Hindawi Publishing Corporation
Mathematical Problems in Engineering
Volume 2013, Article ID 295067, 9 pages
http://dx.doi.org/10.1155/2013/295067

Research Article

Smooth Splicing: A Robust SNN-Based Method for Clustering High-Dimensional Data

JingDong Tan[1,2,3] and RuJing Wang[2,3]

[1] *School of Mathematics, Hefei University of Technology, Hefei 230009, China*
[2] *Institute of Intelligent Machines, Chinese Academy of Sciences, Hefei 230031, China*
[3] *Department of Automation, University of Science and Technology of China, Hefei 230027, China*

Correspondence should be addressed to RuJing Wang; rjwang@iim.ac.cn

Received 8 October 2012; Revised 20 May 2013; Accepted 20 May 2013

Academic Editor: Jun Zhao

Sharing nearest neighbor (SNN) is a novel metric measure of similarity, and it can conquer two hardships: the low similarities between samples and the different densities of classes. At present, there are two popular SNN similarity based clustering methods: JP clustering and SNN density based clustering. Their clustering results highly rely on the weighting value of the single edge, and thus they are very vulnerable. Motivated by the idea of smooth splicing in computing geometry, the authors design a novel SNN similarity based clustering algorithm within the structure of graph theory. Since it inherits complementary intensity-smoothness principle, its generalizing ability surpasses those of the previously mentioned two methods. The experiments on text datasets show its effectiveness.

1. Introduction

In various fields, a vast amount of clustering algorithms have been developed for different types of application. Among these algorithms, there are no algorithm to adapt to all data types and applications. Studying on the clustering algorithm for specific data types or applications seems to be a never-ending process in the current situation. The status quo is that people already have a number of techniques, and they work well for some specific situations. The reason is that for the problem of what is a good set of classes, it is still based on subjective assumption and interpretation of the proponents of algorithm. When the researchers try to use an objective measure to a precise definition of class, they often find that the optimal clustering problem cannot be calculated at all. At present, the existing full-dimensional clustering methods can be roughly divided into these kinds: partition method [1–4], hierarchical method [5–8], density-based method [9, 10], grid-based method [11] and on-line clustering [12, 13]. On clustering high-dimensional data, the popular methods have been roughly divided into these kinds: subspace clustering, pattern-based clustering, and correlation clustering [11, 14, 15]. According to this categorization, the shared-nearest-neighbor approaches discussed in this paper can be referred to as soft-projected clustering algorithms probably, since they neither account for relevance/irrelevance of different attributes nor assign specific (hard) subspaces to the clusters. It seems that all the SNN-based clustering methods for high-dimensional data are marginal in the overview [15]. Nevertheless, recent research by Houle et al. indicates that the discrimination problems only occur when there are a high number of irrelevant dimensions, and that shared-nearest-neighbor approaches can often improve the clustering results in practice [16].

Text, image, and some other data are high-dimensional feature vectors. If we use the traditional k-means method or hierarchical clustering method for their clustering, due to the low similarity between the samples (usually using the cosine similarity), similarity becomes an unreliable guide for clustering. In addition, another common phenomenon of the clustering is the different densities of classes; that is to say, some within-class samples are rare, while the other within-class samples are dense. The standard cohesion metrics (such as SSE) adopted by traditional k-means clustering method

and hierarchical clustering are clearly inappropriate in this case [17]. For efficiently dealing with both cases, a more appropriate measure of proximity for clustering—sharing the nearest neighbor (SNN) similarity—has been presented. Regarding SNN similarity graph as a learning object, two different algorithms—JP clustering [18] and SNN density-based clustering [19, 20]—have been presented. Since the two algorithms are based on the concept of SNN similarity, as the literature [18] stated, they not only are good at dealing with noise and outliers but also can find the classes with different size, shape, and density. What needs to be particularly noted maybe is that they have a very strong high-dimensional data processing capability and specializes in finding compact class of the strongly correlated samples. Therefore these two algorithms are very appropriate alternatives for a certain task such as text clustering. However, they have a common drawback; that is, whether a set is split into two classes or remains unchanged; it may overall depend on the intensity of an edge, which makes them appear somewhat fragile.

Inspired by the idea of smooth splicing between two free curves or surfaces in the computational geometry (i.e., to maintain geometric continuity at connecting point, one keeps the left derivative and the right derivative in consistent with a specified order), the authors propose a new SNN similarity based clustering algorithm: smooth splicing. Compared to JP clustering and the SNN density based clustering algorithm, its robustness can be adjusted adaptively via the smoothness throughout the clustering process. Thus its generalization ability is also promoted accordingly.

The rest of paper will be organized as follows: in Section 2, the method of calculating SNN similarity is introduced; in Section 3, we describe two kinds of similarity-based SNN clustering algorithm in detail; in Section 4, we propose two novel definitions with their properties first, based on them a new clustering algorithm named smooth splicing is proposed; in Sections 5 and 6; the experiments on high-dimensional text datasets are conducted to validate the effectiveness of the proposed method; finally, the conclusions are given in Section 7.

2. SNN Similarity

As long as two samples are in each other's k nearest neighbor list, SNN similarity is the number of their shared neighbor. It is computed through Algorithm 1.

The graph describing SNN similarity of samples is called SNN similarity graph. Since many SNN similarities are zeros, the matrix corresponding to SNN similarity graph is usually very sparse.

