Journal of Theoretical and Applied Information Technology
30th April 2012. Vol. 38 No.2
© 2005 - 2012 JATIT & LLS. All rights reserved·

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

PSNR BASED CLUSTERING AND INDEXING FOR FAST ACCESS

[1] J. SHANMUGA PRATHIPA, [2] T. R. LAKSHMINARAYANAN.

[1]PG Student, Department of CSE, SASTRA University, Thanjavur, India-613401

[2]Professors, School of Computing, SASTRA University, Thanjavur, India -613401

E-mail: [1] spshanmugajsp@gmail.com, [2] shanmugajsp@yahoo.com

ABSTRACT

High dimensional data is inherently more complex in clustering and searching. Searching the whole database without indexing leads to high I/O cost and expensive maintenance. A Peak Signal Noise Ratio (PSNR) based cluster is proposed for efficient access. This indexing method is scalable with data set amount and data dimensionality and outperforms several recently anticipated indexes. Vector approximation (VA-File**)** is used to overcome curse of dimensionality.

Keywords: *Multimedia Databases*, *Peak Signal Noise Ratio* (PSNR), *Indexing Methods*, *Similarity Measures Clustering, Image Databases*

1. INTRODUCTION

In huge amount of image dataset, retrieving an image without any indexing and clustering is cumbersome. Nearest neighbor search queries with the Euclidean distance metric; is not viable at high dimensions which are due to the notorious curse of dimensionality [3]. Conventionally it has been perform by means of bounding spheres and rectangles. However hyper spheres and hyper rectangles are generally not most favorable bounding surfaces for clusters in high-dimensional spaces [1]. Research activity in content-based image retrieval is towards the PSNR values.

Real datasets are demonstrably indexed and able with PSNR values. PSNR value depends upon RGB values. Basic RGB colors contains $(256)^3$ distinct colors. Mean Square Error (MSE) is a function in PSNR value to identify the identical images. If there is an identical images, MSE value is minimized to zero.

The distance is calculated using PSNR based k-means clustering and also the rating is provided for each image. Therefore this paper focus on real-data sets and compares the performances against the targeting real datasets.

2. MULTIMEDIA DATABASE

A multimedia database may contain complex texts, graphics, images, video fragments, map, voice, music and other forms of audio/video information's. Multimedia data's are typically stored as a sequence of bytes with variable lengths and segments of data which are linked together or indexed in multidimensional way for easy reference.

For similarity searching in multimedia data, two main families of multimedia indexing are (1) Description based retrieval systems (2) Content-based retrieval systems [2]. To implement efficient searching in large databases, various kinds of indexing methods have been explored. R-tree and R*-tree have been used to store minimal bounding rectangles so as to speed up similarity search.

In multimedia databases, the fundamental problem resides in the similarity search which is not measured on images directly, but it is based on a number of primitives (histograms of colors, signatures sound etc)

Data sets are predefined clusters and that clusters can be retrieved in a decreasing order of their probability.

3. EXISTING SYSTEM

Density Based Scan (DB) is a clustering algorithm which is based on metric called density. The general idea of DB Scan is defined as the continuous growing number of clusters as long as density (number of data values) in the neighborhood exceeds some threshold.

Pseudo code

```
DBSCAN (DS, ls, Minval)
C = 0
  For unvisited values v in dataset DS
  V as visited
  N = region Query (V, ls)
  If size of (N) < Minval
    Mark   V as outlier
    Else
      C = next cluster
 Expand Cluster (v, N, C, ls, Minval)
       Add   v to cluster C
     For each point v' in N
    If v' is not visited
      Mark v' as visited
      N' = region Query (V', ls)
      If size of (N') >= MinPts
        N = N joined with N'
    If V' is not yet member of any cluster
      Add v' to cluster C
```

This pseudocode is sensitive to user defined parameters. DB Scan effectively works up to minimum-dimensions, whenever dimensionality increases its performance degrades gradually. Indexing structure can be performed efficiently only on SS-tree shaped cluster.

For each point, DB Scan executes its query and neighborhood query in $O(\log n)$. DB Scan does not respond well to data dimensionality with varying densities. Euclidean distance measure is less effective, it is only beneficial for low dimensional data.

