Sizing Sketches: A Rank-Based Analysis for Similarity Search

Zhe Wang, Wei Dong, William Josephson, Qin Lv, Moses Charikar, and Kai Li
Department of Computer Science, Princeton University
Princeton, NJ 08540, USA
zhewang,wdong,wkj,qlv,moses,li@cs.princeton.edu

ABSTRACT

Sketches are compact data structures that can be used to estimate properties of the original data in building large-scale search engines and data analysis systems. Recent theoretical and experimental studies have shown that sketches constructed from feature vectors using randomized projections can effectively approximate ℓ_1 distance on the feature vectors with the Hamming distance on their sketches. Furthermore, such sketches can achieve good filtering accuracy while reducing the metadata space requirement and speeding up similarity searches by an order of magnitude. However, it is not clear how to choose the size of the sketches since it depends on data type, dataset size, and desired filtering quality. In real systems designs, it is necessary to understand how to choose sketch size without the dataset, or at least without the whole dataset.

This paper presents an analytical model and experimental results to help system designers make such design decisions. We present a rank-based filtering model that describes the relationship between sketch size and dataset size based on the dataset distance distribution. Our experimental results with several datasets including images, audio, and 3D shapes show that the model yields good, conservative predictions. We show that the parameters of the model can be set with a small sample dataset and the resulting model can make good predictions for a large dataset. We illustrate how to apply the approach with a concrete example.

Categories and Subject Descriptors

H.3.3 [**Information Search and Retrieval**]: [Search process, Information filtering]

General Terms

Algorithms, Measurement, Design

Keywords

similarity search, feature-rich data, sketch

SIGMETRICS'07, June 12–16, 2007, San Diego, California, USA.

1. INTRODUCTION

Content-based similarity search for massive amounts of feature-rich (non-text) data has been a challenging problem because feature-rich data objects such as images, audio, and other sensor data is typically represented by many feature vectors with tens to hundreds of dimensions each. As a result, the key challenge in designing a content-based similarity search engine is solving the general high-dimensional search problem for very large datasets. In other words, we must understand how to find data objects similar to a query data object quickly with small data structures.

Although past research has made significant progress on the high-dimensional search problem, there is still no satisfactory general solution. Tree data structures such as R-tree [13], K-D tree [2], SR-tree [15], and more recently navigating-nets [17] and cover-tree [3], have been proposed to solve the K-Nearest-Neighbor (KNN) search problem in high-dimensional spaces. But they work well only when the (intrinsic) dimensionality is relatively low. When the dimensionality is beyond 15 or so, such approaches are typically slower than the brute-force approach which scans through all data objects in the dataset. Recently, several indexing methods based on locality sensitive hashing (LSH) [14, 12, 6, 22] have been proposed for approximate KNN search, but they are also limited to relatively low intrinsic dimensionality. When the intrinsic dimensionality is high, the LSH approach typically requires hundreds of hash tables to achieve reasonably good search quality.

A promising approach is to use *sketches* as compact metadata for a similarity search engine. Sketches, which are constructed from domain-specific feature vectors, have two salient properties: their small size and the ability to estimate the distance between two feature vectors from their sketches alone. At search time, sketches are used to filter out unlikely answers, resulting in a much smaller candidate set which is then ranked with a sophisticated distance function on the original feature vectors. This approach is practical for two reasons: the first is that for feature-rich data with high intrinsic dimensionality, filtering metadata is more efficient in both space and time than other known approaches. The second is that content-based similarity search capability is often integrated with traditional search tools based on attributes such as time, location, and other annotations. A typical integration method is to perform an attribute-based search to produce an intermediate dataset which can be filtered on-the-fly efficiently into a small candidate set for final ranking.

Recent theoretical and experimental studies have shown that sketches constructed based on random projections can be used to approximate ℓ_1 distance and that such sketches can achieve good filtering accuracy while reducing the metadata space requirement and speed up similarity searches by an order of magnitude[5, 20]. An important advantage of this approach over other dimension reduction techniques is that sketch construction based on random projections requires no prior knowledge about the content of the datasets. The challenge of designing a real system using this approach is to choose the sketch size wisely.

Properly sized sketches can greatly reduce the storage requirement for metadata and speed up similarity search while maintaining good search quality. An important design decision is sketch size in bits, given the desired filtering quality and the dataset size of a specific data type. Choose too few bits, and the distance estimates computed from the sketches will be inaccurate. Choose too many bits, and the sketches will needlessly waste storage space and CPU time. Ideally, a system designer can determine the sketch size and other parameters of the algorithm at system initialization time when he knows only the targeted data type, dataset size, and perhaps a small sample dataset. In order to achieve this goal, we need to model the relationship between the sketch size and other information and understand how to use the model in real systems designs.

