ISSN (Print) : 2319-5940
ISSN (Online) : 2278-1021

International Journal of Advanced Research in Computer and Communication Engineering
Vol. 2, Issue 4, April 2013

Optimized Clustering Technique for High Dimensional Dataset

Bapusaheb B. Bhusare[1], Dipali G. Mogal[2]

Student, Dept. of Computer Science & Engineering, Govt. College of Engineering, Aurangabad (MH), India[1]

Student, Dept. of Computer Engineering, SSBT College of Engg. & Tech., North Maharashtra University, Jalgaon (MH), India[2].

Abstract: In data mining domain, high-dimensional and correlated data sets are used frequently. Clustering approach is represented for the analysis of similarity between the information within the database of any dimension. In order to have an effective a similarity search on a high dimensional database where exists correlated data, have to improve or extend the conventional clustering methods with different approaches. The existing indexing approaches such as vector approximation has some drawbacks such as ignoring dependencies across dimensions. This results in sub optimality in results. Thus the objective of the system is to perform clustering with exact nearest neighbor search, less number of random Inputs and Outputs over several recently proposed indexes, low computational cost and scales well with dimensions and size of the data set by tightening the cluster-distance bounds, possibly by optimizing the clustering algorithm so as to optimize the cluster distance bounds using optimization techniques like Pillar algorithm. This paper includes a new mechanism for clustering the elements of high-resolution data in order to improve precision and reduce computation time. The system applies K-means clustering after optimized by Pillar Algorithm. The Pillar algorithm considers the pillars placement which should be located as far as possible from each other to withstand against the pressure distribution of a roof, as identical to the number of centroids amongst the data distribution. This algorithm is able to optimize the K-means clustering in aspects of precision and computation time. It designates the initial centroids positions by calculating the accumulated distance metric between each data point and all previous centroids, and then selects data points which have the maximum distance as new initial centroids. This algorithm distributes all initial centroids according to the maximum accumulated distance metric.

Keywords: Multimedia database, Similarity Search, Clustering and KNN Search.

I. INTRODUCTION

During the last decade, multimedia databases have become increasingly important in many application areas such as Geographical Information system (GIS), CAD/CAM, or molecular biology, time series analysis that stores large amount of data periodically and retrieve it from database. Searching and indexing high dimensional databases is a challenging task given the large number of elements to be handled and the high dimensionality of the search space. While searches based on keywords is the current paradigm in many search engines, keywords are not necessarily the most efficient representatives of multimedia information. For example, it would be ineffective to mine databases of medical images based on keywords or “metadata” if the goal is to discover hidden correlations that are unknown and hence have not been quantified through metadata.
Similarity search is the search for elements in the database most similar to the query. A popular query model is the *k*-nearest neighbor (kNN) query, where given a query, the *k* most similar elements are extracted from the database. Since, the feature vectors themselves are large in number and of high-dimensionality; it is more cost effective to store them on a hard-storage device, typically a hard disk.
In the general database search literature, several index structures exist that facilitate search and retrieval of multi-dimensional data, such as the R-tree [11] and in low-dimensional spaces, these outperform sequential scan. But it has been observed that the performance of many multi-dimensional index structures degrades as the dimensions of the features increase and after a certain dimension threshold, they underperform sequential scan [2].
The time incurred in nearest neighbor search is largely dominated by Input Output time, which is determined by the number of sequential and random hard disk accesses. Irrespective of the access strategy, data are always stored and retrieved from the disk in units of disk blocks or pages. Random Inputs Outputs would be faster in retrieving pages that are spaced far apart while less costly sequential access of pages would optimal if the required pages are spaced close together (even if not contiguously). However, due to the exponential growth of hyper volume with dimensionality (”the curse of dimensionality” [12]), a very large portion of the space is actually empty and hence, searching on naive index structures, leads to a large

number of needless and costly random disk accesses, making it slower than the simple sequential scan.

