ISSN : 0976-8491 (Online) | ISSN : 2229-4333 (Print)

Using Grouping and KNN Search Algorithm for High Length Catalo

[1]B. Venkateswarulu, [2]Y. Vinay Kumar

[1,2]Dept. of CSE, Avanthi Institute of Engg. & Tech., Narsipatnam, Visakhapatnam, AP, India

Abstract

We propose a new group-adaptive space bound based on separating hyper plane boundaries of Verona groups to complement our group based catalog. This bound enables efficient spatial filtering, with a relatively small pre-processing storage overhead and is applicable to Euclidean and Mahalanobis similarity measures. Experiments in exact nearest-neighbour set retrieval, conducted on real data sets, show that our cataloguing method is scalable with data set size and data lengthily and outperforms several recently proposed cataloges. Consider approach for likeness search in interrelated, high-length data sets, which are derived within a grouping framework. They note that Catalog by "Vector Approximation" (VA-File),
Which was proposed as a technique to combat the "irritation of Lengthily," employs scalar quantization, and hence necessarily ignores dependencies across dimensions, which represents a source of sub-optimality? Grouping, on the other hand, exploits inter-length correlations and is thus a more compact representation of the data set. However, existing methods to prune irrelevant groups are based on bounding hyper-spheres and/or bounding rectangles, whose lack of tightness compromises their efficiency in exact nearest neighbour search. They propose a new group-adaptive space bound based on separating hyper-plane boundaries of Verona groups to complement their group based catalog.

Keywords

Intrusion Detection, Probabilistic Techniques, Alert Aggregation

I. Introduction

During the last decade, multimedia databases have become increasingly important in many application areas such as medicine, CAD, geography, or molecular biology. An important research issue in the field of multimedia databases is the content based retrieval of similar multimedia objects such as images, text, and videos. However, in contrast to searching data in a relational database, a content based retrieval requires the search of similar objects as a basic functionality of the database system. Most of the approaches addressing similarity search use a so-called feature transformation which transforms important properties of the multimedia objects into high-length points (feature vectors). Thus, the similarity search is transformed into a search of points in the feature space which are close to a given query point in the high-length feature space. Query processing in high-length spaces has therefore been a very active research area over the last few years. A number of new catalog structures and algorithms have been proposed. It has been shown that the new catalog structures considerably improve the performance in querying large multimedia databases. IDS usually focus on detecting attack types, but not on distinguishing between different attack instances.

In addition, even low rates of false alerts could easily result in a high total number of false alerts if thousands of network packets or log file entries are inspected. As a consequence, the IDS create many alerts at a low level of abstraction. It is extremely difficult for a human security expert to inspect this flood of alerts, and decisions that follow from single alerts might be wrong with a relatively high probability. Alerts may originate from low-level IDS such as those mentioned above, from firewalls, etc. Alerts that belong to one attack instance must be grouped together and meta-alerts must be generated for these groups. It is a generative modelling approach using probabilistic methods.

Assuming that attack instances can be regarded as random processes "producing" alerts, we aim at modelling these processes using approximate maximum likelihood parameter estimation techniques. Thus, the beginning as well as the completion of attack instances can be detected. It is a data stream approach, i.e., each observed alert is processed only a few times. Thus, it can be applied on-line and under harsh timing constraints. To outline the preconditions and objectives of alert aggregation we start with a short sketch of our intrusion framework. Then, briefly describe the generation of alerts and the alert format. Finally introduce a grouping algorithm for off-line alert aggregation which is basically a parameter estimation technique for the probabilistic model.

A. Feature-Based Processing Of Similarity

Queries an important aspect of similarity queries is the similarity measure. There is no general definition of the similarity measure since it depends on the needs of the application and is therefore highly application dependent. Any similarity measure, however, takes two objects as input parameters and determines a positive real number, denoting the similarity of the two objects. In defining similarity queries, we have to distinguish between two different tasks, which are both important in multimedia database applications: similarity means that we are interested in all objects of which the similarity to a given search object is below a given threshold e and NN-similarity means that we are only interested in the objects which are the most similar ones with respect to the search object. Our approach to Catalog real high length data sets. We focus on the grouping paradigm for search and retrieval. The data set is grouped, so that groups can be retrieved in decreasing order of their probability of containing entries relevant to the query.