This paper presents two analytical and experimental results to help systems designers achieve the goal above. The first is a rank-based filtering model for the random projection based sketching technique that uses Hamming distance to approximate ℓ_1 distance. We have validated the model with image, audio, and 3D shape datasets and shown that the model can conservatively predict the required sketch size, given desired filtering quality, target dataset size, and filtering result size.

The second is the result of investigating how to use the rank-based filtering model to help systems designers make design decisions without the whole dataset. Experimental results on three real datasets show that the rank-based filtering model performs well, yielding useful, conservative predictions for larger datasets even though the parameters of the model are set with a small sample dataset. This result allows systems designers to build the model into a software tool.

We then show how to apply the analytical results to size sketches in configuring a content-based similarity search tool for a 3D-shape dataset. The case study shows that the analytical model is convenient to use.

2. FILTERING FOR SIMILARITY SEARCH

This section describes the similarity search problem, sketching algorithm, and filtering method using sketches that are considered in our analytical and experimental study.

2.1 Similarity Search

Informally, similarity search refers to searching a collection of objects to find objects similar to a given query object. The objects we will be interested in are noisy, high-dimensional objects such as images and audio recordings. Here, similarity between objects refers to a human-perceived notion of similarity. This informal notion of similarity search is made concrete as follows: objects are represented by high-dimensional feature vectors and similarity is defined in terms of a distance metric on the underlying space of features. Given a query object, q, in this setting, the goal is to find nearby objects, r, such that the distance $d(q, r)$ is small. In particular, we may ask for all objects r within some chosen distance of the query point, or more often, we may ask for the k nearest neighbors of the query point. This latter formulation of the search problem is commonly referred to as the k-nearest neighbor (k-NN) problem.

Although the choice of how to extract features and which distance function to use are domain specific, in practice, it is frequently the case that objects are represented by D-dimensional real-valued vectors and the perceptual distance between objects is modeled by one of the ℓ_p norms. For a pair of points $X = (X_1, \ldots, X_D)$ and $Y = (Y_1, \ldots, Y_D)$, these distance functions have the form:

$$d(X, Y) = \Big(\sum_{i=1}^{D} |X_i - Y_i|^p \Big)^{1/p}$$

2.2 L1 Sketch Construction

In this paper, we focus on a recently proposed sketching technique [18]. The sketches constructed using this technique are bit vectors and the Hamming distance between two bit vectors approximates the weighted ℓ_1 distance between the original feature vectors. This sketching technique has proved useful in several application domains[20].

Briefly, the sketch construction algorithm works by randomly picking a threshold in a particular dimension and checking if the vector's value in that dimension is larger (bit 1) or smaller (bit 0) than the threshold. Let B be the sketch size in bits, and H be the XOR block size in bits. Each sketch is constructed by first generating $B \times H$ bits and then XORing every H consecutive bits, resulting in the final B-bit sketch. By XORing H bits into 1 bit, this algorithm produces a dampening (or thresholding) effect such that smaller distances are approximated with higher resolution, making it suitable for nearest neighbor search[1].

Let w_i be the "weight" (or importance) assigned by the domain-specific feature extraction algorithm to dimension i and let l_i and u_i, respectively, be the minimum and maximum values for the i-th coordinate over all observed feature vectors. Let D be the dimension of the feature vector. At system startup time, $B \times H$ random (i, t_i) pairs are generated using Algorithm 1. At run time, the D-dimensional feature vector x is converted into a B-bit bit vector using Algorithm 2. For further details and a proof of correctness, we refer the reader to [18].

2.3 Filtering using Sketches

Since sketches require little storage space and since the distance between query objects can be estimated from sketches efficiently, sketches can be used to implement a filtering query processor for similarity search. A filtering query processor first constructs a candidate set of result objects for a given query object on the basis of sketch distance. The candidate set size is chosen to be large enough such that it is likely to contain the k-nearest neighbors under the original distance on feature vectors. In effect, the construction of the candidate set "filters out" the vast majority of objects in the system that are far from the query object while still capturing the objects close to the query. Since sketches

[1]See Formula 1 in Section 3 for details.

