Clustering Via Local Regression

Jun Sun[1,2], Zhiyong Shen[1,2], Hui Li[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}@ios.ac.cn, lihui@is.iscas.ac.cn, ydshen@ios.ac.cn

Abstract. This paper deals with the local learning approach for clustering, which is based on the idea that in a good clustering, the cluster label of each data point can be well predicted based on its neighbors and their cluster labels. We propose a novel local learning based clustering algorithm using kernel regression as the local label predictor. Although sum of absolute error is used instead of sum of squared error, we still obtain an algorithm that clusters the data by exploiting the eigen-structure of a sparse matrix. Experimental results on many data sets demonstrate the effectiveness and potential of the proposed method.

Keywords: Clustering, Local Learning, Sum of Absolute Error.

1 Introduction

Data clustering has been extensively studied and practiced across multiple disciplines for several decades [8]. It aims to group objects, usually represented as data points in $\mathbb{R}^d$, into several clusters in a meaningful way. Generally, the clustering objective is formulated to maximize intra-cluster cohesion and inter-cluster separability. Clustering techniques have been applied to many tasks, such as image segmentation [11], unsupervised document organization, grouping genes and proteins with similar functionality, and so on.

Many clustering algorithms have emerged over the years. One of the most popular clustering methods in recent years is the spectral clustering approach which exploits the eigen-structure of a specially constructed matrix. Generally, spectral clustering can be motivated from a graph partitioning perspective. Various graph clustering objectives, including ratio cut [6], normalized cut [11], and min-max cut [7], can be solved effectively by the spectral clustering method.

In this paper, we propose a clustering method that's also based on eigen-decomposing a matrix as is done in spectral clustering. However, we motivate it from the local learning idea, namely, in a good clustering, the cluster label of each data point can be well predicted based on its neighbors and their cluster labels. First, our method constructs local label predictors for each data point using its neighbors and their cluster labels as the training data. Then it minimizes the discrepancy between the data points' cluster labels and the prediction results of all the local predictors to get a final clustering.

W. Daelemans et al. (Eds.): ECML PKDD 2008, Part II, LNAI 5212, pp. 456–471, 2008.

The local learning idea has already been used in supervised learning [3], transductive classification [20] and dimensionality reduction [21]. In supervised learning, for each test data point, data points in the vicinity of the test point are selected for learning an output function. Then this function is used to predict the label of the test point. In practice, this method can achieve better performance than global learning machines since only data points relevant to the test point are used for training [3].

When the local learning idea is used in an unsupervised manner, a clustering objective is formulated in [19] and achieves good performance. This objective is also combined with a global label smoothness regularizer to obtain a method called "clustering with local and global regularization" in [16,17]. However, their local clustering objective uses sum of squared error to measure the discrepancy between the data points' cluster labels and the prediction results of all the local predictors. In regression analysis, a model using the sum of squared error measure can be sensitive to noise and outliers, since large error entries are overemphasized. To obtain a better model, we use sum of absolute error which is more robust and reliable. Traditionally, a regression model using absolute error will lead to a linear programming optimization problem [26]. However, by combining sum of absolute error and kernel regression as the local label predictors, we still obtain an algorithm that clusters the data by exploiting the eigen-structure of a specially constructed matrix, thus inheriting the advantages of the powerful spectral clustering approach. Experimental results on many data sets from real-world domains demonstrate the superiority of our proposed approach in obtaining high quality clusterings.

The rest of the paper is organized as follows. In Section 2, we begin with an introduction to the notations and representation of cluster labels, then formulate the model in detail. The algorithm is derived in Section 3. In Section 4, we evaluate the proposed method on many data sets. We make concluding remarks in Section 5.

2 Model Formulation

2.1 Notations

In this section we introduce the notations adopted in this paper. Boldface lowercase letters, such as $\mathbf{x}$ and $\mathbf{y}$, denote column vectors. Boldface uppercase letters, such as $\mathbf{M}$ and $\mathbf{A}$, denote matrices. The superscript T is used to denote the transpose of a vector or matrix. $\mathbf{M} \geq 0$ means that every entry in $\mathbf{M}$ is nonnegative. For $\mathbf{x} \in \mathbb{R}^d$, $\|\mathbf{x}\|$ denotes the L_2 norm, and $\|\mathbf{x}\|_1$ denotes the L_1 norm. Specifically, $\|\mathbf{x}\| = \sqrt{\sum_{i=1}^{d} x_i^2}$ and $\|\mathbf{x}\|_1 = \sum_{i=1}^{d} |x_i|$. For $\mathbf{M} \in \mathbb{R}^{s \times t}$, $\|\mathbf{M}\|_{\text{F}}$ denotes the Frobenius norm, and $\|\mathbf{M}\|_{\text{SAV}}$ denotes the Sum-Absolute-Value norm [4]. Specifically,

$$\|\mathbf{M}\|_{\text{F}} = \sqrt{\text{trace}(\mathbf{M}^T\mathbf{M})} = \left(\sum_{i=1}^{s}\sum_{j=1}^{t} m_{ij}^2\right)^{\frac{1}{2}} \quad \text{and} \quad \|\mathbf{M}\|_{\text{SAV}} = \sum_{i=1}^{s}\sum_{j=1}^{t} |m_{ij}|$$

Table 1. Summary of notations

Symbols	Description
n	total number of input data points
d	data dimensionality
$\mathbf{X}$	total data matrix, $\mathbf{X} \in \mathbb{R}^{n \times d}$
$\mathcal{X}$	input space from which data is drawn, $\mathcal{X} \subseteq \mathbb{R}^d$
c	number of output clusters
π_l	the set of points in the l-th cluster where $1 \leq l \leq c$
$\|\pi_l\|$	the number of points in the l-th cluster where $1 \leq l \leq c$
$\mathcal{N}_i$	the set of "neighbors" of $\mathbf{x}_i$, here $\mathbf{x}_i \notin \mathcal{N}_i$
$\mathbf{1}_m$	m-dimensional column vector whose entries are all 1's
$\mathbf{I}_m$	the identity matrix of order m
Diag($\mathbf{M}$)	the diagonal matrix whose size and diagonal elements are the same as the square matrix $\mathbf{M}$
Deg($\mathbf{M}$)	the degree matrix of $\mathbf{M}$, i.e., the diagonal matrix whose diagonal elements are the sums of rows of $\mathbf{M}$
Trace($\mathbf{M}$)	the trace of the square matrix $\mathbf{M}$

Other important notations are summarized in Table 1. In addition, neighborhood $\mathcal{N}_i$ simply denotes a set of nearest neighbors (measured by some distance metric) of point $\mathbf{x}_i$. Typically, given $k \ll n$, we define each neighborhood $\mathcal{N}_i$ as the set of k-nearest neighbors of $\mathbf{x}_i$, not including $\mathbf{x}_i$.