II. Prose Analysis

In order to have an effective similarity search on a high-length database where correlated data exists, have to improve or extend the conventional grouping methods with different approaches. A space based bound approach of adaptive grouping in the high length database is implemented. A new group adaptive space bound based on separating hyper plane boundaries of Verona groups to complement our group based catalog is proposed. Even though the hyper plane bounds are better than MBR and MBS bounds, they are still loose when compared with the true query-group space (i.e., the group space bound can be further tightened). [1] The Kmeans algorithm generates the initial centroids randomly and fails to consider a spread out placement of them spreading within the feature space. 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 space 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 [3]. Selecting an appropriate space 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 [4]. A very popular and effective technique employed to overcome the curse of lengthily 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 space from the query vector to each cell are estimated. The bonds are used to prune the data-set of irrelevant vectors. The final set of candidate vectors are then read from the hard disk and the exact nearest neighbors are determined. 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.[6] An effective group space bound using hyper plane bounds is a grouping approach towards similarity search as an alternative to the Vector Approximation (VA) Files. The data set is grouped using a standard grouping or vector Quantization (VQ) technique, e.g., Kmeans or Lloyd's algorithm and during query processing, load the "nearest" groups into the main memory. To retrieve groups till the kth nearest neighbor discovered so far is closer to the query than the remaining groups, this guarantees that the k nearest neighbors has been discovered.

III. Existing System

However, existing methods to prune irrelevant groups are based on bounding hyper spheres and/or bounding rectangles, whose lack of tightness compromises their efficiency in exact nearest neighbour search.
Spatial queries, specifically nearest neighbour queries, in high-length spaces have been studied extensively. While several analyses have concluded that the nearest neighbour search, with Euclidean space metric, is impractical at high dimensions due to the notorious "curse of lengthily", others have suggested that this may be over pessimistic. Specifically, the authors of have shown that what Determines the search performance (at least for R-tree-like structures) is the intrinsic lengthily of the data set and not the lengthily of the address space (or the embedding lengthily). We extend our space bounding technique to the Mahalanobis space metric, and note large gains over existing cataloges.

IV. Proposed System

In existing system Hyperspheres and hyper rectangles are generally not optimal bounding surfaces for groups in high length spaces. The Vector Approximation is somewhat confusing, since what is actually being performed is scalar quantization, where each component of the feature vector is separately and uniformly quantized. The performance of the previous approaches got reduced by increase in the lengthily of the feature. The preprocessing storage required by these approaches is higher.
We propose a new group-adaptive space bound based on separating hyper plane boundaries of Verona groups to complement our group based catalog. This bound enables efficient spatial filtering, with a relatively small pre-processing storage overhead and is applicable to Euclidean and Mahalanobis similarity measures. An experiment in exact nearest-neighbour set retrieval, conducted on real data-sets, shows that our Catalog method is scalable with data-set size and data lengthily and outperforms several recently proposed cataloges.
We note that the Vector Approximation (VA)-file technique implicitly assumes independence across dimensions, and that each component is uniformly distributed. This is an unrealistic assumption for real data-sets that typically exhibit significant correlations across dimensions and non-uniform distributions. To approach optimality, an Catalog technique must take these properties into account. We resort to a Verona grouping framework as it can naturally exploit correlations across dimensions (in fact, such grouping algorithms are the method of choice in the design of vector quantizes). Moreover, we show how our grouping procedure can be combined with any other generic grouping method of choice (such as BIRCH) requiring only one additional scan of the data-set. Lastly, we note that the sequential scan is in fact a special case of grouping based catalog i.e. with only one group. Several catalog structures exist that facilitate search and retrieval of multi-length data. In low length spaces, recursive partitioning of the space with hyper-rectangles hyper-spheres or a combination of hyper-spheres and hyper-rectangles have been found to be effective for nearest neighbour search and retrieval. While the preceding methods specialize to Euclidean space (l2 norm), M-trees have been found to be effective for metric spaces with arbitrary space functions (which are metrics).
Such multi-length cataloges work well in low length 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. In a celebrated result, Weber et. Al have shown that whenever the lengthily is above 10, these methods are outperformed by simple sequential scan. Such performance degradation is attributed to Bellman's 'curse of lengthily', which refers to the exponential growth of hyper-volume with lengthily of the space.

