ADAPTIVE CLUSTER-DISTANCE BOUNDING FOR NEAREST NEIGHBOR SEARCH IN IMAGE DATABASES

Sharadh Ramaswamy and Kenneth Rose

Signal Compression Lab
Dept. of Electrical and Computer Engineering
University of California, Santa Barbara
CA 93106 - 9560
{rsharadh,rose}@ece.ucsb.edu

ABSTRACT

We consider approaches for exact similarity search in a high dimensional space of correlated features representing image datasets, based on principles of clustering and vector quantization. We develop an adaptive cluster distance bound based on separating hyperplanes, that complements our index in selectively retrieving clusters that contain data entries closest to the query. Experiments conducted on real data-sets confirm the efficiency of our approach with random disk IOs reduced by 100X, as compared with the popular Vector Approximation-File (VA-File) approach, when allowed (roughly) the same number of sequential disk accesses, with relatively low pre-processing storage and computational costs.

***Index Terms*—** Similarity search, multi-dimensional indexing, retrieval, vector quantization, clustering

1. INTRODUCTION

With the proliferation of digital multi-media devices, such as digital cameras and video recorders, there has been an explosive growth in multi-media data and new applications that handle these data, such as image search engines, bio-medical imaging etc., hence necessitating efficient storage and data mining solutions. Searching and indexing image 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. Clearly, content-based image search and retrieval (CBIR) would be the appropriate paradigm.

Images are represented by feature vectors and the measure of similarity between two images is assumed to be proportional to the distance between their feature vectors. Recently, a combination of texture features (extracted through Gabor filters) and color features (histograms) have been found to be efficient descriptors of the underlying images and form a part of the MPEG-7 multimedia standard (see [1]). Such feature vectors themselves are typically high-dimensional, such as the 60 dimensional texture descriptors [1] or the 256 dimensional color histograms of QBIC [2].

Similarity search is the search for elements in the database most similar to the query image. A popular query model is the k-nearest neighbor (kNN) query, where given a query image, the k most similar images 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 [3] 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 [4].

The time incurred in nearest neighbor search is largely dominated by IO 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 IOs 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 hypervolume with dimensionality (”the curse of dimensionality” [5]), 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

This work was supported in part by NSF, under grant IIS-0329267, the University of California MICRO program, Applied Signal Technology Inc., Dolby Laboratories Inc., and Qualcomm Inc.

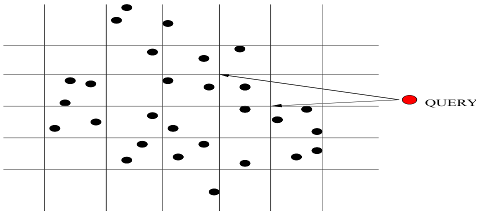

Fig. 1. VA-File: SCALAR QUANTIZATION

than the simple sequential scan.

A very popular and effective technique employed to overcome the curse of dimensionality is the Vector Approximation File (VA-File) [4]. 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. 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* (in contradistinction with vector quantization in the signal compression literature).

In this paper, we consider 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., K-means or Lloyd's algorithm and during query processing, load the "nearest" clusters into the main memory. We motivate such a solution since vector quantization, unlike the scalar quantization of the VA-File, can exploit dependencies across dimensions and hence, would be a more compact representation of the database. We propose 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.

While such a vector quantization and clustering approach to search has been studied in the image database community (see [6, 7, 8]), the earlier approaches have focussed more on *approximate* nearest neighbor search. The distance bounds (based on bounding hyperspheres) derived in [7] are loose and hence the search strategy performs poorly when adapted to exact nearest neighbor search. We next present an effective cluster distance bound that complements our branch-and-bound search algorithm.

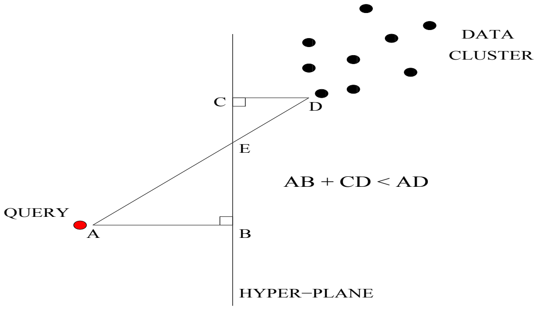

Fig. 2. The Hyperplane Bound

2. THE HYPERPLANE BOUND

Let $d(\mathbf{x}, \mathbf{y})$ be a distance function that estimates the distance between vectors **x** and **y** in the feature space.