Algorithm 1 Generate $B \times H$ Random (i, t_i) Pairs

input: $B, H, D, l[D], u[D], w[D]$
output: $p[D], rnd\_i[B][H], rnd\_t[B][H]$

$p_i = w_i \times (u_i - l_i)$; $\quad for\ i = 0, \ldots, D-1$
normalize p_i $\quad s.t.\ \Sigma_{i=0}^{d-1}\ p_i = 1.0$

for $(b = 0; b < B; b++)$ **do**
  for $(h = 0; h < H; h++)$ **do**
    pick random number $r \in [0, 1)$
    find i $\quad s.t.\ \Sigma_{j=0}^{i-1}\ p_i <= r < \Sigma_{j=0}^{i}\ p_i$
    $rnd\_i[b][h] = i$
    pick random number $t_i \in [l_i, u_i]$
    $rnd\_t[b][h] = t_i$
  end for
end for

Algorithm 2 Convert Feature Vector to B-Bit Vector

input: $v[D], B, H, rnd\_i[B][H], rnd\_t[B][H]$
output: $bits[B]$

for $(b = 0; b < B; b++)$ **do**
  $x = 0$
  for $(h = 0; h < H; h++)$ **do**
    $i = rnd\_i[b][h]$; $\quad t_i = rnd\_t[b][h]$
    $y = (v_i < t_i\ ?\ 0 : 1)$
    $x = x \bigoplus y$
  end for
  $bits[b] = x$
end for

are small and distance estimation on sketches are very efficient, a simple, yet practical approach for generating this candidate set is a linear scan through the set of all sketches.

The second step in a filtering query processor is the ranking of the candidate set by the original distance metric on the original feature vector. This exact computation need only be carried out once for each point in the candidate set. The k-nearest neighbors in the candidate set under the original distance metric is then taken as the query result set. The underlying assumption in a filtering query processor is that the k-nearest neighbors in the candidate set is an accurate estimate of the k-nearest neighbors in the full data set. In practice, one must choose the candidate set to be large enough that it captures a sufficiently large fraction of the k-nearest neighbors under the original distance, but not so large that it adversely affects search engine performance. If the candidate set is too small, the query processor will be fast, but the search quality may be poor. On the other hand, if the candidate set is too big, the processor will waste time and resources on unlikely candidates. We can capture this inherent trade off between search quality and filter set size by asking what filter ratio is necessary to achieve a particular quality goal. If k is the number of results to return, a filter with filter ratio t will return a candidate set of size $t \times k$. A filtering query processor seeks to optimize t for a given fraction of the k-nearest neighbors in the final result set.

A system designer who adopts the filtering approach to similarity search must choose not only a particular domain-specific feature representation and distance function, but also an appropriate sketching algorithm and a set of parameters for sketching and filtering. More specifically, we are mainly interested in answering the following questions:

- What is an appropriate choice for the sketch size, B?
- How to size the sketch if the input data set grows over time?

We are also interested in the other parameters involved for sketching such as the best H value for XORing and best filter ratio t when constructing sketches as they are part of the sketching parameters system designer need to decide at design phase. The rest of the paper presents our analytical and experiment results to answer these questions.

3. ANALYTICAL MODEL

We use the following notation:

- B: sketch size in bits
- k: number of similar objects to return
- t: filter ratio – *i.e.* filtered set size is $k \times t$
- H: XOR block size in bits for sketching
- S: the set of domain-specific feature vectors
- D: the dimensionality of vectors in S
- $d(x, y)$: the domain-specific distance on $x, y \in S$
- $s^0(x)$: the $H \times B$-bit sketch of $x \in S$ before XORing
- $s(x)$: the B-bit sketch of the feature vector $x \in S$
- $d_s(x, y)$: the sketch distance between $x, y \in S$

We now describe a simple analytical model for filtering using the ℓ_1 sketch of Section 2.2. This model provides a basis for system designers to choose appropriate parameter values for a sketch-based filtering similarity search query processor. In particular, for a given data set size, N, and result set size, k, the model predicts the relationship between recall, filter ratio (t), sketch size (B), and XOR block size (H). Thus, the model allows a system designer to choose the system parameters in anticipation of future growth.

In the following description let S be a set of N objects, each represented by a D-dimension feature vector. Given objects q and r, let $d(q, r)$ be the *feature distance* between q and r, $s(q)$ and $s(r)$ be the sketches of q and r, respectively, and $d_s(q, r)$ the *sketch distance* between q and r. We define the *rank* of r given q to be the number of points in S that precede r when objects are ordered in increasing order of *feature* distance from q. For a fixed query q, let r_i denote the ith object in S in this ordering. Similarly, we define the *sketch rank* of r to be the number of points in S that precede r when objects are ordered in increasing order of *sketch* distance to q.

