Clustering in Knowledge Embedded Space

Yungang Zhang[1], Changshui Zhang[1], and Shijun Wang[1]

State Key Laboratory of Intelligent Technology and Systems
Department of Automation, Tsinghua University, Beijing 100084, China
{zyg00,zcs,wsj02}@mails.tsinghua.edu.cn

Abstract. Cluster analysis is a fundamental technique in pattern recognition. It is difficult to cluster data on complex data sets. This paper presents a new algorithm for clustering. There are three key ideas in the algorithm: using mutual neighborhood graphs to discover knowledge and cluster data; using eigenvalues of local covariance matrixes to express knowledge and form a knowledge embedded space; and using a denoising trick in knowledge embedded space to implement clustering. Essentially, it learns a new distance metric by knowledge embedding and makes clustering become easier under this distance metric. The experiment results show that the algorithm can construct a quality neighborhood graph from a complex and noisy data set and well solve clustering problems.

1 Introduction

Cluster analysis is the automatic identification of groups of similar objects. The discovered clusters serve as the foundation for other data mining and analysis techniques. There have been many works on cluster analysis. Existing clustering algorithms, such as K-means [1], PAM [2], CLARANS [3], DBSCAN [4], CURE [5], and ROCK [6] are designed to find clusters that fit some static models. These algorithms will breakdown if the model is not adequate to capture the characteristics of clusters. Most of these algorithms breakdown when the data set consists of clusters that are of different shapes, densities and sizes [7].

This paper presents a new clustering algorithm. There are three key ideas in the algorithm. The first is using mutual neighborhood graphs to discover knowledge and cluster data. The second is using eigenvalues of local covariance matrixes to express knowledge and embedding knowledge into the input space to form a knowledge embedded space. MNN (Mutual Nearest Neighbor) distance in the knowledge embedded space is used as the new distance metric instead of Euclidean distance in the input space. The third is using a denoising trick in knowledge embedded space to implement clustering. Essentially, it learns a new distance metric by knowledge embedding and makes clustering become easier under this distance metric. The experiment results show that the algorithm can cluster data of different shapes, densities and sizes correctly.

The rest of this paper is organized as follows: Section two gives basic notions and an overview of related work. Section three describes the new method in detail. Section four explains several experiments using the new method. Section five gives some discussions. Section six presents conclusions.

N. Lavrač et al. (Eds.): ECML 2003, LNAI 2837, pp. 480–491, 2003.

2 Basic Notions and Related Work

The basic ideas of our algorithm are elicited by LLE [8,9]. LLE showed us an efficient way to use local information to solve nonlinear problems. First, it constructs a neighborhood graph. Next, it discovers the information of reconstruction weights from the neighborhood graph. Finally, it carries out nonlinear dimensionality reduction by using the reconstruction weights. Our algorithm is similar to LLE except it solves clustering problems. First, a mutual neighborhood graph is constructed. For ideal data sets, all points belonging to the same cluster are connected in the neighborhood graph. Any points belonging to different clusters are not connected. However, in practice, due to noise and the complexity of the data set, different clusters are often connected. We must split them from each other. Then, local information useful for clustering is discovered and embedded into input space and a knowledge embedded space is formed. Finally, clustering can be done by the use of this information and different clusters can be split from each other.

The local information used in our algorithm is eigenvalues of local covariance matrixes. This is an extension of local principal component analysis methods [10,11,12]. Our method directly uses all the eigenvalues of local covariance matrixes to represent local knowledge rather than only analyzing local principal components. The advantage is that it contains all the information about local shape and local size rather than local dimension. In section 3.4 and 3.5, we will see that eigenvalues are useful for denoising of input data and λ knowledge that are pivotal steps for clustering.

The Kernel-based method [13,14,15] is a typical solution for nonlinear problems. The key idea of which is to transform nonlinear data sets in the input space into linear data sets in a high dimensional feature space. Essentially, it is still finding a new distance metric by space transformation. The primary difficulty of kernel-based methods is that it is difficult to choose a proper kernel function to perform this task. In our method, we also construct a high dimension space, an easily formed knowledge embedded space. The purpose is to analyze useful information for clustering, not translate nonlinear data sets into linear data sets.

3 NK Algorithm

Our algorithm is called NK algorithm. It means mutual neighborhood graph construction by knowledge embedding. "N" indicates neighborhood and "K" indicates Knowledge. This section describes the NK algorithm in detail. First, some basic concepts are defined, and then the details of the algorithm are given.

3.1 Definitions

Here are the basic concepts used in NK algorithm which will be explained in the next six subsections. The input of the algorithm is a data set X, with N points:

$$X = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N\}, \boldsymbol{x}_i \in R^d \tag{1}$$

N is the number of points and d is the dimension number of input data.