SNN similarity is useful because it addresses some problems when one uses direct similarity. First of all, by using the sharing nearest neighbors, a particular environment in which samples lie is taken into account. For example, a sample and another sample are relatively close, but they belong to different classes. In this case, they generally do not share many neighbors, namely, their SNN similarity value is small. In addition, SNN similarity can also handle the problem of variable density. In the low-density regions, the samples stay further than those in high-density regions. However, the SNN similarity of a pair of samples only depends on the number of the shared nearest neighbors, rather than how far is between these neighbors. It can be seen that SNN similarity can automatically scale to the density of samples points.

(1) identify the k-nearest neighbor of all the sample points.
(2) if two points $\mathbf{x}$ and $\mathbf{y}$ are not in each other's k-nearest neighbor list, then
(3) similarity $(\mathbf{x}, \mathbf{y}) \leftarrow 0$
(4) else
(5) similarity $(\mathbf{x}, \mathbf{y}) \leftarrow$ the number of shared neighbors
(6) end if

Algorithm 1: The algorithm for calculating SNN similarity.

(1) Calculate SNN similarity graph.
(2) Use the threshold of similarity to thin out SNN similarity graph.
(3) Find the connected components (class) of new SNN similarity graph.

Algorithm 2: JP clustering algorithm.

3. SNN-Based Clustering

In theory, all the clustering algorithms using the general similarity can also adopt SNN similarity. However, considering the specific demand of issues and the characteristics of algorithms, there is no need to consider all possible combinations of them. At present, there are two popular SNN similarity based Clustering methods: JP clustering and SNN density based clustering.

3.1. JP Clustering. JP clustering algorithm adopts the SNN similarity, which is a type of proximity between the two points calculated by Algorithm 1. Then it uses a threshold to thin out SNN similarity matrix. Expressed with the terms in graph theory, it is to create and thin out SNN similarity graph. The final category is just the connected component of SNN similarity graph [21]. The framework of the algorithm is described as shown in Algorithm 2.

The time complexity of JP clustering algorithm is $O(n^2)$, and its storage complexity is $O(kn)$, where k is nearest neighbors and n is the total number of samples.

3.2. SNN Density Based Clustering. Incorporating the SNN density into the DBSCAN algorithm, one can derive a new clustering algorithm. This clustering algorithm also starts with SNN similarity graph, while instead of using threshold to thin out SNN similarity graph and then searching the connected component as class in JP clustering, it directly uses DBSCAN method for further clustering. The framework of the algorithm can be described as shown in Algorithm 3.

(1) Calculate SNN similarity graph.
(2) Specify the parameters ε and MinPts, and use DBSCAN method for clustering.

Algorithm 3: SNN density based clustering algorithm.

The algorithm can automatically determine the number of classes in data sets. Noise points and points that lack a strong connection to a group are discarded that is; to say, they do not participate in clustering. Therefore, SNN density-based clustering can find such categories that the sample points of class are highly related. In practice, it is a possible advantage, for example, clustering based on the SNN density can find the subject contained in the text data set in general.

4. Smooth Splicing

As is stated previously, as long as the two samples are in each other's nearest neighbor list, SNN similarity is the number of their shared neighbor. By means of the Algorithm 1, one can calculate the SNN similarity between the samples, which can be formed into a SNN similarity graph. However, regarding the SNN similarity graph as input, the two existing approaches mentioned previously have a common drawback; that is, a sample set is split into two classes or remains unchanged it may depend completely on an edge, which makes them appear somewhat vulnerable. For example, if there are three samples $\mathbf{x}_1$, $\mathbf{x}_2$, and $\mathbf{x}_3$, suppose that both the SNN similarity between $\mathbf{x}_1$ and $\mathbf{x}_2$ and the SNN similarity between $\mathbf{x}_2$ and $\mathbf{x}_3$ are large, yet the SNN similarity between $\mathbf{x}_1$ and $\mathbf{x}_3$ is zero; under these conditions if one uses JP clustering algorithm or SNN density-based clustering algorithm to cluster them, then $\mathbf{x}_1$, $\mathbf{x}_2$, and $\mathbf{x}_3$ generally are assigned to the same class. However, this is obviously not a good clustering. By intuition, if these three samples belong to the same class, then $\mathbf{x}_1$ and $\mathbf{x}_3$ should also have a relatively high SNN similarity, rather than zero. This intuition is similar to the view of smooth splicing in computing geometry. If two adjacent curves meet the higher order derivative continuity at the splice point, then the splicing is considered smoother. Intuitively it is more like a curve as a whole [22]. Such an example is presented in Figure 1.