4. VECTORAPPROXIMATIONINDEXG

The conventional indexing method such as R-Tree, R*-Tree, SR-Tree, K-D-B-Tree and X-Tree are adequately used to solve the problems of K-NN search.

If the dimensionality of applications increases, then the performance is degenerated to worse. To overcome the 'curse of dimensionality' there is a popular and effective technique called vector approximation file (VA-File) [9]. In VA-File, the space is partitioned into hyper rectangular cells. The quantized approximation of data is present inside the group.

The vector approximation file is consecutively scrutinized and the superior and inferior bounds on the distance from the query vector to each cell are anticipated [8]. VA-File eliminates irrelevant cells. Vector node have cluster of neighboring vectors, each node includes the centroid of vector which increases the filtering rate.

4.1 Cluster Distance Bounding

Let us consider q as a query, H_p as a hyper plane and Z_m as a cluster :

$$d\ (q, Z_m) = \min d\ (Z, q)$$

$$d\ (z, q) \geq d\ (q, H_p) + d\ (z, H_p)$$

$$\min d\ (z, q) \geq d\ (q, H_p) + \min d\ (z, H_p)$$

$$d\ (q, z) \geq d\ (q, H_p) + d\ (Z_m, H_p)$$

Distance-to-cluster ≥Query-Hyper plane distance + Cluster-Hyper plane Distance.

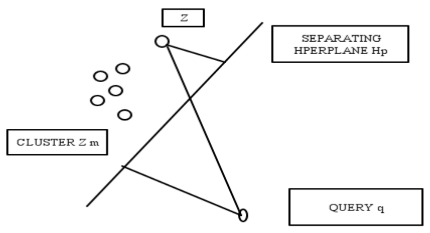


Figure 1: Distance calculating

5. PSNR BASED CLUSTER

Content-based retrieval uses an image features like color, texture, shapes which are matched with the feature vectors of the images in the databases. The images are matched based on color composition such as psnr values.

For each dataset, K-means cluster select the centroid and calculate the distance from the centroid to the query[4].

Modified K-means cluster gives answers which are very effective to the nearest query. In order to find the object, that is closest to a given query point, we find most similar object on a database by using PSNR value. The PSNR value is based on RGB colors.

Algorithm 1: PSNR BASED CLUSTER (x, k)

```
//Generic clustering algorithm returns

//K cluster centroids

//For Index

1: {vm} $^n$=1←Generic Index(x, n)
//For Cluster

2: {vm} $^k$=1←Generic Cluster(x, k)

3:.Set i =0, $x_1=\phi_1$, $x_2=\phi_2$, $x_3=\phi_3$,….$x_k=\phi_n$

//Retrieve the data based on PSNR value

4: $p_s$←RGB values ($x_1$…, $x_k$)

//check the elements in dataset
5: if i <|x| do

6: i = i+1 go to step 2
    Else
    i= empty
   End if
//Move xl to the corresponding K-means partition
7: v= {vm} $^n$ U {vm} $^k$

8: if (v = =$p_s$)
//$M_i$ matched images

9: Return ($M_i$ )

10: end if
```

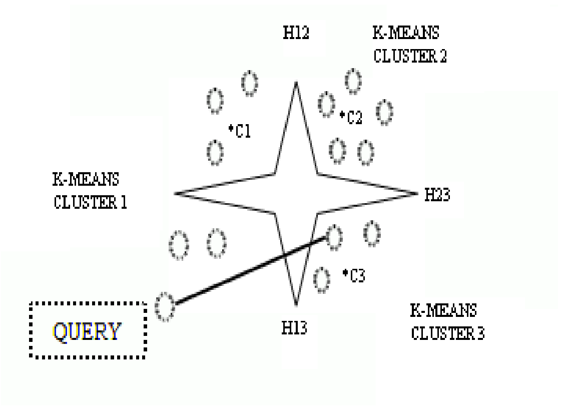


Fig 2: Hyper plane Boundaries of K-means Cluster

In fig2 each K-means cluster with its particular centriod is selected. The distance from centriod to the query is calculated. The condition for a hyper plane lies between cluster and query point. The centriod is a good choice as a pivot[1]. The entire data in the database can be scanned as single scan is achieved using K-means cluster

