Advances in Computer Science and Information Technology (ACSIT)
Print ISSN: 2393-9907; Online ISSN: 2393-9915; Volume 2, Number 11; April-June, 2015 pp. 25-29

http://www.krishisanskriti.org/acsit.html

A Recursive Partitioning Method for Nearest Neighbor Search in High Dimensional Data

Raghunadh Pasunuri[1] and Sobha Rani T[2]

[1,2]University of Hyderabad Hyderabad, Telangana
E-mail: [1]raghunadh_pasunuri@yahoo.co.in, [2]tsrcs@uohyd.ernet.in

Abstract—*Many real world applications require searching a large amount of data to find the objects similar to the search queries. Nearest Neighbour Search is the common operation that has been used to conduct similarity search in vast areas like Content-Based Image Retrieval, Web search engines, micro array data analysis etc. Dimensionality forces us to look at data from a different perspective when dealing with such large data. In this work we propose a recursive partitioning and distance-based indexing scheme for large and high-dimensional data to retrieve the nearest neighbours for a given query. This method works by dividing the data space into disjoint partitions based on the distance from a reference point to the data objects. In the next level for each sub-partition a reference point is selected and again it is partitioned into further sub-sub-partitions. Main advantage of this method is that it reduces the search space. To find the nearest neighbours we use Soergel distance metric which is a dissimilarity-based distance metric to find the association between two data objects. We are able to retrieve the top 100 nearest neighbours (kNN) by searching in only a single bin where all the nearest neighbours lie for the given query according to the distance from a reference point. We have validated our method by conducting experiments with the following data sets: ZINC data set, AT and T Faces Database. Our results show that the proposed method is having a very less construction (off-line) time and search (on-line) time compared to brute-force linear scan. Proposed method reduces or prunes the search space from 100 percent to 10 percent or even less, which will save lot of computation time. We verified the proposed method with other methods and compared the performance and the results are presented.*

1. INTRODUCTION

Many real world applications require searching a large amount of data for finding similar type of objects as the answer for the queries. Nearest Neighbor Search is the common operation that has been used in vast areas like Content-Based Image Retrieval, Web search engines, micro array data analysis etc. It has been a research problem in World Wide Web era of vast data generating by the on-line sites, Content Based Image Retrieval (CBIR), and micro array data analysis in bioinformatics and so on. There are large studies in literature which were focused on these issues. One of the most important challenge in dealing with high-dimensional data is known as "curse of dimensionality", is associated with high dimensional data. As the dimensionality grows the complexity of managing the data also increases and time complexity increases exponentially to retrieve nearest neighbors [1]. High-dimensionality is common in applications like Content Based Image Retrieval and similarity search in chemical structures and so on. Nearest Neighbor Search in such high-dimensional data is gaining more attention in these days.

In this work we attempt to perform nearest neighbor search on large and high-dimensional data sets. In order to find nearest neighbors for a particular query, a variety of similarity metrics are available to quantify the similarity between query and database item, Euclidean distance, hamming distance, cosine similarity, dice coefficient, and Tanimoto Coefficient are some of them. Soergel distance is a dissimilarity-based metric which gives, how dissimilar the given two data objects are. The range of Soergel distance metric is from 0 to 1. For highly similar cases Soergel distance metric value is very near to 0 and for any pair of highly dissimilar cases it is close to one [5].

The rest of this paper is organized as follows. In the next section, we describe related work. In Section 3, we explain about the data sets used. The proposed method is explained in Section 4. Section 5 reports our experimental study and results. Finally, we conclude in Section 6 with directions for future work.

2. LITERATURE SURVEY

Several researchers have been working on the problem of nearest neighbor search in high-dimensional data. Most of them are interested in constructing search/index structures for performing similarity search in high-dimensional data, e.g., image databases, document collections, time-series, genome databases. But unfortunately scalability of these structures with the dimensionality is poor; for e.g., the k-d tree performance degrades even worse than brute-force linear search if the dimensionality of the data exceeds 10 to 20. A novel method for approximate similarity search based on hashing is presented in [6]. In this they used Locality-Sensitive Hashing (LSH) to hash the points from the database ensuring that more similar points will be having higher probability of collision and furthest points are having lower

probability of collision. This scheme is scalable up to 50 or more dimensions.

Survey of a variety of data structures for nearest neighbors in geometric spaces, including variants of k-d trees, R-trees, and structures based on space-filling curves can be found in [7].