The goal is for the analytical model to answer the following question: Given N, k fixed, as a function of t, B, and H, what fraction of the points $p \in S$ with rank at most k have sketch rank at most $k \times t$? We develop the model in a series of steps. First, we describe how we model the distribution of feature distances in the data set. Second, we obtain an expression for the distribution of the sketch distance as a

function of the feature distance. Next, we model the distribution of the sketch rank of an object $r \in S$ for a query q as a function of its feature distance from q. This uses the distribution of feature distances in the data set and distribution of sketch distances. Finally, we use this model for the sketch rank to estimate the recall for a given filter ratio value. Each of these steps is described in the subsections that follow.

3.1 Distance Distribution

Since the sketch distance between two objects is related to the original feature vector distance, we first study the distribution of feature vector distances. For one particular query object, we calculate the feature vector distances of all the other objects in the dataset to this query object. The histogram of all the object feature distances forms the feature vector distance distribution for that particular query object. With the feature distance distribution known, we will be able to predict the sketch distance between the query object and rest of the objects using the analytic model described in the next section. Note that in k-nearest neighbor search, objects that are nearby have much more impact on the overall search quality than the ones that are further away. When we model the distance distribution, we are mostly interested in the distance distribution close to the k nearest neighbors.

One of the goals for our approach is to predict the sketch performance when the dataset size changes. In order to do this, we predict the distribution of object distances in a data set using the distribution of distances in a smaller sample. The basis for this is the hypothesis that every data type is associated with an underlying object distance distribution. The particular distances observed in a specific data set can be viewed as a sample of the distribution associated with the data type. For example, in Figure 1, we compare the average distance distribution of 100 query points with the full dataset with that of a uniformly sampled dataset with only one-tenth of the data points.

One subtlety in modeling distances is that the distribution of distances from different query objects can be different and using a single distribution for them can lead to errors. The distance distributions for different query objects have similar shapes but are peaked at different points. Since our data objects have a natural bound on each dimension, the objects are contained in a high dimensional rectangle. The location of the query object in this high dimensional rectangle will affect the peak of the feature distance distribution. In order to model this variation, we pick a random sample of 100 query objects and use their distance distributions to approximate the overall distance distributions. We compared this approach with using a single average distance distribution. The latter did not perform as well as the approach that explicitly models the variation in object distance distributions.

Further, we approximate the empirical individual query object distance distributions by a distribution with a closed form expression. Due to the nature of k-nearest neighbor search, we are not trying to approximate the full distance distribution. Instead, only the distance distribution close to k-nearest neighbors are considered during fitting. The details of this appear in Section 5.1.

3.2 Sketch Distance Distribution

Given a dataset S, let w_i, u_i, l_i be the weight, upper bound and lower bound of the i-th dimension, respectively. Let $T = \sum_i w_i \times (u_i - l_i)$. Using the sketch algorithm of Section 2.2, for every object $r \in S$, we construct the initial bit vector $s^0(r)$ of length $B \times H$. For a fixed query point q, consider object $r \in S$ and let $x = d(q,r)/T$. The probability that $s^0(q)$ disagrees with $s^0(r)$ in the j-th bit is:

$$\Pr[s_j^0(q) \neq s_j^0(r)] = d(q,r)/T = x$$

After XORing contiguous H-bit blocks of s^0 to produce the final B-bit sketch, the probability that the two sketches differ in bit j is:

$$\Pr[s_j(q) \neq s_j(r)] = p(x) = \frac{1}{2}\left(1 - (1-2x)^H\right) \quad (1)$$

Thus, the probability that the two B-bit sketches $s(q)$ and $s(r)$ differ in exactly b bits is given by the binomial distribution:

$$\Pr[d_s(q,r) = b] = p(x,b) = \binom{B}{b} p(x)^b \left(1 - p(x)\right)^{B-b} \quad (2)$$

where $p(x)$ is given by equation (1). This formula gives the probability distribution of the sketch distance as a function of the feature distance. The proof in detail can be found [18].

3.3 Rank Distribution

Consider an object $r \in S$. We would like to estimate the sketch rank of r, *i.e.* the number of objects that precede r when we order all objects in S in increasing order of sketch distance to query object q. A key assumption in this calculation is that the sketch distances are independent of each other. While this assumption is not completely accurate, it is a reasonable approximation. As we discuss later, this leads to a conservative estimate on the quality of the filtering results. We also assume that in the ordering by sketch distances, objects with the same sketch distance are ordered randomly, that is, for two objects with the same sketch distance, the probability that one precedes the other is exactly 1/2.

