Nonlinear Manifold Clustering By Dimensionality

Wenbo Cao, Robert Haralick
Computer Science
Pattern Recognition Lab
The Graduate Center, City University of New York
365 Fifth Avenue, New York, NY, 10016 U.S.A.
wcao@gc.cuny.edu, haralick@ptah.gc.cuny.edu

Abstract

Because of variable dependence, high dimensional data typically have much lower intrinsic dimensionality than the number of its variables. Hence high dimensional data can be expected to lie in (nonlinear) lower dimensional manifold. In this paper, we describe a nonlinear manifold clustering algorithm. By connecting data vectors with their neighbors in feature space, we construct a neighborhood graph from given set data vectors. Furthermore, geometrical invariance, namely dimensionality, are extracted from the neighborhood of vectors, and used to facilitate the clustering procedure. In addition, we discuss a latent model for data cluster descriptions and an EM algorithm to find such descriptions. Preliminary experiments illustrate that this new algorithm can be used to explore the nonlinear structure of data.

1 Introduction

Clustering algorithms have been extensively used in pattern recognition and machine learning problems. In essence, the goal of clustering algorithm is to categorize internally similar data samples into the same group. Traditional clustering algorithms, for example the K-means algorithm, usually measure similarity by the Minkowski metric which gives equal weight to each dimension. However, many high dimensional data sets often have much lower intrinsic dimensionality, which implies that those data sets indeed lie on lower dimensional manifolds. Therefore one should take into account the geometric structure of data when measuring similarity. Recently some clustering algorithms have been proposed to take the advantage of the geometric structure of data sets, such as subspace clustering algorithms ([1, 3]) and linear manifold clustering algorithms ([2, 7]). Spectral clustering algorithms have been successfully used in computer vision. In the light of recent development of manifold learning ([4]), spectral clustering algorithms ([11, 12, 13]) can be viewed as (1) first finding a low dimensional manifold representation for the data vectors, and (2) then partitioning the data vectors on the manifold into different clusters.

In this study, we extend linear manifold clustering to more general cases, i.e. nonlinear manifold clustering. It is our belief that many data sets have intrinsic nonlinearity, and when data vectors are ample to support nonlinearity, one should explore the nonlinearity of the data set. In this study we assume each cluster forms a nonlinear manifold. As we know, a $m-$dimensional manifold has exact dimensionality m ([10]). Therefore vectors with different dimensional neighborhoods must lie on different manifolds, and should belong to different clusters. On the other hand, nearby vectors with the same dimensional neighborhoods have the possibility of being in the same manifold and the same cluster.

Our paper is organized as follows: in section 2, we formalize the description of data clusters and give an algorithm to find the description of clusters; in section 3, we describe the algorithm to cluster data; in section 4, we show some clustering results of synthetic and real data sets; finally in section 5 we conclude the paper and give the plan of our future work.

2 Data Clusters

Given a set of data vectors $X = \{\mathbf{x}_i | i = 1, 2, \cdots, N, \mathbf{x}_i \in \mathbb{R}^D$, the task of the clustering algorithm is to find a set of clusters $\mathcal{C} = \{C_1, C_2, \cdots, C_J\}$ such that $X = C_1 \cup C_2 \cup \cdots \cup C_J$. Here we assume that each vector in X must belong to at least one cluster. In this section, we introduce our model for data clusters, and formalize the notation of nonlinear data manifold. In this section, we shall denote a data cluster by C and $C = \{\mathbf{y}_i | \mathbf{y}_i \in X, i = 1, 2, \cdots, I\}$.

Proceedings of the 18th International Conference on Pattern Recognition (ICPR'06)
0-7695-2521-0/06 $20.00 © 2006 IEEE

2.1 Cluster Description

High dimensional data are often highly correlated, and the intrinsic dimensionality of high dimensional data is usually much lower than it appears to be. To describe the data more compactly, we propose a latent model as follows. Given a data cluster C, there exists a set of latent variables, $T = \{\mathbf{t}_i | i = 1, 2, \cdots, I, \mathbf{t}_i \in \mathbb{R}^d\}$; and $\mathbf{y}_i$ can be expressed by a (nonlinear) function of latent variable $\mathbf{t}_i$, i.e.

