2009

Towards Solving Similarity Search Problems Using Fuzzy Concept for Multi-Dimensional Data

Yong Shi
Kennesaw State University, yshi5@kennesaw.edu

Recommended Citation

Yong Shi. 2009. Towards solving similarity search problems using fuzzy concept for multi-dimensional data. In Proceedings of the 47th Annual Southeast Regional Conference (ACM-SE 47). ACM, New York, NY, USA, , Article 84 , 3 pages.

Towards Solving Similarity Search Problems Using Fuzzy Concept for Multi-Dimensional Data

Yong Shi
Department of Computer Science and Information Systems
Kennesaw State University
1000 Chastain Road
Kennesaw, GA 30144
yshi5@kennesaw.edu

ABSTRACT

In this paper, we present continuous research on data analysis based on our previous work on similarity search problems. $PanKNN$[13] is a novel technique which explores the meaning of K nearest neighbors from a new perspective, redefines the distances between data points and a given query point Q, and efficiently and effectively select data points which are closest to Q. It can be applied in various data mining fields. In this paper, we applied the Fuzzy concept to improve the performance of PanKNN, targeting the better decision making for the calculation of the distance between a data point and Q. This approach can assist to improve the performance of existing data analysis approaches.

1. INTRODUCTION

With the advance of modern technology, the generation of multi-dimensional data has proceeded at an explosive rate in many disciplines. The similarity search problem has been studied in the last decade, and many algorithms haves been proposed to solve the K nearest neighbor search[10, 12, 2, 9, 8]. $PanKNN$[13] is a novel technique which explores the meaning of K nearest neighbors from a new perspective, redefines the distances between data points and a given query point Q, and efficiently and effectively select data points which are closest to Q. In this paper, we first give a brief introduction about our previous work on PanKNN; then, we propose to use the Fuzzy concept to improve the performance of PanKNN, targeting the better decision making for the calculation of the distance between a data point and Q.

1.1 Related work

In traditional nearest neighbor problems, the similarity between two data points is based on a similarity function such as Euclidean distance which aggregates the difference between each dimension of the two data points. In other words, the nearest neighbor problems are solved based on the distance between the data point and the query point over a fixed set of dimensions (features). However, such approaches only focus on full similarities, i.e., the similarity in full data space of the data set. Also early methods [1, 5, 14] suffer from the "cure of dimensionality". In a high dimensional space the data are usually sparse, and widely used distance metric such as Euclidean distance may not work well as dimensionality goes higher. Recent research [6] shows that in high dimensions nearest neighbor queries become unstable: the difference of the distances of farthest and nearest points to some query point does not increase as fast as the minimum of the two, thus the distance between two data points in high dimensionality is less meaningful. Some approaches [11, 4, 3] are proposed targeting partial similarities. However, they have limitations such as the requirement of the fixed subset of dimensions, or fixed number of dimensions as the input parameter(s) for the algorithms.

1.2 Solving similarity problems

In this subsection, we will briefly introduce our previous work on PanKNN[13]. PanKNN is a novel approach to nearest neighbor problems. We also analyze the nearest neighbor problems for a new perspective. We define the new meaning for the K nearest neighbors problem, and design algorithms accordingly. The similarity between a data point and a query point is not based on the difference aggregation on all the dimensions. We propose self-adaptive strategies to dynamically select dimensions based on the different situation of the comparison.

For a given data point X_i, and a given query point Q, we call the distance between X_i and Q as Pan-distance $PD(X_i, Q)$. $PD(X_i, Q)$ does not calculate the aggregated differences between X_i and Q on all dimensions. Instead, it only take into account those dimensions on which X_i is close enough to Q, and sum them up. This strategy not only avoids the negative impacts from those dimensions on which X_i is far to Q, but also eliminate the curse of dimensionality caused by similarity functions such as Euclidean distance which calculates the square root of the sum of squares of distances on each dimensions.