The sketch rank of r is dependent on the sketch distance $d_s(q,r)$. Consider the event $d_s(q,r) = b$. Note that the probability of this event is a function of the feature distance $d(q,r)$ and is calculated in (2). Consider an object $r' \in S$ such that $d(q,r')/T = x$ and let $s = d_s(q,r')$ be the sketch distance of r'. Let $P(x,b)$ be the probability that r' is ranked lower (*i.e.* closer to q) than r when $d_s(q,r) = b$. Note that this is a function of $x = d(q,r')/T$ and the value b of $d_s(q,r)$.

$$\begin{aligned} P(x,b) &= \Pr[s < b] + \frac{1}{2}\Pr[s = b] \\ &= \sum_{i=0}^{b-1} \Pr[s = i] + \frac{1}{2}\Pr[s = b] \quad (3) \\ &= \sum_{i=0}^{b-1} p(x,i) + \frac{1}{2}p(x,b) \quad (4) \end{aligned}$$

Let $\mathrm{rank}(r)$ denote the sketch rank of r. $\mathrm{rank}(r)$ is the sum of indicator random variables $Y(r_i, r)$, one for every object $r_i \in S$. The indicator variable $Y(r_i, r)$ for $r_i \in S$ corresponds to the event that r_i precedes r in the ordering by sketch distance. Our independence assumption implies that given a value for $d_s(q,r)$, all these variables are independent. Let $x_i = d(q,r_i)/T$. Note that $\Pr[Y(r_i,r) = 1 | d_s(q,r) =$

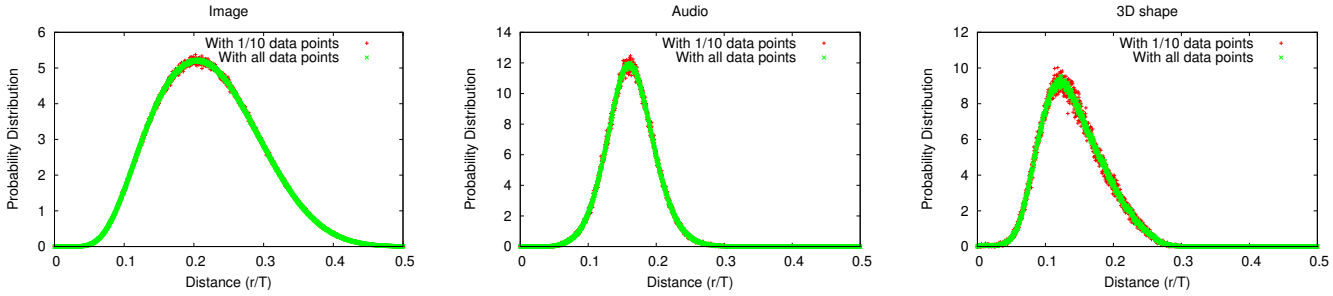

Figure 1: Compare Distance Distribution of Full Dataset and 1/10 of Dataset.

$b] = P(x_i, b)$ computed in (4). The expected value and variance of rank (r) are given by

$$\begin{aligned} \mathrm{E}[\mathrm{rank}\,(r)\,|d_s(q,r) = b] &= \sum_{i=1}^{N} P(x_i, b) \\ \mathrm{Var}[\mathrm{rank}\,(r)\,|d_s(q,r) = b] &= \sum_{i=1}^{N} P(x_i, b) - [P(x_i, b)]^2 \end{aligned}$$

When we use the feature distance distribution model, let $f(x)$ to be the probability density function for the distances, *i.e.* $\int_{x_1}^{x_2} f(x)dx$ is the fraction of points $r' \in S$ such that $d(q, r')/T \in [x_1, x_2]$. We can replace the summation over all data points with an integration over the distance distribution:

$$\begin{aligned} &\mathrm{E}[\mathrm{rank}\,(r)\,|d_s(q,r) = b] = N \int_0^1 P(x, b) f(x)dx \\ &\mathrm{Var}[\mathrm{rank}\,(r)\,|d_s(q,r) = b] = \\ &\qquad N \int_0^1 (P(x, b) - P(x, b)^2) f(x)dx \end{aligned}$$