Definition 4.1: Let the data Z be partitioned into K-means cluster $\{Z_m\}$ pivoted around $\{c_m\}$.Then,

$$Z_m = \{z \in Z: d(z, c_m) \leq d(z, c_n), \forall_n \neq m\}$$

$$Z = \bigcup_m Z_m \quad \text{where } Z_m \cap Z_n = \emptyset, \forall_n \neq m$$

6. KNN SEARCH ALGORITHM

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[10].

K-NN is computationally traceable using an appropriate nearest neighbor search algorithm which is also used for large data sets[11].

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[5].

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.

IF k-NN distance so far < distance to next cluster

STOP (k-NNs found)

ELSE read next cluster (till all clusters read)

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
```

6.1 FLOWCHART OF ALGORITHM

The k-NN algorithm first read the file then calculates the upper and lower bounds to occupy the cluster. It prunes the irrelevant cluster which represents the upper bound and only the lower bound is selected.

Retrieval of the cluster is based on order of distance. If K-NN distance is greater it calculates the distance of the next cluster, otherwise it stops the condition and its process the next cluster until the condition is satisfied

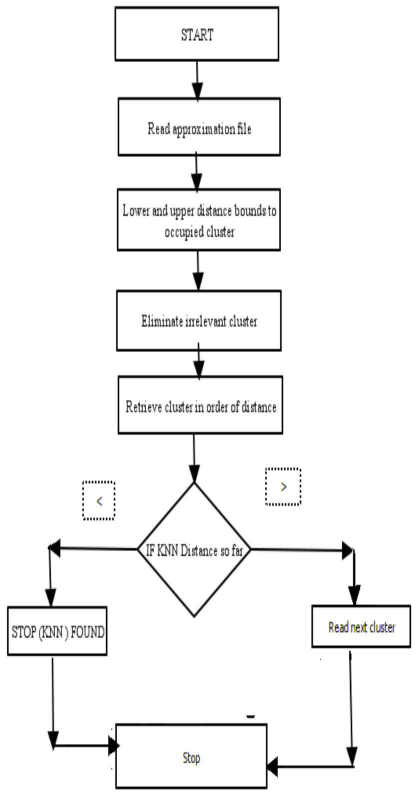

START
Read approximation file
Lower and upper distance bounds to occupied cluster
Eliminate irrelevant cluster
Retrieve cluster in order of distance
<
>
IF KNN Distance so far
STOP (KNN) FOUND
Read next cluster
Stop

Figure 3: K-NN FLOWCHART

7. RELATED WORK

7.1 SS-TREE

The SS-tree is an index structure. This tree is designed for indexing similarity of multidimensional point data. SS- Tree is an improvement over the R*- tree that enhances the performance of the nearest neighbor queries. This improvement is achieved by making modification as explained below. At first, for the region shape it uses bounding spheres instead of bounding rectangle is depicted in figure 4.

The center of the sphere is the centroid of underlying points. The SS-tree uses centroids to divide points into isotropic neighborhoods in the tree consumption algorithms i.e. the insertion and the split algorithm.

When a point is inserted, the insertion algorithm derives the most suitable sub tree for accommodating the new point where the centroid of the sub tree is the tile nearest to the new entry.

In case where a node or leaf is full, the split algorithm calculates completely the variance of its coordinates on each dimension from the centroids of its children. The dimension with highest variance is selected for the purpose of splitting it. The points are divided into isotropic neighborhoods.