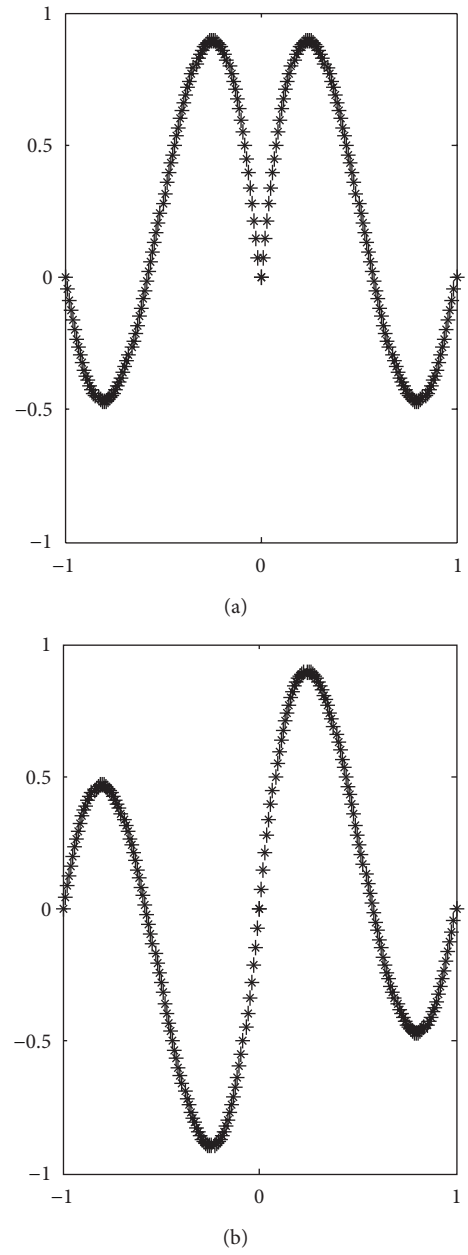

Figure 1: The splicing of two Bézier curves at $x = 0$ ((a) 0 is cusp, (b) smooth splicing).

In view of avoiding the clustering risk resulting from the drawback mentioned before (*Assumption* for clustering), in this section, we first propose the definition of SNN similarity-based n-order smoothness (*Heuristic* for clustering) and then propose the smooth splicing clustering algorithm.

4.1. Definitions and Properties

Definition 1. Suppose that there is a path formed with $2n + 1$ sample points $\{\mathbf{x}_{-n}, \ldots, \mathbf{x}_{-1}, \mathbf{x}_0, \mathbf{x}_1, \ldots, \mathbf{x}_n\}$ and it can be expressed as follows: $\mathbf{x}_{-n}\mathbf{x}_{-n+1} \cdots \mathbf{x}_{-1}\mathbf{x}_0\mathbf{x}_1 \cdots \mathbf{x}_{n-1}\mathbf{x}_n$. The intensity of the edge is the value of SNN similarity between two vertices (greater than zero). We regard this path as the splicing result of one path $\mathbf{x}_{-n}\mathbf{x}_{-n+1} \cdots \mathbf{x}_{-1}\mathbf{x}_0$ with another path $\mathbf{x}_0\mathbf{x}_1 \cdots \mathbf{x}_{n-1}\mathbf{x}_n$ at $\mathbf{x}_0$. If the SNN similarity between $\mathbf{x}_{-1}$ and $\mathbf{x}_1$ is not zero, then we claim that the initial path is 1 order smooth at $\mathbf{x}_0$; otherwise it is named as the 0 order smooth at $\mathbf{x}_0$; if the SNN similarity between two samples selected arbitrarily from $\{\mathbf{x}_{-2}, \mathbf{x}_{-1}, \mathbf{x}_1, \mathbf{x}_2\}$ is not zero, then we claim that the original path is 2 order smooth at $\mathbf{x}_0$; keep on going, if the SNN similarity between two samples selected arbitrarily from $\{\mathbf{x}_{-n}, \ldots, \mathbf{x}_{-1}, \mathbf{x}_1, \ldots, \mathbf{x}_n\}$ is not zero, then we claim that the original path is n order smooth at $\mathbf{x}_0$.

This definition formally is much like the definition of splicing smoothness in computational geometry. In fact, from

the definition it is not difficult to discover the following property about the smoothness.

Property 1. Suppose that a path is s order smooth at $\mathbf{x}_0$, and we select arbitrarily an element in $\{\mathbf{x}_{-s-1}, \mathbf{x}_{s+1}\}$ and arbitrarily an element in $\{\mathbf{x}_{-s}, \ldots, \mathbf{x}_{-1}, \mathbf{x}_1, \ldots, \mathbf{x}_s\}$, if the SNN similarity is not zero, then the original path is $s + 1$ order smooth at $\mathbf{x}_0$.

Definition 1 seems to be defined as the smoothness in a discrete format in computational geometry, and meanwhile Property 1 appears to satisfy some sort of magical symmetry. However, considering that one of paths for splicing is always an edge in the following proposed algorithm, therefore we have to slightly change the previous definition regretfully.

Definition 2. Suppose that there is a path formed with $n + 1$ sample points $\{\mathbf{x}_0, \mathbf{x}_1, \ldots, \mathbf{x}_n\}$ and the path can be expressed as $\mathbf{x}_0\mathbf{x}_1 \cdots \mathbf{x}_{n-1}\mathbf{x}_n$, where the intensity of the edge is the SNN similarity between its two vertices (greater than zero). We regard this path as the splicing result of one edge $\mathbf{x}_0\mathbf{x}_1$ with one path $\mathbf{x}_1 \cdots \mathbf{x}_{n-1}\mathbf{x}_n$ at $\mathbf{x}_1$. If the SNN similarity between $\mathbf{x}_0$ and $\mathbf{x}_2$ is not zero, then we claim that the original path is 1 order unilateral smooth at $\mathbf{x}_1$; otherwise it is named as the 0 order unilateral smooth; if the SNN similarity between $\mathbf{x}_0$ and $\mathbf{x}_3$ is also not zero, then we claim that the original path is 2 order unilateral smooth at $\mathbf{x}_1$; keep on going; if the SNN similarity between two arbitrary samples in $\{\mathbf{x}_0, \mathbf{x}_1, \ldots, \mathbf{x}_n\}$ is not zero, then we claim that the original path is n order unilateral smooth at $\mathbf{x}_1$.