$$\mathbf{y}_i = \mathbf{f}(\mathbf{t}_i). \tag{1}$$

where $i = 1, 2, \cdots, I$ and $\mathbf{f}(\mathbf{t}_i) = (f_1(\mathbf{t}_i), f_2(\mathbf{t}_i), \cdots, f_D(\mathbf{t}_i))^T$. The (nonlinear) data manifold can be defined as $\mathcal{M} = \{\mathbf{f}(\mathbf{t}) | \mathbf{t} \in \mathbb{R}^d\}$. For example, if $\mathbf{f}$ is a vector of linear functions, then $\mathcal{M}$ is a $d-$dimensional hyperplane in $D-$dimensional space. Generally speaking, it is hard to choose an analytic form for $\mathbf{f}$. Nevertheless, under certain conditions, $\mathbf{f}$ can be approximated well enough by the combination of some known functions. In this study we assume that $\mathbf{f}$ can be expanded as a linear function in terms of finite number of basis functions[1]. That is, given a finite set of basis function

$$\mathcal{B} = \{b_1(\mathbf{t}), b_2(\mathbf{t}), \cdots, b_J(\mathbf{t})\}. \tag{2}$$

$\mathbf{f}(\mathbf{t})$ can be expressed as

$$\mathbf{f}(\mathbf{t}) = A(b_1(\mathbf{t}), b_2(\mathbf{t}), \cdots, b_J(\mathbf{t}))^T, \tag{3}$$

where A is $D \times J$ coefficient matrix.

Basis functions can be polynomial, monomial, logrithms, or radial basis functions ([8]).

In this study, we choose the second order polynomial. That is, for a given $\mathbf{t}$, $\mathcal{B} = \{\mathbf{t}^i\mathbf{t}^j | i, j = 1, 2, \cdots d\} \cup \{\mathbf{t}^i | i = 1, 2, \cdots d\} \cup \{1\}$, where $\mathbf{t}^i$ is the $i - th$ component of $\mathbf{t}$.

2.2 Algorithm

Given a cluster of samples $\mathbf{C}$, we can obtain the description of the cluster by solving

$$\min \sum_{i=1}^{I} \|\mathbf{y}_i - A * g(\mathbf{t}_i)\|^2 \tag{4}$$

where $g(\mathbf{t}) = (b_1(\mathbf{t}), b_2(\mathbf{t}), \cdots, b_J(\mathbf{t}))^T$. To measure the goodness-of-fit, we define the mean square error (*MSE*) as

$$\gamma = \frac{1}{I} \sum_{i=1}^{I} \|\mathbf{y}_i - \hat{A} * g(\hat{\mathbf{t}}_i)\|^2, \tag{5}$$

where $\hat{A}$ and $\hat{\mathbf{t}}_i$ are estimated by equation 4

Our algorithm is as follows:

[1] To interpret this model in terms of kernel method [9], we can think the latent manifold is simple enough that can be expressed as a linear manifold in the feature space of the latent variable, where the mapping function can be derived by the chosen basis function.

1. initialize $\mathbf{t}_i$ randomly, denote it by $\mathbf{t}_i^0$;
2. initialize iteration count $j \leftarrow 0$;
3. repeat until the change of MSE is smaller than a threshold value ϵ
   (a) $A^j \leftarrow \arg\min_A \sum_{i=1}^{I} \|\mathbf{y}_i - A * g(\mathbf{t}_i^{j-1})\|^2$;
   (b) $\mathbf{t}_i^j \leftarrow \arg\min_{\mathbf{t}_i} \|\mathbf{y}_i - A^j * g(\mathbf{t}_i)\|^2$;
   (c) $j \leftarrow j + 1$;
   (d) $\gamma^j \leftarrow \frac{1}{I} \sum_{i=1}^{I} \|\mathbf{y}_i - A^j * g(\mathbf{t}_i^j)\|^2$,

In spirit, our algorithm is an EM algorithm. Therefore it guarantees to converge to an optimal solution, though sometimes it may converge to local optimum.