In [8], Indyk and Motwani explored a variant of the nearest neighbor search that is approximate nearest neighbor. Due to the curse of dimensionality, it is always not possible to find the exact nearest neighbors. So this exactness can be compromised by turning into approximate nearest neighbor. Instead of finding the exact nearest neighbors in high-dimensional space, we will find the closest points which are near by the exact nearest neighbor according to some distance. This is called approximate version of nearest neighbor search (NNS).A detailed theoretical study on approximation algorithms can be found in [1].

A distance-based index structure called multi-vantage point (mvp) tree for similarity search on high-dimensional metric spaces is proposed by Tolga Bozkaya and Meral Ozsoyoglu in[10].

A metric-based data structure called iDistance is proposed in[9], which projects the n-dimensional data onto single dimension by calculating the distance of query from a reference point. Here the data is partitioned using the pyramid technique.

Except iDistance[9], all of the above methods are not *adaptive* with respect to data distribution. Scalability to high-dimensions (in hundreds or thousands) and to ultra high-dimensions (in hundred thousands) is poor. Our method is scalable to high-dimensional data.

3. DATA SETS

We used a data set of drug like chemical structures from an online source ZINC [3]. ZINC which stands for a recursive acronym (*ZINC is not commercial*), is a free database of commercially available chemical compounds for virtual screening. For each chemical structure, ZINCID and SMILES notation of the chemical structure are also provided. SMILES (Simplified Molecular Input Line Entry System) representation is one of the many ways to write a chemical structure in a linear format. It is a chemical information system that gives a string representation for 2D or 3D molecule.

The drug-like data set contains a total of 8,783,230 chemical structures. It also provides 9 calculated properties- molecular Weight, logP, apolar desolvation, polar desolvation, Number of HBA, Number of HBD, tPSA, Charge, NRB and ZINCID for each molecule. Apart from these 9 physical properties provided by ZINC, 49 other features have been extracted from the SMILES notation of the chemical structures by Sankara et al. [11]. There are 58 features in total from different classes of features such as physical, atom count, structural and functional groups. From these 58 features we have taken a subset of 28 features excluding the functional groups for our experiments. These are listed in Table 1. Apart from this, we also used another data set that is AT&T Database of Faces (formerly ORL Database of Faces), which contains 400 images of 40 persons. The size of each image is 92 by 112 pixels with 256 gray levels, therefore 10304 dimensions.

Table 1: 28 features [1],[3],[4].

Physical(9)	Atom Count(10)	Structural(9)
1. Mol. Weight	10. Br. Count	20. Cyclic
2. logP	10. C Count	21. Acyclic
3. De_apolar	12. Cl Count	22. Mono Cyclic
4. De_polar	13. F Count	23. Bi Cyclic
5. HBD	14. I Count	24. Tri Cyclic
6. HBA	15. N Count	25. Tet Cyclic
7. tPSA	16. Na Count	26. Hi Cyclic
8. Change	17. O Count	27.Hetero Cyclic
9. NRB	18. P Count	28.ChiralCenters
	19. S Count	

4. APPROACH

4.1 Recursive Partitioning Method

To reduce the search space and the computational complexity we use Recursive Partitioning. It is a data space partitioning method, where in the data space is divided into bins based on the distance from a chosen reference point to the data objects in the sample. There are two phases in this method. One is *construction phase* and the other is *search phase*. In the construction phase, we project the distance onto a straight line and that line is divided into equal parts in some interval range and this will define the bin boundaries. Once the bin boundary is defined, in the construction phase we calculate the distance for a data object from its reference point and check the bin range in which the calculated distance falls, then that data object will be stored in that corresponding bin. The same procedure is followed for all the data objects and that completes the construction phase.

Algorithm 1. Construction

Step 1: Read & preprocess input data X
Step 2: Select a reference point based on selection criteria
Step 3: Find the distance from reference point to all points in X
Step 4: Equally partition the range of Distance into intervals (bin boundaries)
Step 5: Distribute all the data objects among the bins and sort them
Step 6: Repeat from step 2, for each sub-partition until stopping condition met.

In the search phase we are given a query object in the high-dimensional space and asked to return its nearest neighbors. First calculate the distance from the reference point to the query point and check in which bin it is falling so that we will directly eliminate the other bins from searching, concluding that no nearest neighbors will be found in those excluded bins. Search in the corresponding bin only and retrieve the nearest neighbors according to the distance in the order. The same procedure will be followed in the next level also, up to a specified recursion depth. In our experiments we performed this partitioning up to a recursion depth of 3, where the search space is reduced to 5-10% of the original data space. This greatly reduces the computational effort.