$$d : \mathcal{R}^n \times \mathcal{R}^n \to [0, \infty) \tag{1}$$

In subsequent discussion, we shall specialize to the Euclidean distance over (real vector spaces) as the feature similarity measure i.e. $d(\mathbf{x}, \mathbf{y}) = \|\mathbf{x} - \mathbf{y}\|_2$. We define the distance from query **q** and a cluster $\mathcal{X}_m$ as

$$d(\mathbf{q}, \mathcal{X}_m) = \min_{\mathbf{x} \in \mathcal{X}_m} d(\mathbf{q}, \mathbf{x}) \tag{2}$$

The distance of vector **q** to a hyper plane $H(\mathbf{n}, p) = \{\mathbf{y} : \mathbf{y}^T\mathbf{n} + p = 0\}$ is defined in the normal fashion as

$$d(\mathbf{q}, H) = \frac{|\mathbf{q}^T\mathbf{n} + p|}{\|\mathbf{n}\|_2} \tag{3}$$

Given a cluster $\mathcal{X}_m$, the query **q** and a hyperplane H that lies between the cluster and the query (a "*separating hyperplane*", see Figure 2), by simple geometry it is easy to see that for any $\mathbf{x} \in \mathcal{X}_m$

$$\begin{aligned} d(\mathbf{q}, \mathbf{x}) &\geq d(\mathbf{q}, H) + d(\mathbf{x}, H) \\ &\geq d(\mathbf{q}, H) + \min_{\mathbf{x} \in \mathcal{X}_m} d(\mathbf{x}, H) \\ &= d(\mathbf{q}, H) + d(\mathcal{X}_m, H) \\ \Rightarrow d(\mathbf{q}, \mathcal{X}_m) &\geq d(\mathbf{q}, H) + d(\mathcal{X}_m, H) \end{aligned} \tag{4}$$

If $\mathcal{H}_{sep}$ represents a countably finite set of separating hyperplanes (that lie-between the query **q** and the cluster $\mathcal{X}_m$),

$$\Rightarrow d(\mathbf{q}, \mathcal{X}_m) \geq \max_{H \in \mathcal{H}_{sep}} \{d(\mathbf{q}, H) + d(\mathcal{X}_m, H)\} \tag{5}$$

The second lower bound presented in (5) can be used to tighten the lower bound on $d(\mathbf{q}, \mathcal{X}_m)$. Next, we note that the *boundaries* between clusters generated by the K-means algorithm are *linear hyperplanes*. If $\mathbf{c}_1$ and $\mathbf{c}_2$ are centroids of two clusters $\mathcal{X}_1$ and $\mathcal{X}_2$, and $\mathcal{Y}_{12}$ the boundary between them, then $\forall \mathbf{y} \in \mathcal{Y}_{12}$

$$\begin{aligned} d(\mathbf{c}_1, \mathbf{y}) &= d(\mathbf{c}_2, \mathbf{y}) \\ \Rightarrow \|\mathbf{c}_1\|_2^2 - \|\mathbf{c}_2\|_2^2 - 2(\mathbf{c}_1 - \mathbf{c}_2)^T\mathbf{y} &= 0 \end{aligned}$$

Therefore, the hyperplane $H_{12} = H(-2(\mathbf{c}_1 - \mathbf{c}_2), \|\mathbf{c}_1\|_2^2 - \|\mathbf{c}_2\|_2^2)$ is the boundary between the clusters $\mathcal{X}_1$ and $\mathcal{X}_2$. What is to be noted is that these hyperplane boundaries **need not be stored**, rather they can be **generated during run-time from** just **the centroids** $\{\mathbf{c}_m\}_1^M$ themselves. It is straightforward to show that: Given a query $\mathbf{q}$ and a hyperplane H_{mn} that separates clusters $\mathcal{X}_m$ and $\mathcal{X}_n$, it lies between the query and cluster $\mathbf{X}_m$ ***if and only if*** $d(\mathbf{q}, \mathbf{c}_m) \geq d(\mathbf{q}, \mathbf{c}_n)$.

