An Adaptive Index Structure for High-Dimensional Similarity Search

P. Wu, B. S. Manjunath and S. Chandrasekaran
Department of Electrical and Computer Engineering
University of California, Santa Barbara, CA 93106-9560
{peng, manj, shiv}@ece.ucsb.edu

Abstract

A practical method for creating a high dimensional index structure that adapts to the data distribution and scales well with the database size, is presented. Typical media descriptors, such as texture features, are high dimensional and are not uniformly distributed in the feature space. The performance of many existing methods degrade if the data is not uniformly distributed. The proposed method offers an efficient solution to this problem. First, the data's marginal distribution along each dimension is characterized using a Gaussian mixture model. The parameters of this model are estimated using the well known Expectation-Maximization (EM) method. These model parameters can also be estimated sequentially for on-line updating. Using the marginal distribution information, each of the data dimensions can be partitioned such that each bin contains approximately an equal number of objects. Experimental results on a real image texture data set are presented. Comparisons with existing techniques, such as the well known VA-File, demonstrate a significant overall improvement.

1 Introduction

Typical audio-visual descriptors are high dimensional vectors and not uniformly distributed [6]. These descriptors are useful in content based image/video retrieval, data mining and knowledge discovery. To index high dimensional feature vectors, various index structures such as R*-tree, X-tree, TV-tree, etc., have been proposed. A good overview and analysis of these techniques can be found from [5]. The study in [5] also argued that typical tree based index methods are outperformed by a linear search when the search dimensions exceed 10. This has motivated the introduction of approximation methods [2, 5] to speed up the linear search.

Approximation based methods have certain advantages. First, they support different distance measures. This is an important property especially for learning and concept mining related applications. Secondly, the construction of approximation can be made adaptive to the dimensionality of data. However, the approximation method [5] is sensitive to data's distribution. does not perform well when feature vectors are not uniformly distributed.

In [5], the approximation is constructed by partitioning the feature space into hyper rectangles. The grids on each dimension are equally spaced. We refer to such uniform partitioning as "regular approximation" in the following discussion. In [2], the feature space is first transformed using the KL-transform to reduce the correlation of data at different dimensions. Secondly, in the transformed space, the data in each dimension is clustered into a pre-assigned number of grids using the Lloyd's algorithm. But in [3], it is reported that for high dimensional data, transforming the data space by rotation does not result in a significant decorrelation of the data. This implies that global statistics, such as the second order statistics used in [2], may not be able to characterize the data distribution effectively for high dimensional spaces.

In this work, we propose an effective and practical solution to adapt the design of the approximation based index structure to the data's distribution. The main idea of the pro-

posed method is to model the marginal distribution of the data using a mixture of Gaussians and use the estimated parameters of the model to partition the data space. By adapting the construction of approximations to data's marginal distribution, the proposed method overcomes the sensitivity of index performance to data's distribution, thus resulting in a significant improvement compared with regular the VA-File [2, 5].

In the next section, we summarize the construction of regular approximation and its associated indexing. We also discuss the limitations of using regular approximation. Our proposed method is presented in Section 3. Experimental results are provided in Section 4.

2 Regular Approximation

2.1 Construction of regular approximation [5]

In regular approximation methods, such as the VA-File approach [5], the range of the feature vectors in each dimension is uniformly partitioned. Let D denote the total number of dimensions of data space, and let B_i bits ($i = 1, \ldots, D$) be allocated to each of the dimensions. Then the range of feature values on dimension i is segmented to 2^{B_i} partitions by a set of boundary points, denoted as $c^i[k]$ ($k = 0, \ldots, 2^{B_i}$) with equal length, each partition is uniquely identified by a binary string of length B_i. The high dimensional space is in turn segmented into D dimensional hyper cells. Each of them can be uniquely identified by a binary string of length B ($B = \sum B_i$). For a feature vector $v[i][j]$, $j = 0, \ldots, N$, wherein N is the total number of object in a database, its approximation is such a binary string of length B to indicate which hyper cell it is contained in. If $v[i][j]$ falls into a partition bounded by $[c^i[k-1], c^i[k]]$, it satisfies

$$c^i[k-1] \le v[i][j] < c^i[k] \quad (1)$$

So the boundary points provide an approximation of the value of $v[i][j]$. As in [5], a lower bound and a upper bound of the distance between any vector with a query vector can be computed using this property.