Algorithm 2. Search

Step 1: Read the query point q, k (no. of nns)
Step 2: Calculate the distance between reference point and q.
Step 3: Go to the appropriate bin, and if it is further partitioned into bins then repeat from step2, else continue.
Step 4: Return the top k-objects from the bin as nearest neighbors.

4.2 Nearest Neighbor Search

Nearest Neighbor Search problem is defined as retrieving the closest points for a given query from the database. The nearness is defined in terms of distance from a reference point. So the selection of reference point plays an important role in nearest neighbor search. To calculate the similarity (dissimilarity) between any two data objects, we use Tanimoto Coefficient [4] which is basically a similarity coefficient whose value is in the range 0 to 1. The similarity value 1 indicates that the points are similar and the similarity value 0 indicates that the two points are dissimilar. For any two given data points A and B, Tanimoto coefficient is defined as:

$$\boldsymbol{S(A,B)} = \frac{\sum_{i=1}^{n} \boldsymbol{a_i b_i}}{\sum_{i=1}^{n} \boldsymbol{a_i^2} + \sum_{i=1}^{n} \boldsymbol{b_i^2} + \sum_{i=1}^{n} \boldsymbol{a_i b_i}} \quad (1)$$

Where a_i and b_i represent i^{th} feature value of A and B respectively.

In addition to the above similarity measure we also used a dissimilarity-based metric called Soergel distance[5] in our experiments. Soergel Distance value ranges from 0 to 1, value 0 indicates that the two data points are similar and value 1 indicates that the two data points are dissimilar. Soergel distance coefficient is the complement of the Tanimoto (or Jaccard) association coefficient. Distance coefficients are analogous to distances in multidimensional geometric space, though they are not necessarily precisely equivalent to such distances. For any two given data points A and B, Soergel Distance is defined as:

$$\boldsymbol{D(A,B)} = \frac{\sum_{i=1}^{n} |\boldsymbol{a_i - b_i}|}{\sum_{i=1}^{n} \boldsymbol{\max(a_i b_i)}} \quad (2)$$

here a_i and b_i represent i^{th} feature value of A and B respectively.

For a distance coefficient to be described as a metric it must have the following properties:

(1) Distance values must be zero or positive, and the distance from an object to itself must be zero:
$$D_{A,B} \geq 0, D_{A,A} = D_{B,B} = 0$$
(2) Distance values must be symmetric:
$$D_{A,B} = D_{B,A}$$
(3) Distance values must obey the triangular inequality:
$$D_{A,B} \leq D_{A,C} + D_{C,B}$$
(4) The distance between non identical objects must be greater than zero:
$$A \neq B \leftrightarrow D_{A,B} > 0$$

A distance coefficient which has only the first three of these properties is called pseudo metric and one which does not have the third property is not a metric [11].

5. EXPERIMENTS AND RESULTS

Nearest Neighbor Search Using Recursive Partitioning Method

We have taken 50000 molecules randomly from the 8 million data set which is available from the docking site. For this 50000 sample data set we have 2 variants: first one is with only 7 dimensions (features) and the second one is with 28 dimensions. We have extracted the nearest neighbors from this 50000 data set using our method. To check whether these nearest neighbors are valid, we have taken a test molecule with ZINC00000012 as its zincid and given this zincid in the docking site, retrieved all its nearest neighbors which are 114 in number according to 80\% similarity criteria. These 114 nearest neighbors are added into the existing 50K data set and applied the Recursive Partitioning method, and given the query as ZINC00000012, retrieved the nearest neighbors. The result is that the same 114 nearest neighbors are falling in the single bin after partitioning, and can be retrieved by searching only in that single bin only. Here the query point given is taken as the reference point and calculated the Tanimoto Coefficient for all the points in the data set from it. The range of the calculated Tanimoto Coefficient value is 0.7203 to 1, and found the query in the 5th bin along with its 114 nearest neighbors. The reference point for all the experiments is 3/2 times (min+max) unless specified explicitly. Here the min and max are the minimum and maximum values from the distance calculated. The above experiment is also done with another random sample data set of 1048575 molecules with 7 dimensions, and got the same results. From this we can say that our method is showing good scalability when the data set size increased.

To check the dominance of a single feature on the distance distribution of tanimoto coefficient, we have taken 50000 molecules data set with 22 features and counter-part data set

with only a single feature that is molecular weight with the same sample size, and plotted the distance distribution which is the same for both. The Correlation Coefficient for these two distance distributions is 0.99, which tells that both the distributions are same. To reduce the dominance of single feature (Molecular Weight) on the distance distribution, we have done min-max normalization of the data set. After normalization the correlation coefficient value is reduced to 0.893, which is better than the earlier one and this experiment is extended to check how this dominating feature is correlated with the other features in the data set the result is shown in Fig.3. From the figure it clear that after normalization the individual feature is positively correlated with the total features.