V. Module Description

- A New Group Space Bound
- Adaptability to Weighted Euclidean or Mahalanobis Spaces
- An Efficient Search Catalog
- Vector Approximation Files
- Approximate Similarity Search

A. A New Group Space Bound

Key to the success of the grouping-based search strategy is efficient bounding of query-group spaces. This is the mechanism that allows the elimination of irrelevant groups. Traditionally, this has been performed with bounding spheres and rectangles. However, hyper spheres and hyper rectangles are generally not optimal bounding surfaces for groups in high length spaces. In fact, this is a phenomenon observed in the SR-tree, where the authors have used a combination spheres and rectangles, to outperform cataloges using only bounding spheres (like the SS-tree) or bounding rectangles (R-tree).
The premise herein is that, at high dimensions, considerable improvement in efficiency can be achieved by relaxing restrictions on the regularity of bounding surfaces (i.e., spheres or rectangles). Specifically, by creating Verona groups, with piecewise-linear boundaries, we allow for more general convex polygon structures that are able to efficiently bound the group surface. With the

construction of Verona groups under the Euclidean space measure, this is possible. By projection onto these hyper plane boundaries and complementing with the group-hyper plane space, we develop an appropriate lower bound on the space of a query to a group.

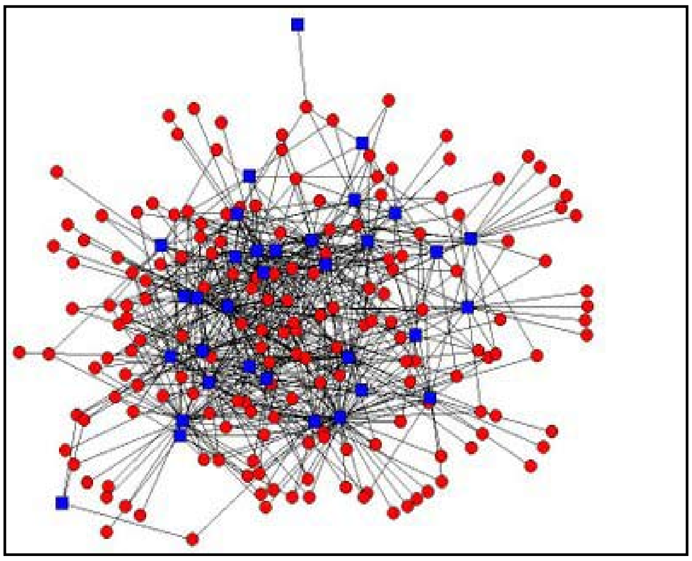
Fig. 1:

B. Adaptability to Weighted Euclidean or Mahalanobis Spaces

Euclidean space metric is popular within the multimedia Catalog community it is by no means the “correct” space measure, in that it may be a poor approximation of user perceived similarities. The Mahalanobis space measure has more degrees of freedom than the Euclidean space and by proper updating (or relevance feedback), has been found to be a much better estimator of user perceptions and more recently) . We extend our space bounding technique to the Mahalanobis space metric, and note large gains over existing cataloges.

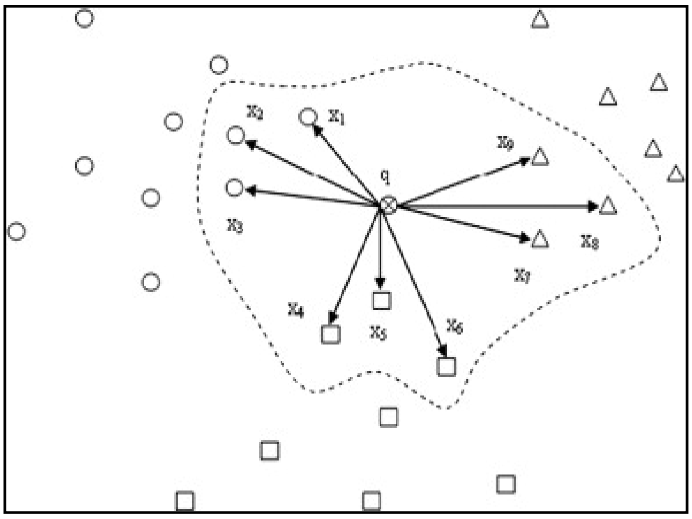

Fig. 2:

C. An Efficient Search Catalog

The data set is partitioned into multiple Verona groups and for any k NN query, the groups are ranked in order of the hyper plane bounds and in this way, the irrelevant groups are filtered out. We note that the sequential scan is a special case of our Catalog, if there were only one group. An important feature of our search catalog is that we do not store the hyper plane boundaries (which form the faces of the bounding polygons), but rather generate them dynamically, from the group centroids. The only storage apart from the centroids are the group-hyper plane boundary spaces (or the smallest group-hyper plane space). Since our bound is relatively tight, our search algorithm is effective in spatial filtering of irrelevant groups, resulting in significant performance gains.

