower-case and upper-case letters, respectively. Sets are denoted by italic upper-case letters. Scalars are denoted by italic lower-case letters.

2 Background

In this section, we revisit two pioneer studies, namely the constrained normalized cuts [15], [16] and the sparse coding based graph construction method [23].

2.1 Constrained Normalized Cuts

Given a vector dataset $X = \{\mathbf{x}_i\}_{i=1}^n$ where $\mathbf{x}_i \in \mathbb{R}^{\mathbb{d}}$, and a constraint set $\{C_=, C_{\neq}\}$ where $(\mathbf{x}_i, \mathbf{x}_j) \in C_=$ if the patterns of $\mathbf{x}_i$ and $\mathbf{x}_j$ are similar and $(\mathbf{x}_i, \mathbf{x}_j) \in C_{\neq}$ otherwise, the aim is to partition X into k clusters biased by the constraint set. Let $\mathbf{W}$ be the similarity matrix over X where $\mathbf{W}_{ij}$ represents the similarity between instances $\mathbf{x}_i$ and $\mathbf{x}_j$. Let $\mathbf{D}$ be the degree matrix over X which is a diagonal matrix with elements $\mathbf{D}_{ii} = \sum_j \mathbf{W}_{ij}$. Let $\overline{\mathbf{L}} = \mathbf{I} - \mathbf{D}^{-1/2}\mathbf{W}\mathbf{D}^{-1/2}$ be the normalized graph Laplacian where $\mathbf{I}$ denotes the identity matrix. Let $\mathbf{Q}$ denote the *constraint matrix* where $\mathbf{Q}_{ij} = 1$ expresses $(\mathbf{x}_i, \mathbf{x}_j) \in C_=$, and $\mathbf{Q}_{ij} = -1$ expresses $(\mathbf{x}_i, \mathbf{x}_j) \in C_{\neq}$, and $\mathbf{Q}_{ij} = 0$ expresses no available side information. Let $\overline{\mathbf{Q}} = \mathbf{D}^{-1/2}\mathbf{Q}\mathbf{D}^{-1/2}$ be the normalized constraint matrix. The *constrained normalized cuts* [15], [16] can be rewritten as:

$$\begin{aligned} &\arg\min_{\mathbf{v}\in\{+\frac{1}{\sqrt{n}},-\frac{1}{\sqrt{n}}\}^n} \mathbf{v}^T\overline{\mathbf{L}}\mathbf{v} \\ &s.t.\ \ \mathbf{v}^T\overline{\mathbf{Q}}\mathbf{v} \geq \alpha, \mathbf{v}^T\mathbf{v} = 1, \mathbf{v}\perp\mathbf{D}^{1/2}\mathbf{1}. \end{aligned} \tag{1}$$

Here the parameter α controls what degree the input side information is respected. After removed $\mathbf{v}^T\overline{\mathbf{Q}}\mathbf{v} \geq \alpha$, Eq. (1) degenerates to the standard normalized cuts [18]. The problem is NP hard and the feasible way is to allow $\mathbf{v}$ to take any real values [21] which is called a relaxed method in the literature. The relaxed solution vector can be given by generalized eigenvalue decomposition [15], [16]. However, it still requires $O(n^2)$ memory cost and $O(n^3)$ time cost.

2.2 Sparse Coding Based Graph Construction

Graph construction amounts to computing a similarity matrix. There exist a variety of methods [18], [21], [23], [24], [25], [26], [27].

- *J. Li is with the College of Computer Science, Zhejiang University, Hangzhou 310058, China, and the Big Data Research Center of Enjoyor Co Ltd, HangZhou 310030, China. E-mail: jylbob@gmail.com.*
- *Y. Xia is with the College of Computer Science, Zhejiang University, Hangzhou 310058, China. E-mail: xiayingjie@zju.edu.cn.*
- *Z. Shan is with the College of Computer Science, Hangzhou Normal University, Hangzhou 310012, China. E-mail: shanzhenyu@zju.edu.cn.*
- *Y. Liu is with the Department of Automation, Shanghai Jiaotong University, Shanghai 200240, China. E-mail: company8110@gmail.com.*

Manuscript received 1 Aug. 2013; revised 13 June 2014; accepted 18 Aug. 2014. Date of publication 9 Sept. 2014; date of current version 23 Dec. 2014.
Recommended for acceptance by S. Chawla.
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.2014.2356471

Among them, the method in [23] is based on *sparse coding* theory [28] and is designed for handling large datasets.