2.2 Cluster Labels

Given a set of data points $\{\mathbf{x}_i\}_{i=1}^n \subseteq \mathcal{X} \subseteq \mathbb{R}^d$, the goal of clustering is to find a disjoint partitioning $\{\pi_l\}_{l=1}^c$ of the data where π_l is the l-th cluster.

We represent a clustering of the data points by a *partition matrix* $\mathbf{P} = [\mathbf{p}_1, \ldots, \mathbf{p}_c] = [p_{il}] \in \{0,1\}^{n \times c}$ and $\mathbf{P}\mathbf{1}_c = \mathbf{1}_n$. Thus, exactly one element in each row of $\mathbf{P}$ is 1. Specifically,

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

Instead of directly using the entries of partition matrix $\mathbf{P}$ as the cluster labels, we use a *Scaled Partition Matrix* $\mathbf{Y} = \mathbf{P}(\mathbf{P}^T\mathbf{P})^{-1}$ where $\mathbf{Y} = [\mathbf{y}_1, \ldots, \mathbf{y}_c] = [y_{il}] \in \mathbb{R}^{n \times c}$. Specifically,

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

Thereby each data point $\mathbf{x}_i$ is associated with a c-dimensional cluster label $[y_{i1}, y_{i2}, \ldots, y_{ic}]$. The scaling is used for obtaining balanced clusters and balanced clusters usually lead to better performance in practice.

2.3 Clustering Objective

In a typical local learning approach for supervised learning [3], for each test data point, data points in the vicinity of the test point is selected for learning

an output function. Then this function is used to predict the label of the test point. Although this method looks "simple and stupid" [3], it can achieve good performance in practice since only data points relevant to the test point are used for training. This approach can be also adapted to the clustering problem [19] and the idea is translated into:

> In a good clustering, the cluster label of a data point can be well estimated based on its neighbors and their cluster labels.

Based on this idea, a clustering objective can be formulated to obtain a clustering that satisfies the above property. For each point $\mathbf{x}_i$, if we construct a local label predictor $o_{il}(\cdot)$ based on its neighborhood information $\{(\mathbf{x}_j, y_{jl}) \,|\, \mathbf{x}_j \in \mathcal{N}_i\}$, then the prediction result $o_{il}(\mathbf{x}_i)$ should be similar to the cluster label y_{il} of $\mathbf{x}_i$. Hence, a brute force approach to select the best final clustering would be:

1. Enumerate all possible labelings of $\{\mathbf{x}_i\}_{i=1}^n$.
2. For each labeling,
   (a) For each neighborhood $\mathcal{N}_i$ $(1 \le i \le n)$ and the corresponding labels, build local learners $[o_{i1}(\cdot), o_{i2}(\cdot), \ldots, o_{ic}(\cdot)]$. Here, $o_{il}(\cdot)$ is the output function learned using the training data $\{(\mathbf{x}_j, y_{jl}) \,|\, \mathbf{x}_j \in \mathcal{N}_i\}$ where $1 \le l \le c$.
   (b) Predict each data point $\mathbf{x}_i$ using the above local label predictors $[o_{i1}(\cdot), o_{i2}(\cdot), \ldots, o_{ic}(\cdot)]$, obtaining $[o_{i1}(\mathbf{x}_i), o_{i2}(\mathbf{x}_i), \ldots, o_{ic}(\mathbf{x}_i)]$. Calculate the error $\sum_{l=1}^{c} |y_{il} - o_{il}(\mathbf{x}_i)|$.
   (c) Calculate the total error sum $\sum_{i=1}^{n} \sum_{l=1}^{c} |y_{il} - o_{il}(\mathbf{x}_i)|$.
3. Pick the labeling with the smallest total error sum.

Obviously, we won't directly use the above exhaustive search approach since the number of possible clusterings is too large. In fact, for n data points and c clusters, there're $\frac{1}{c!}\sum_{l=1}^{c}\binom{c}{l}(-1)^{c-l}l^n$ different clusterings [8].

Suppose $\mathbf{o}_l = [o_{1l}(\mathbf{x}_1), o_{2l}(\mathbf{x}_2), \ldots, o_{nl}(\mathbf{x}_n)]^T \in \mathbb{R}^{n\times 1}$ and $\mathbf{O} = [\mathbf{o}_1, \ldots, \mathbf{o}_c] \in \mathbb{R}^{n\times c}$, which is the combination of all the predictions made by the local label predictors. Then, the clustering objective can be written as follows:

$$\min_{\mathbf{Y}\in\mathbb{R}^{n\times c}} \quad \mathcal{J}(\mathbf{Y}) = \|\mathbf{Y} - \mathbf{O}\|_{\text{SAV}} = \sum_{l=1}^{c} \|\mathbf{y}_l - \mathbf{o}_l\|_1 \tag{3}$$

$$\text{subject to} \quad \mathbf{Y} \text{ is a scaled partition matrix defined in (2)} \tag{4}$$

Note that instead of using the Frobenius norm $\|\cdot\|_{\text{F}}$ as the error measure, we use the Sum-Absolute-Value norm $\|\cdot\|_{\text{SAV}}$ which is more robust. The problem of what kind of local predictor we will use will be addressed in the next subsection.

