Techniques for Similarity Searching in Multimedia Databases*

Hanan Samet
Center for Automation Research, Institute for Advanced Studies,
Department of Computer Science, University of Maryland
College Park, MD 20742 USA
hjs@cs.umd.edu

ABSTRACT

Techniques for similarity searching in multimedia databases are reviewed. This includes a discussion of the curse of dimensionality, as well as multidimensional indexing, distance-based indexing, and the actual search process which is realized by nearest neighbor finding.

1. INTRODUCTION

The representation of multidimensional points and objects, and the development of appropriate indexing methods that enable them to be retrieved efficiently is a well-studied subject (e.g., [5, 6]). Most of these methods were designed for use in application domains where the data usually has a spatial component which has a relatively low dimension. Examples of such application domains include geographic information systems (GIS), spatial databases, solid modeling, computer vision, computational geometry, and robotics. However, there are many application domains where the data is of considerably higher dimensionality, and is not necessarily spatial. This is especially true in multimedia databases where the data is a set of objects and the high dimensionality is a direct result of trying to describe the objects via a collection of features (also known as a *feature vector*). In the case of images, examples of features include color, color moments, textures, shape descriptions, etc. expressed using scalar values. The goal in these applications is often expressed more generally as one of the following:

1. Find objects whose feature values fall within a given range or where the distance from some query object falls into a certain range (range queries).

2. Find objects whose features have values similar to those of a given query object or set of query objects (nearest neighbor queries).

*This work was supported in part by the National Science Foundation under Grants IIS-09-48548, IIS-08-12377, CCF-08-30618, and IIS-07-13501, as well as the Office of Policy Development & Research of the Department of Housing and Development, Microsoft Research, Google, and NVIDIA.

Proceedings of the VLDB Endowment, Vol. 3, No. 2

These queries are collectively referred to as *similarity searching*, and the issues involved in supporting them is the subject of this tutorial, which is organized as follows. Section 2 discusses the curse of dimensionality and its ramifications. Sections 3 and 4 discuss multidimensional indexing and distance-based indexing, respectively. Concluding remarks are drawn in Section 5.

2. CURSE OF DIMENSIONALITY

An apparently straightforward solution to finding the nearest neighbor is to compute a Voronoi diagram for the data points (i.e., a partition of the space into regions where all points in the region are closer to the region's associated data point than to any other data point), and then locate the Voronoi region corresponding to the query point. The problem with this solution is that the combinatorial complexity of the Voronoi diagram in high dimensions is prohibitive — that is, it grows exponentially with its dimension k so that for N points, the time to build and the space requirements can grow as rapidly as $\Theta(N^{k/2})$ [7]. This renders its applicability moot.

The above is typical of the problems that we must face when dealing with high-dimensional data. Generally speaking, multidimensional queries become increasingly more difficult as the dimensionality increases. The problem is characterized as the *curse of dimensionality* (e.g., [7]). This term is used to indicate that the number of samples needed to estimate an arbitrary function with a given level of accuracy grows exponentially with the number of variables (i.e., dimensions) that comprise it. For similarity searching (i.e., finding nearest neighbors), this means that the number of objects (i.e., points) in the data set that need to be examined in deriving the estimate grows exponentially with the underlying dimension.

The curse of dimensionality has a direct bearing on similarity searching in high dimensions as it raises the issue of whether or not nearest neighbor searching is even meaningful in such a domain. In particular, letting d denote a distance function which need not necessarily be a metric (e.g., see . It has been pointed out (e.g., [7]) that nearest neighbor searching is not meaningful when the ratio of the variance of the distance between two random points p and q, drawn from the data and query distributions, and the expected distance between them converges to zero as the dimension k goes to infinity — that is,

$$\lim_{k \to \infty} \frac{\text{Variance}[d(p,q)]}{\text{Expected}[d(p,q)]} = 0.$$

In other words, the distance to the nearest neighbor and the

distance to the farthest neighbor tend to converge as the dimension increases.

3. MULTIDIMENSIONAL INDEXING

Assuming that the curse of dimensionality does not come into play, query responses are facilitated by sorting the objects on the basis of some of their feature values and building appropriate indexes. The high-dimensional feature space is indexed using some multidimensional data structure (termed *multidimensional indexing*) with appropriate modifications to fit the high-dimensional problem environment. Similarity search which finds objects similar to a target object can be performed with a range search or a nearest neighbor search in the multidimensional data structure. However, unlike applications in spatial databases where the distance function between two objects is usually Euclidean, this is not necessarily the case in the high-dimensional feature space where the distance function may even vary from query to query on the same feature .