Figure 1 gives an illustrative example of constructing the regular approximation for two dimension data. The 1856 image objects are collected from the Brodatz album [4]. Figure 1 shows the first two components of the 60 dimensional feature vectors computed in [4]. The feature distribution is clearly not uniform.

2.2 Indexing based on regular approximation

Approximation based nearest neighbor search can be considered as a two phase filtering process. In the first phase, the set of all approximations is scanned sequentially and lower and upper bounds on the distances of each object in the database to the query object are computed. In this phase, if an approximation is encountered such that its lower bound is larger than the k-th smallest upper bound found so far, the corresponding feature vector can be skipped since at least k better candidates exist. At the end of the first phase filtering, the set of vectors that are not skipped are collected as candidates for the second phase filtering. Denote the number of candidates to be N_1.

In the second phase filtering, the actual N_1 feature vectors are examined. The feature vectors are visited in increasing order of their lower bounds and then exact distances to the query vector are computed. If a lower bound is reached that is larger than the k-th actual nearest neighbor distance encountered so far, there is no need to visit the remaining candidates. Let N_2 denote the number of feature vectors visited before the thresholding lower bound is encountered.

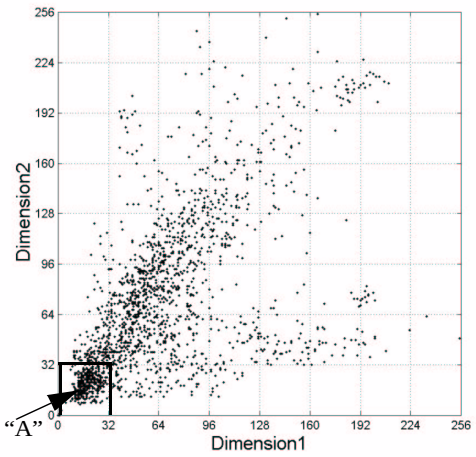


Fig 1. Using regular approximation to uniformly partition a two dimensional space. $N = 1856$, $B_1 = B_2 = 3$.

For approximation based methods, the index performance is measured by N_1 and N_2 (see [5]). Smaller values of N_1 and N_2 indicate a better performance.

2.3 Limitations

The effectiveness of using VA-File based indexing structure is sensitive to data's distribution. For the example given in Figure 1, the data on each dimension is not uniformly distributed. As a result, if a query vector happens to be one that falls into the cell "A" in Figure 1, in which approximately 30% of elements are contained, we will have $N_1 > 0.3N$ and $N_2 > 0.3N$, which indicates a poor block selectivity and a high I/O cost, see [5].

This example illustrates one of the shortcomings of the regular approximation. In such cases, the first phase filtering still results in a large number of items to search. Our proposed method specifically addresses this problem.

3 Approximation Based on Marginal Distribution

Densely populated cells can potentially degrade the indexing performance. For this reason, we propose an approach to adaptively construct the approximation of feature vectors. The general idea is to first estimate data's marginal distribution in each dimension. Secondly, individual axes are partitioned such that the data has equal probability of falling into any partition. The approximation such constructed reduces the possibility of having densely populated cells, which in turn improves the indexing performance.

3.1 pdf modeling using mixture of Gaussians

Denote $p_i(x)$ to be the pdf of data on dimension i. The algorithm introduced below is applied to data in each dimension independently. For notation simplicity, we denote $p(x)$ to be the pdf of an one dimensional signal. The one-dimensional pdf is modeled using a mixture of Gaussians, represented as

$$p(x) = \sum_{j=1}^{M} p(x|j)P(j) \tag{2}$$

where

$$p(x|j) = \frac{1}{\sqrt{2\pi\sigma_j^2}} \exp\left\{-\frac{(x-\mu_j)^2}{2\sigma_j^2}\right\} \tag{3}$$

The coefficients $P(j)$ are called the mixing parameters, which satisfies $\sum_{j=1,\ldots,M} P(j) = 1$ and $0 \le P(j) \le 1$.