2.4 Local Label Predictor

For the clustering objective (3) to be tractable, we want to select a local label predictor that's easy to deal with and yet sufficiently powerful for learning the

local structure. So, we choose *kernel regression* model as our local label predictor. Kernel regression [1,13,18] is a widely used nonparametric technique for nonlinear regression. The prediction of a kernel regression model for a test point takes the form of a weighted average of the target values observed at training points. The weighting coefficients are related to the *kernel function* $K(\mathbf{x}_i, \mathbf{x}_j)$ for $\mathbf{x}_i, \mathbf{x}_j \in \mathcal{X}$ where $\mathcal{X}$ is the data space. With $\mathbf{x}_i$ fixed, $K(\mathbf{x}_i, \mathbf{x})$ can be interpreted as an unnormalized probability density function centered around $\mathbf{x}_i$. Therefore, two key properties of the kernel function are

$$K(\mathbf{x}_i, \mathbf{x}_j) \geq 0 \qquad \text{for all } \mathbf{x}_i, \mathbf{x}_j \in \mathcal{X}. \tag{5}$$

$$\int_{\mathbf{x} \in \mathcal{X}} K(\mathbf{x}_i, \mathbf{x})\, \mathrm{d}\mathbf{x} \in (0, \infty) \qquad \text{for all } \mathbf{x}_i \in \mathcal{X} \tag{6}$$

Given a training set $\{(\mathbf{x}_i, y_i)\}_{i=1}^n$, we want to construct a learner to predict the target value y for a test point $\mathbf{x}$. Motivated from kernel density estimation [2], the target value for a test point $\mathbf{x}$ can be estimated by

$$y = \frac{\sum_{i=1}^n y_i K(\mathbf{x}_i, \mathbf{x})}{\sum_{i=1}^n K(\mathbf{x}_i, \mathbf{x})} \tag{7}$$

which is called the kernel regression formula, also known as Nadaraya-Watson model [10]. This model can also be motivated from the interpolation problem when the input variables are noisy [2].

Generally, a distance-based kernel function [18] can be written as

$$K(\mathbf{x}_i, \mathbf{x}_j) = \varphi(\mathcal{D}(\mathbf{x}_i, \mathbf{x}_j)) \tag{8}$$

where $\mathcal{D}(\cdot, \cdot)$ is a distance metric. $\varphi(\cdot)$ is a nonnegative function and monotonically decreases with increasing $\mathcal{D}(\mathbf{x}_i, \mathbf{x}_j)$. In addition, $\varphi(\cdot)$ often have parameters pertaining to the rate of decay.

Various kernel functions have been studied in the literature, such as Gaussian, Epanechnikov, rectangular, triangular, and so on. The following two kernels will be used in our experiments as in [19]:

- The Gaussian kernel is defined as

$$K(\mathbf{x}_i, \mathbf{x}_j) = \exp\left(-\frac{\mathcal{D}(\mathbf{x}_i, \mathbf{x}_j)^2}{\gamma}\right) \tag{9}$$

  where $\varphi(t) = \exp\left(-\frac{t^2}{\gamma}\right)$ for $\gamma > 0$.

- The cosine kernel defined over nonnegative data points on the unit hypersphere is defined as follows

$$K(\mathbf{x}_i, \mathbf{x}_j) = \mathbf{x}_i^T \mathbf{x}_j = 1 - \frac{\|\mathbf{x}_i - \mathbf{x}_j\|^2}{2} \tag{10}$$

  where $\mathbf{x}_i, \mathbf{x}_j \in \mathcal{X} = \{\mathbf{x} \,|\, \mathbf{x}^T \mathbf{x} = 1, \mathbf{x} \geq 0\}$. Here, $\varphi(t) = 1 - \frac{t^2}{2}$ for $0 \leq t \leq \sqrt{2}$ and $\varphi(t) = 0$ for $t \geq \sqrt{2}$. This kernel generally leads to good performance when used for document data sets [19].

2.5 Constructing and Combining Local Predictors

In our clustering problem, for each data point $\mathbf{x}_i$ and $\{(\mathbf{x}_j, y_{jl}) \,|\, \mathbf{x}_j \in \mathcal{N}_i\}$, we want to construct a local label predictor $o_{il}(\cdot)$ to estimate the cluster label of $\mathbf{x}_i$. Here we choose kernel regression model introduced in the previous subsection as our local label predictor. According to equation (7), the solution of local label prediction for $\mathbf{x}_i$ and l is given by

$$o_{il}(\mathbf{x}_i) = \frac{\sum\limits_{\mathbf{x}_j \in \mathcal{N}_i} K(\mathbf{x}_i, \mathbf{x}_j) y_{jl}}{\sum\limits_{\mathbf{x}_j \in \mathcal{N}_i} K(\mathbf{x}_i, \mathbf{x}_j)} \tag{11}$$

We can construct a matrix $\mathbf{A} = [a_{ij}] \in \mathbb{R}^{n \times n}$ as follows

$$a_{ij} = \begin{cases} \frac{K(\mathbf{x}_i, \mathbf{x}_j)}{\sum\limits_{\mathbf{x}_j \in \mathcal{N}_i} K(\mathbf{x}_i, \mathbf{x}_j)} & \text{if } \mathbf{x}_j \in \mathcal{N}_i \\ 0 & \text{if } \mathbf{x}_j \notin \mathcal{N}_i \end{cases} \tag{12}$$

Two key properties that $\mathbf{A}$ satisfies are

$$\mathbf{A} \geq 0 \tag{13}$$

$$\mathbf{A}\mathbf{1}_n = \mathbf{1}_n \tag{14}$$

These two properties will be used in the next section for the derivation of our main clustering algorithm.

Recall from section 2.3 that

$$\mathbf{o}_l = [o_{1l}(\mathbf{x}_1), o_{2l}(\mathbf{x}_2), \ldots, o_{nl}(\mathbf{x}_n)]^T \in \mathbb{R}^{n \times 1} \tag{15}$$

$$\mathbf{O} = [\mathbf{o}_1, \mathbf{o}_2, \ldots, \mathbf{o}_c] \in \mathbb{R}^{n \times c} \tag{16}$$

where $1 \leq l \leq c$. By combining equations (11) and (12), it's not difficult to see that

$$\mathbf{o}_l = \mathbf{A}\mathbf{y}_l \qquad \text{and} \qquad \mathbf{O} = \mathbf{A}\mathbf{Y} \tag{17}$$

Therefore, the clustering objective in (3) can be rewritten as follows:

$$\min_{\mathbf{Y} \in \mathbb{R}^{n \times c}} \qquad \mathcal{J}(\mathbf{Y}) = \|\mathbf{Y} - \mathbf{A}\mathbf{Y}\|_{\text{SAV}} = \sum_{l=1}^{c} \|\mathbf{y}_l - \mathbf{A}\mathbf{y}_l\|_1 \tag{18}$$

$$\text{subject to} \qquad \mathbf{Y} \text{ is a scaled partition matrix defined in (2)} \tag{19}$$

This is the main objective function we want to optimize. Here, the L_1 norm $\|\cdot\|_1$ in equation (18) makes the function not differentiable and difficult to optimize combined with constraint (2) which is not convex. However, the properties of $\mathbf{A}$ in (13) and (14) ensure that the problem can be optimized by eigen-decomposing some sparse matrix. We will derive the algorithm in the next section.