Definition 1: Set $\omega_i = \{\boldsymbol{x}_i, \boldsymbol{x}_{i_1}, \ldots, \boldsymbol{x}_{i_K}\}$ $(i = 1, .., N)$ is called the local neighborhood of $\boldsymbol{x}_i$, where K is the number of neighbors and $\boldsymbol{x}_{i_l}$ $(l = 1, \ldots, K)$ denotes the l^{th} nearest neighbor of $\boldsymbol{x}_i$. For convenience, $\boldsymbol{x}_{i_0}$ is used to denote $\boldsymbol{x}_i$ and ω_i is rewritten as $\omega_i = \{\boldsymbol{x}_{i_0}, ..., \boldsymbol{x}_{i_K}\}$.

Definition 2: Local covariance matrix of ω_i is:

$$S_i = \frac{1}{K+1} \sum_{l=0}^{K} \left(\boldsymbol{x}_{i_l} - \boldsymbol{m}_i\right) \left(\boldsymbol{x}_{i_l} - \boldsymbol{m}_i\right)^T, \tag{2}$$

where $\boldsymbol{m}_i = \frac{1}{K+1} \sum_{l=0}^{K} \boldsymbol{x}_{i_l}$ is the average of ω_i.

Definition 3: $\boldsymbol{\lambda}_i = [\lambda_{i1}, \ldots, \lambda_{id}]^T$ $(i = 1, \ldots, N, \lambda_{i1} \geq \cdots \geq \lambda_{id})$ is the vector of eigenvalues of S_i and is called the local feature of $\boldsymbol{x}_i$. The knowledge represented by local feature is called λ knowledge.

Definition 4: A neighborhood graph is an undirected weighted graph $G = (X, E)$, where X is the set of data points and E is the set of edges between neighbors with weights e_{ij} to represent the distance. When MNV (Mutual Neighborhood Values) [16] are used as the weights, the neighborhood graph is called a mutual neighborhood graph and the distance represented by MNV is called MNN distance. If $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ are not neighbors, let $e_{ij} = 0$, indicating there is no edge between them; otherwise, e_{ij} is the MNV of $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$:

$$e_{ij} = \begin{cases} L_{ij}, \boldsymbol{x}_j \in \omega_i \\ 0, \boldsymbol{x}_j \notin \omega_i \end{cases}, \tag{3}$$

where L_{ij} is the mutual neighborhood value of the pair of points $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$. If $\boldsymbol{x}_j$ is the p^{th} nearest neighbor of $\boldsymbol{x}_i$ and $\boldsymbol{x}_i$ is the q^{th} nearest neighbor of $\boldsymbol{x}_j$, then $L_{ij} = p + q - 2, p, q = 1, \ldots, K$ [16].

Definition 5: Inadaptability of ω_i is defined as follows:

$$a_i = \frac{1}{d} \sum_{j=1}^{d} \frac{\lambda_{ij}}{\bar{\lambda}_{ij}}, \tag{4}$$

where λ_{ij} is the j^{th} element of $\boldsymbol{\lambda}_i$, $\bar{\lambda}_{ij} = \frac{1}{K} \sum_{t \in \{i_1, \ldots, i_K\}} \lambda_{tj}$, $\boldsymbol{\lambda}_t$ $(t = i_1, \ldots, i_K)$ is the vector of eigenvalues of S_t. i_l $(l = 1, \ldots, K)$ is the subscript of $\boldsymbol{x}_{i_l}$ which is the l^{th} nearest neighbor of $\boldsymbol{x}_i$.

3.2 Mutual Neighborhood Graph Construction

The first step is mutual neighborhood graph construction: Calculate the Euclidean distance of each pair of $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$; Calculate the mutual neighborhood

value $L = \{L_{ij}\}$; Find the K nearest neighbors for each point $\boldsymbol{x}_i$ according to mutual distance L_{ij}.

This is a K-NN method which is suitable for our algorithm, when K is fixed, the eigenvalues of a local covariance matrix represent the distribution of the local data, which helps us to learn knowledge from the data. Furthermore, mutual nearest neighbor distance is used instead of Euclidean distance. As we know, MNN distance contains important knowledge for nonlinear problems and makes it easier for the points with similar densities to cluster together. We regard local density as important knowledge for clustering, so MNN distance is more suitable here than traditional Euclidean distance.