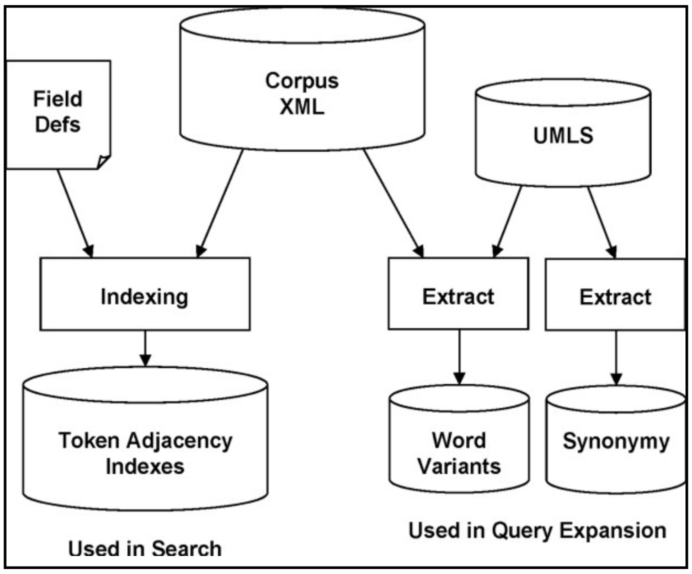

Fig. 3:

We expand on the results and techniques initially presented in, with comparison against several recently proposed Catalog techniques.

D. Vector Approximation Files

A popular and effective technique to overcome the curse of lengthily is the vector approximation file (VA-File). VA-File partitions the space into hyper-rectangular cells, to obtain a quantized approximation for the data that reside inside the cells. Non-empty cell locations are encoded into bit strings and stored in a separate approximation file, on the hard-disk. During a nearest neighbour search, the vector approximation file is sequentially scanned and upper and lower bounds on the space from the query vector to each cell are estimated. The bonds are used to prune irrelevant cells. The final set of candidate vectors are then read from the hard disk and the exact nearest neighbours are determined. At this point, we note that the terminology “Vector Approximation” is somewhat confusing, since what is actually being performed is scalar quantization, where each component of the feature vectors separately and uniformly quantized (in contradistinction with vector quantization in the signal compression literature). VA-File was followed by several more recent techniques to overcome the curse of lengthily. In the VA+-File, the data-set is rotated into a set of uncorrelated dimensions, with more approximation bits being provided for dimensions with higher variance. The approximation cells are adaptively spaced according to the data distribution. Methods such as LDR and the recently proposed non-linear approximations aim to outperform sequential scan by a combination of grouping and lengthily reduction. There also exist a few hybrid methods, such as the A-Tree, and IQ-Tree, which combine VA-style approximations within a tree based catalog.

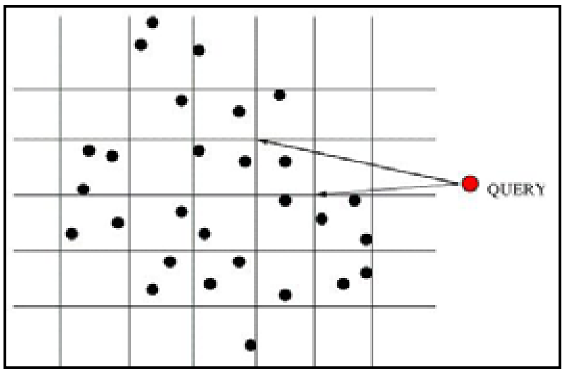

Fig. 4:

E. Approximate Similarity Search

Lastly, it has been argued that the feature vectors and space functions are often only approximations of user perception of similarity. Hence, even the results of an exact similarity search are inevitably perceptually approximate, with additional rounds of query refinement necessary. Conversely, by performing an approximate search, for a small penalty in accuracy, considerable savings in query processing time would be possible. Examples of such search strategies are MMDR probabilistic searches and locality sensitive hashing .The reader is directed to for a more detailed survey of approximate similarity search. The limits of approximate Catalog i.e. the optimal tradeoffs between search quality and search time has also been studied within an information theoretic framework.