For large data set we have compared our method with the brute-force linear search, and result is our method is taking only 8 minutes where as linear scan is taking approximately 700 minutes. This is a good achievement in computational time reduction. For another high-dimensional data set AT &T Faces Database we have got a cluster purity of 80% that is after partitioning a bin is may contain 80% of the images from the nearest neighbors of given query and 20% of images from other than query's nearest neighbors.

Search time of AT & T Faces Data set are compared for the two methods namely Linear scan and Recursive partitioning method, and are shown in the table 5. From these tables, it is clear that for high-dimensional data our method is taking more time for construction phase where as for search (online) phase it is taking less time compared to brute force linear scan. The time units are in seconds for both the tables.

Table 2: Construction time(in seconds) for different data sets

Data set	Construction time
ZINC50K7D	15
ZINC50K28D	20.05
ATT40	14.05

Table 3: Search time(in seconds) for ZINC data set of size 50000 points with 7 dimensions

Query sample size	Linear Scan	Rec. Partitioning
50	2.41956	0.02184
100	1.291056	0.02184
150	1.369888	0.02465

Table 4: Search time(in seconds) for ZINC data set of size 50000 points with 28 dimensions

Query sample size	Linear Scan	Rec. Partitioning
50	1.452672	0.021216
100	1.635972	0.023244
150	1.534208	0.020488

Table 5: Search time(in seconds) for AT&T Faces data set of size 400 points with 10304 dimensions

Query sample size	Linear Scan	Rec. Partitioning
10	1.65672	0.001563
20	2.06154	0.002344
30	2.0566	0.005729
40	2.01123	0.005078

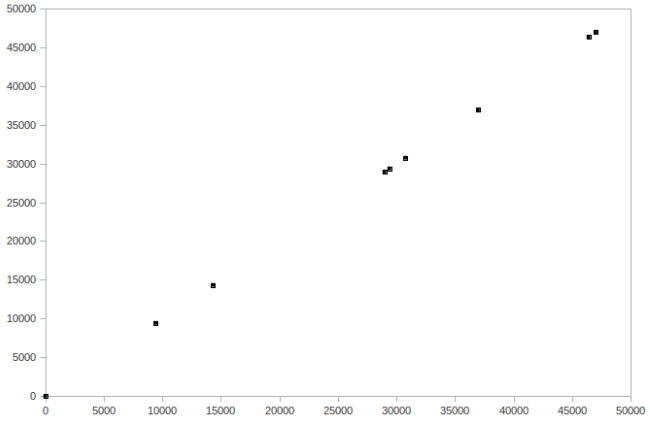

Fig. 1: Matches between nearest neighbors from Linear Scan and Recursive Partitioning Method

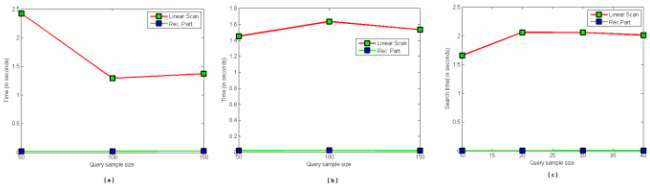

Fig. 2: Comparing Search time: a) ZINC7 dataset b) ZINC28 dataset c) AT&T Faces data set. From the figure it is evident that Recursive Partitioning is taking less than 2% of total time taken by Linear Scan(brute force), and it takes less than 0.3% for AT&T Faces data set when compared with Linear Scan search time.