II. LITERATURE REVIEW

In order to have an effective similarity search on a high-dimensional database where correlated data exists, have to improve or extend the conventional clustering methods with different approaches. Even though the hyper plane bounds are better than MBR and MBS bounds, they are still loose when compared with the true query-cluster distance (i.e., the cluster distance bound can be further tightened).The Kmeans algorithm generates the initial centroids randomly and fails to consider a spread out placement of them spreading within the feature space [1]. In this case, the initial centroids may be placed so close together that some become inconsequential. Because of this, the initial centroids generated by K-means may be trapped in the local optima. A method of placing the initial centroids whereby each of them has a farthest accumulated distance between them. The proposed algorithm in this paper is inspired by the thought process of determining a set of pillars locations in order to make a stable house or building. [8]Selecting an appropriate distance measure (or metric) is fundamental to many learning algorithms such as k-means, nearest neighbor searches, and others. However, choosing such a measure is highly problem specific and ultimately dictates the success or failure of the learning algorithm.

A very popular and effective technique employed to overcome the curse of dimensionality is the Vector Approximation File (VA-File). In the VA-File, the space is partitioned into a number of hyper-rectangular cells, which approximate the data that reside inside the cells. The non-empty cell locations are encoded into bit strings and stored in a separate approximation file, on the hard-disk. In the search for the nearest neighbors, first the vector approximation file is sequentially scanned and upper and lower bounds on the distance from the query vector to each cell are estimated. The bounds are used to prune the data-set of irrelevant vectors. At this point, we note that the name “Vector Approximation” is somewhat misleading, since what is actually being performed is scalar quantization, where each component of the feature vector is separately and uniformly quantized. [4] An effective cluster distance bound using hyperplane bounds is a clustering approach towards similarity search as an alternative to the Vector Approximation (VA) Files. The data set is clustered using a standard clustering or vector Quantization (VQ) technique, e.g., Kmeans and during query processing, load the”nearest” clusters into the main memory. To retrieve clusters till the k^{th} nearest neighbor discovered so far is closer to the query than the remaining clusters, which guarantees that the *k* nearest neighbors have been discovered. [4]

III. EXISTING SYSTEM

Several index structures exist that facilitate search and retrieval of multidimensional data. In low-dimensional spaces, recursive partitioning of the space with hyper rectangles or a combination of hyper spheres and hyper rectangles, have been found to be effective for nearest neighbor search and retrieval. [5]While the preceding methods specialize to Euclidean distance, Mtree has been found to be effective for metric spaces with arbitrary distance functions. Such multidimensional indexes work well in low dimensional spaces, where they outperform sequential scan. But it has been observed that the performance degrades with increase in feature dimensions and, after a certain dimension threshold, becomes inferior to sequential scan.

Bellman's “curse of dimensionality" [12] explains why it is difficult to design clustering algorithms for high-dimensional data, for the computational complexity grows exponentially fast as the dimensionality of data is increased. Reducing dimensionality has been a common approach in dealing with high dimensional data. It builds clusters in selected subspaces so that high-dimensional data can be addressed or even visualized. When certain dimensions are irrelevant for the application at hand, eliminating them to reduce data dimensionality would seem reasonable. This is equivalent to considering only the set of data at the projection of the subspace of the remaining dimensions. But considering only one subspace would inevitably cause loss of information. One can only hope that a clustering acquired from the subspace is reasonably close to the real world. We want to have efficient clustering algorithms that are suitable for database applications involving large sets of high-dimensional data. In particular, we want to devise a clustering algorithm that satisfies the following criteria:

Efficiency: It should produce a clustering in sub quadratic time.

Scalability: It should be I/O efficient, capable of dealing with a large data set.

Accuracy: It should give each dimension full and equal consideration.

Other techniques, such as LDC, iDistance, and Pyramid Tree, are based on local dimensionality reducing transformations. The data set is partitioned and, in each partition, the distances of the resident vectors to some reference point, typically the centroids, are evaluated. The feature vectors in a partition are now indexed by their centroids distance, using ubiquitous one-dimensional indexes such as the B+-tree. During query processing, spheres of gradually increasing radii are drawn around the query, until they intersect a cluster sphere. Now, the relevant elements in the partition, identified by centroids distances which lie in the intersecting region, are retrieved for finer scrutiny. The search radius is set to such a value that the exact NNs are returned. In another existing approach based on the transformation to and indexing, LDC, another approximation layer is created, by generating a box identification code for each resident point. Once an initial set of candidates have been identified with the B+-tree, the corresponding approximations are scanned to further filter out irrelevant points within the partition. The surviving elements are finally retrieved from the hard drive to determine the