Here we briefly introduce the sparse coding based graph construction. Given a dataset X, of the form d-by-n matrix, $\mathbf{X}$, sparse coding aims to find a pair of matrices, $\mathbf{U} \in \mathbb{R}^{d\times p}$ and $\mathbf{Z} \in \mathbb{R}^{p\times n}$, such that $\mathbf{UZ}$ could best approximate $\mathbf{X}$ where $\mathbf{U}$'s columns represent the desired base vectors and $\mathbf{Z}$'s columns represent *sparse* coefficient vectors—each vector has few non-zero components. The cost function to be minimized is

$$f(\mathbf{U},\mathbf{Z}) = \|\mathbf{X}-\mathbf{UZ}\|_F^2. \tag{2}$$

Unfortunately, it is costly to precisely solve for $\mathbf{U}^*$ and $\mathbf{Z}^*$[29]. In practice, an approximate and efficient way [23] is to *randomly* choose p instances among the input dataset and act them as base vectors, and then to estimate each column vector of $\mathbf{Z}$ according to

$$\mathbf{Z}_{ij} = \frac{K_\sigma(\mathbf{x}_j,\mathbf{u}_i)}{\sum_{i\in rNB(j)} K_\sigma(\mathbf{x}_j,\mathbf{u}_i)}, \tag{3}$$

where σ denotes the bandwidth of Gaussian kernel $K_\sigma(\cdot,\cdot)$ and $i \in rNB(j)$ means that the base vector $\mathbf{u}_i$ is among the r nearest base (rNB) vectors of instance $\mathbf{x}_j$. The sparseness of the matrix $\mathbf{Z}$ is controlled by the parameter r.

After obtained $\mathbf{Z}$, one can construct two forms of graph matrices: $\mathbf{G} = \mathbf{Z}^T\mathbf{Z}$ and $\mathbf{S} = \mathbf{ZZ}^T$. Choose $\hat{\mathbf{Z}} = \mathbf{D}^{-1/2}\mathbf{Z}$ where $\mathbf{D}_{ii} = \sum_j \mathbf{Z}_{ij}$. We can obtain the normalized graph matrices $\hat{\mathbf{G}} = \hat{\mathbf{Z}}^T\hat{\mathbf{Z}} \in \mathbb{R}^{n\times n}$ and $\hat{\mathbf{S}} = \hat{\mathbf{Z}}\hat{\mathbf{Z}}^T \in \mathbb{R}^{p\times p}$. It is easy to check that the normalized graph Laplacian over X is $(\mathbf{I}-\hat{\mathbf{G}})$. The time complexity for computing $\hat{\mathbf{G}}$ is $O(pn^2)$, but the one for computing $\hat{\mathbf{S}}$ is just $O(np^2)$ ($p \ll n$).

3 Problem Formulation

In this section, we formulate a scalable constrained normalized-cuts problem and demonstrate how to solve it.

3.1 The Scalable Constrained Normalized Cuts

In the following, Problem 1 refers to a straightforward integration of the constrained normalized cuts and the sparse coding based graph construction, and Problem 2 refers to the formulated scalable constrained normalized-cuts problem.

Problem 1. *Let $\hat{\mathbf{G}}$ be the similarity matrix over X. The relaxed constrained normalized-cuts problem can be formulated as*

$$\min_{\mathbf{v}\in\mathbb{R}^n} \mathbf{v}^T\mathbf{L}\mathbf{v}, s.t.\ \ \mathbf{v}^T\mathbf{Q}\mathbf{v} \geq \alpha, \mathbf{v}^T\mathbf{v} = 1, \mathbf{v}\perp\mathbf{1}, \tag{4}$$

where $\mathbf{L} = \mathbf{I} - \hat{\mathbf{G}}$ is the normalized graph Laplacian.

Based on [15], the normal approach for finding the solution vector of Problem 1 can be reduced to the following generalized eigenvalue problem:

$$\mathbf{L}\mathbf{v} = \lambda(\mathbf{Q}-\beta\mathbf{I})\mathbf{v}, \tag{5}$$

where β is a lower bound of α. The time cost for solving this problem is $O(n^3)$, infeasible for handling large datasets.

To seek a scalable solution, we write $\mathbf{v} \in \mathbb{R}^n$ as $\hat{\mathbf{Z}}^T\mathbf{u}$ where $\mathbf{u} \in \mathbb{R}^p$. Plugging $\mathbf{v} = \hat{\mathbf{Z}}^T\mathbf{u}$ into Eq. (4), we reformulate Problem 1 as the following Problem 2.