It is obvious that unilateral smooth also has the similar property mentioned previous. In addition, the smoothness in the rest of paper will denote unilateral smoothness specially.

Property 2. If path $\mathbf{x}_0\mathbf{x}_1 \cdots \mathbf{x}_{n-1}\mathbf{x}_n$ is $n - 1$ order unilateral smooth at $\mathbf{x}_1$, $\mathrm{SNN}(\mathbf{x}_{-1}, \mathbf{x}_0) > 0$, the SNN similarities between $\mathbf{x}_{-1}$ and arbitrarily an element in $\{\mathbf{x}_1, \ldots, \mathbf{x}_s\}$ are not zero, then $\mathbf{x}_{-1}\mathbf{x}_0 \cdots \mathbf{x}_{n-1}\mathbf{x}_n$ is n order unilateral smooth at $\mathbf{x}_0$.

4.2. Algorithm Design. By means of Algorithm 1, we can calculate the SNN similarity between any two samples in the data set $\{\mathbf{x}_i \mid 1 \leq i \leq n\}$, which form an $n \times n$ SNN similarity matrix finally. From the matrix, we can find C_m^2 edges, where the intensity of each edge is the SNN similarity of its two vertices. In fact, due to high degree sparseness of SNN similarity matrix, many edges with zero intensity do not participate in clustering. If two paths contain the same sample point, then we can consider merging them. Specifically, we should check whether this splicing meets the s order smooth condition or not at splicing point (if the two are edge, then s can only take 1), if it is satisfied, then splicing performs; otherwise, everything remains unchanged. In addition, since splicing needs to satisfy some degree of smoothness, thus this type of clustering based on the smooth splicing is not completely dependent on the strength of edge. Compared to JP clustering and SNN density-based clustering algorithm, the generalizing ability of this algorithm will increase along with the increasing value of smoothness. However, the too large value of smoothness will induce a number of edges failing to participate in splicing. This may result in a too large number of small scale categories, especially in face of noncompact clusters or large-scale dataset.

Assume that there already exists a connected graph and an edge tries to join the connected graph through splicing at a vertex. Starting from this vertex, we can find a number of paths in the connected graph, and an edge must meet various degrees of smoothness with every path at splicing point before it successfully joins the graph. The order of edges should be predetermined in order to maintain the stability of the experimental results, so we can sort the edges according to their intensities (i.e., SNN values) in descending order. The prior edges participating in splicing have higher strength, while the smoothness required is lower at this moment. The strength of edge to participate in clustering afterwards (within the same graph) becomes smaller gradually, while the demand of smoothness increases. So, it can be seen that the splicing clustering method inherits a complementary intensity-smoothness principle, which is also a trade-off between local assumption and global assumption. This is obviously justified as a clustering criterion, which can be stated more explicitly as follows: if a sample point want to join a class, then either the SNN similarity between it and a certain sample in this class should be high or it should have a shared nearest neighbor at least with many enough samples in this class (SNN value is 1). Since the algorithm inherits the complementary strength, smoothness operating mechanism, it does not need to set the parameters for itself. Although there has no trouble of tuning parameters in this case, the algorithm also losses its flexibility, and so we consider setting parameter k in Algorithm 1 as the only parameter of new algorithm. In fact, the Algorithm 1 will be also the first step in our algorithm as the other SNN similarity based clustering algorithms. The following Figure 2 roughly describes the process of generating a single connected graph (on behalf of a class or a set of noise).

In the process of splicing, the vertices of single connected graph gradually expand outwards. As a vertex may appear several times in the same layer (but due to the deletion operation afterwards, a vertex cannot simultaneously appear in different layers), so there maybe exists a cycle generated from smooth splicing. For this reason, the result maybe is a single connected graph, while not necessarily a tree. We can make use of Property 2 to decide whether the splicing condition is met or not for any edge in edge set (in fact, SNN similarity between the vertices in the same layer should not be zero too.). This simple performing avoids the trouble of searching all paths in a single connected graph, while the latter is an NP-complete problem in graph theory. If the condition is fulfilled, then the splicing goes on, and now the new joined vertex is regarded as the new splicing points to expand the graph continuously. Before the further expansion form the vertices in the kth layer, we can prune those edges which contain the vertices in the $(k - 1)$th layer for these vertices have been fall into this category decidedly. It also means that those edges including these vertices either have been joined into the connected graph or have no need to be considered to participate in clustering (in fact, it is also impossible that they appear at the other connected graphs on