ISSN (Print) : 2319-5940
ISSN (Online) : 2278-1021

International Journal of Advanced Research in Computer and Communication Engineering
Vol. 2, Issue 4, April 2013

nearest neighbors. In order to reduce disk IO, care is taken to control the maximal fraction of space searched, yet return the exact nearest neighbors.
Spatial queries, specifically nearest neighbor queries, in high-dimensional spaces have been studied extensively. While several analyses have concluded that the nearest neighbor search, with euclidean distance metric, is impractical at high dimensions due to the notorious "curse of dimensionality," others have suggested that this may be over pessimistic. Specifically, what determine the search performance are the intrinsic dimensionality of the data set and not the dimensionality of the address space.

IV. PROPOSED SYSTEM

In our proposed system we have to optimize the clustering algorithm by tightening the cluster distance bounds by applying the technique called Pillar algorithm for optimization. This is inspired by the thought process of determining a set of pillars' locations in order to make a stable house or building. We consider the pillars which should be located as far as possible from each other to withstand against the pressure distribution of a roof, as number of centroids among the gravity weight of data distribution in the vector space. Therefore, our enhanced approach in this paper designates positions of initial centroids in the farthest accumulated distance between them in the data distribution. This approach is composed of by determining the initial centroids and then represents the outlier detection method.
The main issue of the proposed approach is to have an efficient similarity search based on the indexing for the high dimensional data. Here in this approach they have proposed a new cluster-adaptive distance bound based on separating hyperplane boundaries of Voronoi clusters to complement our cluster based index. Initially the index construction is performed by the creation of Nearest Neighbour clusters. Thus the elements obtained from the same clusters are stores together by a storage strategy concept. Next the kNN based search algorithm is presented which is based on the branch and bound algorithm, by accessing the clusters in the order of the lower bounds to the query distance.
The proposed approach performs clustering with exact nearest neighbour search. So the search time will be reduced. Input Output cost will reduce because of the less number of random IOs over several recently proposed indexes. Low computational cost and scales well with dimensions and size of the data set, by tightening the cluster-distance bounds by optimizing the algorithm.

The architecture is divided into four modules:

A. Data Set Extraction
B. Implementation of pillar optimization
C. KNN Search result implementation
D. Performance measure