Problem 2.

$$\min_{\mathbf{u}\in\mathbb{R}^p} \mathbf{u}^T\mathbf{A}\mathbf{u}, s.t.\ \ \mathbf{u}^T\hat{\mathbf{Q}}\mathbf{u} \geq \alpha, \mathbf{u}^T\hat{\mathbf{S}}\mathbf{u} = 1, \mathbf{1}^T\hat{\mathbf{S}}\mathbf{u} = 0, \tag{6}$$

here,

$$\mathbf{A} = \hat{\mathbf{S}} - \hat{\mathbf{S}}\hat{\mathbf{S}}, \hat{\mathbf{Q}} = \hat{\mathbf{Z}}\mathbf{Q}\hat{\mathbf{Z}}^T. \tag{7}$$

This problem is mathematically equivalent to Problem 1, but it results in two significant changes: (1) the n-by-n normalized graph Laplacian $\mathbf{L}$ is compressed as the p-by-p matrix $\mathbf{A}$; (2) the n-by-n constraint matrix $\mathbf{Q}$ is naturally compressed as the p-by-p matrix $\hat{\mathbf{Q}}$. Consequently, the solution of Problem 1 might be efficiently recovered from the solution of Problem 2 considering $p \ll n$.

3.2 Solving Problem 2

To solve Problem 2, we use Lagrange multiplier and obtain that

$$\Lambda(\mathbf{u},\lambda,\mu) = \mathbf{u}^T\mathbf{A}\mathbf{u} - \lambda(\mathbf{u}^T\hat{\mathbf{Q}}\mathbf{u}-\alpha) - \mu(\mathbf{u}^T\hat{\mathbf{S}}\mathbf{u}-1). \tag{8}$$

Using the KKT Theorem [30], the feasible solutions must satisfy the the following conditions:

$$\mathbf{A}\mathbf{u} - \lambda\hat{\mathbf{Q}}\mathbf{u} - \mu\hat{\mathbf{S}}\mathbf{u} = 0 \tag{9}$$

$$\mathbf{u}^T\hat{\mathbf{Q}}\mathbf{u} \geq \alpha \tag{10}$$

$$\mathbf{u}^T\hat{\mathbf{S}}\mathbf{u} = 1, \tag{11}$$

$$\mathbf{1}^T\hat{\mathbf{S}}\mathbf{u} = 0, \tag{12}$$

$$\lambda \geq 0, \tag{13}$$

$$\lambda(\mathbf{u}^T\hat{\mathbf{Q}}\mathbf{u}-\alpha) = 0. \tag{14}$$

When $\lambda = 0$, Eq. (9) degenerates to the standard eigen-system $\hat{\mathbf{S}}\mathbf{u} = (1-\mu)\mathbf{u}$, namely side information does not work. To use side information, we limit $\lambda > 0$ which reduces Eq. (10) and Eq. (14) to

$$\mathbf{u}^T\hat{\mathbf{Q}}\mathbf{u} = \alpha, \tag{15}$$

Assuming that $-\mu/\lambda = \beta$, Eq. (9) can be reduced to

$$\mathbf{A}\mathbf{u} = \lambda(\hat{\mathbf{Q}} - \beta\hat{\mathbf{S}})\mathbf{u}, \tag{16}$$

which is a generalized eigenvalue problem that can be solved very efficiently due to $p \ll n$.

To derive a scalable constrained spectral clustering algorithm (SCACS), it is essential to discuss how to set the parameters β and α. Via a few algebraic manipulations, we find two facts which are useful for algorithm design. First, the matrix $\mathbf{A}$ is positive semi-definite, hence we have

$$\lambda\mathbf{u}^T(\hat{\mathbf{Q}} - \beta\hat{\mathbf{S}})\mathbf{u} = \lambda(\alpha-\beta) \geq 0, \tag{17}$$

meaning that the parameter α is lower-bounded by β. Thus, users just need to specify β, regardless of α. Second, for ensuring that Eq. (16) has at least i meaningful solution vectors in terms of non-negative normalized cuts, $\beta < \gamma_i$ is a sufficient condition where γ_i denotes the ith largest eigenvalue of the generalized eigen-system $\hat{\mathbf{Q}}\mathbf{x} = \gamma\hat{\mathbf{S}}\mathbf{x}$.

4 Algorithm and Analysis

Based on the mathematical analysis above, in this section we derive the algorithm and analyze the complexity.

4.1 The Proposed Algorithm

Problem 2 just indicates a binary constrained spectral clustering problem where the solution vector $\mathbf{v}^*$ plays the role of grouping indicator. Without loss of generality, here we directly derive an algorithm for k-class problems ($k \geq 2$). We call the proposed algorithm *scalable constrained spectral clustering* and list 13 steps in Algorithm 1.