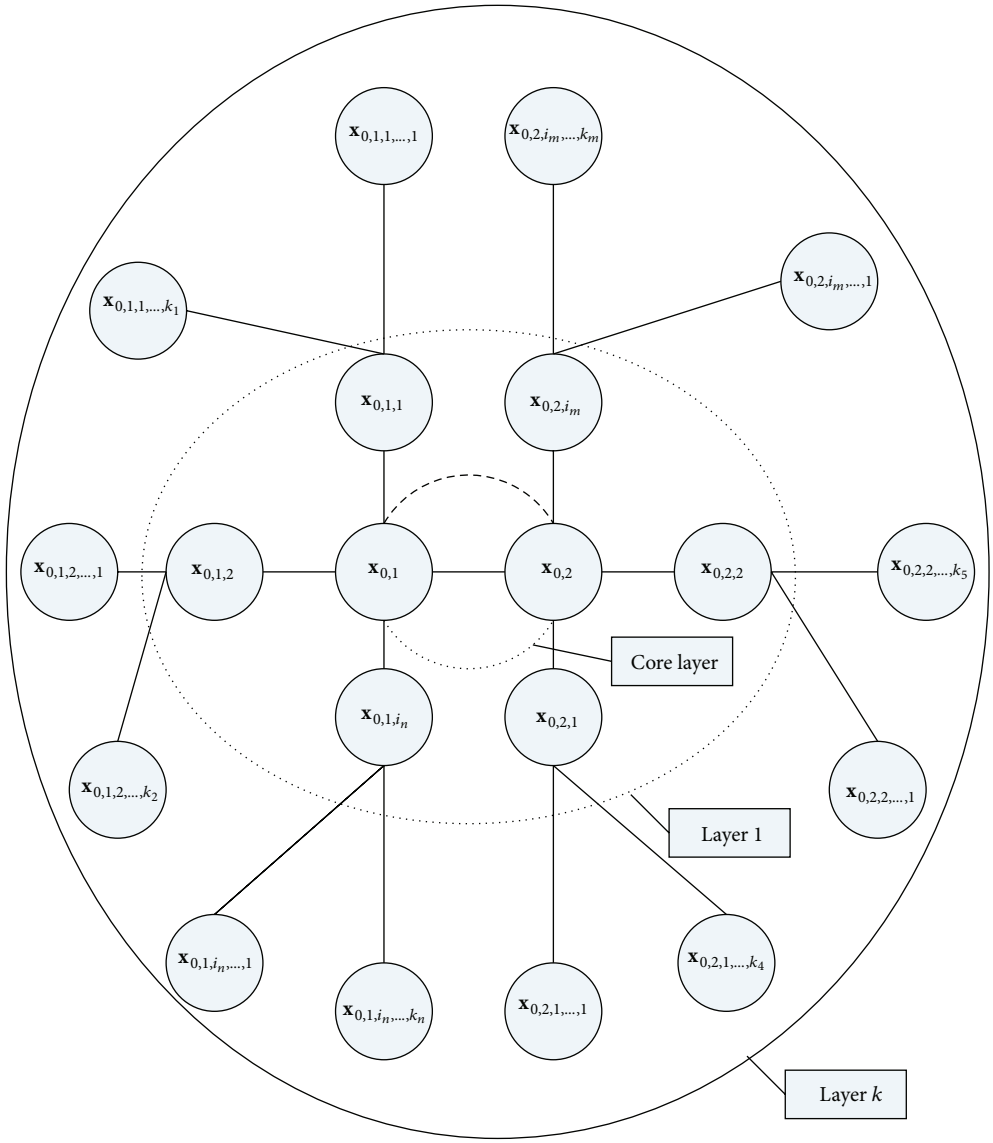

Figure 2: The process of smooth splicing clustering.

behalf of the other classes), so the capacity of edges set will diminish quickly and the algorithm will speed up.

4.3. Algorithm Complexity. Suppose that the number of samples is n and the sparse ratio of SNN matrix is τ, while $t = 1-\tau$, and then the number of edges is $N = tn^2/2-n$. Since each edge can be expressed as a 1×3 matrix (two vertex labels and one edge value), so the storage complexity of the algorithm is $O(3tn^2/2-3n)$. Since the computational complexity of sorting the edge set is $O(N \log N)$ and each vertex to join a single connected graph needs to go through $O(N)$ comparisons and testing operations respectively, therefore the computational complexity of the algorithm is $O(N \log N + N^2)$.

4.4. Algorithm Framework. In summary, SNN similarity-based smooth splicing clustering algorithm can be described as shown in Algorithm 4.

5. Clustering on TDT2

As the SNN similarity based clustering algorithm is particularly effective in dealing with high-dimensional data, so here we only conduct experiments on high-dimensional data set. The methods used are three kinds of SNN similarity based approaches mentioned previously. Since the popular SSE measure is not suitable for evaluating the results of SNN similarity based clustering methods, so we have to evaluate the experimental results using the manual method.

5.1. TDT2 Dataset. TDT2 data set used in experiment (Nist topic detection and tracking data set) is collected in the first half of 1998 from the total six sources, including two news columns (APW, NYT), two audio programs (VOA, PRI), and two television programs (CNN, ABC). It contains 11201 topics of the text and is divided into 96 semantic categories.

(1) Calculate SNN similarity graph.
(2) Identify the C_m^2 edges, remove the edges with zero intensity, and sort the rest edges according to their strengths in descending order.
(3) While edge set is not empty, then the smoothness is set as 1, starting from the first edge, study the following edges one by one whether they satisfy the 1 order splicing conditions, if
  (3.1) Yes,
    (3.1.1) Two edges splice;
    (3.1.2) Regard the newly joined vertexes as the splicing point, and make use of the property 2 to splice continuously, meanwhile smoothness gradually increases, until they do not meet the conditions for splicing;
    (3.1.3) Prune the edges containing the vertexes in layer $k-1$ from the edge set when the layer k is completed;
    (3.1.4) When no edge in edge set meets the conditions for splicing, then prune the edges containing the outermost vertexes of the single-connected graph from the edge set, and go to 3.
  (3.2) No, start from the first edge in edge set, repeat 3.
(4) If edge set is empty, then each single-connected graph represents a category, where some small sample sets can sets can be regarded as noise category.