3 Clustering Algorithm

We argue that a good similarity measure for clustering algorithms should consider not only the statistical properties but also the geometrical properties of given data samples. Traditional similarity measure often only reflects people's belief about the statistical properties of given data samples, but ignore the geometrical properties of data samples. For example, The Euclidean distance is widely used as one feature of similarity among data vectors: the nearer two vectors are, the more similar they are and the more likely they belong to the same cluster. Nearest neighbor algorithms implements this belief by assigning vectors to their closest cluster centers, i.e. the most likely clusters. Graph theoretical algorithms enforce this idea by connecting data vectors only when their distance (or the rank of their distance) is smaller than a threshold value. However, we believe that, in additional to statistical properties, high dimensional data sets often exhibit geometrical structures. Similar data samples should have similar geometrical properties. Our algorithm reflects these beliefs. It is based on a single-linkage graph theoretical clustering algorithm, and uses geometrical information to refine the graph and make more reasonable clustering decisions.

To derive statistical and geometrical properties of data vectors, we need to introduce the definition of neighborhood for each vector. From a statistical standpoint, defining a local area enables us to enforce a certain level of confidence on whether data vectors belong to one cluster. From a geometrical standpoint, we assume that the data manifolds investigated in this study are smooth enough; therefore locally they can be approximated by linear manifolds, and we can use well established linear methods to obtain the geometrical properties. The neighborhood of a sample is defined as : given a data set $X = \{\mathbf{x}_i \in \mathbb{R}^D | i = 1, 2, \cdots, N\}$, the neighbor set of a given sample may be

$k-$**neighborhood** $\mathcal{N}_i(k)$ is a set of points that contains k nearest points of $\mathbf{x}_i$;

$r-$**neighborhood** $\mathcal{N}_i(r) = \{\mathbf{x}_j | \|\mathbf{x}_i - \mathbf{x}_j\| < r, \mathbf{x}_j \in X\}$.

In the following discussing, we shall not differentiate these two notations, and denote the neighborhood of sample $\mathbf{x}_i$ by $\mathcal{N}_i$.

With no prior information, we think that nearby vectors are more likely to be in the same clusters than vectors far away from each other. Therefore, we construct a weighted undirected graph $G =< V, E >$ from the given data set X as follows:

1. $V = X$, i.e. that vertex set of G is the given data set X;
2. $E = \{\mathbf{x}_i\mathbf{x}_j | \mathbf{x}_i \in \mathcal{N}_j\}$, that is two vectors, $\mathbf{x}_i$ and $\mathbf{x}_j$, are connected if and only if one is in the neighborhood of the other;
3. the weight of edge $\mathbf{x}_i\mathbf{x}_j$, w_{ij}, is the Euclidean distance between $\mathbf{x}_i$ and $\mathbf{x}_j$.

When the data manifold is smooth enough, it can be considered as locally linear. That is, for a small patch of the data manifold, we can use a linear manifold to approximate it. Therefore the intrinsic dimensionality of the data manifold can be estimated by the well known principal component analysis method (*PCA*). To estimate the dimensionality of a given data sample $\mathbf{x}_i$, we simply apply PCA on all samples in $\mathcal{N}_i$, and obtain the number of significant eigenvalues as the dimensionality of $\mathbf{x}_i$. Denote it by $dim(\mathbf{x}_i)$.

The dimensionality of a data manifold is a geometrical invariant quantity for all vectors that lie on the manifold. We expect data vectors that are in the same cluster should have the same dimensional neighborhoods. Define two kinds of connected components of graph G,

1. A *heterogeneous connected component* (*HeC*) is a connected component of G, i.e. $HeC =< V', E' >$, where $V' \subseteq V$, and $\forall \mathbf{x}_i \in V', \mathbf{x}_j \in V'$, there exists a path that connects $\mathbf{x}_i$ and $\mathbf{x}_j$;
2. A *homogeneous connected component* (*HoC*) is a connected component of G, i.e. $HoC =< V'', E'' >$ and satisfies

$$\forall \mathbf{x} \in V'', dim(\mathbf{x}) = constant.$$

We then assume that, (1) data vectors that are in the same *HoC* are in the same cluster, and (2) data vectors that are in different *HeC*'s are in different clusters. *HeC*'s and *HoC*'s can be easily found by standard graph algorithms and therefore are not described here.

3.1 Separated Clusters

If no two clustes are *connected*, i.e. for any two clusters C_i and C_j of data set X, there are no two vectors $\mathbf{x}_i \in C_i$ and $\mathbf{x}_j \in C_j$ such that $\mathbf{x}_i$ and $\mathbf{x}_j$ are in the same *HeC*. For this case, finding clusters is an easy task, because each *HeC* is indeed a cluster.