Algorithm 1. Scalable Constrained Spectral Clustering

1: **Input**: a dataset $\mathbf{X} \in \mathbb{R}^{d\times n}$, the base-vector number p, the n-by-n constraint matrix $\mathbf{Q}$, β, cluster number k.
2: Choose p vector data among the input dataset at random, and stack them in the columns of matrix $\mathbf{U} \in \mathbb{R}^{d\times p}$.
3: Compute $\mathbf{Z} \in \mathbb{R}^{p\times n}$ using Eq. (3) and then compute $\hat{\mathbf{Z}} = \mathbf{D}^{-1/2}\mathbf{Z}$ where $\mathbf{D}$ denotes a diagonal matrix with elements $\mathbf{D}_{ii} = \sum_j \mathbf{Z}_{ij}$.
4: Compute $\hat{\mathbf{S}} = \hat{\mathbf{Z}}\hat{\mathbf{Z}}^T$ and $\hat{\mathbf{Q}} = \hat{\mathbf{Z}}\mathbf{Q}\hat{\mathbf{Z}}^T$.
5: Find the largest eigenvalue γ_{max} of the generalized eigen-system $\hat{\mathbf{Q}}\mathbf{x} = \gamma\hat{\mathbf{S}}x$.
6: If $\beta \geq \gamma_{max}$, return $\{\mathbf{v}^*\} = \emptyset$; otherwise, find all the eigenvectors $\{\mathbf{u}_i\}$ by solving generalized eigen-system Eq. (16), where $\mathbf{u}_i$ denotes the ith eigenvector, $1 \leq i \leq p$.
7: Find among $\{\mathbf{u}_i\}$ the eigenvectors $\{\mathbf{u}_i\}^+$ associated with positive eigenvalues.
8: Normalize each $\mathbf{u}_i \in \{\mathbf{u}_i\}^+$ by multiplying a factor $\sqrt{\frac{1}{\mathbf{u}_i^T\hat{\mathbf{S}}\mathbf{u}_i}}$.
9: Remove the eigenvectors from $\{\mathbf{u}_i\}^+$ that are not orthogonal to the vector $\mathbf{1}^T\hat{\mathbf{S}}$.
10: Find among $\{\mathbf{u}_i\}^+$ the m eigenvectors that lead to the smallest values of $\mathbf{u}_i^T\mathbf{A}\mathbf{u}_i$ where $m = \min\{k-1, |\{\mathbf{u}_i\}^+|\}$, then stack them in columns of matrix $\mathbf{V}$.
11: Compute $\mathbf{V}^{(r)} = \hat{\mathbf{Z}}^T\mathbf{V}(\mathbf{I} - \mathbf{V}^T\mathbf{A}\mathbf{V})$.
12: Normalize $\mathbf{V}^{(r)}$'s rows to have unit length, then feed it to the k-means algorithm.
13: **Output**: the grouping indicator.

Several key steps are interpreted as follows: (1) Step 7 aims to satisfy the condition $\lambda > 0$; (2) Step 8 aims to scale each eigenvectors for satisfying the condition of Eq. (11); (3) Step 9 aims to satisfy the condition of Eq. (12); (4) In Step 11, we recover the solution vectors by the linear transformation $\hat{\mathbf{Z}}^T\mathbf{u}$ and we weight each solution vector by one minus the associated value of the objective function.

It is worth mentioning that the input parameter β is tunable, making the algorithm flexible to noisy side information or inappropriate mathematical expressions for side information. Usually, the larger β is given, the more side information is respected.

4.2 Complexity Analysis

The algorithm complexity is analyzed as follows. The time cost for computing the matrix $\hat{\mathbf{S}}$ is in general $O(np^2)$, and that for computing the generalized eigenvalue decomposition in Step 5 or in Step 6 is $O(p^3)$, and that for computing $\hat{\mathbf{Z}}\mathbf{Q}\hat{\mathbf{Z}}^\mathbf{T}$ is $O(kp^2 + kpn)$. Hence the general time complexity of our algorithm is

$$O(kpn + kp^2 + np^2 + p^3). \tag{18}$$

In practical applications, p can be chosen as a value far smaller than n so that the running time grows almost linearly to n. The memory cost for storing the matrices $\hat{\mathbf{Z}}$ and $\hat{\mathbf{S}}$ are $O(np)$ and $O(p^2)$, respectively. The general memory complexity is

$$O(np + p^2). \tag{19}$$

5 Experiment

In this section, we carry out experiments for assessing the proposed SCACS algorithm.