Algorithm 4: SNN similarity based smooth splicing clustering algorithm.

In this data set, the texts which can belong to two classes or more classes were removed lately, as a result the total number of the text becomes 9,394 and the total number of the classes becomes 30. This data set can be obtained via the following link: http://www.cad.zju.edu.cn/home/dengcai/Data/TextData.html. In order to facilitate labor statistics, we only select 4 categories whose labels are 21, 22, 23, and 24 respectively, to form the data set, where the total number of samples is 293 and each sample is a 36772-dimensional vector.

5.2. Evaluating Method. For a data set, the manual identification and clustering results as are shown in Table 1.

Then the calculating formula for clustering accuracy is $P = a/(a+b)$ and recall rate is calculated as follows: $R = a/(a+c)$. The calculating result of each formula is related to a special class in data set.

5.3. Parameters Setting. The value of parameter k in Algorithms 2 and 3 takes constant 12, which is an approximate parameter value in many k nearest neighbors related algorithm in practice. The SNN similarity threshold in the Algorithm 2 is selected from $[1, 12]$. In Algorithm 3, the neighborhood radius ε is selected from $[1, 10]$ and the least point parameter MinPts is selected from $[1, 10]$. In Algorithm 4, the only the nearest neighbor parameter k is selected from $[10, 30]$.

5.4. Experimental Results. The optimal parameter of SNN similarity threshold in Algorithm 2 is 1. In Algorithm 3, the optimal radius of neighborhood domain $\varepsilon = 5$ and the optimal MinPts $= 4$, and they are obtained by cross-validation method. In Algorithm 4, the optimal parameters of the nearest neighbor $k = 29$. In the results of JP clustering, we regard the categories whose labels are 44, 36, 56, and 55 respectively as four categories discovered by the algorithm. They correspond to four classes in the training data set, and the other categories containing a small amount of samples are seen as noise. In the results of SNN density-based clustering, we select categories whose labels are 1, 3, 5, and 7 respectively, as 4 classes discovered by algorithm. They also correspond to four categories in the training data set, and the other categories containing a small amount of samples are looked on as noise. In clustering results of the smooth splicing, we treat categories whose labels are 6, 5, 7, and 4 respectively, as four categories discovered by the algorithm. They still correspond to four classes in the training data set, and the other types containing a small amount of samples are regarded as noise. In addition, we also find that all the noise samples and all the samples wrongly clustered belong to a variety of new small scale categories; that is to say, all the elements of confusion matrix are zeros. Under this special situation, the clustering accuracy and recall rate are the same for every class, that is, $P = R$. Clustering results are shown in Table 2.

Table 1

	Manual identification	
	True	False
Clustering results		
Yes	a	b
No	c	d

As is shown from Table 2, in experiments on text data set containing four types of samples, the smooth splicing clustering algorithm obtains the best clustering accuracy and recall rate in three classes, and only in class 21, JP clustering and SNN density based clustering outperform our method, which means that our method is a fairly good clustering algorithm based on SNN similarity. In addition, we also study the relationship between the number of classes discovered by splicing clustering algorithm and the parameter of the nearest neighbor k, and the experimental result is shown in Figure 3.

As can be seen from Figure 3, along with the increasing nearest neighbor values, the number of classes gradually decreases. When the parameter of the nearest neighbor takes 29, the number of classes reaches the minimum value 7, and meantime, the best clustering accuracy and recall rate are

Table 2: Clustering precision and recall accuracy of three algorithms.

Clustering method	Class in TDT2	Precision/recall
JP clustering	Class 21	**1.0000**
	Class 22	0.8243
	Class 23	0.7222
	Class 24	0.4225
SNN density based clustering	Class 21	0.9605
	Class 22	0.6622
	Class 23	0.7222
	Class 24	0.9014
Smooth splicing clustering	Class 21	0.7368
	Class 22	**0.9459**
	Class 23	**0.8194**
	Class 24	**0.9577**

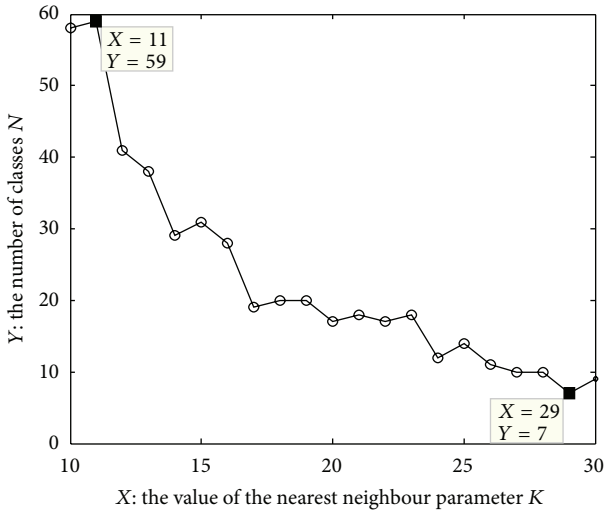

Figure 3: The relation between the no. of classes and the value of the nearest neighbors.

obtained. In fact, along with increasing values of the nearest neighbor, the sparse ratio of SNN matrix is getting small correspondingly, and, thus the number of edges, which can participate in splicing clustering every time increases, at the same time the number of generated categories reduces. However, when the values of the nearest neighbor increase, the computational complexity of the algorithm correspondingly increases and the running time becomes longer.