The task of estimating the pdf is then converted to be a problem of parameter estimation. The parameters we need to estimate are $\phi_j = \{P(j), \mu_j, \sigma_j^2\}$, for $j = 1, \ldots, M$.

3.2 Parameter estimation using the EM algorithm

The classical maximum likelihood (ML) approach is used to estimate the parameters. The task is to find ϕ_j, $j = 1, \ldots, M$, to maximize

$$\Phi(\phi_1, \ldots, \phi_M) = \prod_{l=1}^{N} p(v[l] | (\phi_1, \ldots, \phi_M)) \tag{4}$$

where $v[l]$, $l = 1, \ldots, N$, are the given data set.

A simple and practical method of solving this optimization problem is to use the Expectation-Maximization algorithm [1]. Given N data $v[l]$ available as the input for the estimation, EM algorithm estimates the parameters iteratively using all the N data in each iteration. Let t denote the iteration number. Then the following equations are used to update the parameters

$$\mu_j^{t+1} = \left(\sum_{l=1}^{N} p(j|v[l])^t v[l] \right) \left(\sum_{l=1}^{N} p(j|v[l])^t \right)^{-1} \tag{5}$$

$$(\sigma_j^2)^{t+1} = \left(\sum_{l=1}^{N} p(j|v[l])^t (v[l] - \mu_j^t)^2 \right) \left(\sum_{l=1}^{N} p(j|v[l])^t \right)^{-1} \tag{6}$$

$$P(j)^{t+1} = \frac{1}{N} \sum_{l=1}^{N} p(j|v[l])^t \tag{7}$$

Using Bayes' theorem [1], the $p(j|v[l])^t$ is computed as

$$p(j|v[l])^t = (p(v[l]|j)^t P(j)^t) \left(\sum_{j=1}^{M} p(v[l]|j)^t P(j)^t \right)^{-1} \tag{8}$$

where

$$p(v[l]|j)^t = \frac{1}{\sqrt{2\pi(\sigma_j^2)^t}} \exp\left\{ -\frac{(v[l] - \mu_j^t)^2}{2(\sigma_j^2)^t} \right\} \tag{9}$$

3.3 Sequential updating of parameters

Using the formulas given in equations (5), (6) and (7), the parameters can be estimated given that the N data are available. For a large database, N is usually only a small portion of the total number of elements in the database. In practice, it is desirable to have an incremental pdf update scheme that can track the changes of the data distribution. The EM algorithm can be modified for sequential updating [1]. Given that $\{P(j)^N, \mu_j^N, (\sigma_j^2)^N\}$ is the parameter set estimated form using the N data $v[l]$, the updated parameter set, when there is a new data $v[N+1]$ coming in, can be computed as

$$\mu_j^{N+1} = \mu_j^N + \theta_j^{N+1} (v[N+1] - \mu_j^N) \tag{10}$$

$$(\sigma_j^2)^{N+1} = (\sigma_j^2)^N + \theta_j^{N+1} [(v[N+1] - \mu_j^N)^2 - (\sigma_j^2)^N] \tag{11}$$

$$P(j)^{N+1} = P(j)^N + \frac{1}{N+1} (p(j|v[N+1]) - P(j)^N) \tag{12}$$

where

$$(\theta_j^{N+1})^{-1} = \frac{p(j|v[N])}{p(j|v[N+1])}(\theta_j^{N})^{-1} + 1 \quad (13)$$

The conditional probability $p(j|v[N+1])$ is computed as in (8) and (9).