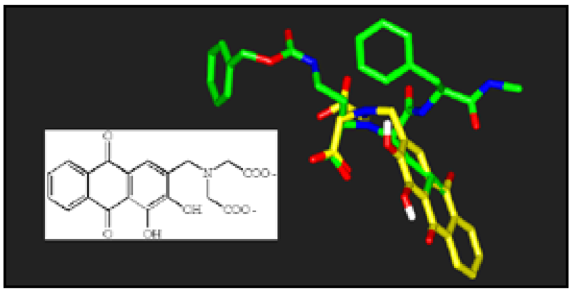

Fig. 5:

VI. Grouping and Catalog Structure

There exist several techniques of grouping the data-set, from the fast K-means algorithm (which requires multiple scans of the data-set) and Generalized Lloyd Algorithm (GLA) [8] to methods such as BIRCH [9], which require only a single scan of the data-set. The output of any of these algorithms can be a starting point. From each of the K groups detected by a generic grouping algorithm, a pivot is chosen i.e. K pivot points in all.

```
Algorithm 1 VORONOI-CLUSTERS($\mathcal{X}$,K)
1: //Generic clustering algorithm returns
   //K cluster centroids
   $\{\mathbf{c}_m\}_{=1}^{K} \leftarrow$ GenericCluster($\mathcal{X}$,$K$)
2: set $l = 0$, $\mathcal{X}_1 = \phi, \mathcal{X}_2 = \phi, \ldots, \mathcal{X}_K = \phi$
3: while $l < |\mathcal{X}|$ do
4:     $l = l + 1$
5:     //Find the centroid nearest to data element $\mathbf{x}_l$
       $k = \arg\min_m d(\mathbf{x}_l, \mathbf{c}_m)$
6:     //Move $\mathbf{x}_l$ to the corresponding Voronoi partition
       $\mathcal{X}_k = \mathcal{X}_k \bigcup \{\mathbf{x}_l\}$
7: end while
8: return $\{\mathcal{X}_m\}_{=1}^{K}, \{\mathbf{c}_m\}_{=1}^{K}$
```

Then the entire data-set is scanned and each data-element is mapped to the nearest pivot. Data mapping to the same pivot are grouped together to form Verona groups (see Algorithm 1). This would lead to slight re-arrangement of groups, but this is necessary to retain piecewise linear hyper plane boundaries between groups. We believe the centroid is a good choice as a pivot. Thus, quick Verona grouping, with possibly only a single scan of the entire data-set, can be achieved using any generic grouping algorithm. Lastly, also note that any Catalog scheme would need at least one scan of the database, which indicates that catalog construction times for our scheme are as very close to the minimum possible.

We note that the K-means, GLA and BIRCH algorithms are fast and can generate reliable estimates of group centroids from sub-samples of the data-set. Typically, for K groups, even a sub-sample of size 100K is sufficient. As we shall see, for the range of groups we are considering, this would be overwhelmingly smaller than the data-set. Faster catalog construction would be possible by allowing for hierarchical and multi-stage grouping. However, only the groups at the leaf level are returned. We tested several grouping techniques including GLA and BIRCH, and the results were largely similar. While it is possible to also optimize the grouping itself, that is not our goal in these experiments.

A. Storage Strategy

Elements within the same group are stored together (contiguously). We retain the group centroids cm and maintain pointers from each centroid to the location of the corresponding group on the hard-disk. We also maintain in a separate file the space (bounds) of each group from its bounding hyper planes.

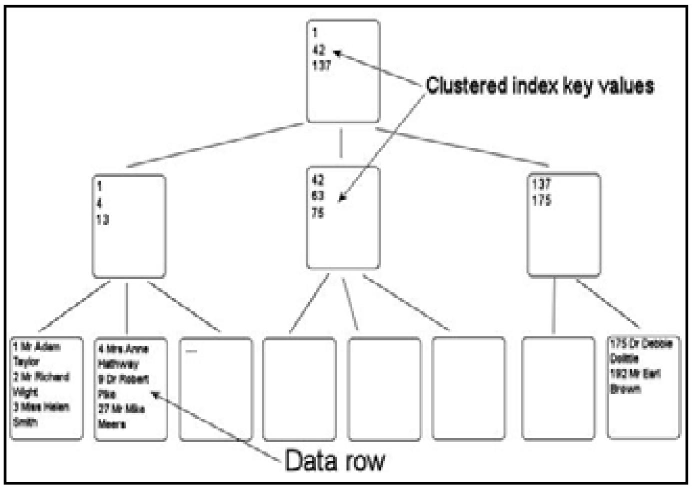

Fig. 6:

We note that the total storage is O(Kd+K2) and O(K(d+1)) real numbers, for the full and reduced complexity hyper plane bounds respectively, where K is the number of groups. Lastly, note that while the elements of each group are stored contiguously, the individual groups are stored on different parts of the disk, with enough space provided to allow them to grow and shrink.