2.1. Reduced Complexity Hyperplane Bound

For evaluation of the lower-bound presented in (4) and (5), we would need to pre-calculate and store $d(H_{mn}, \mathcal{X}_m)$ for all cluster pairs (m, n). With M clusters, there are $M(M-1)$ distances that need to be pre-calculated and stored, in addition to the cluster centroids themselves. The total storage for all clusters would be $O(M^2 + Md)$, where d is the dimensionality. This heavy storage overhead makes the hyperplane bound, in this form, impractical for a large number of clusters. We can loosen the bound in (5) as follows:

$$\begin{aligned} d(\mathbf{q}, \mathcal{X}_m) &\geq \max_{H \in \mathcal{H}_{sep}} \{d(\mathbf{q}, H) + d(H, \mathcal{X}_m)\} \\ &\geq \max_{H \in \mathcal{H}_{sep}} d(\mathbf{q}, H) + \min_{H \in \mathcal{H}_{sep}} d(H, \mathcal{X}_m) \end{aligned}$$

This means that for every cluster $\mathcal{X}_m$ we would only need to store one distance term

$$d_m = \min_{1 \leq n \leq M, n \neq m} d(H_{mn}, \mathcal{X}_m)$$

3. EXPERIMENTAL RESULTS

We compared the performance of our index (henceforth referred to as 'VQ-Hyperplane') with that of the VA-File and clustering based search technique presented in [7] (henceforth referred to as 'VQ-Sphere') on two real image data-sets. Our feature vectors were MPEG-7 texture feature descriptors extracted from 64 × 64 blocks of the images. The first data-set AERIAL was extracted from 40 large aerial photographs [1], is 60-dimensional and consists of 275,465 vectors. The second data-set BIO-RETINA [2] was generated from images of tissue sections of feline retinas as a part of an ongoing project at the Center for Bio-Image Informatics, UCSB. It is 208,506 elements long and 62 dimensional. We also assumed a *page size* of 8kB. The query sets themselves were generated by randomly selecting 1000 elements from the relevant data-sets. For each query, the 10 nearest neighbors (10NN) were mined.

3.1. IO Performance Comparison

We evaluated the performance of the VA-File at different quantization levels (3-8 bits per dimension) and the VQ methods for different numbers of clusters (10-600 clusters). We note that our index 'VQ-Hyperplane' is able to consistently reduce the number of random IO reads as compared with the VA-File and the 'VQ-Sphere', when allowed (roughly) the same number of sequential disk accesses. For BIO-RETINA (Figure 4), at 3 bit quantization for the VA-File, a nearly $3000X$ reduction in random disk accesses is possible with the vector quantization/clustering approach with 60 clusters. A nearly 100X reduction in IO reads is possible over the VA-File at 5-bits per dimension quantization (30 clusters in our method). We notice similar large reductions of $\approx 100X$ in random IO reads for the AERIAL data-set (Figure 3), at 5-bit per dimension quantization for the VA-File. We also note that the 'VQ-Sphere' method [7] *underperforms* the VA-File on the AERIAL dataset and outperforms the VA-File on the BIO-RETINA dataset.

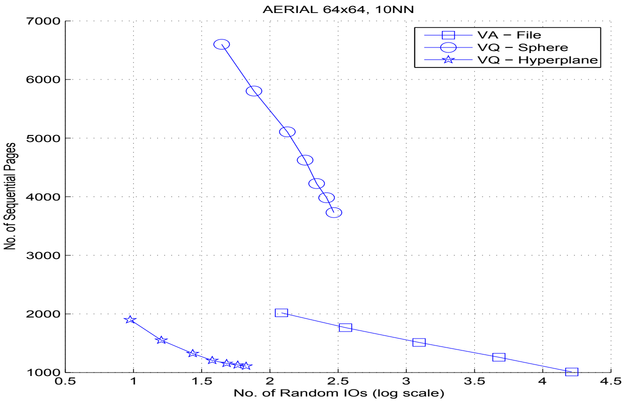

Fig. 3. IO Performance - AERIAL

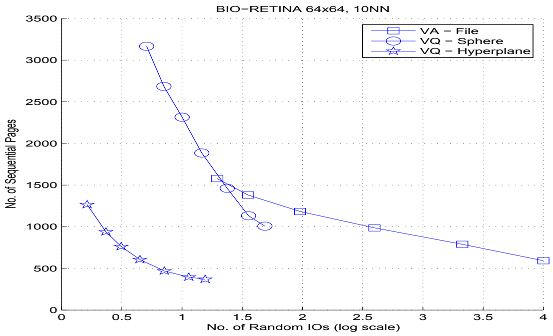

Fig. 4. IO Performance - BIO-RETINA

3.2. Computations and Pre-processing Storage Cost

We also compared the pre-processing storage and average computation costs (distance evaluations) of the different methods (Figures 5 and 6). Since the VA-File maintains a separate compressed representation for each element of the database, the approximation file size grows with the size of the database.