3.3 Distance Metric Learning by Knowledge Embedding

The second step is to discover knowledge from the mutual neighborhood graph. After mutual neighborhood graph construction, the neighborhood of each data point is identified. As a result, local covariance matrix S_i and its eigenvalues $\boldsymbol{\lambda}_i$ can be computed. Then we get local knowledge for each point and the knowledge embedded space is formed by combine $\boldsymbol{x}_i$ with $\boldsymbol{\lambda}_i$: $\boldsymbol{y}_i = \begin{bmatrix} \boldsymbol{x}_i \\ \boldsymbol{\lambda}_i \end{bmatrix}$, where $\boldsymbol{y}_i$ is the corresponding point of $\boldsymbol{x}_i$ in knowledge embedded space. In the next subsections two steps of denoising are performed which are the pivotal steps in NK algorithm.

The distance metric used in our algorithm is MNN distance in the knowledge embedded space rather than Euclidean distance in input space. If two points are close in this distance metric, it means that in the input space, their coordinates are close and local features are similar. How this metric is used for clustering will be discussed in section 5.

3.4 Denoising of Input Data

When data set has some background noise, clustering becomes difficult. We should remove background noise from data set. By the use of λ knowledge, this task can be done easily. As we know, background noise is usually very sparse. So its eigenvalues of local covariance matrix are much larger than other points. Then, background noise can be removed by using a threshold $E(\boldsymbol{\lambda}_i) + P_{noise} * D(\boldsymbol{\lambda}_i)$, where $E(\boldsymbol{\lambda}_i)$ and $D(\boldsymbol{\lambda}_i)$ are the mean and variance of $\boldsymbol{\lambda}_i$ respectively, and P_{noise} is a parameter. How to choose P_{noise} will be discussed in section 5.

3.5 Denoising of $\boldsymbol{\lambda}$ Knowledge

In neighborhood graphs, there are often some edges connecting two points that are not actually neighbors. These are called false edges. In Fig. 1, the edges between layers are false edges. Where there is a false edge, the corresponding eigenvalues of the local covariance matrix cannot represent the correct local feature of the neighborhood. We consider this as some kind of noise of λ knowledge. An efficient algorithm must be used to remove the false edges and this is called the denoising of λ knowledge.

Inadaptability a_i is defined for false edge finding. In Fig. 2, $\boldsymbol{x}_i$ is a point with false edges. ω_i is the corresponding neighborhood. $\boldsymbol{x}_j$ is a neighbor of $\boldsymbol{x}_i$ who has no false edges and ω_j is $\boldsymbol{x}_j$'s neighborhood. $\boldsymbol{\lambda}_i$ and $\boldsymbol{\lambda}_j$ are corresponding eigenvalues. Obviously, the elements of $\boldsymbol{\lambda}_i$ tend to be larger than $\boldsymbol{\lambda}_j$'s. So neighborhoods with false edges can be found by comparing their eigenvalues. There may be clusters with different densities and eigenvalues in a sparse cluster will be larger than those in a dense cluster. So we compare eigenvalues only between neighbors and define inadaptability as definition 5, where λ_{ij} is the j^{th} element of $\boldsymbol{\lambda}_i$, $\bar{\lambda}_{ij}$ is the mean of the j^{th} eigenvalue of $\boldsymbol{\lambda}_{i_l}(l = 1, \ldots, K)$. So the similarity between ω_i and ω_{i_l} can be measured by a_i. If ω_i and ω_{i_l} are similar, a_i is small; otherwise a_i is large. Then ω_i with false edges whose corresponding a_i is larger than a threshold is selected. $E(a_i)+P_{false}*D(a_i)$ is used as the threshold, where $E(a_i)$ and $D(a_i)$ are the mean and variance of $a_i \quad (i = 1, \ldots, N)$ and P_{false} is a parameter. How to choose P_{false} will be discussed in section 5.

Each ω_i with false edges is denoised by a steepest descent method. First, calculate the mean of the neighborhood $\boldsymbol{m}_i = \frac{1}{K+1} \sum_{l=0}^{K} \boldsymbol{x}_{i_l}$. Then, calculate the distance between $\boldsymbol{x}_{i_l}$ and $\boldsymbol{m}_i$. Next, remove the point with the maximal distance from ω_i and repeat this procedure until the inadaptability on the rest points is smaller than the threshold $E(a_i) + P_{false} * D(a_i)$.

After denoising, a well constructed mutual neighborhood graph is obtained. It is called a denoised mutual neighborhood graph.

3.6 Clustering in Knowledge Embedded Space

In the last step, we cluster data in knowledge embedded space. Start from arbitrary node $\boldsymbol{x}_i$, find its K nearest neighbors, $\boldsymbol{x}_{i_1}, \ldots, \boldsymbol{x}_{i_K}$. Then find the K nearest neighbors of each $\boldsymbol{x}_{i_l}$, etc. Combining all these points together results in a cluster. All the clusters can be obtained in the same way. It is obvious that the number of clusters is determined automatically.