VII. The KNN Search Algorithm

We now present KNN-SEARCH, our procedure for k-NN search. Our algorithm is a branch-and-bound algorithm, and since the groups are accessed in order of the lower bounds to the query space, it is guaranteed to return the k-nearest neighbors. The main search algorithm KNN-SEARCH (see Algorithm 2) calls on four functions

- HyperplaneBound(q) - returns lower bounds on the space between query q and all groups, using separating hyper plane boundaries (as explained in section V-A).
- Sort Array(a[],'ascend') - sorts an array a in ascending order and returns the sorted array and sorting "rank" (order).
- Farthest(x,A) - returns the space of the element in A furthest from x.
- FindkNNsIn(q,A, I) - for query q and initial candidate list I, finds and returns the kNNs in group A, as well the number of elements in the group
- LoadNextPage (F) - loads a page of a group into main memory (RAM), where F is the pointer to the group file
- FindkNN(x,A) - finds the k-nearest neighbors of x in set A.

Algorithm 2 KNN-SEARCH(q)

```
1: //Initialize
   set FLAG=0, $count = 0, N = 0, kNN = \phi$
2: //Evaluate query-cluster distance bounds
   $d_{LB}[]$ ←HyperplaneBound(q)
3: //Sort the query-cluster distance bounds in ascending
   //order
   $\{d_{LB}^{sort}[], o[]\}$ ←SortArray($d_{LB}$,'ascend')
4: while FLAG==0 do
5:    $count = count + 1$
6:    //Find the kNNs upto current cluster
      $\{N_c, kNN\}$ ←FindkNNsIn(q, $\mathcal{X}_{o[count]}$, $kNN$)
7:    //Update number of elements scanned
      $N = N + N_c$
8:    //Find the kNN radius
      $d_{kNN}$=Farthest(q, $kNN$)
9:    if $count < K$ then
10:      if $N > k$ then
11:         if $d_{kNN} < d_{LB}^{sort}[count + 1]$ then
12:            set FLAG=1 //kNNs found, search ends
13:         end if
14:      end if
15:   else
16:      set FLAG=1 //all clusters scanned, search ends
17:   end if
18: end while
19: return $kNN$
```

Algorithm 3 FindkNNsIn(q, $\mathcal{A}, \mathcal{I}$)

```
1: set $N_c = 0$, $F$=Open($\mathcal{A}$), $kNN = \mathcal{I}$
2: while !(EOF($F$)) do
3:    // Load the next cluster page
      $\mathcal{C}$=LoadNextPage(F)
4:    //Merge kNN list with current page
      $\mathcal{X}_{cand} = \mathcal{C} \bigcup kNN$
5:    //Find the kNNs within the candidate list
      $kNN[]$ ←Find$_{kNN}$(q, $\mathcal{X}_{cand}$)
6:    //Update number of elements scanned
      $N_c = N_c + |\mathcal{C}|$
7: end while
8: return $N_c, kNN$
```

For every query, the processing starts with a call to Hyperplane Bound (q). The centroids and the group hyper plane spaces are loaded into the main memory. The hyper plane bounds are calculated and returned. These lower bounds are sorted in ascending order and the groups are correspondingly ranked (line 3). Then, the first (nearest) group is scanned from the hard-disk (one page at a time2, see Algorithm 3) and the kNNs within this subset of the data-set (line 6) are identified. Additionally, dkNN, the space of the kth-NN from the query, is evaluated (line 8) and stored in main memory. If this space is less than the space of the next closest group, then the search ends as the kNNs have been found. In other words, this is the stopping condition (line 12). Otherwise, the second group is scanned from the disk. The previous estimate of the kNNs is merged with the entries of the current group to form a new candidate set and the kNNs within this new candidate set are evaluated. This search procedure continues till the stopping condition is reached or all groups have been searched .

A. Advantages of the Proposed System

The proposed approach performs grouping with exact nearest neighbor search. So the search time will be reduced. IO cost will reduce because of the less number of random IOs over several recently proposed cataloges. Low computational cost and scales well with dimensions and size of the data set, by tightening the group-space bounds by optimizing the algorithm.

VIII. Conclusion and Future Work

As we are tightening the group-space 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 IOs over several recently proposed cataloges. Because of the group with exact nearest neighbor search, the search time reduces. Possibly by optimizing the grouping algorithm, the query group space bounds can be further tightened, so as to optimize the group space bounds. Future efforts would be directed toward this and other related problems.

