Regularized Local Reconstruction for Clustering

Jun Sun[1,2], Zhiyong Shen[1,2], Bai Su[1,2], and Yidong Shen[1]

[1]State Key Laboratory of Computer Science,
Institute of Software, Chinese Academy of Sciences, Beijing 100190, China
[2]Graduate University, Chinese Academy of Sciences, Beijing 100049, China
{junsun,zyshen,subai,ydshen}@ios.ac.cn

Abstract. Motivated by the local reconstruction approach to discovering low dimensional structure in high dimensional data, we propose a novel clustering algorithm that effectively utilizes local reconstruction information. We obtain the local reconstruction weights by minimizing the reconstruction error between each data point and the reconstruction from its neighbors. An entropy regularization term is incorporated into the reconstruction objective function so that the smoothness of the reconstruction weights can be explicitly controlled. The reconstruction weights are then used to obtain the clustering result by employing spectral clustering techniques. Experimental results on a number of datasets demonstrate that our algorithm performs well relative to other approaches, which validate the effectiveness of our approach for clustering.

Keywords: Clustering, Local Reconstruction, Entropy Regularization.

1 Introduction

Data clustering, also known as cluster analysis, is one of the most fundamental problems and an active and important research area in data mining and machine learning. Generally speaking, the goal of clustering is to discover meaningful and natural groupings in data automatically. Usually it involves partitioning a finite set of data objects into several disjoint clusters so that intra-cluster similarity and inter-cluster separability are maximized. Since no labeled data (supervised information) are available to guide the partitioning process, data clustering falls under the category of unsupervised learning.

Various clustering techniques have been proposed over the years. Among these techniques, one of the most interesting approaches is spectral clustering [11], which has received considerable attention in recent years. Spectral clustering algorithms aim to recover the clustering structure in data by exploiting the top eigenvectors of a specially constructed matrix. Spectral clustering algorithms are often easy to implement using linear algebra software packages and can be very efficient if the specially constructed matrix is sparse. Motivated from a graph partitioning perspective, various spectral clustering algorithms have been proposed based on different graph cut objectives such as ratio cut [4] and normalized cut [11]. Spectral clustering techniques are also used in clustering methods which

T. Theeramunkong et al. (Eds.): PAKDD 2009, LNAI 5476, pp. 110–121, 2009.

are based on the local learning idea [13,16]. A lot of applications exist for spectral clustering techniques, such as image segmentation [11,17], circuit layout [4], speech separation [1] and so on.

In this paper, we derive a new clustering algorithm that also utilizes the top eigenvectors of a specially constructed matrix, hence inheriting the advantages of spectral clustering. However, our algorithm is motivated by the local reconstruction approach to dealing with high dimensional data that lie on or near a low dimensional manifold [9]. The local reconstruction approach tries to capture geometric properties of the underlying data manifold by locally reconstructing each data point from its neighbors. The local reconstruction perspective has already proved very useful in dimensionality reduction [8,9] and semi-supervised learning [15].

Based on the local reconstruction perspective, we propose a novel clustering algorithm that effectively utilizes local reconstruction information. Our proposed approach consists of two parts. In the first part, the local reconstruction weights are obtained by minimizing the reconstruction error between each data point and the reconstruction from its neighbors. Important geometric characteristics of local neighborhoods can be preserved by the local reconstruction weights [9]. In the second part of our approach, the reconstruction weights are then used to produce the final clustering result.

In our clustering approach, the reconstruction of each data point is performed in the reproducing kernel Hilbert space. Thus, nonlinear relations between each data point and its neighbors can be captured. Besides, an entropy regularization term is incorporated into the reconstruction objective function so that the smoothness of the reconstruction weights can be explicitly controlled. In the second part of our algorithm, the special structures of the scaled cluster label matrix and the reconstruction weight matrix make the optimization problem much easier to analyze, and spectral clustering techniques can be employed to solve the clustering problem efficiently. Experimental results on a number of real-world datasets demonstrate that our algorithm performs well relative to other spectral clustering approaches, which validate the effectiveness of our approach in obtaining good clusterings.

The remainder of the paper is organized as follows. We introduce the formulation of the proposed clustering model in Section 2. In Section 3, we derive the detailed clustering algorithm. Experimental results are presented in Section 4. Section 5 concludes the paper and discusses future works.