3.7 Summarization of the Algorithm

The algorithm is presented in detail in Table 1.

4 Experiments

4.1 Clustering

Fig. 1–4 are experiment results of a two-layer Swiss roll which is a typical nonlinear problem. In the first step, there were many false edges between layers. All the points will be clustered into one cluster, see Fig. 1. Our algorithm can cluster all the points correctly, because it uses a denoising trick to remove the false edges, see Fig. 4.

Fig. 5–8 are experiments on data sets of many clusters with different shapes, densities, sizes and also with some background noise. The data sets comes from

Table 1. NK algorithm

Step 1: Mutual neighborhood graph construction

- Calculate the Euclidean distance matrix $D = \{d_{ij}\}, i, j = 1, \ldots, N$.
- Calculate the mutual neighborhood value matrix, $L = \{L_{ij}\}$.
- Find the K nearest neighbors for each point $\boldsymbol{x}_i$ under MNN distance.

Step 2: Knowledge embedding

- Calculate local covariance matrix S_i of each point $\boldsymbol{x}_i$.
- Calculate eigenvalues $\boldsymbol{\lambda}_i$ of S_i.

Step 3: Denoising of input data.

- Calculate the threshold $E(\boldsymbol{\lambda}_i) + P_{noise} * D(\boldsymbol{\lambda}_i)$ and remove noise points.
- Construct a new mutual neighborhood graph on the denoised input data.

Step 4: Denoising of λ knowledge.

- Recalculate eigenvalues $\boldsymbol{\lambda}_i$ of each point $\boldsymbol{x}_i$.
- Calculate inadaptability a_i of each point $\boldsymbol{x}_i$.
- Calculate the threshold $E(a_i) + P_{false} * D(a_i)$ and select out ω_i with false edges whose corresponding a_i is larger than the threshold $E(a_i) + P_{false} * D(a_i)$.
- Denoising ω_i with false edges by a steepest descent method:

  a) calculate the mean of the neighborhood $\boldsymbol{m}_i = \frac{1}{K+1} \sum_{l=0}^{K} \boldsymbol{x}_{i_l}$

  b) calculate the distance d_i^l between $\boldsymbol{x}_{i_l}$ and $\boldsymbol{m}_i$,$\boldsymbol{x}_{i_l}$ is the l^{th} neighbor of $\boldsymbol{x}_i$.

  c) Find $\boldsymbol{x}_{i_s}$ whose corresponding d_i^s is the maximum in all $d_i^l, l = 1, \ldots, K$. Remove the point $\boldsymbol{x}_{i_s}$ from ω_i and remove the point $\boldsymbol{x}_i$ from ω_{i_s}. It means to break the edges between $\boldsymbol{x}_i$ and $\boldsymbol{x}_{i_s}$

  d) Repeat c) until the inadaptability on the rest points is smaller than the threshold.

Step 5: Clustering in knowledge embedded space.

[17]. All the points can be correctly clustered and background noise is removed. To keep the figure legible, only parts of the points are shown and background noise is not shown.

4.2 Enhanced Isomap

The algorithm was also used to construct the neighborhood graph for ISOMAP [18]. For complex data sets, there are often some false edges in the neighborhood graph which prevent ISOMAP from reducing dimensionality correctly. After removing these false edges, ISOMAP was able to find the structure of complex data sets much more accurately (see Fig. 9).