On more dimensions X_i is close (within the sets of K nearest neighbor) to Q, the smaller Pan-distance X_i has to Q. If we have two data points X_i and X_j, we judge which data point is closer to Q based on how many dimensions on which they are close enough (within dimension-wise K nearest neighbors) to Q, as well as their average distances

ACMSE '09 March 19-21, 2009, Clemson, SC, USA.
©2009 ACM 978-1-60558-421-8/09/03 ...$10.00

to Q on such dimensions.

Given a data set DS, we first calculate the difference δ_{il} of each data point X_i to the query point Q on each dimension D_l. Then we sort the *ids* on each dimension D_l based on δ_{il}, and select the first K *ids* on each dimension D_l and put them into KS_l. We put all the *id*s in all KS_l to the set GS, and calculate the PD(X_i, Q) for each data point if its *id* is in GS. Finally, we sort the *id*s based on the Pan-distance and select the first K *ids* in the sorted list as the *id*s of K nearest neighbors of Q. We do not need to calculate the difference using different number of dimensions. The *number* of dimensions and the *subset* of dimensions associated with data point X_i are both dynamically decided depending on the values of X_i and their rankings on different dimensions.

1.3 Fuzzy concept

Some data in the real world are not naturally well organized. Clusters in the data may overlap each other. Fuzzy concept can be applied to further improve the PanKNN algorithm.

The concept of fuzzy sets was first introduced by Zadeh [15] to represent vagueness. The use of fuzzy set theory is becoming popular because it produces not only crisp decision when necessary but also corresponding degree of membership. Usually, membership functions are defined based on a distance function, such that membership degrees express proximities of entities to cluster centers. In conventional clustering, sample is either assigned to or not assigned to a group. Assigning each data point to exactly one cluster often causes problems, because in real world problems a crisp separation of clusters is rarely possible due to overlapping of classes. Also there are exceptions which cannot be suitably assigned to any cluster. Fuzzy sets extend to clustering in that object of the data set may be fractionally assigned to multiple clusters, that is, each point of data set belongs to groups by a membership function. This allows for ambiguity in the data and yields detailed information about the structure of the data, and the algorithms adapt to noisy data and classes that are not well separated. Most fuzzy cluster analysis methods optimize a subjective function that evaluates a given fuzzy assignment of data to clusters.

One of the classic fuzzy clustering approach is the Fuzzy C-means Method designed by Bezdek, J. C [7]. In brief, for a data set X with size of n and cluster number of c, it extends the classical within groups sum of squared error objective function to a fuzzy version by minimizing the objective function with weighting exponent m, $1 \leq m < \infty$:

$$J_m(U, V) = \sum_{k=1}^{n} \sum_{i=1}^{c} (u_{ik})^m d^2(x_k, v_i), \tag{1}$$

where U is a partition of X in c part, $V = v = (v_1, v_2, ..., v_c)$ are the cluster centers in R^p , and A is any $(p \times p)$ symmetric positive definite matrix defined as the following:

$$d(x_k, v_i) = \sqrt{(x_k - v_i)^\top (x_k - v_i)}, \tag{2}$$

where $d(x_k, v_i)$ is an inner product induced norm on R^p, u_{ik} is referred to as the grade of membership of x_k to the cluster i.

The fuzzy C-Means (FCM) uses an iterative optimization of the objective function, based on the weighted similarity measure between x_k and the cluster center v_i. During each iteration, it calculates the c cluster centers $\{v_{i,t}\}, i = 1, ..., c$

$$v_{i,t} = \frac{\sum_{k=1}^{n} u_{ik,t-1}^m x_k}{\sum_{k=1}^{n} u_{ik,t-1}^m}, \tag{3}$$

for those data points not of any current cluster center, it calculate the following

$$u_{ik,t} = \frac{1}{\sum_{j=1}^{c} \left(\frac{d_{ik,t}}{d_{jk,t}} \right)^{\frac{2}{m-1}}}. \tag{4}$$

When a predefined termination condition is satisfied, the algorithm is terminated.

1.4 Fuzzy-based PanKNN

In this subsection we propose to use the fuzzy concept to improve the performance of PanKNN by modifying the calculation of the distance between a data point and a query point.