Given the fact that N is usually on the order of hundreds of thousands, the distribution of rank (r) (for a specific value of $d_s(q, r)$) is approximately normal by the Central Limit Theorem. The normal distribution parameters can be determined by $\mathrm{E}[\mathrm{rank}\,(r)\,|d_s(q,r) = b]$ and $\mathrm{Var}[\mathrm{rank}\,(r)\,|d_s(q,r) = b]$. Thus the probability that rank (r) is at most M can be expressed as:

$$\Pr[\mathrm{rank}\,(r_k) \le M | d_s(q,r) = b] = \int_0^M f(y; \mu_b, \sigma_b)dy$$

where

$$\begin{aligned} \mu_b &= \mathrm{E}[\mathrm{rank}\,(r)\,|d_s(q,r) = b] \\ \sigma_b &= \sqrt{\mathrm{Var}[\mathrm{rank}\,(r)\,|d_s(q,r) = b]} \\ f(y; \mu, \sigma) &= \frac{1}{\sigma\sqrt{2\pi}} e^{-(y-\mu)^2/2\sigma^2} \end{aligned}$$

Now, we can write the distribution of rank (r) as a mixture of normal distributions, one for each value of $d_s(q, r)$. The distribution for b is weighted by $\Pr[d_s(q, r) = b]$. This gives us the distribution of rank (r) and allows us to calculate the probability that rank (r) is at most M as follows:

$$\Pr[\mathrm{rank}\,(r) \le M] = \sum_{b=0}^{B} \Pr[d_s(q,r) = b] \int_0^M f(y; \mu_b, \sigma_b)dy$$

We overload the notation somewhat and use rank (x) for $x \in [0, 1]$ to denote the sketch rank of an object $r \in S$ such that $d_s(q, r)/T = x$. Note that $\Pr[d_s(q, r) = b] = p(x, b)$. Using the previous expression for $\Pr[\mathrm{rank}\,(r) \le M]$, we get

$$\Pr[\mathrm{rank}\,(x) \le M] = \sum_{b=0}^{B} p(x, b) \int_0^M f(y; \mu_b, \sigma_b)dy$$

Given this expression for the rank distribution, we can now estimate search quality for a given filter set size, M.

3.4 Search Quality Estimation

Once we have an expression for the rank distribution for objects $r \in S$, for a given filter set size M, the expected fraction of the k nearest neighbors being included in the filtered set (*i.e.* the recall) can be computed as:

$$\mathrm{Recall} = \frac{1}{k} \sum_{j=1}^{k} \Pr[\mathrm{rank}\,(r_j) \le M]$$

When the feature distance distribution is used, the recall can be calculated as:

$$\mathrm{Recall} = \frac{N}{k} \int_0^{x_0} \Pr[\mathrm{rank}\,(x) \le M] f(x)dx$$

where x_0 can be derived from:

$$k = N \int_0^{x_0} f(x)dx$$

Note that the sketch distributions and rank distributions are computed based on a single query object q. As a result, the search quality estimate (recall) may vary for different query points. To ensure that the results are representative of the entire data set, we use multiple representative query objects to model the distance distribution. To estimate overall search quality, we average the recall value computed using the distance distributions for each of these query objects.

4. EVALUATION

We have employed three kinds of feature-rich datasets to validate our models. To evaluate the filtering quality, we have used average recall in which the "gold standard" for comparison is the results computed with the original distance function.

4.1 Datasets

We have studied three kinds of data: images, audio, and 3D shapes. Table 1 provides a summary of the dataset sizes

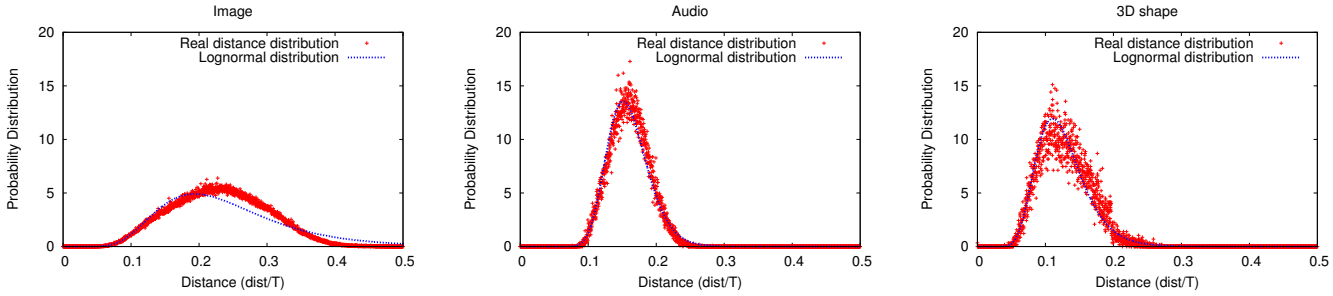

Figure 2: Compare the Real Distance Distribution with Lognormal Distribution: We only fit the initial part of real distribution for k-nearest neighbour search.