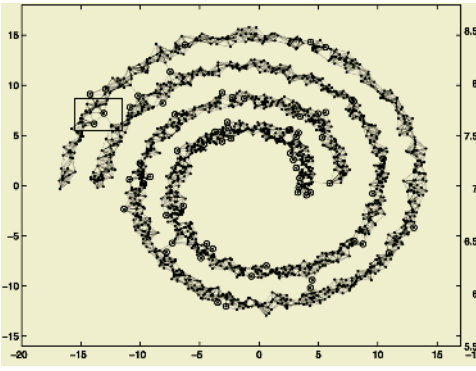

Fig. 1. Mutual neighborhood graph of a two-layer Swiss roll of 2000 points

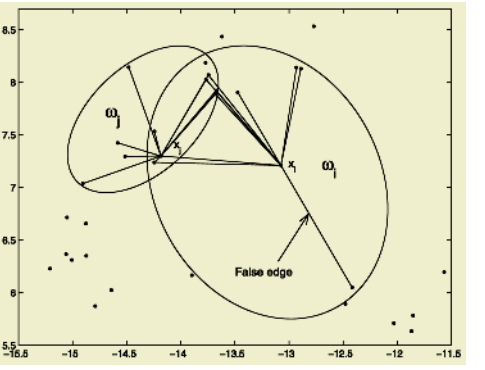

Fig. 2. The rectangle area in Fig. 1

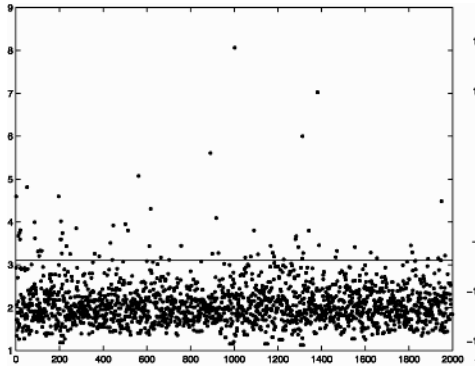

Fig. 3. Inadaptability of all points. The horizontal line is the threshold

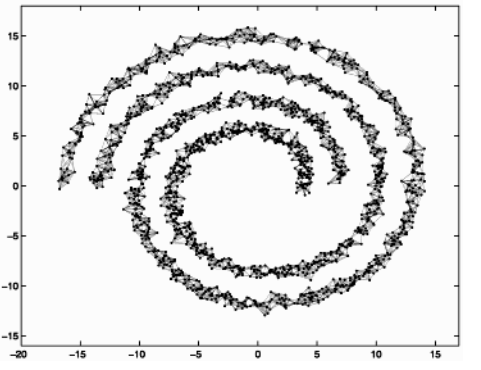

Fig. 4. Mutual neighborhood graph after denoising of λ knowledge. $K = 10, P_{noise} = 8, P_{false} = 2$