Given each fixed value of the nearest neighbor, the procedure repeats independently for 10 times, and the mean value of running time is computed. The experimental results are shown in Figure 4. Clearly, the running time of algorithm approximately linearly increases accompanyied with the increase of the nearest neighbor, and so it shows the good scalability of our algorithm.

6. Clustering on Chinese Text Dataset

In this section, we will further conduct the clustering experiments to validate the effectiveness of smooth splicing clustering. Here, we will use another high-dimensional dataset and another evaluating method. The management maybe is not delicate as Section 5, but it is closer to the true operation in practice.

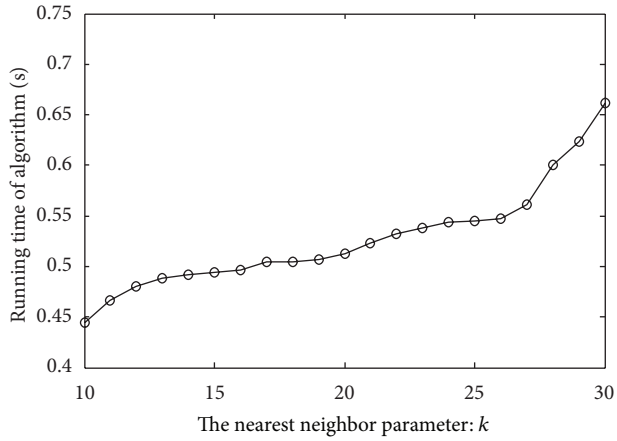

Figure 4: The relation between the running time and the value of the nearest neighbors.

6.1. Chinese Text Dataset. We select 931 news reports from the People's Daily (1st–30th, January, 1998), which belong to ten classes and they are politics, economics, sports, art, medicine, education, transportation, computer, environment, and military. Then the text data set is transformed into a vector data set using the ICTCLAS software (developed by Institute of Computing Technology, Chinese Academy of Sciences, to be our Chinese segmentation tool here) to segment the text [23] and TFIDF formula to compute the weights of the keywords [24], where each vector has 1000 attributes.

6.2. Evaluating Method. Here we try to use a novel method to evaluate the experimental result. We discard the classes which include few samples, and they can be regarded as noisy classes again. Then we count the numbers of the rest efficient classes and compare them with the true number of classes in dataset. Without doubt, those statistical results approximating the true number of classes will be appealing.

6.3. Parameters Setting. The SNN similarity threshold in the Algorithm 2 is selected from $[1, 12]$. In Algorithm 3, the neighborhood radius ε is selected from $[1, 10]$ and the least point parameter MinPts is selected from $[1, 10]$. In all the algorithms, the nearest neighbor parameter k is selected from $[10, 25]$. The classes including only one sample are discarded as noisy classes. With every fixed parameter k, we compare the optional experimental result of every algorithm to the true number of classes in dataset, that is, 10.

6.4. Experimental Results. We conduct the experiment on this text dataset for comparing the previously mentioned three methods. The results are shown in Table 3.

As shown in Table 3, although all noisy classes are discarded, Algorithm 2 (i.e., JP clustering) still presents over

Table 3: Comparison of the optimal class number for three algorithms.

The nearest neighbor parameter	Class no. for Algorithm 2	Class no. for Algorithm 2	Class no. for Algorithm 2
10	106	24	22
11	131	27	20
12	125	26	20
13	239	24	19
14	169	24	17
15	184	19	18
16	170	18	17
17	207	17	15
18	190	17	15
19	232	12	14
20	246	12	11
21	182	11	11
22	246	9	10
23	260	10	10
24	302	7	10
25	249	5	8

100 classes with the every nearest neighbor. It means that JP clustering is highly sensitive to the intensity of single edge. Meanwhile, it is also demonstrated that JP clustering is difficult to find the true class numbers in high-dimensional dataset. Algorithm 3 (i.e., SNN density based clustering) searches the least 25 classes with that every nearest neighbor, and in particular it presents very approximate class numbers when the nearest neighbor is selected from 19 to 23. Due to the varied class density is taken into account, SNN density based clustering partly avoids the disadvantage of relying on single edge intensity and accordingly obtains better performance in approximating the true class numbers. As shown in last column of Table 3, Algorithm 4 (i.e., smooth splicing clustering) fast approximate the true class number when the nearest neighbor descends, and in particular it presents highly appealing output when the nearest neighbor varies from 20 to 24. Except for the nearest neighbor 20, smooth splicing shows better performance compared to SNN density based clustering in the other cases. As smooth splicing clustering can be effective in discovering classes in high-dimensional dataset, we believe that it benefits much from its character of smooth linkage.

7. Conclusions

This paper presents a new SNN similarity based smooth splicing clustering algorithm, which includes a complementary intensity-smoothness mechanism. Comparing the traditional JP clustering and SNN density based clustering algorithm, its dependence on the strength of single edge is weakened, meanwhile its generalizing ability is improved correspondingly. The experimental results verify the effectiveness of the proposed algorithm. Considering that image, voice, video, and gene expression data also have high-dimensional features, and in the future, the algorithm can be used to cluster these datasets to further examine its validity.