By these algorithms the performance of the tile on the nearest neighbor queries are enhanced. The advantage of using bounding spheres is the storage space allocation .It requires nearly half of the storage space which is compared with the bounding rectangles.

Since a sphere determines the center and radius,the same may be represented with as many parameters as tile dimensionality plus one. But where as to determine a rectangle, the number of parameters required is tile double of the dimensionality, this is because a rectangle is determined by lower and upper bound of every dimension. This advantage allows almost doubling the fan out of nodes, and the height of the trees are reduced. The second advantage of the SS-tree is that it modifies the forced reinsertion mechanism of the R*-tree [6]. In case a node or a leaf is full, R*-tree is used for splitting. A portion of its entries are reinserted and the splitting is done only when the reinsertion is made on the same tree level. On the other hand, the SS-tree reinserts the entries. Unless the reinsertion has been made at the same node, SS-tree promotes the dynamic reorganization of the tree structure.

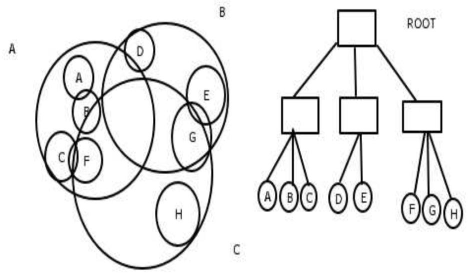


Figure 4: The SS-tree structure

7.2 SR-TREE

Through the insertion of the bounding sphere and bounding rectangle, the region of the SR-tree is specified. The neighborhoods are partitioned into smaller regions instead of incorporating bounding rectangle using SS-tree [7]. So this improves the disjointness among region. This enhances the performance on nearest neighbor queries especially for high dimensional and non-uniform data which can be practical in actual image/video similarity indexing. The SR-tree is unique because a region is specified by the intersection of the bounding sphere and bounding rectangle of the underlying point.

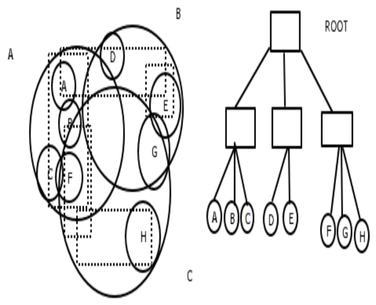


Figure 5: The SR-tree structure

The insertion algorithm of the SR-tree and SS-tree are different due to updating regions when a new entry is inserted. In the case of SR-tree mutually the bounding spheres and the bounding rectangles are required to be updated and updating bounding sphere is alone needed in the case of SS-tree.

The manner of updating bounding rectangle of the R-tree and R*-tree are same. But, the manner of updating bounding spheres is different from that of the SS-tree. This is because a region of the SR-tree is the meeting point of a bounding rectangle and a bounding sphere. The bounding sphere of a father node is determined by SR-tree by using both the bounding spheres and bounding rectangle of its kid's nodes. The deletion of an entry causes no utilization of any leaf node. The entry is simply removed from the tree. Otherwise the under-utilized leaf or node is removed from the tree and then orphaned entries are reinserted to tile tree.