5.1 Experiment Setup

We implement our algorithm and other compared algorithms over Linux machines, the machine for recording computational time with 3.10 GHz CPU and 8 GB main memory.

TABLE 1
Dataset Description

Dataset	# Instances	# Attributes	# Classes
Image Segmentation	2,310	19	7
USPS	9,298	256	10
Pen Digits	10,992	16	10
Letter Recognition	20,000	16	26
MNIST	70,000	784	10
CoverType	581,012	54	7

The datasets that we used are downloaded from UCI machine learning repository[1] and the other two web sites.[2,3] Basic information is listed in Table 1.

Side information is generated based on the ground truth labels of the datasets. The following expression illustrates the encoding rules of $\mathbf{Q}$:

$$\mathbf{Q}_{ij} = \begin{cases} 1 & \text{if } \mathbf{x}_i \text{ and } \mathbf{x}_j \text{ have consistent labels} \\ 1 & \text{if } i = j \\ -1 & \text{if } \mathbf{x}_i \text{ and } \mathbf{x}_j \text{ have different labels} \\ 0 & \text{no side information.} \end{cases} \tag{20}$$

In our experiment, we randomly sample c labelled instances from a given input dataset, and then obtain $\mathbf{Q}$ based on the rules of Eq. (20).

The clustering accuracy is evaluated by the best matching rate (ACC). Let $\mathbf{h}$ be the resulting label vector obtained from a clustering algorithm. Let $\mathbf{g}$ be the ground truth label vector. Then, the best matching rate is defined as

$$ACC = \frac{\sum_{i=1}^{n}\delta(g_i, map(h_i))}{n}, \tag{21}$$

where $\delta(a, b)$ denotes the delta function that returns 1 if $a = b$ and returns 0 otherwise, and map(h_i) is the permutation mapping function that maps each cluster label h_i to the equivalent label from the data corpus.

5.2 Comparisons with Benchmarks and FCSC

In this part, we compare our SCACS algorithm with three spectral algorithms (LSC-R, SL and FCSC). Among them, "LSC-R" [23] is the unsupervised baseline that constructs similarity graphs based on sparse coding. "SL" [10] is the CSC baseline which encodes side information by directly modifying similarity matrices. "FCSC" [16] is the state-of-the-art in terms of accuracy. FCSC is too slow to handle large datasets, hence in this experiment we used four small datasets:

1) The Image Segmentation dataset.
2) A subset sampled from the USPS dataset. We used the first 300 instances of each class, 3,000 instances in total.
3) A subset sampled from the Pen Digits dataset. We used the first 300 instances of each class, 3000 instances in total.
4) A subset sampled from the Letter Recognition dataset. We used the first five classes of instances, 3,864 instances in total.

We set the Gaussian kernel width σ to the average Euclidean distance between all instances and base vectors. We use $p = 500$, $r = 3$ and $\beta = \beta_0\gamma_{k-1}$ where $\beta_0 = 0.5 + 0.4 \times \frac{c}{n}$ and γ_{k-1} denotes the (k-1) th largest eigenvalue of the generalized eigen-system $\hat{\mathbf{Q}}\mathbf{x} = \gamma\hat{\mathbf{S}}\mathbf{x}$. For fair comparison, the randomly selected labelled instances are

1. http://archive.ics.uci.edu/ml
2. http://www.zjucadcg.cn/dengcai
3. http://yann.lecun.com/exdb/mnist