Real multilength data-sets exhibit significant correlations and non-uniform distributions. Hence, Catalog with the VA-File, by performing uniform, scalar quantization, is suboptimal. We proposed an Catalog method, based upon principles of vector quantization instead, where the data set is partitioned into Verona groups and groups are accessed in order of the query-group spaces. We developed group space bounds based on separating hyper plane boundaries and our search catalog, complemented by these bounds, is applicable to Euclidean and Mahalanobis space metrics. It obtained significant reductions in number of random IOs over several recently proposed cataloges, when allowed (roughly) the same number of sequential pages, has a low computational cost and scales well with dimensions and size of the data-set. We note that while the hyper plane bounds are better than MBR and MBS bounds, they are still loose compared with the true query-group space (see Figures 6 and 7). Conceivably, the group-space bounds can be further tightened, possibly by optimizing the grouping algorithm so as to optimize the group space bounds. Future efforts would be directed toward this and other related problems.

References

[1] S. Ramaswamy, K. Rose,"Adaptive Group Space Bounding for High Length Catalog", Transactions on Knowledge and data engineering, Vol. 23, pp. 815 - 830, 2011.

[2] A.R. Barambah, K. Arai,"Hierarchical K-means: an algorithm for centroids initialization for K-means", Reports of the Faculty of Science and Engineering, Saga University, Japan, Vol. 36, No. 1, 2007.

[3] Ali Ridho Barakbah, Yasushi Kiyoki,"A Pillar Algorithm for KMeans Optimization by Space Maximization for Initial Centroid Designation", Inf. Technol.Dept., Electron. Eng. Polytech. Inst. of Technol., Surabaya, pp. 61-68, 2009.