8. EXPERIMENTAL RESULT

Real-data sets are collected and the extensive test is conducted, for analysis the performance of indexing and clustering are measured.

8.1 PERFORMANCE MEASURES

Test is conducted for the image datasets for examining the cluster and indexing. Datasets ranges from 1 to1000.Each group of dataset consist of 100 images. The performance is measured based on how effectively the similarity content are retrieved based on the search content and also it measures how reduction occurs in the dissimilarity of retrieved image in high-dimensional spaces .In previous method is used only for low-dimensionality.

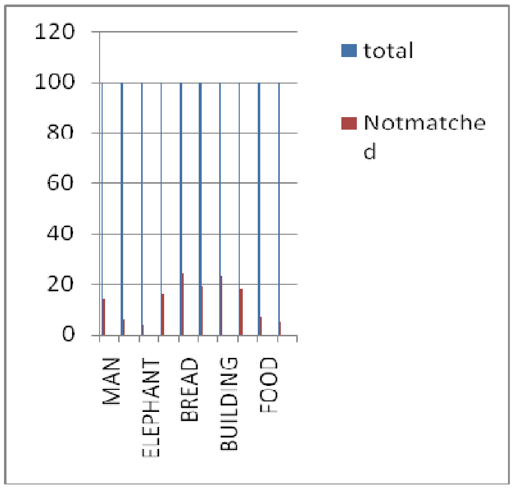


Figure 6: Not matched

Data present in the dataset are not matched with the search content i.e. dissimilarity image retrieved is reduced by using psnr based clustering algorithm. It is represented in the above figure6.

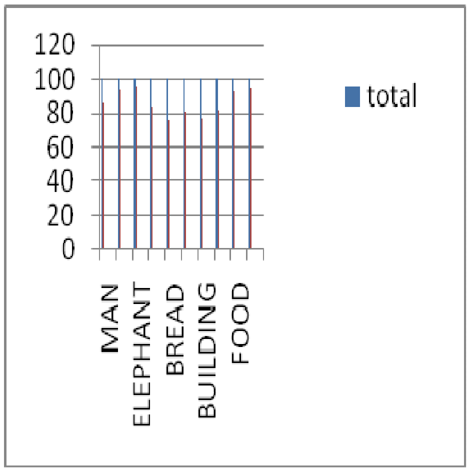


Figure 7: Matched content

Matching of data present in the dataset with the searched content is represented in the figure7

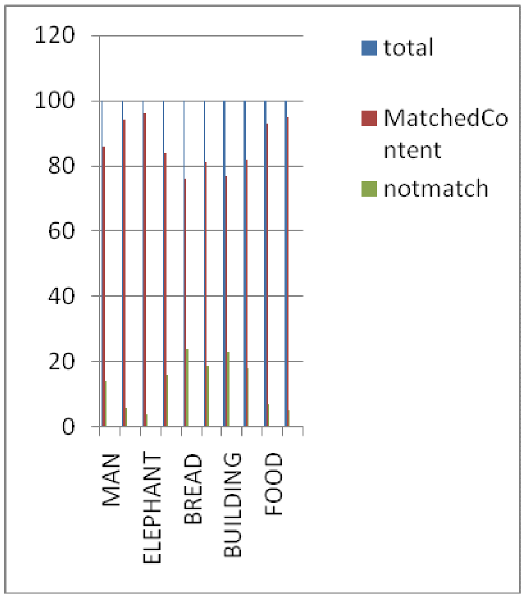


Figure 8: Not Matched vs. Matched content

The combinations both matched and not matched searched content are represent in the figure8

Our clustering and indexing reduce the complexity of searching the content in high dimensional space which is proved in the above figures. Datasets are downloaded from http:/www.cs.cmu.edu/datasets.

9. CONCLUSION

Scalar quantization indexing provides only suboptimal solution. But proposed indexing method based on vector quantization provides effectively better solution than the indexing based on scalar quantization. Psnr algorithm clusters the data more efficiently based on color, so the correspondence image retrieval is high-speed.

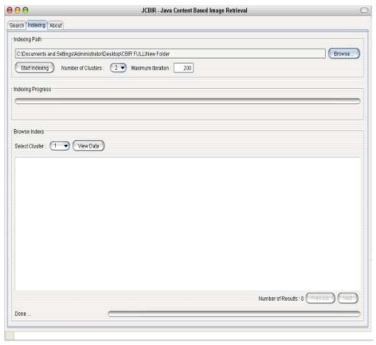
The clusters are accessed based on query-cluster distance; proposed indexing has a low computational I/O cost and scalable with high dimensional data set. Proposed hyper plane bound is tight and provides efficient clustering and indexing. Huge gains in IO complexity are possible over VA-File.

10. RESULT

The figure 9 denotes indexing the data. This process is achieved by specifying the number of clusters and iterations .Then select the data and click indexing process, the image dataset are indexed based on vector approximation.

Journal of Theoretical and Applied Information Technology
30th April 2012. Vol. 38 No.2

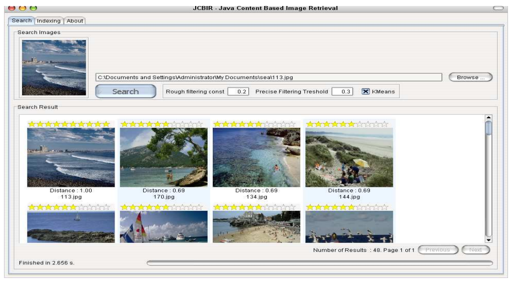
ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195



Indexing Path: C:\Documents and Settings\Administrator\Desktop\CBIR FULL\New Folder
Number of Clusters : 2
Maximum Iteration : 200
Select Cluster : 1
Number of Results : 0
Done ...

Figure 9: Indexing the data