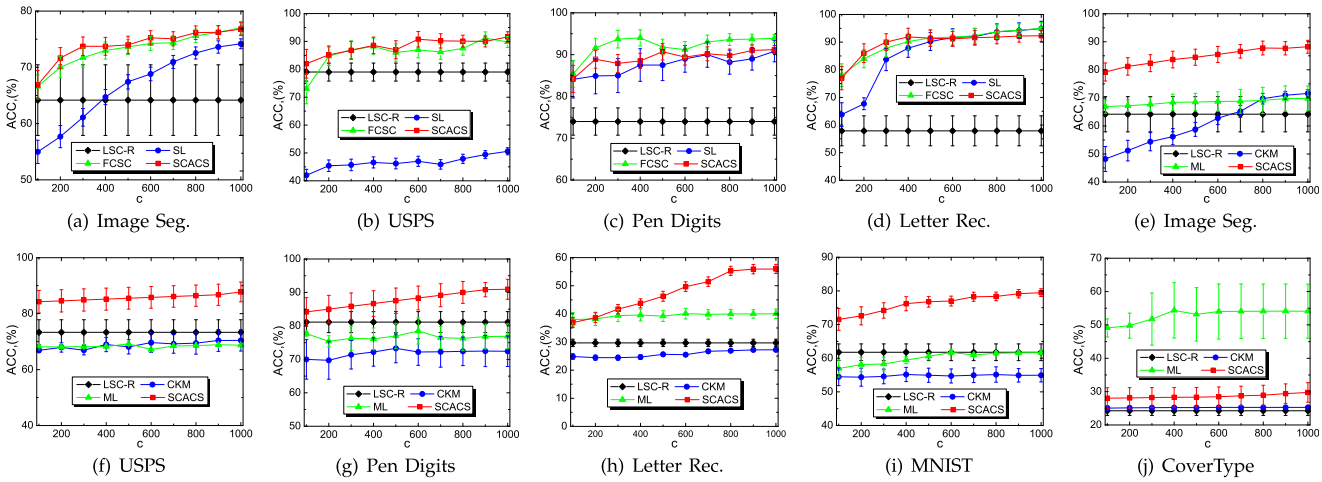


Fig. 1. Accuracy comparisons among different algorithms. Here c denotes the number of labelled data.

consistent for SL, SCACS and FCSC. In order to show how the results vary with c, we use 10 different values ranging from 100 to 1,000 by a increasing step size 100; for each value, we carry out 20 trials in terms of 20 different constraint sets. For SL, we use the k-nearest neighbor method to compute similarity matrices ($k = 30$), and we use radial basis function (RBF) to compute the similarities between pairwise instances. For FCSC, we directly use RBF kernel function to compute similarity matrices as used by the authors.

We demonstrate the algorithm accuracies and the standard deviations in Figs. 1a, 1b, 1c, and 1d. One can see that our algorithm outperforms LSC-R over all the datasets with large margins, indicating that our algorithm in encoding side information is appropriate. In addition, FCSC significantly outperforms SL, indicating that the constrained normalized-cuts framework performs much better in encoding side information. Over the four small datasets, our algorithm outperforms SL on average and is close to FCSC.

5.3 Comparisons with Efficient Algorithms

In this part, we compare our algorithm with three efficient algorithms (LSC-R, CKM, and ML). Among them, "CKM" refers to the k-means based constrained clustering algorithm [2]. "ML" refers to the side information based metric learning method [4] and we use their efficient version that learns a diagonal matrix. The datasets that we used are listed in Table 1.

For fair comparison, we use the same base-vector selections for LSC-R and SCACS, and we use the same random selections of constrained instances for CKM, ML and SCACS. We carry out 20 trials in terms of 20 different constrained instance sets, and report the average clustering accuracies and the standard deviations. The parameter that we used are $p = 500$, $\beta_0 = 0.1$, $r = 3$, respectively.

The results are illustrated in Figs. 1e, 1f, 1g, 1h, 1i, and 1j. Over five datasets (Image Segmentation, USPS, Pen Digits, LetterRec and MNIST), our algorithm significantly outperforms other three algorithms; and over the remaining CoverType dataset, our algorithm performs better than LSC-R and CKM. Note that ML performs well on low-dimensional datasets such as Letter Recognition and CoverType, with a decreased performance on high-dimensional datasets such as USPS and MNIST. However, our algorithm performs much better on high-dimensional datasets.

The running time is recorded in Table 2. The results show that SCACS is much faster than FCSC. For the Pen Digits dataset, FCSC demands more than 11 hours, however, our algorithm demands only 3.22 seconds. For the two largest datasets, MNIST and CoverType, FCSC could not return results within a week, however, our algorithm only demands 12.02 and 64.47 seconds, respectively.

5.4 Influence of Parameters

Below we show how the parameters (r, p, β_0) influence SCACS. We use two datasets, Pen Digits and MNIST, with a lower dimensionality and a higher dimensionality, respectively. For computing the influence of r, we fix $c = 500$ and vary r from 3 to 30 by a step size 1. For computing the influence of p, we fix $c = 500$ and vary p ranging from 100 to 1,000 by a step size 100. For computing the influence of β_0, we vary β_0 from 0.1 to 0.9 by a step size 0.1.