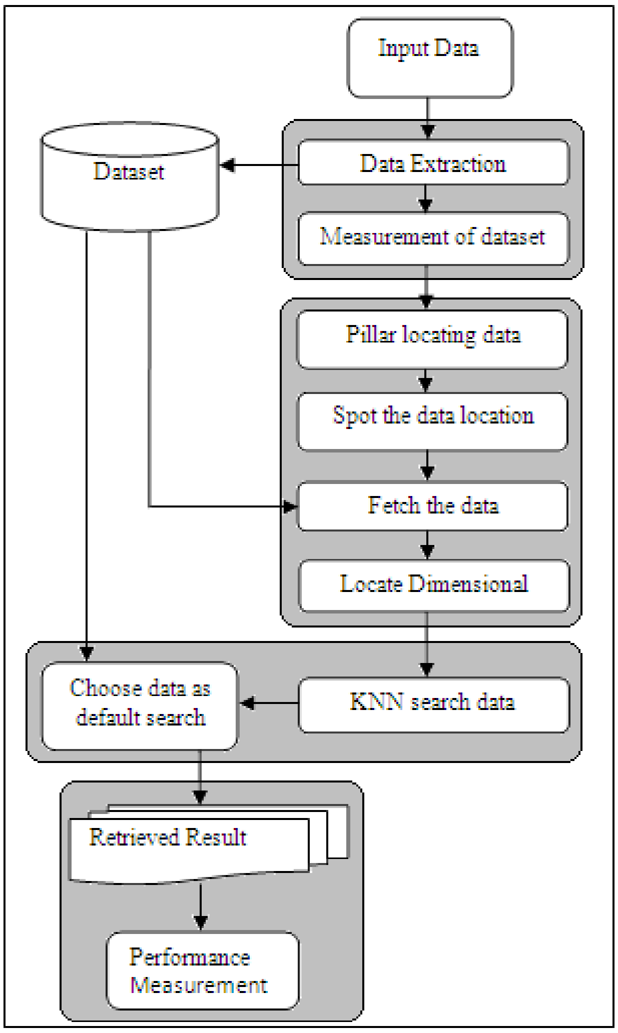

Fig. 1 Proposed System Architecture

A. Data Set Extraction

We use the real data-set, SENSORS, which was generated by the Intel Berkeley Research Lab4. Data were collected from 54 sensors deployed in the Intel Berkeley Research lab between February 28 and April 5, 004. Each sensor measures humidity, temperature, light and voltage values once every 31 seconds. We retain data from those sensors that generated in excess of 50,000 readings. This corresponds to 4 types of measurements - temperature, light, and humidity and voltage readings - from 15 sensors which is equivalent to 60 correlated sources.

B. Implementation of pillar optimization

This is inspired by the thought process of determining a set of pillars locations in order to make a stable house or building. We consider the pillars which should be located as far as possible from each other to withstand against the pressure distribution of a roof, as number of centroids among the gravity weight of data distribution in the vector space.Here we consider with specify location of data in a large dataset user construct the specify centroids. And thus the data is get clustered and viewed as a specify dimension of data.

ISSN (Print) : 2319-5940
ISSN (Online) : 2278-1021

International Journal of Advanced Research in Computer and Communication Engineering
Vol. 2, Issue 4, April 2013

The K-means algorithm for clustering data considering that its ability to cluster huge data, and also outliers, quickly and efficiently [8]. However, Because of initial starting points generated randomly, K-means algorithm is difficult to reach global optimum, but only to one of local minima [17] which it will lead to incorrect clustering results. Barakbah and Helen [15] performed that the error ratio of K-means is more than 60% for well-separated datasets. To avoid this phenomenon, we use our previous work regarding initial clusters optimization for K-means using Pillar algorithm [15]. The Pillar algorithm is very robust and superior for initial centroids optimization for K-means by positioning all centroids far separately among them in the data distribution.

Therefore, this algorithm designates positions of initial centroids in the farthest accumulated distance between them in the data distribution.

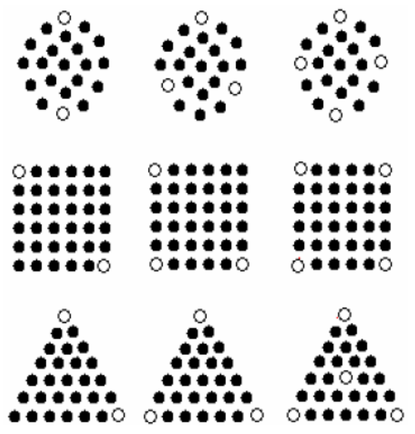

Fig. 2 Illustration of locating a set of pillar (white points) with standing against different pressure distribution of roofs.

The Pillar algorithm is described as follows. Let $X=\{xi \mid i=1,\dots,n\}$ be data, k be number of clusters, $C=\{ci \mid i=1,\dots,k\}$ be initial centroids, $SX \subseteq X$ be identification for X which are already selected in the sequence of process, $DM=\{xi \mid i=1,\dots,n\}$ be accumulated distance metric, $D=\{xi \mid i=1,\dots,n\}$be distance metric for each iteration, and m be the grand mean of X. The following execution steps of the proposed algorithm are described as:

```
Algorithm 1: Pillar
1. Set $C=\emptyset$, $SX=\emptyset$, and $DM=[]$
2. Calculate $D \leftarrow \text{dis}(X, m)$
3. Set number of neighbors $nmin = \alpha \cdot n / k$
4. Assign $dmax \leftarrow argmax\ (D)$
5. Set neighborhood boundary $nbdis = \beta \cdot dmax$
6. Set $i=1$ as counter to determine the $i$-th initial centroids
7. $DM = DM + D$
8. Select $ж \leftarrow xargmax\ (DM)$ as the candidate for $i$th initial centroids
9. $SX=SX \cup ж$
10. Set $D$ as the distance metric between $X$ to $ж$.
11. Set $no \leftarrow$ number of data points fulfilling $D \le nbdis$
12. Assign $DM(ж)=0$
13. If $no < nmin$, go to step 8
14. Assign $D(SX)=0$
15. $C = C \cup ж$
16. $i = i + 1$
17. If $i \le k$, go back to step 7
18. Finish in which $C$ is the solution as optimized initial centroids.
```