[1] Available for download from http://vision.ece.ucsb.edu/datasets

[2] Available for download from http://scl.ece.ucsb.edu/datasets/features.txt

Secondly, in order to reduce the costly random access reads, the quantization resolution in each dimension needs to be increased, which again results in larger approximation files. However,, in the VQ methods, random IO reads are reduced by *reducing* the number of clusters. Hence, we note that the VQ methods have significantly ($\approx 10X - 100X$) lower storage. And between the two VQ-methods, our index 'VQ-Hyperplane' generates lower number of random IOs given the same pre-processing storage.

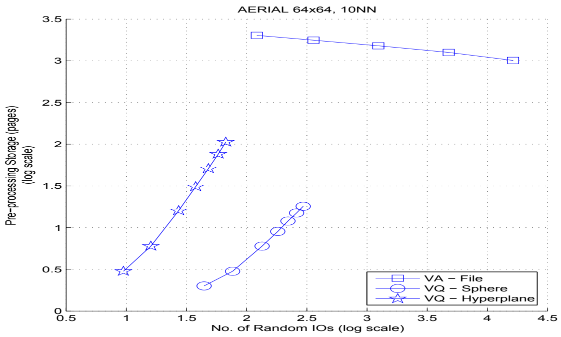

Fig. 5. Pre-processing Storage Cost - AERIAL

For each element of the data set the VA-File needs to compute an upper and a lower bound, hence the computational costs double with the data-set size. Since vector quantization exploits correlation across dimensions, it is a much more compact representation of the database. Additionally, as seen before, the filtering in our VQ method is tighter. Hence less number of vectors need to be tested, leading to ($\approx 10X$) lower number of distance evaluations, given the same number of random disk accesses. We also note that our method has lower computational cost as compared with the VQ-Sphere method.

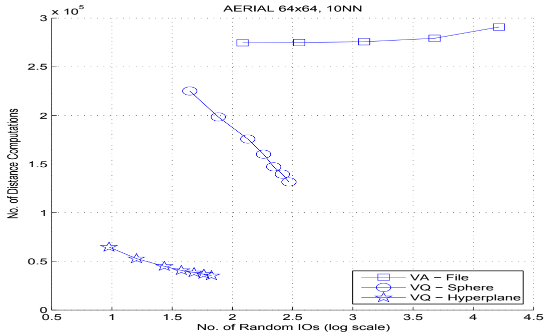

Fig. 6. Computational Costs - AERIAL

4. CONCLUSIONS

We proposed an image database indexing technique for exact nearest neighbor search, based on the principles of clustering and vector quantization. The image feature vectors are clustered and during query processing, the nearest clusters are visited in order. We developed an adaptive cluster distance bound, based on separating hyperplanes, that complements our branch-and-bound search. Our index has low storage and computation costs and is able to provide significant reduction in random disk accesses over known methods.

5. ACKNOWLEDGEMENTS

The authors would like to thank Dr. Jayant Nayak, Dr. Navneet Panda, Vinay Melkote and Ankur Saxena for their valuable suggestions. The authors are extremely grateful to Balakrishnan Selvam for his help in performing the experiments.

6. REFERENCES

[1] B.S. Manjunath, J.-R. Ohm, V.V. Vasudevan, and A. Yamada, "Color and texture descriptors.," *IEEE Trans. on Circuits and Systems for Video Technology*, vol. 11, no. 6, pp. 703–715, June 2001.

[2] M. Flickner, H. Sawhney, W. Niblack, J. Ashley, Q. Huang, B. Dom, M. Gorkani, J. Hafner, D. Lee, D. Petkovic, D. Steele, and P. Yanker, "Query by image and video content: The QBIC system," *Computer*, vol. 28, no. 9, pp. 23–32, 1995.

[3] A. Guttman, "R-trees: A dynamic index structure for spatial searching.," in *SIGMOD Conference*, 1984, pp. 47–57.

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

[5] R.E. Bellman, *Adaptive Control Processes: A Guided Tour*, Princeton University Press, Princeton, NJ, 1961.

[6] J.Y. Chen, C.A. Bouman, and J.C. Dalton, "Hierarchical browsing and search of large image databases.," *IEEE Transactions on Image Processing*, vol. 9, no. 3, pp. 442–455, March 2000.

[7] J.Y. Chen, C.A. Bouman, and J.P. Allebach, "Fast image database search using tree-structured VQ.," in *Proc. of IEEE International Conference on Image Processing*, 1997, vol. 2, pp. 827–8305.

[8] V. Castelli, A. Thomasian, and C.S. Li, "CSVD: Clustering and singular value decomposition for approximate similarity search in high-dimensional spaces.," *IEEE Transactions on Knowledge and Data Engineering*, vol. 15, no. 3, pp. 671–685, 2003.