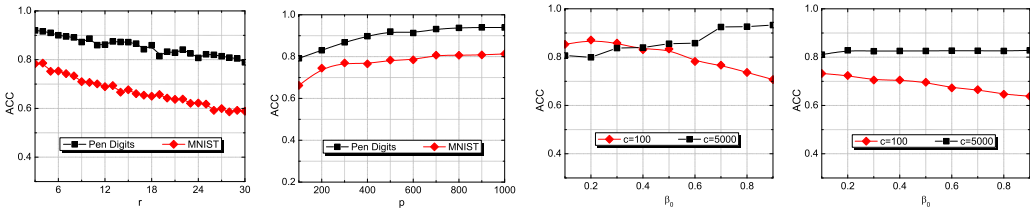
Fig. 2a shows the influence of r. The accuracies of SCACS tend to decrease as r increases, indicating that sparseness plays a key role in our algorithm. This suggests that a relative small r is preferable. The accuracy decrease pace over the MNIST dataset is faster than that over the Pen Digits dataset. This might support that sparseness is more important for grouping high-dimensional datasets than for grouping low-dimensional datasets. Fig. 2b shows that the accuracies of SCACS vary mildly with p; in particular in the interval of, e.g., p ranging from 500 to 1,000, our algorithm is significantly robust. This states that it is easy to set a usable value for p. Figs. 2c and 2d show how the clustering accuracies vary with β_0. One may notice that the clustering accuracies drop seriously if constrained instances are very few (e.g., $c = 100$) but β_0 uses a larger value (e.g., $\beta_0 = 0.8$). Conversely, if constrained instances are abundant, a larger β_0 might result in a significant improvement in terms of accuracy, e.g., for the Pen Digits datasets, the clustering accuracies can be significantly improved as β_0 increases when $c = 5{,}000$.

6 Conclusion and Future Work

We have developed a new k-way scalable constrained spectral clustering algorithm based on a closed-form integration of the constrained normalized cuts and the sparse coding based graph construction. Experimental results show that (1) with less side information, our algorithm can obtain significant improvements in

TABLE 2
Running Time (seconds), $c = 100$, $p = 500$

Dataset	LSC-R	CKM	ML	SCACS	FCSC
ImageSeg	0.41	0.01	3.46	2.35	153
USPS	0.96	0.82	21.05	3.29	14,461
PenDigits	0.82	0.31	5.42	3.22	39,948
LetterRec	4.12	4.10	15.17	6.59	212,600
MNIST	9.32	56.70	492.9	12.02	-
CoverType	49.12	33.31	190.58	64.47	-



(a) Influence of r (b) Influence of p (c) Pen Digits (d) MNIST

Fig. 2. The influence of parameters (r,p,β_0) to the proposed algorithm.

accuracy compared to the unsupervised baseline; (2) with less computational time, our algorithm can obtain high clustering accuracies close to those of the state-of-the-art; (3) It is easy to select the input parameters; (4) our algorithm performs well in grouping high-dimensional image data. In the future, we are considering an active selection of pairwise instances for labelling; we will also apply our algorithm to group urban transportation big data, which might significantly boost sensor placement optimization.

Acknowledgments

This paper was supported in part by the following funds: National High Technology Research and Development Program of China (2011AA010101), National Natural Science Foundation of China (61002009 and 61304188), Key Science and Technology Program of Zhejiang Province of China (2012C01035-1), and Zhejiang Provincial Natural Science Foundation of China (LZ13F020004 and LR14F020003), Postdoctoral fund of China. The authors would like to thank Prof. Xiaofei He for beneficial suggestions. Yingjie Xia is the corresponding author.