2 Clustering Model

2.1 Notations

First we introduce some notations. Boldface lowercase letters, such as $\mathbf{x}$ and $\mathbf{y}$, denote column vectors. Boldface uppercase letters, such as $\mathbf{M}$ and $\mathbf{W}$, denote matrices. $\mathbf{M}^T$ denotes the transpose of $\mathbf{M}$. $\mathbf{W} \geq 0$ means that every entry in $\mathbf{W}$ is nonnegative. $\mathbf{1}_m \in \mathbb{R}^m$ is a vector of 1's. $\Delta_m = \left\{\mathbf{w} \in \mathbb{R}_m \,|\, \mathbf{w}^T\mathbf{1}_m = 1,\, \mathbf{w} \geq 0\right\}$

is the probability simplex. For any $\mathbf{w} \in \Delta_m$, the elements of $\mathbf{w}$ are nonnegative and sum to one. $\mathbf{I}_m \in \mathbb{R}^{m\times m}$ is the identity matrix of order m. For $\mathbf{x} = [x_1, \ldots, x_m]^T \in \mathbb{R}^m$, $\|\mathbf{x}\|_1 = \sum_{i=1}^m |x_i|$ denotes the ℓ_1 norm. The set of nonnegative real numbers is denoted as $\mathbb{R}_+$. For a square matrix $\mathbf{A} \in \mathbb{R}^{m\times m}$, Trace($\mathbf{A}$) is the trace of $\mathbf{A}$, i.e., the sum of the diagonal elements of $\mathbf{A}$.

2.2 Data and Label Representation

Let $\mathcal{X} \subseteq \mathbb{R}^d$ denote the input space from which n data points, $\mathbf{x}_1, \cdots, \mathbf{x}_n$, where $\mathbf{x}_i = [x_{i1}, \ldots, x_{id}]^T$, are sampled. Given a set of data points $\{\mathbf{x}_i\}_{i=1}^n \subseteq \mathcal{X} \subseteq \mathbb{R}^d$, the goal of data clustering is to find a set of disjoint and exhaustive clusters $\{\pi_l\}_{l=1}^c$ from the data where π_l is the l-th cluster. c is the number of clusters. $|\pi_l|$ is the number of points in the l-th cluster.

Given a positive semi-definite kernel function $K(\cdot,\cdot)$, the data points from the original input space $\mathcal{X}$ are mapped to a possibly infinite dimensional reproducing kernel Hilbert space $\mathcal{F}$ [10]. The mapping is denoted as $\phi : \mathcal{X} \mapsto \mathcal{F}$. The reconstruction of each data point $\mathbf{x}_i$ is then performed in $\mathcal{F}$ so that nonlinear relations between $\mathbf{x}_i$ and its neighbors can be captured. We denote the inner product in $\mathcal{F}$ as $\langle\cdot,\cdot\rangle_{\mathcal{F}}$, so

$$\langle\phi(\mathbf{x}_i), \phi(\mathbf{x}_j)\rangle_{\mathcal{F}} = K(\mathbf{x}_i, \mathbf{x}_j) \qquad \text{and} \qquad \|\phi(\mathbf{x}_i)\|_{\mathcal{F}}^2 = \langle\phi(\mathbf{x}_i), \phi(\mathbf{x}_i)\rangle_{\mathcal{F}} \tag{1}$$

A clustering solution is represented by a *Scaled Cluster Label Matrix* $\mathbf{Y} = [\mathbf{y}_1, \ldots, \mathbf{y}_n]^T = [y_{il}] \in \mathbb{R}^{n\times c}$ where $\mathbf{y}_i = [y_{i1}, y_{i2}, \ldots, y_{ic}]^T \in \mathbb{R}^c$. Specifically,

$$y_{il} = \begin{cases} \frac{1}{|\pi_l|} & \text{if } \mathbf{x}_i \in \pi_l, \\ 0 & \text{if } \mathbf{x}_i \notin \pi_l. \end{cases} \tag{2}$$

$\mathbf{Y}$ is the unknown variable in our clustering model. Once $\mathbf{Y}$ is known, the resultant clustering can be obtained easily. So each data point $\mathbf{x}_i$ is associated with a cluster label $\mathbf{y}_i \in \mathbb{R}^c$. The scaling in $\mathbf{Y}$ is used for producing balanced clusters.

In the following, neighborhood $\mathcal{N}_i$ denotes a set of nearest neighbors of point $\mathbf{x}_i$, not including $\mathbf{x}_i$. The number of neighbors of $\mathbf{x}_i$ is $n_i = |\mathcal{N}_i|$. Usually, $n_i \ll n$.

2.3 Clustering Model

The proposed clustering model consists of two parts, which produce the local reconstruction weights (represented by $\mathbf{W}$) and final clustering ($\mathbf{Y}$) respectively.

In the first part, we try to locally reconstruct each data point $\phi(\mathbf{x}_i)$ from its neighbors. The local reconstruction for data point $\phi(\mathbf{x}_i)$ is $\sum_{\mathbf{x}_j\in\mathcal{N}_i} w_{ij}\phi(\mathbf{x}_j)$ where $w_{ij} \geq 0$ and $\sum_{\mathbf{x}_j\in\mathcal{N}_i} w_{ij} = 1$. All the reconstruction weights form a weight matrix $\mathbf{W} = [w_{ij}] \in \mathbb{R}^{n\times n}$ where $w_{ij} = 0$ if $\mathbf{x}_j \notin \mathcal{N}_i$. Besides, we assume that $\mathbf{w}_i \in \mathbb{R}^{n_i}$ is composed of w_{ij} where $\mathbf{x}_j \in \mathcal{N}_i$, so $\mathbf{w}_i \in \Delta_{n_i}$. The reconstruction error for $\phi(\mathbf{x}_i)$ is denoted by $L_x\left(\phi(\mathbf{x}_i), \sum_{\mathbf{x}_j\in\mathcal{N}_i} w_{ij}\phi(\mathbf{x}_j)\right)$, the value of which is to be minimized. Important geometric characteristics of local neighborhoods