3 Algorithm Derivation

In this section, we simplify the optimization problem (18) and obtain an algorithm that's based on exploiting the eigen-structure of a sparse matrix.

3.1 Main Theorem

Suppose $\widehat{\mathbf{a}}_i \in \mathbb{R}^{n\times 1}$ denotes the transpose of the i-th row vector of $\mathbf{A}$, so $\mathbf{A} = [\widehat{\mathbf{a}}_1, \widehat{\mathbf{a}}_2, \ldots, \widehat{\mathbf{a}}_n]^T$. Then we have

$$\mathcal{J}(\mathbf{Y}) = \|\mathbf{Y} - \mathbf{A}\mathbf{Y}\|_{\text{SAV}} = \sum_{l=1}^{c}\sum_{i=1}^{n} |y_{il} - \widehat{\mathbf{a}}_i^T \mathbf{y}_l| \tag{20}$$

Substituting (2) into (20), we have

$$\mathcal{J}(\mathbf{Y}) = \sum_{l=1}^{c}\left(\sum_{\mathbf{x}_i\in\pi_l} \frac{|1 - \sum_{\mathbf{x}_j\in\pi_l} a_{ij}|}{|\pi_l|} + \sum_{\mathbf{x}_i\notin\pi_l} \frac{|0 - \sum_{\mathbf{x}_j\in\pi_l} a_{ij}|}{|\pi_l|}\right) \tag{21}$$

According to the properties of $\mathbf{A}$ in (13) and (14), we obtain

$$\mathcal{J}(\mathbf{Y}) = \sum_{l=1}^{c}\left(\sum_{\mathbf{x}_i\in\pi_l} \frac{\sum_{\mathbf{x}_j\notin\pi_l} a_{ij}}{|\pi_l|} + \sum_{\mathbf{x}_i\notin\pi_l} \frac{\sum_{\mathbf{x}_j\in\pi_l} a_{ij}}{|\pi_l|}\right) \tag{22}$$

$$= \sum_{l=1}^{c} \frac{\sum_{\mathbf{x}_i\in\pi_l}\sum_{\mathbf{x}_j\notin\pi_l}(a_{ij} + a_{ji})}{|\pi_l|} \tag{23}$$

$$= \sum_{l=1}^{c} \frac{\mathbf{p}_l^T\left(\text{Deg}(\mathbf{A} + \mathbf{A}^T) - \mathbf{A} - \mathbf{A}^T\right)\mathbf{p}_l}{|\pi_l|} \tag{24}$$

where $\mathbf{p}_l$ is the l-th column of partition matrix $\mathbf{P}$ defined in (1).

Define $\mathbf{F} = [\mathbf{f}_1, \ldots, \mathbf{f}_c] = [f_{il}] \in \mathbb{R}^{n\times c}$ as $\mathbf{F} = \mathbf{P}(\mathbf{P}^T\mathbf{P})^{-\frac{1}{2}}$, so $\mathbf{f}_l = \mathbf{p}_l(\mathbf{p}_l^T\mathbf{p}_l)^{-\frac{1}{2}}$. Specifically,

$$f_{il} = \begin{cases} \sqrt{\frac{1}{|\pi_l|}} & \text{if } \mathbf{x}_i \in \pi_l, \\ 0 & \text{if } \mathbf{x}_i \notin \pi_l. \end{cases} \tag{25}$$

Obviously, we have

$$\mathbf{F}^T\mathbf{F} = \left((\mathbf{P}^T\mathbf{P})^{-\frac{1}{2}}\mathbf{P}^T\right)\mathbf{P}(\mathbf{P}^T\mathbf{P})^{-\frac{1}{2}} = \mathbf{I}_c \tag{26}$$

where $\mathbf{I}_c \in \mathbb{R}^{c\times c}$ is the identity matrix of order c.

Using $\mathbf{F}$, the clustering objective can be simplified as

$$\mathcal{J}(\mathbf{F}) = \sum_{l=1}^{c} \mathbf{f}_l^T\left(\text{Deg}(\mathbf{A} + \mathbf{A}^T) - \mathbf{A} - \mathbf{A}^T\right)\mathbf{f}_l \tag{27}$$

$$= \text{Trace}\left(\mathbf{F}^T\left(\text{Deg}(\mathbf{A} + \mathbf{A}^T) - \mathbf{A} - \mathbf{A}^T\right)\mathbf{F}\right) \tag{28}$$

Therefore, we obtain the following theorem:

Table 2. Clustering via LOcal Regression (CLOR)

Input :
Data set $\{\mathbf{x}_i\}_{i=1}^n$, number of clusters c, neighborhood size k.
Output :
Partition matrix $\mathbf{P}$ as defined in (1).
Procedure :
1. Compute the k nearest neighbors ($\mathcal{N}_i$) for each $\mathbf{x}_i$.
2. Construct the matrix $\mathbf{A}$ as defined in (12).
3. Construct the matrix $\mathbf{M} = \mathrm{Deg}(\mathbf{A} + \mathbf{A}^T) - \mathbf{A} - \mathbf{A}^T$.
4. Compute the eigenvectors corresponding to the c smallest eigenvalues of $\mathbf{M}$, thus obtaining $\mathbf{F}^*$.
5. Discretize $\mathbf{F}^*$ to get the partition matrix $\mathbf{P}$.