References

[1] K. Wagstaff and C. Cardie, "Clustering with instance-level constraints," in *Proc. 17th Int. Conf. Mach. Learn.*, 2000, pp. 1103–1110.
[2] K. Wagstaff, C. Cardie, S. Rogers, and S. Schroedl, "Constrained k-means clustering with background knowledge," in *Proc. 18th Int. Conf. Mach. Learn.*, 2001, pp. 577–584.
[3] S. Basu, A. Banerjee, and R. Mooney, "Semi-supervised clustering by seeding," in *Proc. 19th Int. Conf. Mach. Learn.*, 2002, pp. 27–34.
[4] E. P. Xing, A. Y. Ng, M. I. Jordan, and S. J. Russell, "Distance metric learning with application to clustering with side-information," in *Proc. Adv. Neural Inf. Process. Syst.*, 2003, pp. 505–512.
[5] S. Basu, B. I. Lenko, and R. J. Mooney, "A probabilistic framework for semi-supervised clustering," in *Proc. ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2004, pp. 59–68.
[6] N. Shental, A. Bar-hillel, T. Hertz, and D. Weinshall, "Computing gaussian mixture models with em using equivalence constraints," in *Proc. Adv. Neural Inf. Process. Syst. 16*, 2003, pp. 505–512.
[7] B. Kulis, S. Basu, I. Dhillon, and R. Mooney, "Semi-supervised graph clustering: A kernel approach," in *Proc. 22nd Int. Conf. Mach. Learn.*, 2005, pp. 457–464.
[8] Y.-M. Cheung and H. Zeng, "Semi-supervised maximum margin clustering with pairwise constraints," *IEEE Trans. Knowl. Data Eng.*, vol. 24, no. 5, pp. 926–939, May 2012.
[9] S. X. Yu and J. B. Shi, "Grouping with bias," in *Proc. Adv. Neural Inf. Process. Syst.*, 2001, pp. 1327–1334.
[10] S. D. Kamvar, D. Klein, and C. D. Manning, "Spectral learning," in *Proc. Int. Joint Conf. Artif. Intell.*, 2003, pp. 561–566.
[11] X. Ji and W. Xu, "Document clustering with prior knowledge," in *Proc. 29th Annu. Int. Conf. Res. Develop. Inf. Retrieval*, 2006, pp. 405–412.
[12] Z. Lu and M. A. Carreira-Perpinan, "Constrained spectral clustering through affinity propagation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2008, pp. 1–8.
[13] Z. Li, J. Liu, and X. Tang, "Constrained clustering via spectral regularization," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2009, pp. 421–428.
[14] T. Coleman, J. Saunderson, and A. Wirth, "Spectral clustering with inconsistent advice," in *Proc. 25th Int. Conf. Mach. Learn.*, 2008, pp. 152–159.
[15] X. Wang and I. Davidson, "Flexible constrained spectral clustering," in *Proc. 16th ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2010, pp. 563–572.
[16] X. Wang, B. Qian, and I. Davidson, "On constrained spectral clustering and its applications," *Data Mining Knowl. Discov.*, vol. 28, no. 1, pp. 1–30, 2012.
[17] M. Schultz and T. Joachims, "Learning a distance metric from relative comparisons," in *Proc. Adv. Neural Inf. Process. Syst.*, 2003, pp. 41–48.
[18] J. B. Shi and J. Malik, "Normalized cuts and image segmentation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 22, no. 8, pp. 888–905, Aug. 2000.
[19] A. Y. Ng, M. I. Jordan, and Y. Weiss, "On spectral clustering: Analysis and an algorithm," in *Proc. Adv. Neural Inf. Process. Syst. 14*, 2001, pp. 849–856.
[20] C. Fowlkes, S. Belongie, F. R. K. Chung, and J. Malik, "Spectral grouping using the Nyström method," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 26, no. 2, pp. 214–225, Feb. 2004.
[21] U. V. Luxburg, "A tutorial on spectral clustering," *Statist. Comput.*, vol. 17, no. 4, pp. 395–416, 2007.
[22] Y. Yang, Y. Yang, H. Shen, Y. Zhang, X. Du, and X. Zhou, "Discriminative nonnegative spectral clustering with out-of-sample extension," *IEEE Trans. Knowl. Data Eng.*, vol. 25, no. 8, pp. 1760–1771, Aug. 2013.
[23] X. Chen and D. Cai, "Large scale spectral clustering with landmark-based representation," in *Proc. 25th Conf. Artif. Intell.*, 2011, p. 1.
[24] S. I. Daitch, J. A. Kelner, and D. A. Spielman, "Fitting a graph to vector data," in *Proc. 26th Annu. Int. Conf. Mach. Learn.*, 2009, p. 26.
[25] T. Jebara J. Wang, and S. Chang, "Graph construction and *b*-matching for semi-supervised learning," in *Proc. Int. Conf. Mach. Learn.*, 2009, p. 56.
[26] B. Cheng, J. Yang S. Yan Y. Fu, and T. S. Huang, "Learning with l1-graph for image analysis," *IEEE Trans. Image Process.*, vol. 19, no. 4, pp. 858–866, Apr. 2010.
[27] W. Liu, J. He, and S. Chang, "Large graph construction for scalable semi-supervised learning," in *Proc. Int. Conf. Mach. Learn.*, 2010, pp. 679–686.
[28] B. A. Olshausen, "Emergence of simple-cell receptive field properties by learning a sparse code for natural images," *Nature*, vol. 381, pp. 607–609, 1996.
[29] H. Lee A. Battle R. Raina, and A. Y. Ng, "Efficient sparse coding algorithms," in *Proc. Adv. Neural Inf. Process. Syst. 19*, 2006, pp. 801–808.
[30] H. W. Kuhn and A. W. Tucker, "Nonlinear programming," in *Proc. 2nd Berkeley Symp.*, 1951, pp. 481–492.