can be preserved by the local reconstruction weights w_{ij}, which in turn can be used to obtain the final clustering result. Here, a natural choice for the error measure L_x is $L_x\left(\phi(\mathbf{x}_i), \sum_{\mathbf{x}_j \in \mathcal{N}_i} w_{ij}\phi(\mathbf{x}_j)\right) = \left\|\phi(\mathbf{x}_i) - \sum_{\mathbf{x}_j \in \mathcal{N}_i} w_{ij}\phi(\mathbf{x}_j)\right\|_{\mathcal{F}}^2$.

In the second part, the reconstruction weights are used to obtain the clustering result. The objective is to minimize the cluster label reconstruction error $\mathcal{E}(\mathbf{Y}) = \sum_{i=1}^{n} L_y\left(\mathbf{y}_i, \sum_{\mathbf{x}_j \in \mathcal{N}_i} w_{ij}\mathbf{y}_j\right)$, which measures how well the cluster label of each data point can be reconstructed from its' neighbors' cluster labels, using the same reconstruction weights obtained in the first part of our clustering model. Here, we use sum of absolute error as the discrepancy measure, namely, $L_y\left(\mathbf{y}_i, \sum_{\mathbf{x}_j \in \mathcal{N}_i} w_{ij}\mathbf{y}_j\right) = \left\|\mathbf{y}_i - \sum_{\mathbf{x}_j \in \mathcal{N}_i} w_{ij}\mathbf{y}_j\right\|_1$.

The two parts of our clustering model are formalized as follows.

1. Compute the local reconstruction weights. For each $\mathbf{x}_i$ where $1 \leq i \leq n$, solve the following problem:

$$\begin{aligned} \min_{\mathbf{w}_i \in \mathbb{R}^{n_i}} \quad & \mathcal{E}_i(\mathbf{w}_i) = \left\|\phi(\mathbf{x}_i) - \sum_{\mathbf{x}_j \in \mathcal{N}_i} w_{ij}\phi(\mathbf{x}_j)\right\|_{\mathcal{F}}^2 - \gamma\, \widehat{H}(\mathbf{w}_i) \\ \text{subject to} \quad & \mathbf{w}_i \in \Delta_{n_i} \end{aligned} \tag{3}$$

2. Compute the label matrix $\mathbf{Y}$ by solving the following problem:

$$\begin{aligned} \min_{\mathbf{Y} \in \mathbb{R}^{n \times c}} \quad & \mathcal{E}(\mathbf{Y}) = \sum_{i=1}^{n} \left\|\mathbf{y}_i - \sum_{\mathbf{x}_j \in \mathcal{N}_i} w_{ij}\mathbf{y}_j\right\|_1 \\ \text{subject to} \quad & \mathbf{Y} \text{ is defined in (2)} \end{aligned} \tag{4}$$

In Eq.(3), $-\widehat{H}(\mathbf{w}_i)$ is the entropy regularization term which explicitly controls the smoothness of the weight components in $\mathbf{w}_i$. $\gamma \geq 0$ is a pre-specified parameter. A larger value of γ means that more uniform weights are preferable. $\widehat{H}(\cdot)$ is the generalized entropy which will be defined and discussed in Section 2.4. A key property of $\widehat{H}(\cdot)$ is that the more uniform the weights in $\mathbf{w}_i$ are, the larger the value of $\widehat{H}(\mathbf{w}_i)$ becomes.

2.4 Generalized Entropy

We introduce the concept of generalized entropy as a measure of the degree of uncertainty within a discrete probability distribution (over a set of m elements) that can be represented by a vector $\mathbf{w} = [w_1, \ldots, w_m]^T \in \Delta_m$. A definition of generalized entropy recently proposed in [7] is provided in the following:

Definition 1. Generalized entropy *is defined as a mapping*

$$\widehat{H} : \Delta_m \mapsto \mathbb{R}_+$$

that satisfies the following two criteria (symmetry and concavity):

1. *For any* $\mathbf{w}_a \in \Delta_m$*, and any* $\mathbf{w}_b \in \Delta_m$ *whose elements are a permutation of the elements of* $\mathbf{w}_a$*,* $\widehat{H}(\mathbf{w}_a) = \widehat{H}(\mathbf{w}_b)$*.*
2. $\widehat{H}(\cdot)$ *is a concave function.*

This definition reflects an important property of generalized entropy $\widehat{H}(\cdot)$: more uniform elements of $\mathbf{w}$ indicate larger values of $\widehat{H}(\mathbf{w})$. For example, if $\mathbf{v}_1 = [0.1\ 0.3\ 0.6]^T \in \Delta_3$ and $\mathbf{v}_2 = [0.3\ 0.3\ 0.4]^T \in \Delta_3$, then $\widehat{H}(\mathbf{v}_1) \leq \widehat{H}(\mathbf{v}_2)$. Many entropies proposed in the literature are special cases of Definition 1 [7]. Some examples are given as follows:

1. $\widehat{H}(\mathbf{w}) = \sum_{i=1}^{m} -w_i \log(w_i)$, which is the *Shannon entropy.*
2. $\widehat{H}(\mathbf{w}) = 1 - \mathbf{w}^T\mathbf{w}$, which will be referred to as ℓ_2*-entropy.*

3 Algorithm

In this section, we derive a clustering algorithm based on the clustering model presented in the previous section. Our algorithm consists of two parts. The first part computes the local reconstruction weight matrix $\mathbf{W}$ by solving the optimization problem (3). The second part computes the final clustering by solving problem (4).

3.1 The Computation of W

The kernel matrix over $\{\mathbf{x}_i\}_{i=1}^n$ is denoted as $\mathbf{K} \in \mathbb{R}^{n\times n}$ so that $K(\mathbf{x}_i, \mathbf{x}_j)$ is the element in the i-th row and j-th column of $\mathbf{K}$. The kernel matrix over $\mathcal{N}_i$ is denoted as $\mathbf{K}_i \in \mathbb{R}^{n_i\times n_i}$, which is a submatrix of $\mathbf{K}$ corresponding to the data points in $\mathcal{N}_i$. If $N_j(\mathbf{x}_i)$ denotes the j-th neighbor of $\mathbf{x}_i$, then the vector $\mathbf{k}_i = [K(\mathbf{x}_i, N_1(\mathbf{x}_i)), K(\mathbf{x}_i, N_2(\mathbf{x}_i)), \ldots, K(\mathbf{x}_i, N_{n_i}(\mathbf{x}_i))]^T \in \mathbb{R}^{n_i}$ is used to denote the kernel function values between data point $\mathbf{x}_i$ and its neighbors.

After some algebraic operations using Eq.(1), problem (3) can be simplified to an equivalent problem as follows.

$$\begin{aligned} \min_{\mathbf{w}_i\in\mathbb{R}^{n_i}} \quad & \mathbf{w}_i^T\mathbf{K}_i\mathbf{w}_i - 2\mathbf{k}_i^T\mathbf{w}_i - \gamma\,\widehat{H}(\mathbf{w}_i) \\ \text{subject to} \quad & \mathbf{w}_i \in \Delta_{n_i} \end{aligned} \tag{5}$$

Even with generalized entropy defined in Section 2.4, this sub-problem for computing $\mathbf{w}_i$ is still a small-scale (since $n_i \ll n$) convex programming problem, since $\widehat{H}(\cdot)$ is a concave function and all the constraints with respect to $\mathbf{w}_i$ are linear. There're very effective and efficient algorithms that can solve convex programs reliably [2]. For the algorithm derived in this section, we'll use $\widehat{H}(\mathbf{w}) = 1 - \mathbf{w}^T\mathbf{w}$ which is the ℓ_2*-entropy.* Problem (3) is then equivalent to the following quadratic programming problem.

$$\begin{aligned} \min_{\mathbf{w}_i\in\mathbb{R}^{n_i}} \quad & \mathbf{w}_i^T\left(\mathbf{K}_i + \gamma\mathbf{I}_{n_i}\right)\mathbf{w}_i - 2\mathbf{k}_i^T\mathbf{w}_i \\ \text{subject to} \quad & \mathbf{w}_i \in \Delta_{n_i} \end{aligned} \tag{6}$$

3.2 The Computation of Y

In this subsection, we want to obtain the final clustering by optimizing problem (4). This objective function appears difficult to optimize because of the constraint (2) and the ℓ_1 norm $\|\cdot\|_1$ in problem (4). However, two key properties of $\mathbf{A}$ in (7) ensure that the problem can be optimized using spectral clustering techniques. The two key properties of $\mathbf{W}$ are

$$\mathbf{W} \geq 0 \qquad \text{and} \qquad \mathbf{W}\mathbf{1}_n = \mathbf{1}_n \tag{7}$$

For simplifying problem (4), we define $\mathbf{H} = [\tilde{\mathbf{h}}_1, \ldots, \tilde{\mathbf{h}}_c] = [h_{il}] \in \mathbb{R}^{n \times c}$ as follows

$$h_{il} = \begin{cases} \sqrt{\frac{1}{|\pi_l|}} & \text{if } \mathbf{x}_i \in \pi_l, \\ 0 & \text{if } \mathbf{x}_i \notin \pi_l. \end{cases} \tag{8}$$

An important property of $\mathbf{H}$ is that $\mathbf{H}^T\mathbf{H} = \mathbf{I}_c$.

Also, we define $\mathbf{A} = \mathbf{W} + \mathbf{W}^T$ and $\mathbf{M} = \mathrm{Deg}(\mathbf{A}) - \mathbf{A}$ where $\mathrm{Deg}(\mathbf{A})$ is the degree matrix of $\mathbf{A}$, i.e., the diagonal matrix whose diagonal elements are the sums of rows of $\mathbf{A}$.