[4] J. Davis, B. Kulis, P. Jain, S. Sra, and I.Dhillon,(2007), "Information-Theoretic Metric Learning", Proc. Int'l Conf. Machine Learning (ICML), pp. 209-216.
[5] R. Weber, H. Schek, S. Blott,"A Quantitative Analysis and Performance Study for Similarity-Search Methods in High-Length Spaces", Proc. Int'l Conf. Very Large Data Bases (VLDB), pp. 194-205, 1998.
[6] S. Ramaswamy, K. Rose,"Adaptive Group-Space Bounding forSimilarity Search in Image Databases", Proc.Int'l Conf. Image Processing (ICIP), Vol. 6, pp. 381-384, 2007.
[7] K. S. Beyer, J. Goldstein, R. Ramakrishnan, U. Shaft, "When is "nearest neighbor" meaningful?", in ICDT, 1999, pp. 217–235.
[8] C. C. Aggarwal, A. Hinneburg, D. A. Keim,"On the surprisingbehavior of space metrics in high length spaces", in ICDT, 2001, pp. 420–434.
[9] B. U. Pagel, F. Korn, C. Faloutsos,"Deflating the lengthity curse using multiple fractal dimensions", in ICDE, 2000, pp. 589–598.
[10] T. Huang, X. S. Zhou,"Image retrieval with relevance feedback:From heuristic weight adjustment to optimal learning methods", in ICIP, Vol. 3, pp. 2–5, 2001.
[11] J. Davis, B. Kulis, P. Jain, S. Sra, I. Dhillon,"Information-theoretic metric learning", in ICML, 2007, pp. 209–216.
[12] R. Weber, H. Schek, S. Blott,"A quantitative analysis and performance study for similarity-search methods in high-length spaces", in VLDB, August 1998, pp. 194–205.
[13] A. Gersho, R. M. Gray,"Vector Quantization and Signal Compression.Kluwer Academic Publishers, 1992.
[14] T. Zhang, R. Ramakrishnan, M. Livny,"BIRCH: An efficient data grouping method for very large databases", in SIGMOD, 1996, pp. 103114.
[15] P. Ciaccia, M. Patella, P. Zezula,"M-tree: An efficient access method for similarity search in metric spaces", in VLDB, 1997, pp. 426–435.
[16] R. Bellman, Adaptive Control Processes: A Guided Tour. Princeton, NJ: Princeton University Press, 1961.
[17] H. Ferhatosmanoglu, E. Tuncel, D. Agrawal, A. E. Abbadi, "Vector approximation based Catalog for non-uniform high length data sets", in CIKM, 2000, pp. 202–209.
[18] K. Chakrabarti, S. Mehrotra,"Local lengthity reduction: A new approach to Catalog high length spaces", in VLDB, September 2000, pp. 89–100.
[19] K. Vu, K. Hua, H. Cheng, S. Lang,"A non-linear lengthityreduction technique for fast similarity search in large databases", in SIGMOD, 2006, pp. 527–538.
[20] S. Berchtold, C. Bohm, H. Kriegel,"The Pyramid-technique:Towards breaking the curse of lengthity", in SIGMOD, 1998, pp. 142–153.
[21] H. Jin, B. C. Ooi, H. T. Shen, C. Yu, A. Zhou,"An adaptive and efficient lengthity reduction algorithm for high-length Catalog", in ICDE, March 2003, pp. 87–98.
[22] P. Ciaccia, M. Patella,"PAC nearest neighbor queries: Approximate and controlled search in high-length and metric spaces", in ICDE, 2000, pp. 244–255.
[23] R. Weber, K. B¨ohm,"Trading quality for time with nearest neighbor search", in EDBT, 2000, pp. 21–35.
[24] E. Tuncel, H. Ferhatosmanoglu, K. Rose,"VQ-Catalog: An catalog structure for similarity searching in multimedia databases", in ACM Multimedia, 2002, pp. 543–552.
[25] E. Tuncel, K. Rose,"Towards optimal grouping for approximate similarity searching", in ICME, Vol. 2, August 2002, pp. 497–500.
[26] H. Ferhatosmanoglu, E. Tuncel, D. Agrawal, A. E. Abbadi,"Approximate nearest neighbor searching in multimedia databases", in ICDE, April 2001, pp. 503–511.
[27] A. Gionis, P. Indyk, R. Motwani,"Similarity search in high dimensions via hashing", in VLDB, September 1999, pp. 518–529.
[28] P. Ciaccia, M. Patella,"Approximate similarity queries: A survey", CSITE-08-01 Technical Report, May 2001.
[29] N. Katayama, S. Satoh,"The SR-tree: An catalog structure for highlength nearest neighbor queries", in SIGMOD, May 1997, pp. 369-380.
[30] D. A. White, R. Jain,"Similarity Catalog with the SS-tree", in ICDE, 1996, pp. 516–523.
[31] N. Beckmann, H. Kriegel, R. Schneider, B. Seeger, "The R*-tree:an efficient and robust access method for points and rectangles", in SIGMOD, 1990, pp. 322–331.
[32] Y. Sakurai, M. Yoshikawa, S. Uemura, H. Kojima,"The A-tree: An catalog structure for high-length spaces using relative approximation", in VLDB, September 2000, pp. 516–526.
[33] S. Berchtold, C. Bohm, H. V. Jagadish, H. P. Kriegel, J. Sander,"Independent Quantization: An catalog compression technique for highlength data spaces", in ICDE, 2000, pp. 577–588.
[34] C. Yu, B. C. Ooi, K. L. Tan, H. V. Jagadish,"Catalog the space:An efficient method to knn processing", in VLDB, September 2001, pp. 421–430.
[35] Y. Ishikawa, R. Subramanya, C. Faloutsos,"Mindreader: Querying databases through multiple examples", in VLDB, August 1998, pp. 218227.
[36] Y. Rui, T. Huang,"Optimizing learning in image retrieval", CVPR, Vol. 1, pp. 1236–1243, 2000.
[37] S. Ramaswamy, K. Rose,"Adaptive group-space bounding forsimilarity search in image databases", in ICIP, Vol. 6, 2007, pp. 381–384.
[38] N. Koudas, B. C. Ooi, H. T. Shen, A. K. H. Tung,"LDC: Enabling search by partial space in a hyper-length space", in ICDE, 2004, pp. 6–17.
[39] A. Guttman,"R-trees: A dynamic catalog structure for spatial searching", in SIGMOD, 1984, pp. 47–57.

Venkateswarlu Bondu received the Master's Degree in Computer Science and Systems Engineering from Andhra University College of Engineering, pursuing Ph.D in Computer Science in Andhra University. He is an Associate Professor in the Department of Compute Science in Avanthi Institute of Engineering and Technology. His research areas of interests are Software Engineering & Data Modeling.

Vinay Kumar Yalamanchili received the BTech degree inComputer Science from JNTUK, Jawarharlal Nehru Technological University. He pursuing his M.Tech Software Engineering from JNTU KAKINADA. His Research interest includeobjectorientedprogramming,cloud computing and iphone programming