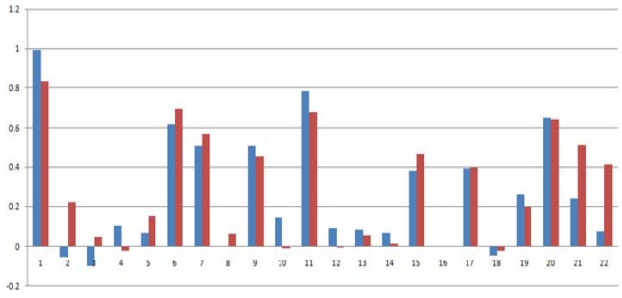

Fig. 3: Correlation between the features of data set

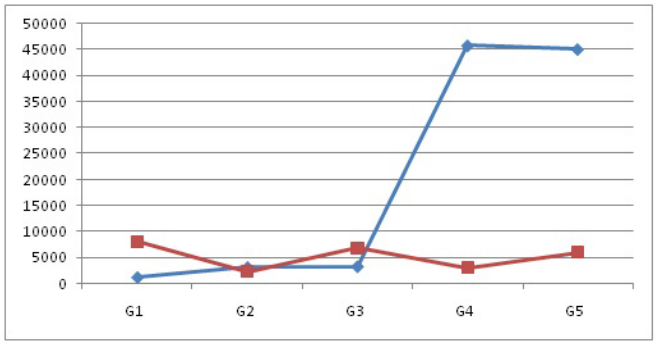

Fig. 4: The validation of the proposed method

To validate the proposed method we have taken a random two dimensional data set and checked the shuffling of points when the reference point is changed from mean to the proposed one from the subgroups. The result is that when mean is reference point, the shuffling is more from level1 to level2, where as this shuffling is less for the proposed reference point. This result is shown in Fig. 4.

6. CONCLUSION

In this paper we have presented a metric-based method for Nearest Neighbor processing. Our results show that the proposed method is effective and efficient in retrieving the nearest neighbors for a given query with less time and space. Our method is performing well when compared to brute-force linear scan and able to reduce 90% of the search space. We also studied the domination of a dense feature in the distance distribution and compared it with the other features which are sparsely distributed, which may be useful in feature selection problem that is needed for dimensionality reduction. To validate the proposed method we have taken the nearest neighbors of ZINC00000012 molecule from the docking site and those are included in the existing data set and applied our method on the data set to retrieve the nearest neighbors. The result is that the nearest neighbors from docking site and nearest neighbors from our method both are almost equal. One more validation experiment is to compare nearest neighbors both from the brute-force linear scan and our method, we plotted a graph for this which is almost linear, and so the nearest neighbors we retrieved from both the methods are same. From this we can say that our method is working well in reducing the search space. Regarding the selection of reference point, after an extensive experimentation we can say that proposed method is giving good results.

In future we want to test our method with a dimensionality reduction technique for large and high-dimensional data. For the ultra high-dimensional data set also we want to apply our method along with a dimensionality reduction method to check the performance in ultra high-dimensional space.

7. ACKNOWLEDGEMENTS

The authors would like to thank Prof. Chakravarthy Bhagvathi and Dr. S. Durga Bhavani for their valuable comments and suggestions.

REFERENCES

[1] P. Indyk, "Nearest Neighbors in High-Dimensional Spaces", Eds. J.E. Goodman and J. O'Rourke, In *Handbook of Discrete and Computational Geometr,*, chapter 39. CRC Press, 2nd edition, 2004.

[2] Poorna Chandrasekhar A., Sobha Rani T., " Storage and Retrieval of Large Data Sets: Dimensionality Reduction and Nearest Neighbour Search", *IC3 2012*, pp.262-272.

[3] ZINC- A free database for virtual screening, http://www.zinc.docking.org

[4] Sankara Rao A., Durga Bhavani S., Sobha Rani T., Bapi Raju S., Sastry G.N., "Study of Diversity and Similarity of Large Chemical Databases using Tanimoto Measure", *Proceedings of ICIP-2011, Springer (LNCS) in Communications in Computer and Information Science (CCIS)* Series (2010).

[5] Cha Sung-Hyuk., "Comprehensive survey on distance/similarity measures between probability density functions", *Int. J Math Models and Appl. Sci.* 2007;1:300–307.

[6] A. Gionis, P. Indyk, and R. Motwani, "Similarity Search in High Dimensions via Hashing", *Proceedings of the 25th VLDB Conference*, Edindurgh, Scotland,1999.

[7] H. Samet, "The Design and Analysis of Spatial Data Structures", *Addison-Wesley, Reading*, MA, 1989.

[8] P. Indyk and R. Motwani, "Approximate Nearest Neighbor - Towards Removing the Curse of Dimensionality", *In Proceedings of the 30th Symposium on Theory of Computing*, 1998, pp. 604- 613.

[9] Cui Yu, Beng Chin Ooi, Kian-Lee Tan, H. V. Jagadish.: Indexing the Distance: An Efficient Method to KNN Processing. In Proceedings of the 27th VLDB Conference, Rome, Italy, 2001.

[10] T. Bozkaya, M. Ozsoyoglu, "Distance-based Indexing for High-dimensional metric spaces", *In Proc. SIGMOD International Conference on Management of Data*, 1997, pp. 357-368.

[11] Kamichety H. M., Pradeep Natarajan, Subrata Rakshit, " An Empirical Framework to Evaluate Performance of Dissimilarity Metrics in Content Based Image Retrieval Systems", *Technical Report, Center of Artificial Intelligence and Robotics, Bangalore*, 2002.