Then the optimization problem in (4) can be proved to be equivalent to the following problem [13]:

$$\min_{\mathbf{H} \in \mathbb{R}^{n \times c}} \mathrm{Trace}\left(\mathbf{H}^T\mathbf{M}\mathbf{H}\right) \qquad \text{s.t. } \mathbf{H} \text{ is defined in (8)} \tag{9}$$

A sketched proof can be found in the Appendix.

Relaxation. As is done in a typical spectral clustering algorithm, $\mathbf{H}$ defined in (8) is relaxed to be any matrix in $\left\{\mathbf{H} \in \mathbb{R}^{n \times c} \,|\, \mathbf{H}^T\mathbf{H} = \mathbf{I}_c\right\}$. The relaxed optimization problem is in the following:

$$\min_{\mathbf{H} \in \mathbb{R}^{n \times c}} \mathrm{Trace}\left(\mathbf{H}^T\mathbf{M}\mathbf{H}\right) \qquad \text{s.t. } \mathbf{H}^T\mathbf{H} = \mathbf{I}_c \tag{10}$$

According to the *Ky Fan* Theorem [18], the globally optimal solution set of problem (10) is as follows:

$$\left\{\mathbf{H}^\star\mathbf{Q} \,|\, \mathbf{Q} \in \mathbb{R}^{c \times c}, \mathbf{Q}^T\mathbf{Q} = \mathbf{I}_c\right\} \tag{11}$$

where $\mathbf{H}^\star \in \mathbb{R}^{n \times c}$ is formed by the eigenvectors corresponding to the c smallest eigenvalues of $\mathbf{M}$.

Discretization. In this step, the solution $\mathbf{H}^\star$ to the relaxed problem (10) needs to be discretized to produce a clustering result. We use the discretization method proposed in [17], which tries to rotate $\mathbf{H}^\star$ so that it's close to a cluster indicator matrix. The details of this method can be found in [17].

Algorithm: RLRC ($\mathbf{K}$, c, k, γ)
Input: Kernel matrix $\mathbf{K} \in \mathbb{R}^{n \times n}$, number of output clusters c, number of nearest neighbors k and parameter γ.
Output: Clustering solution $\{\pi_l\}_{l=1}^{c}$.
Procedure:
1. For each $\mathbf{x}_i$, compute the k nearest neighbors ($\mathcal{N}_i$) based on kernel matrix $\mathbf{K}$.
2. **for** each $1 \leq i \leq n$
   Compute the reconstruction weight vector $\mathbf{w}_i$ by solving the quadratic programming problem (6).
   end for
3. Construct the sparse matrix $\mathbf{W}$ by combining the $\mathbf{w}_i$'s together.
4. Construct the sparse matrix $\mathbf{M} = \mathrm{Deg}(\mathbf{W} + \mathbf{W}^T) - \mathbf{W} - \mathbf{W}^T$.
5. Construct the matrix $\mathbf{H}^\star$ by computing the eigenvectors corresponding to the c smallest eigenvalues of $\mathbf{M}$.
6. Discretize $\mathbf{H}^\star$ to get the clustering solution $\{\pi_l\}_{l=1}^{c}$.
7. **return** $\{\pi_l\}_{l=1}^{c}$;

Fig. 1. RLRC **algorithm**

3.3 Main Algorithm

In practice, the number of neighbors n_i for each data point $\mathbf{x}_i$ is often fixed to a small value $k \ll n$, i.e., $n_i = k$ for all $1 \leq i \leq n$. Thus, each neighborhood $\mathcal{N}_i$ is defined as the set of k-nearest neighbors of $\mathbf{x}_i$, not including $\mathbf{x}_i$, using some distance metric. The distance between $\mathbf{x}_i$ and $\mathbf{x}_j$ is measured by

$$\mathcal{D}\left(\phi(\mathbf{x}_i), \phi(\mathbf{x}_j)\right) = \left\|\phi(\mathbf{x}_i) - \phi(\mathbf{x}_j)\right\|_{\mathcal{F}} = \sqrt{K(\mathbf{x}_i, \mathbf{x}_i) + K(\mathbf{x}_j, \mathbf{x}_j) - 2K(\mathbf{x}_i, \mathbf{x}_j)}$$

Therefore, given a kernel matrix, the neighborhood of each data point can be computed easily. Here, k is provided by domain experts.

The main algorithm is summarized in Fig.1. We name the algorithm Regularized Local Reconstruction for Clustering (RLRC).

3.4 Computational Complexity

The main computational load comes from step 1, 2 and 5. Given a kernel matrix, the k-nearest neighbors of each data point can be computed with complexity $O(n)$. So the overall complexity for step 1 is $O(n^2)$. For step 2, since each quadratic programming problem requires time $O(k^3)$, the overall time complexity for step 2 is $O(nk^3)$. For step 5 which eigen-decomposes a $n \times n$ matrix with $O(nk)$ nonzero elements to obtain the top c eigenvectors, the time complexity is $O(n^2c)$ without special optimizations. Therefore, the time complexity of RLRC algorithm in Fig.1 is $O(nk^3 + n^2c)$. Note that the complexity of step 5 can be reduced to subquadratic in n by adopting more efficient methods for sparse eigen-decomposition problem [5].