3.2 Connected Clusters

A more interesting case is when clusters are not separated, i.e. they are connected. There might be many reasons for the existence of such kind of clusters. For example, noise can make separated clusters become non-separated. Or indeed, clusters may pass through each other. In this case, we cannot simply say that each *HeC* is a cluster. Nor can we say a *HoC* is a cluster. In the presence of noise, boundary effects and the estimation error of dimensionality, a cluster may consist two or more *HoC*'s.

Formally speaking, two connected clusters, C_i and C_j, means that there exists $\mathbf{x}_i \in C_i$, and $\mathbf{x}_j \in C_j$ such that $\mathbf{x}_i$ and $\mathbf{x}_j$ are in the same *HeC*. It is easy to show that C_i and C_j are in the same *HeC*.

Since by our assumption *HoC*'s in different *HeC*'s cannot belong to the same cluster, in the following discussion we only discuss *HoC*'s in the same *HeC*. Define the distance of two HoC's, HoC_1 and HoC_2, as

$$d(HoC_1, HoC_2) = \inf\{d_G(\mathbf{x}'_i, \mathbf{x}''_i) | \mathbf{x}'_i \in HoC_1, \mathbf{x}''_i \in HoC_2\}, \quad (6)$$

where $d_G(\mathbf{x}_i, \mathbf{x}_j)$ is the length of shortest path between verteice $\mathbf{x}_i$ and $\mathbf{x}_j$ in graph G. Also define $\gamma(HoC_1, HoC_2)$ the mean square error if we describe HoC_1 and HoC_2 by one cluster. We consider that two *HoC*'s are in the same cluster if

1. If the distance between them, $d(HoC_1, HoC_2)$, is less than a given value r;
2. If $\gamma(HoC_1, HoC_2)$ is less than a given value ϵ.

To automatically select a threshold value ϵ for MSE, we first find a cluster description for HoC_1 and calculate the MSE; denoted it by $\gamma(HoC_1)$. Then we find a cluster description for HoC_2 and calculate the MSE; denote it by $\gamma(HoC_2)$. If HoC_1 and HoC_2 belong to the same cluster, then $\gamma(HoC_1, HoC_2)$ should be comparable with $\gamma(HoC_1)$ and $\gamma(HoC_2)$. In practice, we choose $\epsilon = \alpha \max(\gamma(HoC_1), \gamma(HoC_2))$, where $\alpha > 1$.

4 Experiment

In this section, we present the results of applying our algorithm on synthetic and real data sets. We shall use ac-

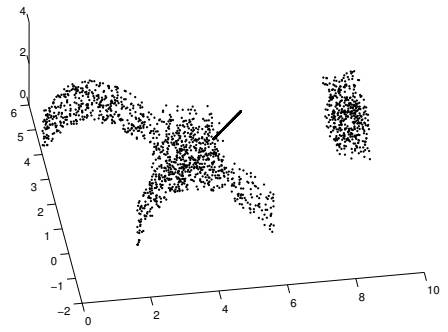

Figure 1. Synthetical Data Generated: $M = 400, \sigma^2 = 0.005$

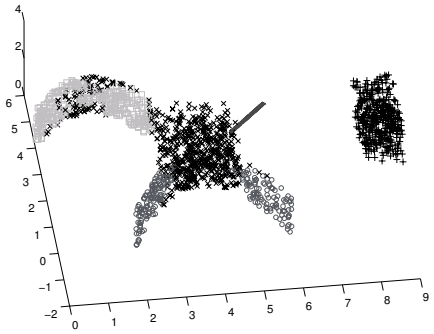

Figure 2. 5 **clusters are found:** $M = 400, \sigma^2 = 0.005$

	$\sigma^2 = 0.001$	$\sigma^2 = 0.005$	$\sigma^2 = 0.01$
$M = 200$	$0.63(0.03)$	$0.62(0.03)$	$0.60(0.02)$
$M = 300$	$0.82(0.01)$	$0.79(0.03)$	$0.70(0.04)$
$M = 400$	$0.90(0.01)$	$0.85(0.03)$	$0.77(0.04)$

Table 1. The mean and standard deviation of accuracy for different data size and noise

curacy to measure the performance of our algorithm. Accuracy is calculated by means of a *confusion matrix*. To simplify the calculation of accuracy, we require that one vector can only be assigned to one cluster [2].