Let n denote the total number of data points and d be the dimensionality of the data space. Let D_l be the lth dimension, where l = 1, 2, ..., d. Let the input d-dimensional data set be $\mathbf{X}$

$$\mathbf{X} = \{X_1, X_2, ..., X_n\},$$

which is normalized to be within the hypercube $[0, 1]^d \subset R^d$. Each data point X_i is a d-dimensional vector:

$$X_i = [x_{i1}, x_{i2}, ..., x_{id}]. \tag{5}$$

Data point X_i has the *id* number i. Let Q be the query point: $Q = [q_1, q_2, ..., q_d]$. Let $\Delta_i = [\delta_{i1}, \delta_{i2}, ..., \delta_{id}]$ as the array of differences between the data point X_i and the query point Q on each dimension.

Given a data set DS of n data points $\mathbf{X} = \{X_1, X_2, ..., X_n\}$ with d dimensions D_1, D_2, ..., D_d, and a query point Q in the same data space, we first sort the data points on each dimension D_l, l=1, 2, ..., d, based on δ_{il} which is the difference between data point X_i and Q on dimension D_l. On each dimension D_l, l=1, 2, ..., d, let KS_l be the set which contains the *id*s of the first K data points in the sorted list. We call these first K data points as *dimension-wise K nearest neighbor* to Q on D_l.

For each data point X_i, i=1, 2, ... n, let U_i be a binary array associated with X_i. $U_i = [u_{i1}, u_{i2}, ..., u_{id}]$. The value of u_{il} is calculated as following:

if $i \notin KS_l$, $u_{il} = 0$; if $i \in KS_l$, $u_{il} = \frac{\delta_{il}}{\sum_{j \in KS_l} \delta_{jl}}$.

u_{il} is the grade of closeness (instead of "membership" in the traditional fuzzy concept) X_i to Q on dimension l.

Definition 1: Pan-distance

Given two d-dimensional points $X_i = [x_{i1}, x_{i2}, ..., x_{id}]$ and $Q = [q_1, q_2, ..., q_d]$, with D_l as the dimension l, l=1, 2, ..., d, the Pan-distance of X_i to Q

$$PD(X_i, Q) = \frac{\sum_{l=1}^{d} u_{il} * \delta_{il}}{(\sum_{l=1}^{d} u_{il})^2} \tag{6}$$

where δ_{il} is the difference between X_i and Q on D_l, u_{il} is defined as above.

PD(X_i, Q) can also be defined as

$$PD(X_i, Q) = \frac{\sum_{l=1}^{d} u_{il} * \delta_{il}}{\sum_{l=1}^{d} u_{il}} * \frac{1}{\sum_{l=1}^{d} u_{il}}, \tag{7}$$

where $\frac{\sum_{l=1}^{d} u_{il} * \delta_{il}}{\sum_{l=1}^{d} u_{il}}$ is the average distance of X_i to Q on those dimensions on which X_i is within the set of *dimension-*

```
Algorithm Fuzzy PanKNN (DS: data set, Q: query point, D: dimensions, K: number of data points required)
Begin
  For each $X_i \in DS$, calculate $\Delta_i = [\delta_{i1}, \delta_{i2}, ..., \delta_{id}]$ in which $\delta_{il} = |x_{il} - q_l|$;
  On each dimension $D_l$, l=1, 2, ..., d, sort the ids of the data points in $DS$, based on $\delta_{il}$ for $X_i$. Let $S_l$ be the sorted list on $D_l$;
  Let $KS_l$ be the subset of $S_l$ which contains the first K ids in $S_l$;
  For each data point $X_i$, i=1, 2, ... n, generate $U_i = [u_{i1}, u_{i2}, ..., u_{id}]$ in which $u_{il} = 0$, if $i \notin KS_l$; $u_{il} = \frac{\delta_{il}}{\sum_{j \in KS_l} \delta_{jl}}$, if $i \in KS_l$;
  Let set $GS = \{i\}$ in which i $\in KS_l$, l=1, 2, ..., d;
  For each data point $X_i$, where $i \in GS$, calculate $PD(X_i, Q)$;
  Sort $GS = \{i\}$ based on $PD(X_i, Q)$;
  Let set PKNNS contain the first K ids $\in GS$;
  Return PKNNS.
End.
```

Figure 1: Proc: Primitive PanKNN

wise K nearest neighbor to Q, and $\frac{1}{\sum_{l=1}^{d} u_{il}}$ is the weight to the average difference based on on how many dimensions on which X_i is within the set of K nearest neighbor to Q.