Table 1. Summary of datasets

Name	Source	n	d	c
cranmed	CRANFIELD/MEDLINE	2431	41681	2
k1a	WebACE	2340	21839	20
k1b	WebACE	2340	21839	6
re0	Reuters-21578	1504	2886	13
re1	Reuters-21578	1657	3758	25
tr11	TREC	414	6429	9
tr12	TREC	313	5804	8
tr23	TREC	204	5832	6
tr31	TREC	927	10128	7
tr41	TREC	878	7454	10
tr45	TREC	690	8261	10
wap	WebACE	1560	8460	20

4 Experimental Results

In this section, we conduct experiments on a number of real-world datasets to evaluate the effectiveness of our clustering algorithm by comparing its performance with other related methods.

4.1 Summary of Datasets

In this subsection, we will introduce the datasets used in our experiments. We use 12 document datasets[1] from the CLUTO toolkit [19]. Table 1 summarizes the basic information of the datasets.

Dataset *cranmed* is composed of the CRANFIELD and MEDLINE abstracts[2]. The three datasets *k1a*, *k1b* and *wap* are from the WebACE project where each document corresponds to a web page listed in the subject hierarchy of Yahoo!. Datasets *re0* and *re1* are derived from Reuters-21578 text collection [6]. The six datasets *tr11*, *tr12*, *tr23*, *tr31*, *tr41*, and *tr45* are from the TREC collection [14]. The processed datasets are also available from `http://shi-zhong.com/software/docdata.zip`. These datasets provide a good representation of different data distributions and characteristics, and so are good candidates for evaluating different clustering algorithms.

4.2 Clustering Evaluation Criteria

In all the experiments, the "true" number of classes c is provided to all the considered clustering algorithms. Given class labels, we evaluate the clustering results using two external validity measures, Normalized Mutual Information (*NMI*) [12] and Clustering Accuracy (*Acc*) [13,16]. *NMI* is an information-theoretic

[1] `http://glaros.dtc.umn.edu/gkhome/fetch/sw/cluto/datasets.tar.gz`
[2] Available from `ftp://ftp.cs.cornell.edu/pub/smart`

measure that's previously defined in [12] to compare different clusterings. *Acc* [13,16] is calculated based on the one-to-one correspondence between the clusters and the classes. We denote a permutation function as $\sigma(\cdot) : \{i\}_{i=1}^{c} \mapsto \{j\}_{j=1}^{c}$ which maps each cluster index i to a class index $\sigma(i)$. *Acc* is calculated as follows:

$$Acc = \frac{\max\left(\sum_{i=1}^{c} n_{i,\sigma(i)}\right)}{n} \tag{12}$$

where $n_{i,\sigma(i)}$ is the number of points which are in both the i-th cluster and $\sigma(i)$-th class. Note that larger values of *NMI* and *Acc* suggest better clustering solutions.

4.3 Experimental Settings

We normalize each document vector (Bag-of-Words) to unit norm and then the cosine kernel [16] is adopted as the kernel function in all the experiments. We compare the performance of the following clustering algorithms:

- Spectral clustering with normalized cut (NCut) [11]. The symmetric weighted k-nearest neighbor graph is used as the affinity graph. The edge weight between two connected data points in the graph is calculated using the kernel function (cosine similarity).
- Local Learning based Clustering Algorithm[3] (LLCA) proposed in [16]. There's a regularization parameter λ in LLCA. As is done in [16], we choose this parameter from: $\lambda \in \{0.1, 1, 1.5\}$. We report the best performance among the results produced when different values of λ are used, which gives LLCA an unfair advantage over others.
- Clustering via local regression (CLOR) proposed in [13]. This algorithm is also based on the local learning idea as LLCA.
- Our proposed algorithm RLRC. We fix the regularization parameter $\gamma = 1$ to avoid the difficult parameter selection procedure.

All the above algorithms cluster the data by utilizing local neighborhood information (k-nearest neighbors) of the data points. Besides, all of them use spectral clustering techniques to optimize the objective functions. We use the same discretization method[4] for all of them. Note that in all the experiments, the number of classes c is provided to all the clustering algorithms.

4.4 Performance Results

In this subsection, we will provide an empirical study of the considered algorithms. Clustering performance results on the datasets in terms of both *NMI* and *Acc* values will be compared and discussed .