Theorem 1. *The optimization problem in* (18) *is equivalent to the following one:*

$$\min_{\mathbf{F}\in\mathbb{R}^{n\times c}} \quad \mathrm{Trace}\left(\mathbf{F}^T\left(\mathrm{Deg}(\mathbf{A}+\mathbf{A}^T)-\mathbf{A}-\mathbf{A}^T\right)\mathbf{F}\right) \tag{29}$$

$$\text{subject to} \quad \mathbf{F} \textit{ is defined in } (25) \tag{30}$$

3.2 Relaxation and Discretization

Following the standard spectral clustering procedure, we relax the $\mathbf{F}$ defined in (25) to be any matrix from $\mathbb{R}^{n\times c}$ that satisfies $\mathbf{F}^T\mathbf{F} = \mathbf{I}_c$. Thus, the optimization problem is as follows:

$$\min_{\mathbf{F}\in\mathbb{R}^{n\times c}} \quad \mathrm{Trace}\left(\mathbf{F}^T\left(\mathrm{Deg}(\mathbf{A}+\mathbf{A}^T)-\mathbf{A}-\mathbf{A}^T\right)\mathbf{F}\right) \tag{31}$$

$$\text{subject to} \quad \mathbf{F}^T\mathbf{F} = \mathbf{I}_c \tag{32}$$

From the *Ky Fan* Theorem [23], the global optimal solution of the above relaxed problem is given by any matrix from the following set

$$\left\{\mathbf{F}^*\mathbf{Q} \,|\, \mathbf{Q}\in\mathbb{R}^{c\times c}, \mathbf{Q}^T\mathbf{Q} = \mathbf{I}_c\right\} \tag{33}$$

where the columns of $\mathbf{F}^* \in \mathbb{R}^{n\times c}$ are the c eigenvectors corresponding to the c smallest eigenvalues of the matrix $\mathrm{Deg}(\mathbf{A}+\mathbf{A}^T) - \mathbf{A} - \mathbf{A}^T$.

After obtaining the relaxed solution, we have to discretize it to get a final solution defined in (1). The discretization approach used in [22] is adopted to obtain the final partition matrix $\mathbf{P}$ since it's previously reported to produce satisfactory results [22,19]. Using an iterative procedure, this discretization method tries to rotate $\mathbf{F}^*$ (after normalizing the rows of $\mathbf{F}^*$ to unit norm) so that it's close to a partition matrix as defined in (1). Details can be found in [22].

The main algorithm is summarized in Table 2. We name our algorithm Clustering via LOcal Regression (CLOR).

3.3 Relation to Other Approaches

Although the final optimization problem uses the spectral methods for finding the optimal clustering, our clustering criterion is motivated from a local learning approach. In our method, we adopt sum of absolute error as the discrepancy measure instead of sum of squared error which is used in [19]. In addition, in order for the optimization to be tractable, we use kernel regression as the local label predictor while [19] uses "kernel ridge regression". We will empirically demonstrate that our clustering algorithm usually achieves better performance than the algorithm proposed in [19].

Our algorithm is different from spectral clustering with ratio cut [6] or normalized cut (NCut) [11] on the k-nearest neighbor similarity graph. We construct the following k-nearest neighbor similarity matrix $\mathbf{G} = [g_{ij}] \in \mathbb{R}^{n \times n}$

$$g_{ij} = \begin{cases} K(\mathbf{x}_i, \mathbf{x}_j) & \text{if } \mathbf{x}_j \in \mathcal{N}_i \text{ or } \mathbf{x}_i \in \mathcal{N}_j \\ 0 & \text{otherwise} \end{cases} \tag{34}$$

Spectral clustering algorithms typically use the affinity matrix $\mathbf{G}$ which is symmetric. In general, the graph laplacian (used by ratio cut) or normalized graph laplacian (used by NCut) derived from this affinity matrix are not equal to $\mathrm{Deg}(\mathbf{A} + \mathbf{A}^T) - \mathbf{A} - \mathbf{A}^T$. Since NCut usually outperforms ratio cut in practice, we compare our algorithm CLOR with NCut in the next section and demonstrate the better performance of our algorithm.

4 Experiments

In this section, we present an empirical evaluation of our clustering method in comparison with representative algorithms on a number of data sets.

4.1 Data Sets

In this subsection, we introduce the basic information of the data sets used in our experiments. We use 14 document data sets[1] from the CLUTO toolkit [24].

We briefly introduce the basic information of the data sets as follows.

- *cranmed*: This data set comprises the CRANFIELD and MEDLINE abstracts which were previously used to evaluate information retrieval systems.
- *fbis*: The *fbis* data set is derived from the Foreign Broadcast Information Service data of TREC-5 [15].
- *hitech*: This data set is from the San Jose Mercury newspaper articles that are distributed as part of the TREC collection.
- *k1a*, *k1b* and *wap*: These data set are from the WebACE project (WAP).
- *re0* and *re1*: These data sets are from Reuters-21578 text collection [9].
- *tr11*, *tr12*, *tr23*, *tr31*, *tr41*, and *tr45*: These six data sets are derived from the TREC collection [15].

[1] `http://glaros.dtc.umn.edu/gkhome/fetch/sw/cluto/datasets.tar.gz`

Table 3. Summary of data sets

Name	Source	Points(n)	Dims(d)	Classes(c)
cranmed	CRANFIELD/MEDLINE	2431	41681	2
fbis	FBIS (TREC)	2463	12674	17
k1a	WebACE	2340	21839	20
k1b	WebACE	2340	21839	6
hitech	San Jose Mercury (TREC)	2301	10080	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

Table 3 summarizes the basic properties of the data sets. The smallest of these data sets contains 204 data points and the largest consists of 2463 points. The number of classes ranges from 2 to 25. These data sets are widely used in the literature to evaluate different clustering systems [14,24,25].

4.2 Evaluation Criteria

In all the experiments of this paper, the class labels of data points are known to the evaluation process and external validity measures are used to evaluate how much class structure is recovered by a clustering. Besides, the true number of classes c is provided to all the clustering algorithms. We use two popular external validity measures, Normalized Mutual Information (*NMI*) [12] and Clustering Accuracy (*Acc*), as our criteria.

Given a clustering $\mathcal{C}$ and the "true" partitioning $\mathcal{B}$ (class labels). The number of clusters in $\mathcal{C}$ and classes in $\mathcal{B}$ are both c. Suppose n_i is the number of objects in the i-th cluster, $n_j^{'}$ is the number of objects in the j-th class and n_{ij} is the number of objects which are both in the i-th cluster and j-th class. *NMI* between $\mathcal{C}$ and $\mathcal{B}$ is calculated as follows [12]:

$$NMI(\mathcal{C},\mathcal{B}) = \frac{\sum_{i=1}^{c}\sum_{j=1}^{c} n_{ij}\log\frac{n\cdot n_{ij}}{n_i\cdot n_j^{'}}}{\sqrt{\sum_{i=1}^{c} n_i\log\frac{n_i}{n}\sum_{j=1}^{c} n_j^{'}\log\frac{n_j^{'}}{n}}}. \tag{35}$$

The value of *NMI* equals 1 if and only if $\mathcal{C}$ and $\mathcal{B}$ are identical and is close to 0 if $\mathcal{C}$ is a random partitioning. Larger values of *NMI* indicate better clustering performance.