After getting the optimized initial centroids, we apply clustering using the K-means algorithm and then obtain the position of final centroids. We use these final centroids as the initial centroids for the real dataset, and then apply the data point clustering using K-means.

C. KNN Search result implementation

Many nearest neighbor search algorithms have been proposed over many past years; these generally seek to reduce the number of distance evaluations is actually performed [17]. K-NN is computationally traceable using an appropriate nearest neighbor search algorithm which is also used for large data sets[18]. Approximation file is scanned in the upper region and lower region which is used for calculating the distance from query vectors to the candidate region [19]. The data is accessed in order of the lower bounds to the query distance. The search procedure continued till stopping condition is attained. All clusters have been searched to retrieve clusters in order of distance.

The kNN based search algorithm is presented which is based on the branch and bound algorithm. By accessing the clusters in the order of the lower bounds to the query distance.At the particular location of user view the user cluster the data and fetch the information. It's measured with different dimension of search location.

```
Algorithm 2: KNN-SEARCH
1: Declare n, f [n], i=0, upper, lower;
2: While (i <= n)
3: Read f [i]
4: Determine upper and lower
5: Delete upper bounds
Set upper =0
6: Take lower bound
 7: if (Lower< K-NN)
Increment I Go to step2
  Else
Stop
```

D. Performance measurement

We measure the performance analysis of pillar algorithm and constructed its variation at stage 1.And at the next level we measure the KNN search performance metrics. At the final level we measure the comparison of both algorithms.

In our Sensor dataset which consist of 5000 data we measure the Distance, Hyperbound and File length and number of Vectors which is shown in given below chart.

ISSN (Print) : 2319-5940
ISSN (Online) : 2278-1021

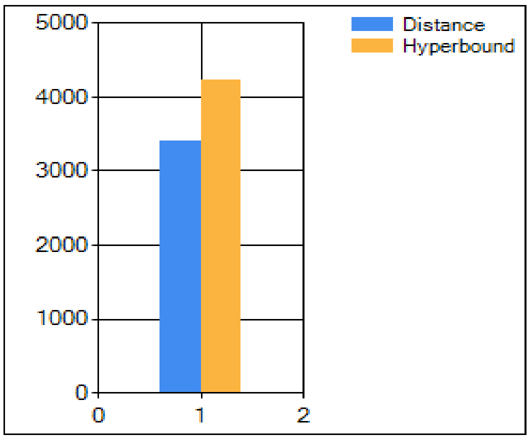

Chart1. Distance and Hyperbound

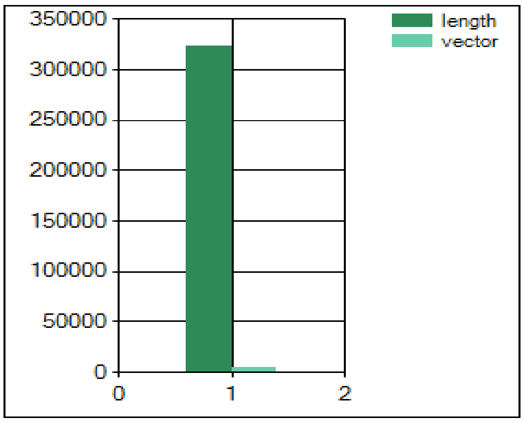

Chart2. No. of Length and Vector

V. CONCLUSION AND FUTURE WORK

Real multidimensional data sets exhibit non uniform distributions and significant correlations. Hence, indexing with VA-File, by performing uniform, scalar quantization is suboptimal. To overcome this problem, we are tightening the cluster-distance bounds by optimizing the algorithm, the computational cost will be reduced and the dimensions and size of the data set can be scaled. Less number of random Inputs Outputs over several recently proposed indexes. Because of the cluster with exact nearest neighbor search, the search time reduces. Possibly by optimizing the clustering algorithm, the query cluster distance bounds can be further tightened, so as to optimize the cluster distance bounds. Future efforts would be directed toward this and other related problems.