4.1 Synthetic Data

As shown in figure 1, we randomly generate five clusters in $\mathbb{R}^3$. Of 5 clusters generated, 2 are $3D$ clusters, 2 are $2D$ clusters, and 1 is $1D$ cluster; one of the $2D$ clusters actually passes through a $3D$ cluster. Each cluster has M vectors and noise $\eta \sim \mathcal{N}(0, \sigma^2 I)$, where I is a 3×3 identity matrix. In figure 2, we show the clusters found by our algorithm. To test the effect of different size and noise on our algorithm, we repeat our experiments 100 times for several selected M and σ^2. The mean and standard derivation of accuracy is shown in table 1. These experiments show that (1) our algorithm can be used to explore the nonlinear structure of given data set (as shown in figure 2), and (2) with reasonable data vectors and noise level, our algorithm can be used to generate clusters within a certain level of accuracy.

[2]This option is used here solely for the sake of simplicity. More rationally, our algorithm should be modified to be a fuzzy clustering algorithm, i.e. one vector can appear in more than one cluster, since by assumption one cluster may pass through another cluster.

4.2 Real Data

To provide a real-world example we used the handwritten digit data set from UCI machine learning repository ([5]). From 3823 handwritten digits, we extract 364 samples of digit 7 and 336 samples of digit 8. Applying our algorithm on this data set, we obtained 4 clusters. The size of clusters 1, 2, 3 and 4 are 335, 271, 67, and 26, respectively. Cluster 1 contains only digit 8. And Clusters 3 and 4 only contain digit 7. Cluster 2 contains 1 digit 8 and 270 digit 7. In this experiment we generate 4 clusters instead of 2 clusters. Nonetheless, the result show that our algorithm can differentiate digits 7 and 8 well.

5 Conclusion

In this paper, we discussed a nonlinear manifold clustering algorithm. Using the geometrical properties of data, we described a new graph clustering algorithm. In the preliminary experiments, we show that our algorithm can be used to explore the nonlinearity of data. In the future we plan to investigate the applicability of our algorithm to real data, and do empirical comparisons to other clustering algorithms. Also, we plan to apply our algorithm for the purpose of dimensionality reduction.

References

[1] C. Aggarwal, C. Procopiuc, J. Wolf, P. Yu, and J. Park. Fast algorithms for projected clustering. In *Proceedings of the 1999 ACM SIGMOD international conference on management of data*, pages 61–72, 1999.

[2] C. Aggarwal and P. Yu. Finding generalized projected clusters in high dimensional spaces. In *Proceedings of the 2000 ACM SIGMOD international conference on management of data*, pages 70–81, 2000.

[3] R. Agrawal, J. Gehrke, D. Gunopulos, and P. Raghavan. Automatic subspace clustering of high dimensional data for data mining applications. In *Proceedings of the ACM SIGMOD international conference on management of data*, pages 94–105, 1998.

[4] M. Belkin and P. Niyogi. Laplacian eigenmaps and spectral techniques for embedding and clustering. *Neural Information Processing Systems*, 14:585–591, 2001.

[5] C. Blake and C. Merz. Uci repositroy of machine learning databases. http://www.ics.uci.edu/mlearn/MLRepository.html.

[6] R. Duda, P. Hart, and D. Stork. *Pattern Classification*. Wiley Interscience, 2nd edition, 2001.

[7] R. Haralick and R. Harpaz. Linear manifold clustering. In *4th International Conference on Machine Learning and Data Mining in Pattern Recognition*, pages 132–141, 2005.

[8] T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning*. Springer Verlag, New York, 2001.

[9] K. Müler, S. Mika, G. Rätsch, K. Tsuda, and B. Schölkopf. An introduction to kernel-based learning algorithms. *IEEE Transaction on Neural Networks*, 12(2):181–202, 2001.

[10] J. Munkres. *Topology*. Prentice Hall, 1999.

[11] A. Ng, M. Jordan, and Y. Weiss. On spectral clustering: analysis and an algorithm. *Advances in Neural Information Processing Systems*, 14:849–856, 2002.

[12] J. Shi and J. Malik. Normalized cuts and image segmentation. In *Proc. IEEE Conference of Computer Vision and Pattern Recognition*, pages 731–737, 1997.

[13] Y. Weiss. Segmentation using eigenvectors: a unifying view. In *International Conference on Computer Vision*, 1999.