Suppose n is the total number of objects and other notations are the same as above. Clustering Accuracy (*Acc*) builds a one-to-one correspondence between

the clusters and the classes. Suppose the permutation function $\mathrm{Map}(\cdot) : \{i\}_{i=1}^{c} \mapsto \{j\}_{j=1}^{c}$ maps each cluster index to a class index, i.e., $\mathrm{Map}(i)$ is the class index that corresponds to the i-th cluster. Acc between $\mathcal{C}$ and $\mathcal{B}$ is calculated as follows:

$$Acc(\mathcal{C}, \mathcal{B}) = \frac{\max\left(\sum_{i=1}^{c} n_{i,\mathrm{Map}(i)}\right)}{n} \tag{36}$$

The value of Acc equals 1 if and only if $\mathcal{C}$ and $\mathcal{B}$ are identical. Larger values of Acc indicate better clustering performance.

4.3 Comparison Settings

Each data point (document) is originally represented by a term-frequency vector (Bag-of-Words). We normalize each vector to unit norm so that every point $\mathbf{x}_i$ lies on the nonnegative unit hypersphere, that is, $\mathbf{x}_i \in \mathcal{X} = \{\mathbf{x} \,|\, \mathbf{x}^T\mathbf{x} = 1, \mathbf{x} \geq 0\}$ for all $1 \leq i \leq n$. The cosine kernel defined in (10) is generally considered very suitable for documents. So it is adopted as the kernel function in most experiments unless other kernels are explicitly mentioned to be used. For each point $\mathbf{x}_i$, the neighborhood $\mathcal{N}_i$ consists of k-nearest neighbors of $\mathbf{x}_i$ measured by the cosine similarity. Here, k is provided by domain experts.

We test our Clustering via LOcal Regression (CLOR) algorithm in comparison with another two algorithms which are listed as follows:

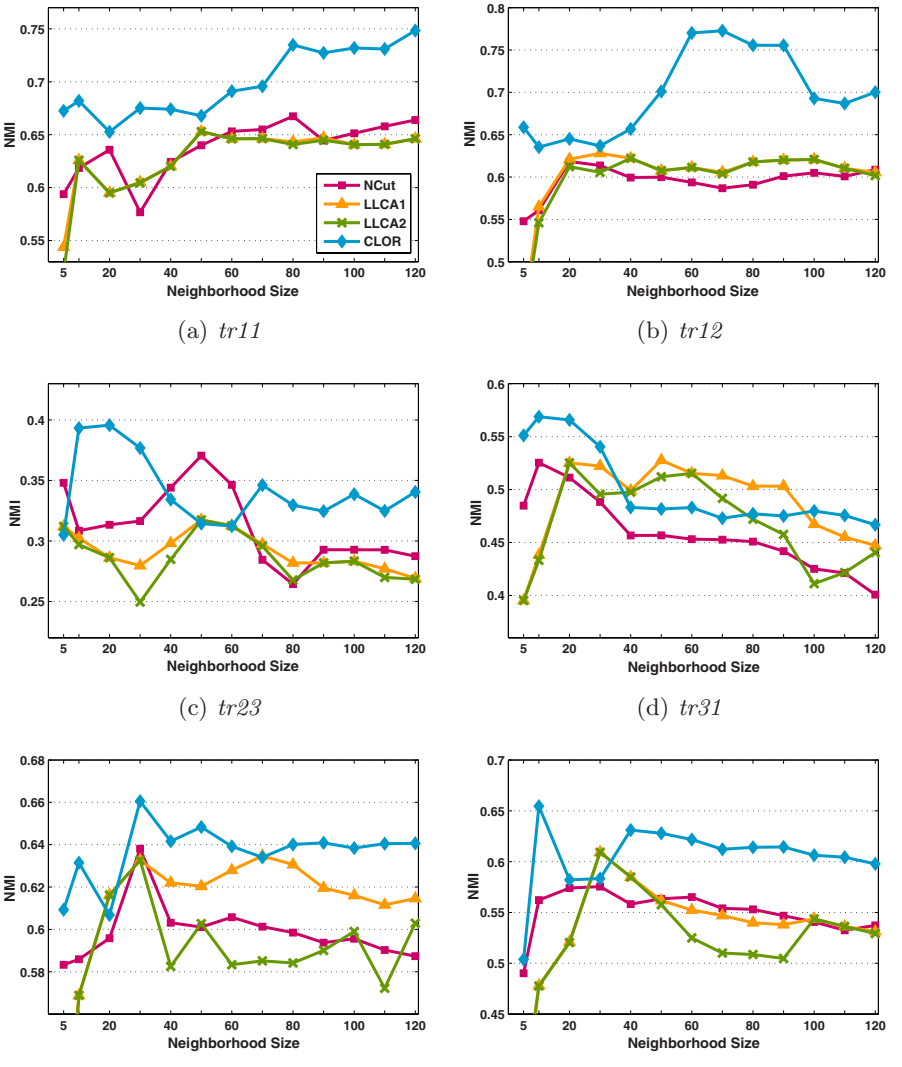
- Spectral clustering with normalized cut (NCut) [11]. The affinity graph is constructed using the weighted k-nearest neighbor graph. The edge weight between two connected data points is calculated with the kernel function.
- Local Learning based Clustering Algorithm[2] (LLCA) proposed in [19]. There is a regularization parameter λ in LLCA. As is done in [19], we choose this parameter from: $\lambda \in \{0.1, 1, 1.5\}$. For each data set and k, we report the best performance among the results produced when different values of λ are used (LLCA1). We also report the result obtained when LLCA automatically select λ using the parameter selection method in [19] (LLCA2). Therefore, LLCA1 has an unfair advantage over others.

For each algorithm, we assume that the "true" number of clusters c is given. All the algorithms use the same discretization method whose code is available at `http://www.cis.upenn.edu/~jshi/software/`. Note that the main computational load of all the above algorithms is to eigen-decompose (or singular value decomposition) a sparse $n \times n$ matrix with $O(nk)$ non-zero elements.