In particular, we aim to address the following two important questions:

[3] The code is available at `www.kyb.tuebingen.mpg.de/bs/people/mingrui/LLCA.zip`
[4] The code is available at `http://www.cis.upenn.edu/~jshi/software/`

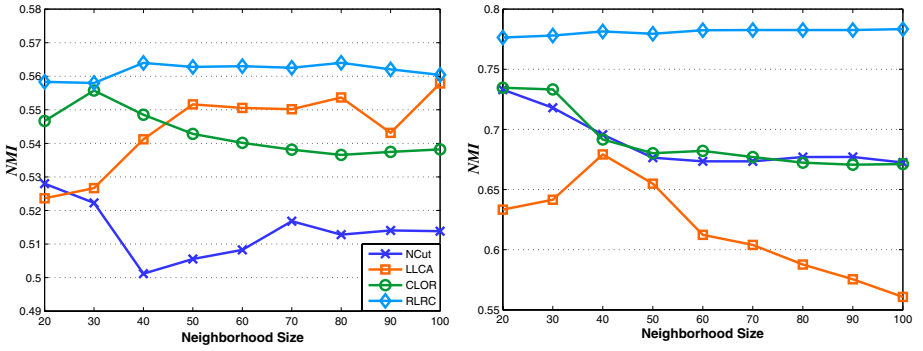
(1) How effective is the proposed algorithm RLRC compared with the other three considered algorithms?
(2) How will the performance of RLRC change with different neighborhood sizes?

Generally, the optimal value of k is not easy to determine. In practice, we choose k on the order of $\log(n)$ so that the k-nearest neighbor graph is asymptotically connected under some special conditions [3]. Therefore, for datasets with size on the order of 1000, we can choose k to be a small multiple of 10. The experimental results on the 12 datasets when neighborhood size $k = 40$ are presented in Table 2. It can be observed that CLOR and the proposed algorithm RLRC achieve the best *NMI* and *Acc* values. On 8 out of 12 datasets, our algorithm RLRC achieves the best *NMI* and *Acc* values. On dataset *k1a*, our algorithm achieves the best *NMI* value but CLOR achieves the best *Acc* value. On dataset *re0* and *re1*, our algorithm achieves the best *Acc* values but CLOR achieves the best *NMI* values. On dataset *tr41*, CLOR outperforms RLRC with respect to both *NMI* and *Acc* values. However, the *NMI* and *Acc* values achieved by our algorithm RLRC are very close to those achieved by CLOR on dataset *tr41*. The clustering results in Table 2 demonstrate the effectiveness and potential of our proposed clustering approach in discovering accurate clusterings.

We also conduct experiments on the considered datasets with different neighborhood sizes k. Figure 2 shows how clustering performance results (*NMI*) vary with different values of k on dataset *k1a* and *k1b*. The horizontal axis denotes the neighborhood size k and the vertical axis denotes the clustering performance measured by *NMI*. With different values of k, our algorithm RLRC achieves more stable results compared with other algorithms. Among the remaining three algorithms, CLOR is more stable than the other two algorithms with respect to different values of k.

Table 2. Experimental results of the considered algorithms on the datasets. Both *NMI* and *Acc* results are listed in the table. For each dataset, the results with highest *NMI* and *Acc* values are shown in boldface. Here, the neighborhood size $k = 40$ for all the considered algorithms.

	NMI				*Acc*			
	NCut	LLCA	CLOR	RLRC	NCut	LLCA	CLOR	RLRC
cranmed	0.8628	0.4163	0.8906	**0.9413**	0.9778	0.8198	0.9835	**0.9930**
k1a	0.5012	0.5412	0.5485	**0.5640**	0.3970	0.4132	**0.4432**	0.4282
k1b	0.6957	0.6792	0.6916	**0.7814**	0.7987	0.8162	0.7368	**0.8688**
re0	0.4005	0.4085	**0.4264**	0.3954	0.3265	0.3883	0.3291	**0.4402**
re1	0.4842	0.4854	**0.4982**	0.4957	0.3621	0.3784	0.3856	**0.4683**
tr11	0.6242	0.6201	0.6740	**0.7234**	0.5725	0.5652	0.6522	**0.7271**
tr12	0.5994	0.6221	0.6568	**0.7373**	0.6454	0.6613	0.7380	**0.7796**
tr23	0.3441	0.2982	0.3343	**0.3574**	0.4118	0.3775	0.4510	**0.4608**
tr31	0.4566	0.4990	0.4832	**0.5341**	0.5491	0.5620	0.5383	**0.6095**
tr41	0.6031	0.6220	**0.6416**	0.6203	0.5809	0.5763	**0.6355**	0.6253
tr45	0.5582	0.5848	0.6311	**0.6641**	0.5783	0.6681	0.6493	**0.6739**
wap	0.5249	0.5420	0.5406	**0.5768**	0.3904	0.4231	0.4199	**0.4795**



(a) *k1a* (b) *k1b*

Fig. 2. The clustering results (*NMI*) by the considered algorithms on datasets *k1a* and *k1b* with different neighborhood sizes (k). The legend is shown only in subfigure (a) for clarity.

5 Conclusion

In this paper, we propose a new clustering algorithm, namely, Regularized Local Reconstruction for Clustering (RLRC), which is motivated by the local reconstruction approach to dealing with high dimensional data that lie on or near a low dimensional manifold [9]. Our clustering method effectively utilizes local reconstruction information which captures important geometric characteristics of local neighborhoods by locally reconstructing each data point from its neighbors. Spectral clustering techniques can be employed to solve the clustering problem efficiently. Experimental results on a number of datasets demonstrate the effectiveness of our proposed algorithm. Future work includes how to select the neighborhood size and the regularization parameter automatically.