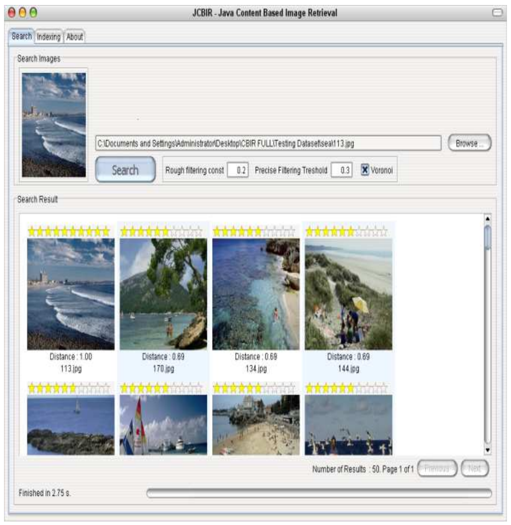
The figure 10 specifies the searching the data. In that the search content is specified as input. The searching method achieved by using K-NN searching. It retrieves the content based on psnr values. The psnr value calculates the average value of RGB color and then it starts the searching process.



C:\Documents and Settings\Administrator\My Documents\sea\113.jpg
Rough filtering const 0.2
Precise Filtering Treshold 0.3
[✓ KMeans]
Distance : 1.00 113.jpg
Distance : 0.69 170.jpg
Distance : 0.69 134.jpg
Distance : 0.69 144.jpg
Number of Results : 48. Page 1 of 1
Finished in 2.656 s.

Figure 10: Searching the data

The figure 11 shows the retrieved content. The input of searched content is retrieved based on color composition so it successful access the similarities images. That is shown in below figure.



C:\Documents and Settings\Administrator\Desktop\CBIR FULL\Testing Dataset\sea\113.jpg
Rough filtering const 0.2
Precise Filtering Treshold 0.3
[✓ Voronoi]
Distance : 1.00 113.jpg
Distance : 0.69 170.jpg
Distance : 0.69 134.jpg
Distance : 0.69 144.jpg
Number of Results : 50. Page 1 of 1
Finished in 2.75 s.

Figure 11: Retrieved content

REFRENCES:

[1] Adaptive Cluster Distance Bounding for High-Dimensional Indexing-Sharadh Ramaswamy, Student Member, IEEE, and Kenneth Rose.2011

[2] C.Faloutsos, Searching in Multimedia Databases by Content. Kluwer Academic Press, 1996.

[3] K.S. Beyer, J. Goldstein, R. Ramakrishnan, and U. Shaft, “When is “Nearest Neighbor” Meaningful,” Proc. Int’l Conf. Database Theory(ICDT), pp.217235, 1999.

[4] Daniela Stan & Ishwar K. Sethi, Image Retrieval Using a Hierarchy of Clusters.

[5] P. Indyk and R. Motwani, “Approximate nearest neighbors: Toward removing the curse of dimensionality”, in Proc. ACM symp. Theory of Computing, 1998

[6] N. Beckmann, H.-P. Kriegel , R. Schneider, and B. Seeger. The R*-tree: An efficient and robust access method for points and rectangles.*ACM SIGMOD*, pages 322–331,May 23-25 1990.

[7] A. Guttman. R-trees: A dynamic index structure for spatial searching .In Proc. of the ACM SIGMOD Int. Conf. on Management of Data, pages 47-57, Boston, MA, June 1984.

[8] D. R. Heisterkamp and J. Peng, “Kernel Vector Approximation Files for Relevance Feedback Retrieval in Large Image Databases, “Multimedia Tools and Applications, vol 25, pp. 175-189, June, 2005.

[9] Hung-Yi, Lin and P. Huang, "Perfect KDB-Tree Structure for Indexing Multidimensional Data", hired International Conference on Information technology and applications (ICITA 2005), 4-7 July 2005, Sydney, Australia.

[10] 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, pp. 573-582.

[11] Arya S.: ‘Nearest Neighbor Searching and Applications’, Ph.D. thesis, University of Maryland, 1995.