4.4 Comparison of Clustering Performance

In this section, we compare the clustering performance of the investigated algorithms. We have tested the clustering performance of the algorithms with various neighborhood sizes.

[2] The code is available at `www.kyb.tuebingen.mpg.de/bs/people/mingrui/LLCA.zip` The code for NCut is also included in this package.



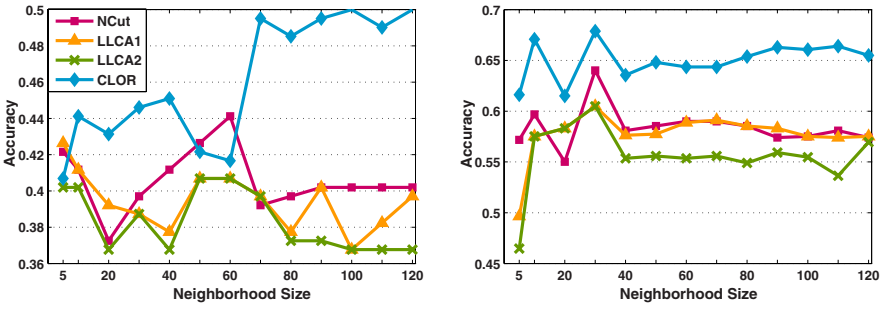
(a) *tr11* (b) *tr12*

(c) *tr23* (d) *tr31*

(e) *tr41* (f) *tr45*

Fig. 1. The clustering performance (*NMI*) on six TREC data sets with different number of nearest neighbors (k). The legend is shown only in subfigure (a) for clarity.

For the six TREC data sets (*tr11*, *tr12*, *tr23*, *tr31*, *tr41*, and *tr45*), Figure 1 shows the the clustering performance of the algorithms. The x-axis denotes the neighborhood size k and the y-axis denotes the clustering performance measured by *NMI*. When the neighborhood size is too small ($k = 5$), LLCA1 and LLCA2 produce unsatisfactory clustering results on data sets such as *tr12*, *tr41* and *tr45*. These *NMI* results are not displayed in the subfigures for clarity.



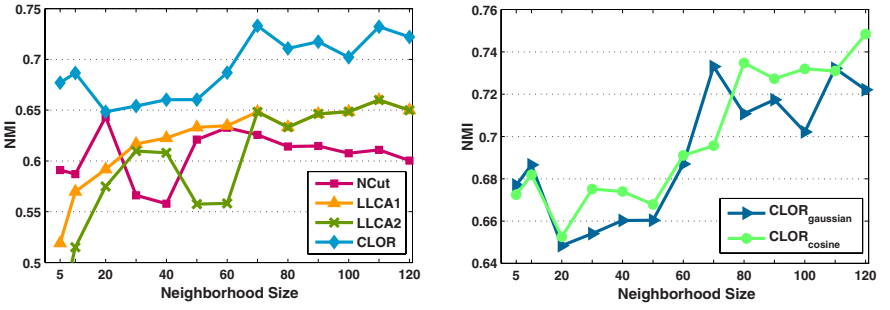
(a) *tr23* (b) *tr41*

Fig. 2. The clustering performance (*Acc*) on data sets *tr23* and *tr41* with different neighborhood sizes. The legend is shown only in subfigure (a) for clarity.

The results in Figure 1 show that our algorithm CLOR very often achieves the best performance on all the six data sets and when different neighborhood sizes are chosen. Even though the algorithm LLCA1 has the extra edge of selecting the best *NMI* value when different values of the regularization parameter λ are used, our algorithm CLOR has a clear advantage over it. Generally in practice, spectral clustering with normalized cut (NCut) can produce very good results when an appropriate similarity measure is chosen. Here we see that the performance of LLCA is comparable to that of NCut, which demonstrates the effectiveness of the local learning approach to clustering. The better performance of CLOR in Figure 1 shows that our proposed algorithm is more suitable for the task of clustering based on the local learning idea.

We also compare the clustering performance of the algorithms on the six TREC data sets in terms of *Acc* as defined in equation (36). Most of the performance figures are somewhat similar to the ones displayed in Figure 1. The two most different figures are shown in Figure 2. Note that here, LLCA1 represents the best *Acc* value when different regularization parameters γ are used. And the value of γ associated with the best *Acc* is not necessarily the same as the value of γ that corresponds to the best *NMI*. It's observed that our algorithm CLOR still outperforms others in most cases. Particularly, on the data set *tr41*, CLOR seems to have a larger advantage over others in terms of *Acc* than *NMI*.

Besides, we conduct experiments on the six TREC data sets when Gaussian kernel is used instead of cosine kernel. Due to the space limit, we only show the results on data set *tr11* in Figure 3. The left subfigure shows the *NMI* values of the algorithms on *tr11* when different neighborhood sizes k are chosen. Here the algorithms use Gaussian kernel ($\gamma = 1$). The right subfigure displays the performance of CLOR with the two kernels on data set *tr11*. It can be observed from the left subfigure that our algorithm still outperforms the other three. The right subfigure shows that cosine kernel is slightly better than Gaussian kernel for document clustering. However, the result from Gaussian kernel doesn't deviate too much from the result with cosine kernel on *tr11*. Experiments with Gaussian



(a) *tr11*, Gaussian Kernel ($\gamma = 1$) (b) Gaussian vs. Cosine

Fig. 3. The left subfigure shows the clustering performance (*NMI*) of the considered algorithms on data set *tr11*. Here Gaussian kernel ($\gamma = 1$) is used. The right subfigure displays the performance of CLOR with different kernels on data set *tr11*.

kernel have also been conducted on the other five TREC data sets. From these experiments, we can draw similar conclusions as above. Therefore, the superiority of our algorithm CLOR is not sensitive to what type of kernel is used.

In general, theoretical guidance on how to choose the best neighborhood size k is rare. A rule of thumb is to choose k so that the number of connected components of the k-nearest neighbor graph is close to one. An asymptotic result is that if k is chosen on the order of $\log(n)$, then the k-nearest neighbor graph will be connected for n points drawn i.i.d. from some probability distribution with a connected support in $\mathbb{R}^d$ [5]. Therefore, for data sets with size around 1000, we can choose k to be a small multiple of 10. We list the experimental results on the remaining 8 data sets when neighborhood size $k = 30$ in Table 4. It can be observed that our algorithm CLOR consistently outperforms the other three. We also note that different performance measures may lead to divergent performance ranks of the clustering algorithms, which is reflected by the results on the data sets *re0* and *re1*.