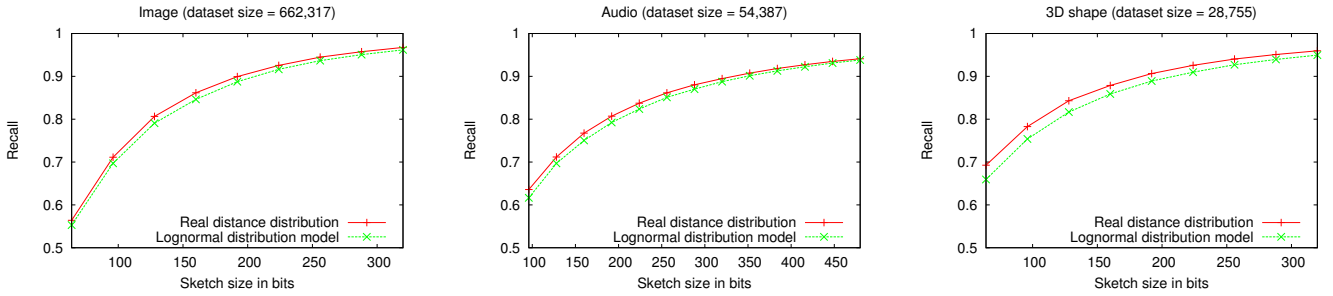

Figure 3: Filter Quality with Different Distribution Models (H = 3): Lognormal distribution gives conservative filtering quality prediction.

and the number of dimensions in the domain-specific feature vector representations.

Dataset	Number of Feature Vectors	Dimension
image	662,317	14
audio	54,387	192
3D shape	28,775	544

Table 1: Dataset Sizes and Dimensions.

4.1.1 *Image Data*

The image dataset used in our study is drawn from the Corel Stock Photo Library, which contains about 60,000 images. The main reason to choose this dataset is that it has become a standard dataset for evaluating content-based image retrieval algorithms.

We used the JSEG [7] image segmentation tool to segment each image into multiple homogeneous regions based on color and texture. On average, each image is segmented into 10 regions, resulting in about 660,000 regions in total. The image region representation we use is similar to the one proposed in [18], where each region is represented by a 14-dimensional feature vector: nine dimensions for color moments and five dimensions for bounding box information. The bounding box is the minimum rectangle covering a segment and is characterized by five features: aspect ratio (width/height), bounding box size, area ratio (segment size/bounding box size), and region centroids. The similarity between two regions is determined by weighted ℓ_1 distance on their 14-dimensional feature vectors.

4.1.2 *Audio Data*

Our audio dataset is drawn from the DARPA TIMIT collection [11]. The TIMIT collection is an audio speech database that contains 6,300 English sentences spoken by 630 different speakers with a variety of regional accents. We chose this dataset also because it is available to the research community.

We break each sentence into smaller segments and extract features from each segment. For each audio segment, we use the Marsyas library [24] to extract feature vectors. We begin by using a 512-sample sliding window with variable stride to obtain 32 windows for each segment and then extract the first six MFCC (Mel Frequency Cepstral Coefficients) parameters from each window to obtain a 192 dimensional feature vector for each segment. We use weighted ℓ_1 distance on the 192-dimensional feature vectors to determine similarity. As a result, the sentences of the dataset are partitioned into about 54,000 word segments, and we extract one feature vector per word segment.

4.1.3 *3D Shape Models*

The third dataset we use in our study contains about 29,000 3D shape models, which is a mixture of 3D polygonal models gathered from commercial viewpoint models, De Espona Models, Cacheforce models and from the Web. Each model is represented by a single feature vector, yielding about 29,000 feature vectors in total.

To represent the 3D shape models, we use the Spherical Harmonic Descriptor (SHD) [16], which has been shown to provide good search quality for similarity search of 3D shape models. The models are first normalized, then placed on a $64 \times 64 \times 64$ axial grid. Thirty-two spheres of different di-

ameters are used to decompose each model. Up to order 16 spherical harmonic coefficients are derived from the intersection of model with each of the 32 spherical shells. By concatenating all the spherical descriptors of a 3D model in a predefined order, we get a $32 \times 17 = 544$-dimensional shape descriptor for each 3D model. Although the original SHD algorithms used ℓ_2 distance as the similarity metric, we have found that ℓ_1 distance delivers similar search quality. As a result, in this study we use ℓ_1 distance on the 544-dimensional feature vector.