Acknowledgments. This work is supported in part by NSFC grants 60673103 and 60721061.

References

1. Bach, F.R., Jordan, M.I.: Learning Spectral Clustering, with Application To Speech Separation. Journal of Machine Learning Research 7, 1963–2001 (2006)
2. Boyd, S., Vandenberghe, V.: Convex Optimization. Cambridge University Press, Cambridge (2004)
3. Brito, M., Chavez, E., Quiroz, A., Yukich, J.: Connectivity of the Mutual K-nearest-neighbor Graph in Clustering and Outlier Detection. Statistics and Probability Letters 35(1), 33–42 (1997)
4. Chan, P.K., Schlag, M.D.F., Zien, J.Y.: Spectral K-way Ratio-cut Partitioning and Clustering. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems 13(9), 1088–1096 (1994)

5. Fokkema, D.R., Sleijpen, G.L.G., van der Vorst, H.A.: Jacobi–Davidson Style QR and QZ Algorithms for the Reduction of Matrix Pencils. SIAM Journal on Scientific Computing 20(1), 94–125 (1998)
6. Lewis, D.D.: Reuters-21578 text categorization test collection
7. Luo, P., Zhan, G., He, Q., Shi, Z., Lü, K.: On defining partition entropy by inequalities. IEEE Transactions on Information Theory 53(9), 3233–3239 (2007)
8. Roweis, S., Saul, L.: Nonlinear Dimensionality Reduction by Locally Linear Embedding. Science 290(5500), 2323–2326 (2000)
9. Saul, L., Roweis, S.: Think Globally, Fit Locally: Unsupervised Learning of Low Dimensional Manifold. Journal of Machine Learning Research 4, 119–155 (2003)
10. Schölkopf, B., Smola, A.: Learning with Kernels. MIT Press, Cambridge (2002)
11. Shi, J., Malik, J.: Normalized Cuts and Image Segmentation. IEEE Transactions on Pattern Analysis and Machine Intelligence 22(8), 888–905 (2000)
12. Strehl, A., Ghosh, J.: Cluster Ensembles — A Knowledge Reuse Framework for Combining Multiple Partitions. Journal of Machine Learning Research 3, 583–617 (2002)
13. Sun, J., Shen, Z., Li, H., Shen, Y.: Clustering via Local Regression. In: Proceedings of the European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases (2008)
14. TREC: Text REtrieval Conference, `http://trec.nist.gov`
15. Wang, F., Zhang, C.: Label Propagation through Linear Neighborhoods. IEEE Transactions on Knowledge and Data Engineering 20(1), 55–67 (2008)
16. Wu, M., Schölkopf, B.: A Local Learning Approach for Clustering. Advances in Neural Information Processing Systems 19 (2006)
17. Yu, S.X., Shi, J.: Multiclass Spectral Clustering. In: Proceedings of the 9th International Conference on Computer Vision (2003)
18. Zha, H., He, X., Ding, C., Gu, M., Simon, H.D.: Spectral Relaxation for K-means Clustering. Advances in Neural Information Processing Systems 14 (2001)
19. Zhao, Y., Karypis, G.: Empirical and Theoretical Comparisons of Selected Criterion Functions for Document Clustering. Machine Learning 55, 311–331 (2004)

Appendix

Proof:

$$\mathcal{E}(\mathbf{Y}) = \sum_{i=1}^{n} \left\| \mathbf{y}_i - \sum_{\mathbf{x}_j \in \mathcal{N}_i} w_{ij}\mathbf{y}_j \right\|_1 = \sum_{l=1}^{c}\sum_{i=1}^{n} \left| y_{il} - \sum_{j=1}^{n} w_{ij} y_{jl} \right| \tag{13}$$

$$= \sum_{l=1}^{c} \left(\sum_{\mathbf{x}_i \in \pi_l} \frac{|1 - \sum_{\mathbf{x}_j \in \pi_l} w_{ij}|}{|\pi_l|} + \sum_{\mathbf{x}_i \notin \pi_l} \frac{|0 - \sum_{\mathbf{x}_j \in \pi_l} w_{ij}|}{|\pi_l|} \right) \tag{14}$$

$$= \sum_{l=1}^{c} \frac{\sum_{\mathbf{x}_i \in \pi_l} \sum_{\mathbf{x}_j \notin \pi_l} (w_{ij} + w_{ji})}{|\pi_l|} = \sum_{l=1}^{c} \tilde{\mathbf{h}}_l^T \left(\mathrm{Deg}(\mathbf{A}) - \mathbf{A} \right) \tilde{\mathbf{h}}_l \tag{15}$$

$$= \sum_{l=1}^{c} \tilde{\mathbf{h}}_l^T \mathbf{M} \tilde{\mathbf{h}}_l = \mathrm{Trace}\left(\mathbf{H}^T \mathbf{M} \mathbf{H} \right) \tag{16}$$