Table 4. Empirical results when neighborhood size $k = 30$. Both *NMI* and *Acc* results are provided. On each data set, the best results of *NMI* and *Acc* are shown in boldface.

	NMI				*Acc*			
	LLCA1	LLCA2	NCut	CLOR	LLCA1	LLCA2	NCut	CLOR
cranmed	0.8479	0.8479	0.8568	**0.8927**	0.9745	0.9745	0.9770	**0.9840**
fbis	0.5816	0.5757	0.5832	**0.5909**	0.5108	0.5108	0.5428	**0.5534**
hitech	0.3373	0.3373	0.2956	**0.3439**	0.4920	0.4920	0.4207	**0.5502**
k1a	0.5267	0.5267	0.5223	**0.5557**	0.4060	0.4060	0.3987	**0.4897**
k1b	0.6416	0.6416	0.7180	**0.7332**	0.8120	0.8120	0.8466	**0.8846**
re0	0.3905	0.3847	0.4030	**0.4302**	**0.3863**	0.3138	0.3324	0.3318
re1	0.4942	0.4598	0.4967	**0.5043**	**0.3959**	0.3693	0.3730	0.3953
wap	0.5258	0.5093	0.5173	**0.5426**	0.3962	0.3333	0.3859	**0.4314**

5 Conclusions and Future Work

In this paper, we propose a new local learning based clustering algorithm, namely, Clustering via LOcal Regression (CLOR). This algorithm is based on the local learning idea, i.e., in a good clustering, the cluster label of each data point can be well predicted based on its neighbors and their cluster labels. When using kernel regression model as the local label predictor and sum of absolute error as the discrepancy measure, we obtain an algorithm that still inherits the advantages of spectral clustering. Experimental results on many data sets show that our algorithm consistently outperforms the previously proposed local learning approach for clustering [19] and spectral clustering based on normalized cut, which demonstrate the effectiveness and potential of our proposed algorithm in obtaining accurate clusterings.

In the future, we want to continue this work on several issues as follows. First, we want to gain a deeper understanding on the underlying reasons for the good performance of our algorithm. We will start this work by investigating the capacity and locality control issues of the local label predictor in our approach and the one in [19]. Second, we will try to derive a good and stable automatic parameter selection procedure for neighborhood size k and parameters in the kernel function. Third, instead of using k-nearest neighbors measured by a distance metric which is provided by domain experts, we plan to learn the neighborhood $\mathcal{N}_i$ using semi-supervised information such as instance-level constraints [14]. This problem seems to be easier than learning a distance metric and thus hopefully we'll obtain good results.

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

References

1. Benedetti, J.K.: On the Nonparametric Estimation of Regression Functions. Journal of the Royal Statistical Society 39(2), 248–253 (1977)
2. Bishop, C.M.: Pattern Recognition and Machine Learning. Springer, New York (2006)
3. Bottou, L., Vapnik., V.: Local learning algorithms. Neural Computation 4(6), 888–900 (1992)
4. Boyd, S., Vandenberghe, V.: Convex Optimization. Cambridge University Press, Cambridge (2004)
5. 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)
6. 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)
7. Ding, C., He, X., Zha, H., Gu, M., Simon, H.D.: A Min-max Cut Algorithm for Graph Partitioning and Data Clustering. In: Proceedings of the 2001 IEEE International Conference on Data Mining (2001)

8. Jain, A.K., Dubes, R.C.: Algorithms for Clustering Data. Prentice-Hall, Englewood Cliffs (1988)
9. Lewis, D.D.: Reuters-21578 text categorization test collection, http://www.daviddlewis.com/resources/testcollections/reuters21578/
10. Nadaraya, E.A.: On Estimating Regression. Theory of Probability and its Applications 9(1), 141–142 (1964)
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. Takeda, H., Farsiu, S., Milanfar, P.: Kernel Regression for Image Processing and Reconstruction. IEEE Transactions on Image Processing 16(2), 349–366 (2007)
14. Tang, W., Xiong, H., Zhong, S., Wu, J.: Enhancing Semi-Supervised Clustering: A Feature Projection Perspective. In: Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (2007)
15. TREC: Text REtrieval Conference, http://trec.nist.gov
16. Wang, F., Zhang, C., Li, T.: Clustering with Local and Global Regularization. In: Proceedings of the Twenty-Second AAAI Conference on Artificial Intelligence (2007)
17. Wang, F., Zhang, C., Li, T.: Regularized Clustering for Documents. In: Proceedings of the 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (2007)
18. Weinberger, K.Q., Tesauro, G.: Metric Learning for Kernel Regression. In: Proceedings of the Eleventh International Conference on Artificial Intelligence and Statistics (2007)
19. Wu, M., Schölkopf, B.: A Local Learning Approach for Clustering. In: Advances in Neural Information Processing Systems 19 (2006)
20. Wu, M., Schölkopf, B.: Transductive Classification via Local Learning Regularization. In: Proceedings of the Eleventh International Conference on Artificial Intelligence and Statistics (2007)
21. Wu, M., Yu, K., Yu, S., Schölkopf, B.: Local Learning Projections. In: Proceedings of the Twenty-Fourth International Conference on Machine Learning (2007)
22. Yu, S.X., Shi, J.: Multiclass Spectral Clustering. In: Proceedings of the 9th International Conference On Computer Vision (2003)
23. Zha, H., He, X., Ding, C., Gu, M., Simon, H.D.: Spectral Relaxation for K-means Clustering. In: Advances in Neural Information Processing Systems 14 (2001)
24. Zhao, Y., Karypis, G.: Empirical and Theoretical Comparisons of Selected Criterion Functions for Document Clustering. Machine Learning 55, 311–331 (2004)
25. Zhong, S., Ghosh, J.: A Unified Framework for Model-based Clustering. Journal of Machine Learning Research 4, 1001–1037 (2003)
26. Zhu, X., Goldberg, A.: Kernel Regression with Order Preferences. In: Proceedings of the Twenty-Second AAAI Conference on Artificial Intelligence (2007)