Searching in high-dimensional spaces is time-consuming. Performing range queries in high dimensions is considerably easier, from the standpoint of computational complexity, than performing similarity queries as range queries do not involve the computation of distance. In particular, searches through an indexed space usually involve relatively simple comparison tests. However, if we have to examine all of the index nodes, then the process is again time-consuming. In contrast, computing similarity in terms of nearest neighbor search makes use of distance and the process of computing the distance can be computationally complex. For example, computing the Euclidean distance between two points in a high-dimensional space, say d, requires d multiplication operations and $d-1$ addition operations, as well as a square root operation (which can be omitted). Note also that computing similarity requires the definition of what it means for two objects to be similar, which is not always so obvious.

4. DISTANCE-BASED INDEXING

Often, the only information that we have available is a distance function that indicates the degree of similarity (or dis-similarity) between all pairs of the N objects. Usually the distance function d is required to obey the triangle inequality, be non-negative, and be symmetric, in which case it is known as a *metric* and also referred to as a *distance metric*. However, at times, the distance function is not a metric (e.g., SASH[4, 7]). Often, the degree of similarity is expressed using a similarity matrix which contains interobject distance values, for all possible pairs of the N objects

Given a distance function, we usually index the objects with respect to their distance from a few selected objects. We use the term *distance-based indexing* to describe such methods (e.g., [2]). There are two basic partitioning schemes: *ball partitioning* and *generalized hyperplane partitioning*.

In ball partitioning, the data set is partitioned based on distances from one distinguished object, sometimes called a *vantage point*, into the subset that is inside and the subset that is outside a ball around the object In generalized hyperplane partitioning, two distinguished objects p_1 and p_2 are chosen and the data set is partitioned based on which of the two distinguished objects is the closest — that is, all the objects in subset A are closer to p_1 than to p_2, while the objects in subset B are closer to p_2. The asymmetry of ball partitioning is a potential drawback of this method as the outer shell tends to be very narrow for metric spaces typically used in similarity search In contrast, generalized hyperplane partitioning is more symmetric, in that both partitions form a "ball" around an object.

The advantage of distance-based indexing methods is that distance computations are used to build the index, but once the index has been built, similarity queries can often be performed with a significantly lower number of distance computations than a sequential scan of the entire dataset. Of course, in situations where we may want to apply several different distance metrics, then the drawback of the distance-based indexing techniques is that they require that the index be rebuilt for each different distance metric, which may be nontrivial. This is not the case for the multidimensional indexing methods which have the advantage of supporting arbitrary distance metrics (however, this comparison is not entirely fair, since the assumption, when using distance-based indexing, is that often we do not have any feature values as for example in DNA sequences).

5. CONCLUDING REMARKS

Providing indexing support for similarity searching is an important area where much work remains to be done. Some of the more promising research directions lie in developing techniques to identify the important features in the applications so that the dimension of the problem domain can be reduced. An alternative is to find an embedding for the distance function in a vector space (e.g, [3]) thereby enabling us to properly utilize the vast array of existing indexing and nearest neighbor techniques (e.g., [1, 2, 8]).

6. REFERENCES

[1] G. R. Hjaltason and H. Samet. Distance browsing in spatial databases. *ACM Transactions on Database Systems*, 24(2):265–318, June 1999. Also University of Maryland Computer Science Technical Report TR–3919, July 1998.

[2] G. R. Hjaltason and H. Samet. Index-driven similarity search in metric spaces. *ACM Transactions on Database Systems*, 28(4):517–580, Dec. 2003.

[3] G. R. Hjaltason and H. Samet. Properties of embedding methods for similarity searching in metric spaces. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(5):530–549, May 2003. Also an expanded version in University of Maryland Computer Science Technical Report TR–4102, January 2000.

[4] M. E. Houle and J. Sakuma. Fast approximate similarity search in extremely high-dimensional data sets. In *Proceedings of the 21st IEEE International Conference on Data Engineering*, pages 619–630, Tokyo, Japan, Apr. 2005.

[5] H. Samet. *Applications of Spatial Data Structures: Computer Graphics, Image Processing, and GIS*. Addison-Wesley, Reading, MA, 1990.

[6] H. Samet. *The Design and Analysis of Spatial Data Structures*. Addison-Wesley, Reading, MA, 1990.

[7] H. Samet. *Foundations of Multidimensional and Metric Data Structures*. Morgan-Kaufmann, San Francisco, 2006.

[8] H. Samet, J. Sankaranarayanan, and H. Alborzi. Scalable network distance browsing in spatial databases. In *Proceedings of the ACM SIGMOD Conference*, pages 43–54, Vancouver, Canada, June 2008. Also see University of Maryland Computer Science Technical Report TR–4865, April 2007.