Definition 2: Pan-K Nearest Neighbors Problem

Given a data set DS of n data points $\mathbf{X} = \{X_1, X_2, ..., X_n\}$ *with* D_l *as the dimension l, l=1, 2, ..., d, and a query point Q in the same data space, find a set PKS which consists of k data points from DS so that for any data point* $X_i \in PKS$ *and any data point* $X_j \in DS - PKS$*, the Pan-distance of* X_i *to Q is less than or equal to the Pan-distance of* X_j *to Q. The set PKS is the Pan-K Nearest Neighbor set of Q in DS.*

Figure 1 presents the Fuzzy PanKNN algorithm.

2. REFERENCES

[1] White D.A. and Jain R. Similarity Indexing with the SS-tree. In *Proceedings of the 12th Intl. Conf. on Data Engineering*, pages 516–523, New Orleans, Louisiana, February 1996.

[2] E. Achtert, C. Böhm, P. Kröger, P. Kunath, A. Pryakhin, and M. Renz. Efficient reverse k-nearest neighbor search in arbitrary metric spaces. In *SIGMOD '06*, pages 515–526, New York, NY, USA, 2006. ACM.

[3] C. C. Aggarwal. Towards meaningful high-dimensional nearest neighbor search by human-computer interaction. In *ICDE*, 2002.

[4] C. C. Aggarwal, A. Hinneburg, and D. A. Keim. On the surprising behavior of distance metrics in high dimensional space. *Lecture Notes in Computer Science*, 1973, 2001.

[5] D. A. Berchtold S., Keim and H.-P. Kriegel. The X-tree : An index structure for high-dimensional data. In *VLDB'96*, pages 28–39, Bombay, India, 1996.

[6] K. Beyer, J. Goldstein, R. Ramakrishnan, and U. Shaft. When is "nearest neighbor" meaningful? In *International Conference on Database Theory 99*, pages 217–235, Jerusalem, Israel, 1999.

[7] J. C. Bezdek. *Pattern Recognition with Fuzzy Objective Function Algorithms.* Kluwer Academic Publishers, Norwell, MA, USA, 1981.

[8] B. Cui, H. Shen, J. Shen, and K. Tan. Exploring bit-difference for approximate KNN search in high-dimensional databases. In *Australasian Database Conference, 2005.*, 2005.

[9] R. Fagin, R. Kumar, and D. Sivakumar. Efficient similarity search and classification via rank aggregation, 2003.

[10] A. Gionis, P. Indyk, and R. Motwani. Similarity search in high dimensions via hashing. In *The VLDB Journal*, pages 518–529, 1999.

[11] A. Hinneburg, C. C. Aggarwal, and D. A. Keim. What is the nearest neighbor in high dimensional spaces? In *The VLDB Journal*, pages 506–515, 2000.

[12] T. Seidl and H.-P. Kriegel. Optimal multi-step k-nearest neighbor search. *SIGMOD Rec.*, 27(2):154–165, 1998.

[13] Y. Shi and L. Zhang. A dimension-wise approach to similarity search problems. In *the 4th International Conference on Data Mining (DMIN'08)*, 2008.

[14] R. Weber, H.-J. Schek, and S. Blott. A quantitative analysis and performance study for similarity-search methods in high-dimensional spaces. In *Proc. 24th Int. Conf. Very Large Data Bases, VLDB*, pages 194–205, 24–27 1998.

[15] L. A. Zadeh. Fuzzy sets. *Information and Control*, 8(3):338–353, 1965.