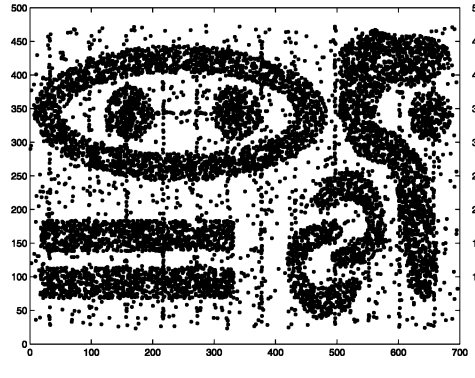

Fig. 5. A data set of 10000 points

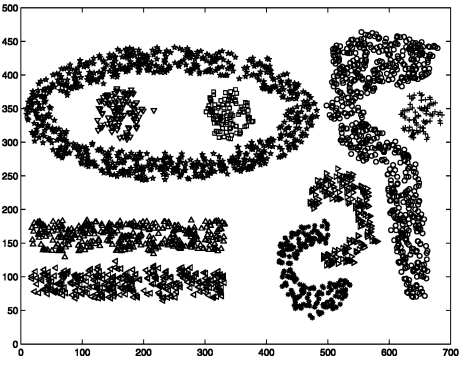

Fig. 6. Clustering result of Fig. 5. $K = 11, P_{noise} = 0.8, P_{false} = 1$

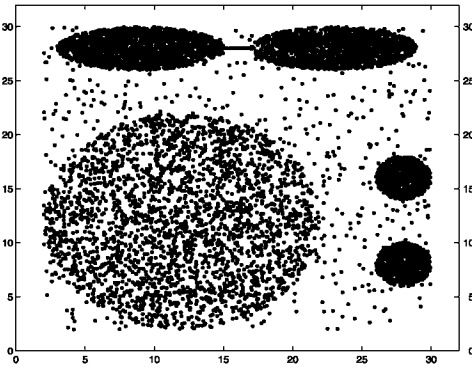

Fig. 7. A data set of 8000 points

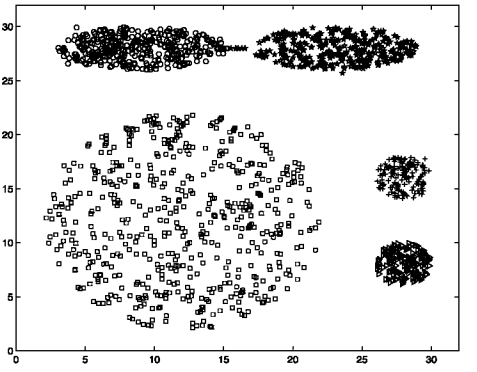

Fig. 8. Clustering result of Fig. 7. $K = 6, P_{noise} = 1, P_{false} = 2$

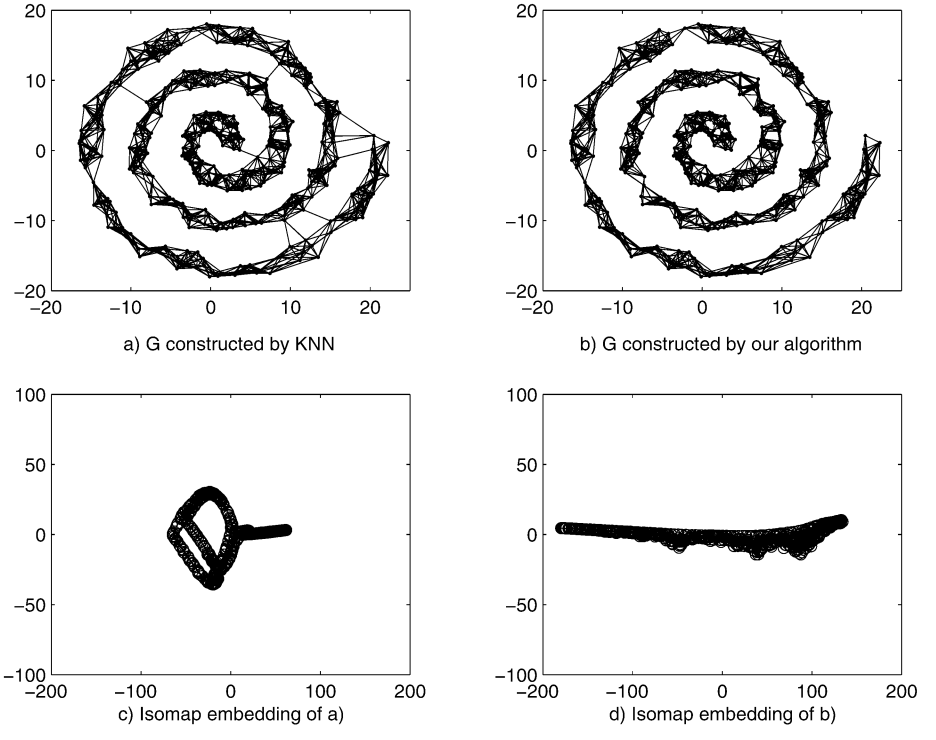

Fig. 9. Enhanced ISOMAP

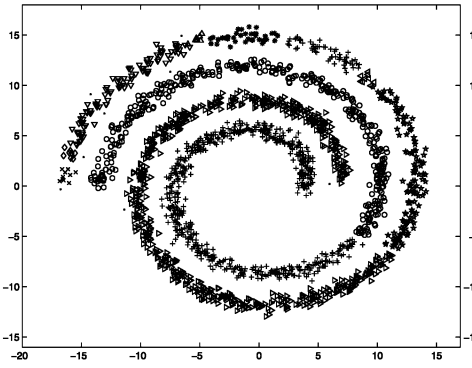

Fig. 10. $Eps = 0.772$

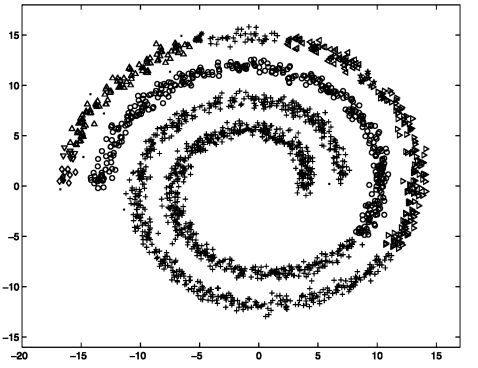

Fig. 11. $Eps = 0.773$

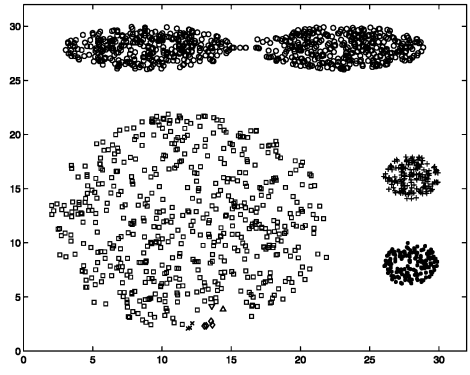

Fig. 12. $Eps = 0.5$

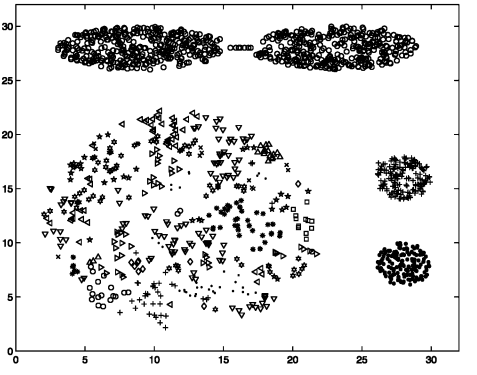

Fig. 13. $Eps = 0.4$

4.3 Compare with DBSCAN

DBSCAN [4] is a well-known spatial clustering algorithm that has been shown to find clusters of arbitrary shapes. We have done some experiments to compare NK algorithm and DBSCAN.