3.4 Bit allocation

Denote the estimated pdf to be $p(x)$ as the approximation of $p(x)$. The objective of nonlinear quantization is to segment the pdf into grids of equal area. If the boundary points are denoted by $c[l]$, $l = 0, \ldots, 2^b$, b is the number of bits allocated, the boundary points should satisfy

$$\int_{c[l]}^{c[l+1]} \hat{p}(x)dx = \frac{1}{2^b}\int_{c[0]}^{c[2^b]} \hat{p}(x)dx \quad (14)$$

Using this criterion, the boundary points can be determined efficiently from a single scan of the estimated pdf.

3.5 Approximation updating

For dynamic databases, the pdf estimates are to be updated periodically. In our implementation, we update the estimates whenever a certain number of new data items are added. The approximation and pdf quantization are updated only when the new pdf, denoted as $p_{new}(x)$, differs significantly from the current pdf, denoted as $p_{old}(x)$. We use the following measure to quantify this change

$$\rho = (\int (\hat{p}_{old}(x) - \hat{p}_{new}(x))^2 dx)(\int \hat{p}_{old}(x)^2 dx)^{-1} \quad (15)$$

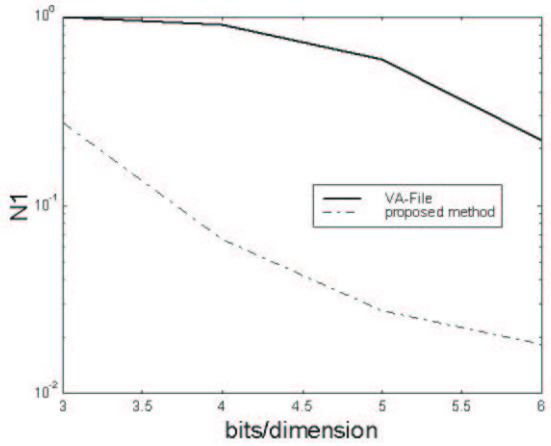
The approximation on that dimension is updated when ρ is larger than a certain threshold.

4 Experiments and Discussions

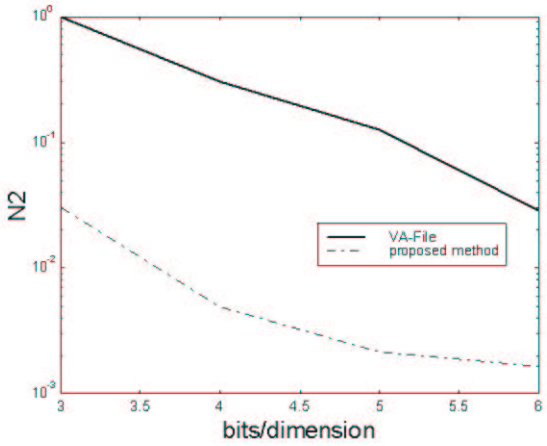
Our evaluation is performed on a database containing 275,465 aerial photo images. For each image, the method developed in [4] is used to extract a 60 dimensional texture feature descriptor. Initially, around 10% of total number of image objects are used to initialize the pdf estimation using the algorithms proposed in Section 3.2. Based on the estimated pdf, the approximation is constructed to support nearest neighbor search of the whole database. Meanwhile, an updating strategy is developed to take the rest of the data as input to the on-line estimation. For each dimension, when the change of the estimated pdf, which is measured by (15), is beyond the threshold $\rho > 0.15$, the approximation is adjusted accordingly.

The approach that uses the regular approximation, VA-File, is also implemented for comparison purpose. The number of candidate N_1 and the number of visited feature vectors N_2 are used to evaluate the performance. We tested using 3, 4, 5 and 6 bits for each dimension to construct the approximation. For each approximation, we consider the queries to be all image items in the database. For each query, the 10 nearest neighbor search is performed and results in a N_1 and a N_2. The average performances of N_1 and N_2 are computed by averaging N_1 and N_2 from all queries. The results are shown in Figure 2(a) and (b) for N_1 and N_2, respectively. Note that the figures are plotted on a logarithmic Y-axis. To normalize the results into the same range for displaying, the maximum value of $N_1(N_2)$ of VA-File is used to normalize all the values of $N_1(N_2)$. As observed, N_1 for the VA-File is about 3 to 20 times more than the proposed adaptive method (Figure 2(a)). After the second phase filtering, the VA-File visits 16 to 60 times more feature vectors than the proposed method (Figure 2(b). One can conclude that the adaptive pdf quantization results in an order of magnitude performance improvement over the regular VA-File.