REFERENCES

[1] S. Ramaswamy and K. Rose, "Adaptive Cluster Distance Bounding for High Dimensional indexing" Transactions on Knowledge and data engineering, vol. 23. June 2011.
[2] R. Weber, H. Scheck, and S. Blott, "A quantitative analysis and performance study for similarity-search methods in high-dimensional spaces." in *VLDB*, August 1998.
[3] A.R. Barakbah, K. Arai, "Hierarchical K-means: an algorithm for centroids initialization for K-means," Reports of the Faculty of
Science and Engineering, Saga University, Japan, 2007 Vol. 36.
[4] S. Ramaswamy and K. Rose, "Adaptive Cluster-Distance Bounding for Similarity Search in Image Databases," Proc. Int'l Conf. Image Processing (ICIP), vol. 6, 2007
[5] P. Ciaccia, M. Patella, and P. Zezula, "M-tree: An efficient access method for similarity search in metric spaces." in *VLDB*, 1997.
[6] E. Tuncel, P. Koulgi, and K. Rose, "Rate-Distortion Approach to Databases: Storage and Content-Based Retrieval," IEEE Trans. Information Theory, vol. 50, no.6, June 2004.
[7] M.Divya and T.Senthil Prakash "Achieving True Query-Cluster Distance Bound by Optimizing Clustering Algorithm" International Journal of Communications and Engineering Volume 02, March 2012.
[8] Ali Ridho Barakbah and Yasushi Kiyoki "A Pillar Algorithm for Kmeans Optimization by Distance Maximization for Initial Centroid Designation", Inf. Technol. Dept., Electron. Eng. Polytech. Inst. Of Technol., Surabaya, 2009.
[9] H. Ferhatosmanoglu, E. Tuncel, D. Agrawal, and A. E. Abbadi, "Vector approximation based indexing for non-uniform high dimensional data sets," in *CIKM*, 2000.
[10] E. Tuncel, H. Ferhatosmanoglu, and K. Rose, "VQ-Index: An index structure for similarity searching in multimedia databases." in *ACM Multimedia*, 2002.
[11] A. Guttman, "R-trees: A dynamic index structure for spatial searching." in *SIGMOD Conference*, 1984.
[12] R.E. Bellman, "*Adaptive Control Processes: A GuidedTour*", Princeton University Press, Princeton, NJ, 1961.
[13] C.C. Aggarwal, A. Hinneburg, and D.A. Keim, "On the Surprising Behavior of Distance Metrics in High Dimensional Spaces," Proc. Int'l Conf. Database Theory (ICDT), 2001.
[14] Ali Ridho Barakbah and Yasushi Kiyoki "A New Approach for Image Segmentation using Pillar-Kmeans Algorithm" International Journal of Information and Communication Engineering 2010.
[15] A.R. Barakbah, A. Helen, "Optimized K-means: an algorithm of initial centroids optimization for K-means", Proc. Seminar on Soft Computing, Intelligent System, and Information Technology (SIIT), Surabaya, 2005.
[16] B. Kövesi, J.M. Boucher, S. Saoudi, "Stochastic K-means algorithm for vector quantization", Pattern Recognition Letters, Vol. 22, pp. 603-610, 2001.
[17] S. Arya, D.M. Mount, N.S. Netanyahu, R. Silverman, and A. Wu. "An optimal algorithm for approximate nearest neighbor searching", In Proceedings of the 5th Annual ACM-SIAM Symposium on Discrete Algorithms, 1994.
[18] Arya S.: "Nearest Neighbor Searching and Applications", Ph.D. Thesis, University of Maryland, 1995.
[19] P. Indyk and R. Motwani, "Approximate nearest neighbors: Toward removing the curse of dimensionality", in Proc. ACM symp.Theory of Computing, 1998