In most of our experiments, DBSCAN works very well, but it fails to perform well in some cases while NK algorithm can work well. Fig. 10-13 are some results of DBSACN. Following the recommendation of [4], the $MinPts$ was fixed to 4 and Eps was changed in these experiments. Fig. 10 is the best result of DBSCAN with $Eps = 0.772$ and there are 11 clusters. If Eps is increased to 0.773, some part of Swiss roll of different layers will be clustering into the same clustering, see Fig. 11. In Fig. 12 and Fig. 13 the data set contains clusters of different densities and the figures illustrate that DBSCAN cannot effectively find clusters of different densities [7,19] while NK algorithm works well on the same data set. To keep the figure legible, only parts of the points are shown, and background noise is not shown.

4.4 Run-Time Analysis

The overall computational complexity of NK algorithm mainly depends on the amount of time it requires to compute nearest neighbors and compute eigenvalues of local covariance matrixes. The complexity of computing nearest neighbors is $O(dN^2)$ [9]. However, some other nearest neighbors computing algorithms such as K-D trees can be used to compute the neighbors in time $O(N \log N)$ [20]. Computing the eigenvalues of one local covariance matrix scales as $O(d^3)$ [9]. As a result, the overall complexity of NK algorithm is $O(dN^2 + d^3N)$. It will be greatly sped up if a faster neighbors computing algorithm is used.

We have implemented NK algorithm in MATLAB, running on a Pentium 4 2.0GHz processor. Table 2 gives the actual running times of the algorithm on different data sets of the same dimension of 2.

Table 2. Running time in seconds

Data Set	Size	Graph construction	Denoising	Overall
1	2000	2.4	0.8	3.2
2	4000	11.1	1.6	12.7
3	6000	27.5	2.5	30
4	8000	41	4	45

5 Discussion

Distance metrics are very important in many learning and data mining algorithms. MNN distance in knowledge embedding space is used as the new distance metric in NK algorithm. Here some explanation of how this distance metric is used for clustering will be given. If two points $\boldsymbol{y}_i$ and $\boldsymbol{y}_j$ are close under this distance metric, that is $\boldsymbol{x}_i$ is close to $\boldsymbol{x}_j$ and $\boldsymbol{\lambda}_i$ is close to $\boldsymbol{\lambda}_j$, it means that in the input space, the coordinates of these two points are close and their local features are similar. In mutual neighborhood graph construction, each point is connected with its neighbors. This ensures that $\boldsymbol{x}_i$ is close to $\boldsymbol{x}_j$ if they are neighbors. Although, corresponding $\boldsymbol{\lambda}_i$ and $\boldsymbol{\lambda}_j$ are not always close since there are some false edges. So denoising is performed to remove false edges and after denoising, $\boldsymbol{\lambda}_i$ and $\boldsymbol{\lambda}_j$ become close. Because of this, in a denoised neighborhood graph, each pair of neighbors is close in knowledge embedded space. Note, the new distance metric is not used to measure the distance between each pair of points in the same cluster, but only the distance of neighbors.

There are three main parameters that are determined experimentally. The most important parameter is the number of neighbors K. The other two are P_{noise} and P_{false}. In practice, first decide P_{noise} and P_{false}, and then choose K. They can be chosen almost independently.

For data set without noise, P_{noise} should be a large number and easy to choose, such as 6 to 8 or even larger. If there is background noise, P_{noise} should be a small number, such as 0.1-1.5. We can set P_{noise} equal to 1 and then reduce it if not all the noise is removed or increase it if too many data points are removed. When choosing P_{noise}, the value of K is not important as long as it is not too small or too large. P_{false} is relative easy to choose. We can set $P_{false} = 2$ in most cases. If some points with false edges are not detected, then P_{false} should be smaller.

As we know, in neighborhood graph construction algorithm, K is difficult to choose, especially, when data set are complex. In our algorithm, K is easier to choose than many other algorithms profiting from the denoising trick. When false edges can't be removed, even if the points with false edges are detected, K should be smaller. When a data set is sparse or it is asymmetrical, sometimes a cluster will break where the data is very sparse. At this time, increasing K has some effect, but not a thorough solution. This is still a problem to be solved. Most of other clustering algorithm will still fail in the same environment. In our experiments, NK algorithm works much better than DBSCAN when the data set is sparse or it is asymmetrical.