We have also investigated the indexing performance as the size of the database grows. For this purpose, we construct 4 databases, including 10%, 25%, 50% and 100% percent of the total 275,645 image objects. For each dimension, 6 bits are assigned to construct the approximation. In measuring the scalability, we are mainly concerned with N_1 as this



(a)



(b)

Fig 2. Comparison between VA-File and the proposed method: (a) Number of candidates (N1); (b) Number of visited feature vectors (N2) after the second phase filtering.The proposed methods offers a significant reduction in N1 and N2 compared to the VA files.

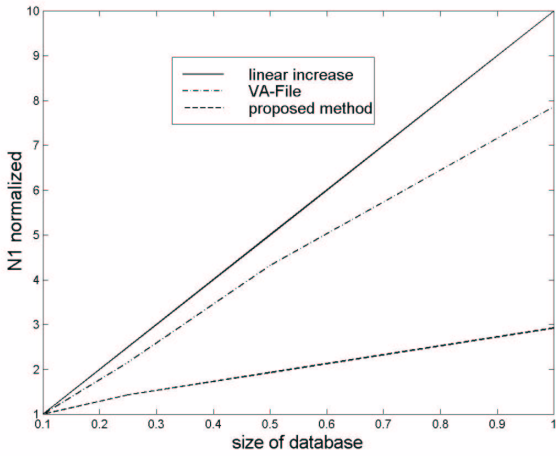


Fig 3. The increase of the number of candidates (N1) vs. the size of databases, the solid line is plotted to illustrate a linear increase of N1 as the database grows.

relates directly to the I/O cost of the index structure. Figure 3 illustrates that as the size of the database grows, how fast the number of candidates N_1 increase for the VA-File and the proposed approach. As can be observed, in terms of the scalability with the size of databases, the proposed method tends to maintain a much better sub-linear behavior as the database grows.

We have presented a novel adaptive indexing scheme for high-dimensional feature vectors. The method is adaptive to the data distribution. The indexing performance scales well with the size of the database. Experiments demonstrate a significant improvement over the VA-file data structures for high dimensional datasets.

Ackowledgement:

This research was in part supported by the following grants/awards: LLNL/ISCR award #0108, NSF-IRI 9704785, NSF Instrumentation #EIA-9986057, NSF Infrastructure NSF#EIA-0080134, and by Samsung Electronics.

References

[1] Christopher M. Bishop, *Neural networks for pattern recognition*, Oxford: Clarendon Press, 1995.

[2] H. Ferhatosmanoglu, E. Tuncel, D. Agrawal, A. E. Abbadi, "Vector approximation based indexing for non-uniform high dimensional data sets," Proc. Int'l Conf. Information and Knowledge Management (CIKM)**,** pp. 202-209, Washington, DC, USA. November 2000.

[3] S. Kaski, "Dimensionality reduction by random mapping: Fast similarity computation for clustering," Proc. Int'l Joint Conference on Neural Networks, volume 1, pages 413-418. IEEE Service Center, Piscataway, NJ, 1998.

[4] B. S. Manjunath and W. Y. Ma, "Texture features for browsing and retrieval of image data," IEEE Trans. on Pattern Analysis and Machine Intelligence, 18(8), pp. 837-842, 1996.

[5] R. Webber, J.-J. Schek and S. Blott, "A quantitative analysis and performance study for similarity-search methods in high-dimensional space," Proc. Int'l Conf. Very Large Data bases, pp. 194-205, New York City, New York, August 1998.

[6] ISO/IEC JTC1/SC29/WG11, "MPEG-7 Working Draft 4.0," Beijing, July, 2000.