Acknowledgments

This research was supported by the National Natural Science Foundation of China under Grant no. 31171456 and the National Key Technology R&D Program under Grant no. 2013BAD15B03.

References

[1] L. Kaufman and P. J. Rousseeuw, *Finding Groups in Data: An Introduction to Cluster Analysis*, Wiley-Interscience Press, New York, NY, USA, 1990.

[2] M. Liu, X. Jiang, and A. C. Kot, "A multi-prototype clustering algorithm," *Pattern Recognition*, vol. 42, no. 5, pp. 689–698, 2009.

[3] J. MacQueen, "Some methods for classification and analysis of multivariate observations," in *Proceedings of the 5th Berkeley Symposium on Mathematical Statistics and Probability*, 1967.

[4] P. J. Rousseeuw, "Silhouettes: a graphical aid to the interpretation and validation of cluster analysis," *Journal of Computational and Applied Mathematics*, vol. 20, pp. 53–65, 1987.

[5] S. Guha, R. Rastogi, and K. Shim, "CURE: an efficient clustering algorithm for large databases," in *Proceedings of the ACM SIGMOD International Conference on Management of Data*, pp. 73–84, 1998.

[6] S. Guha, R. Rastogi, and K. Shim, "Rock: a robust clustering algorithm for categorical attributes," *Information Systems*, vol. 25, no. 5, pp. 345–366, 2000.

[7] G. Karypis, E. Han, and V. Kumar, "Chameleon: hierarchical clustering using dynamic modeling," *Computer*, vol. 32, no. 8, pp. 68–75, 1999.

[8] T. Zhang and R. Ramakrishnan, "BIRCH: an efficient data clustering method for very large databases," in *Proceedings of the ACM SIGMOD International Conference on Management of Data*, pp. 103–114, 1996.

[9] M. Ankerst, M. M. Breunig, H. Kriegel, and J. Sander, "OPTICS: ordering points to identify the clustering structure," in *Proceedings of the ACM SIGMOD International Conference on Management of Data*, pp. 49–60, 1999.

[10] M. Ester, H. P. Kriegel, J. Sander, and X. Xu, "A density-based algorithm for discovering clusters in large spatial databases with noise," in *Proceedings of the 2nd International Conference on Knowledge Discovery and Data Mining*, pp. 226–231, 1996.

[11] R. Agrawal, J. Gehrke, D. Gunopulos, and P. Raghavan, "Automatic subspace clustering of high dimensional data," *Data Mining and Knowledge Discovery*, vol. 11, no. 1, pp. 5–33, 2005.

[12] L. Moussiades and A. Vakali, "Clustering dense graphs: a web site graph paradigm," *Information Processing and Management*, vol. 46, no. 3, pp. 247–267, 2010.

[13] O. Etzioni and M. Perkowitz, "Category translation: learning to understand information on the internet," in *Proceedings of the 15th International Joint Conference on Artificial Intelligence*, 1995.

[14] M. E. Houle, "The relevant-set correlation model for data clustering," *Statistical Analysis and Data Mining*, vol. 1, no. 3, pp. 157–176, 2008.

[15] H. Kriegel, P. Kröger, and A. Zimek, "Clustering high-dimensional data: a survey on subspace clustering, pattern-based clustering, and correlation clustering," *ACM Transactions on Knowledge Discovery from Data*, vol. 3, no. 1, article 1, pp. 1–58, 2009.

[16] M. E. Houle, H. Kriegel, P. Kröger, E. Schubert, and A. Zimek, "Can shared-neighbor distances defeat the curse of dimensionality?" in *Proceedings of the 22nd International Conference on Scientific and Statistical Database Management*, 2009.

[17] J. Han, M. Kamber, and J. Pei, *Data Mining: Concepts and Techniques*, Morgan Kaufmann, 2006.

[18] R. A. Jarvis and E. A. Patrick, "Clustering using a similarity measure based on shared nearest neighbors," *IEEE Transactions on Computers*, vol. 22, no. 11, pp. 1025–1034, 1973.

[19] L. Ertoz, M. Steinbach, and V. Kumar, "A new shared nearest neighbor clustering algorithm and its applications," in *Proceedings of the Workshop on Clustering High Dimensional Data and Its Applications at 2nd SIAM International Conference on Data Mining*, 2001.

[20] L. Ertoz, M. Steinbach, and V. Kumar, "Finding clusters of different sizes, shapes, and densities in noisy, high dimensional data," in *Proceedings of the 2nd SIAM International Conference on Data Mining*, San Francisco, Calif, USA, May 2003.

[21] D. Reinhard, *Graph Theory*, Springer, Berlin, Germany, 2005.

[22] S. Hu, R. Tong, T. Ju, and J. Sun, "Approximate merging of a pair of Bezier curves," *Computer-Aided Design*, vol. 33, no. 2, pp. 125–136, 2001.

[23] H. P. Zhang, H. K. Yu, D. Y. Xiong, and Q. Liu, "HHMM-based Chinese lexical analyzer ICTCLAS," in *Proceedings of the 2nd SIGHAN Workshop on Chinese Language Processing (SIGHAN '03)*, 2003.

[24] G. Salton and M. J. McGill, *Introduction to Modern Information Retrieval*, McGraw-Hill, 1983.