6 Conclusion

This paper presents a new algorithm for clustering by knowledge embedding. Mutual neighborhood graphing is used for knowledge discovery and clustering. Eigenvalues of local covariance matrixes are used to represent local features. Denoising is needed because data sets are usually complex. The experiment results show that the algorithm can construct a quality neighborhood graph from a complex and noisy data set and it efficiently solves clustering problems.

Acknowledgements

We would like to thank Zhongbao Kou, Baibo Zhang, Shifeng Weng, Jun Wang, Tao Ban, and Jian'guo Li for a number of helpful discussions and suggestions.

References

1. Jain, A.K., Dubes, R.C.: Algorithms for Clustering Data. Prentice Hall (1988)
2. Kaufman, L., Rousseeuw, P.J.: Finding Groups in Data: an Introduction to Cluster Analysis. John Wiley & Sons (1990)
3. Ng, R., Han, J.: Efficient and effective clustering method for spatial data mining. In Proc. of the 20th VLDB Conference, Santiago, Chile (1994) 144-155
4. Martin Ester, Hans-Peter Kriegel, Jörg Sander, Xiaowei Xu: A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise. KDD (1996) 226-231
5. Sudipto Guha, Rajeev Rastogi, Kyuseok Shim: CURE: An efficient clustering algorithm for large databases. In Proc. of 1998 ACM-SIGMOD Int. Conf. on Management of Data (1998)

6. Sudipto Guha, Rajeev Rastogi, Kyuseok Shim: ROCK: a robust clustering algorithm for categorical attributes. In Proc. of the 15th Intl Conf. on Data Eng. (1999)
7. Karypis, G., Han, E., Kumar, V.: CHAMELEON: A Hierarchical Clustering Algorithm Using Dynamic Modeling. IEEE Computer, 32, (1999) 68-75
8. Roweis, S.T., Saul, L.K.: Nonlinear dimensionality reduction by locally linear embedding. Science, Vol. 290 (2000) 2323-2326
9. Saul, L.K., Roweis, S.T.: An introduction to locally linear embedding. Tech. rep., AT&T Labs - Research (2001)
10. Fukunaga, K., Olsen, D.R.: An algorithm for finding intrinsic dimensionality of data. IEEE Transactions on Computers, Vol. 20 (1971) 176-183
11. Pettis, K., Bailey, I., Jain, T., Dubes, R.: An intrinsic dimensionality estimator from near-neighbor information. IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 1 (1979) 25-37
12. Kambhatla, N., Leen, T.K.: Dimension reduction by local principal component analysis. Neural Computation, Vol. 9, num. 7 (1997) 1493-1516
13. Schölkopf, B., Smola, A.J., Müller, K.R.: Nonlinear Component Analysis as a Kernel Eigenvalue Problem. Neural Computation, 10 (1998) 1299-1319
14. Schölkopf, B., Mika, S., Burges, C.J.C., Knirsch, P., Müller, K.R., Raetsch, G., Smola, A.: Input Space vs. Feature Space in Kernel-Based Methods. IEEE Trans. on NN, Vol. 10, No. 5 (1999) 1000-1017
15. Schölkopf, B., Smola, A.J.: Learning with Kernels: Support Vector Machines, Regularization and Beyond. Cambridge, Massachusetts: MIT Press (2002)
16. Jain, A.K., Robert, P.W., Duin, Jianchang Mao: Statistical Pattern Recognition: A Review. IEEE Transactions on Pattern Analysis and Machine Intelligence (1999)
17. Harel, D., Koren, Y.: Clustering Spatial Data Using Random Walks. Proceedings of The 7th ACM Int. Conference on Knowledge Discovery and Data Mining (KDD'01). ACM press (2001) 281–286
18. Tenenbaum, J.B., Silvam, V.de., Langford J.C.: A global geometric framework for nonlinear dimensionality reduction. Science, Vol. 290 (2000) 2319-2323
19. Osmar, R., Zaïane, Andrew Foss, Chi-Hoon Lee, Weinan Wang: On Data Clustering Analysis: Scalability, Constraints and Validation, Sixth Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD'02), Taipei, Taiwan, May (2002)
20. Friedman, J.H., Bentley J.L., Finkel R.A.: An algorithm for finding best matches in logarithmic expected time. ACM Transactions on Mathematical Software, Vol. 3 (1997) 209-226