4.2 Evaluation Metrics and Method

We have conducted two types of experimental studies in this paper: distribution model fitting and filtering model validation.

For distribution model fitting, we use some common distribution functions to fit the distance distribution and compare the fitted distance function with the real distribution. In order to validate the fit, we compare the residuals after the least squared fitting and also plot the result for visual inspection. Moreover, we put each fitted distribution function into our model and compare their results with the result using real distance distribution to determine the best distribution function.

For filtering model validation, we compare the filtering results predicted by the model with those by an implementation of the sketch-based filtering algorithm. The filtering qualities are measured against the *gold standard* of each dataset, which are computed by a brute-force approach over the entire dataset using the original distance function. Specifically, we compare the recall value at filter ratio t predicted by our model with that computed experimentally. The recall at filter ratio t is the fraction of the k nearest neighbors to the query point that appear in the first $t \times k$ objects found by the filtering search algorithm. An ideal filtering algorithm will have a recall value of 1.0 at a filter ratio of 1. In practice, a good filter is one that achieves a satisfactory recall with a small filter ratio. Since re-ranking of the candidate objects filter using the original feature vectors is done after filtering, we do not need to report the precision of our filtering method here.

The method used in our experimental evaluation is to pick one hundred objects uniformly at random from each dataset as queries. For each query object, we use the domain-specific feature vector distance to compute the k nearest neighbors. We then fix the size of the sketch produced by the sketching algorithm and for each object in the dataset generate a sketch of that size, and use the sketches to compute the filtered candidate set of $t \times k$ objects and calculate the fraction of the k nearest neighbors appearing in this set. Since the sketching algorithm is itself randomized, we take the average recall over ten instances of the sketch algorithm. Finally, for each sketch size, we report the average recall value at a fixed filter ratio t over the one hundred randomly chosen query objects and compare that recall to the recall predicted by our model.

5. EXPERIMENTAL RESULTS

We are interested in answering the following three questions:

- How shall we model the distance distribution of real datasets to be used in the analytical model?
- How well can the analytical model help system designers choose design parameters such as sketch size?
- How well can the analytical model predict for large dataset if its parameters are set with a small sample dataset?

This section reports the experimental answers to these questions.

5.1 Distance Distribution Model

In this section, we explore the possibility of using a simple distribution – with closed form expression – to model the real data distance distribution. This allows us to model the system with fewer parameters (typically two) rather than the full distance distribution. Furthermore, we can reuse the distribution model when the dataset size grows.

We have investigated several common distance distributions which may be used to model the observed real distance distribution from the dataset and decided to choose lognormal distribution in our experiments. Note that in k-nearest neighbor search, objects that are much further away than the k-th nearest neighbor have much less impact on the overall search quality than the ones that are closer. As a result, the distance distribution close to the k nearest neighbors are the most important in the overall result.

Based on this observation, we use the distance values of the $2 \times k \times t$ closest data points to fit the statistical models, and then use the models to extrapolate to the full set of distances. We use GNU Scientific Library [9]'s nonlinear least-square fitting library to find fitting parameters. We only use initial part of the distance distribution corresponding to the $2 \times k \times t$ closest data to do the fitting. It tries to minimize the error between the real distance distribution and the corresponding portion of lognormal distance distribution. As shown in Figure 2, the lognormal distribution fits the data, even when extrapolated to the full dataset. Since this shows the distance distribution of just a single query point, we can see the distance distribution is not as smooth as in Figure 1 where the average distance distribution of 100 query points is used.

To model the real dataset, we use 100 randomly chosen query points to generate 100 distance distributions. After that, each distance distribution is fitted with the lognormal distribution. We then use these 100 sets of lognormal distribution parameters to model the distance distribution of the whole dataset. This helps us to model the variation of distance distributions as seen from different data points in the real dataset.

We have also validated the choice of the distribution model by comparing the filtering result generated from the real distance distribution with that generated by the model distribution. Figure 3 shows the filtering results using lognormal distribution model, together with the filtering results when the real distance distribution is used[2]. We can see that the lognormal distribution generates close trend to the real distance distribution. It is important to note that using lognormal distribution gives a conservative estimate of the recall compared to the real distribution. From the system designer's perspective, this is desirable behavior since the recall is bounded by the model at a small cost in sketch size.

[2]The real distance distribution is directly computed from the dataset and no parameter fitting is performed.

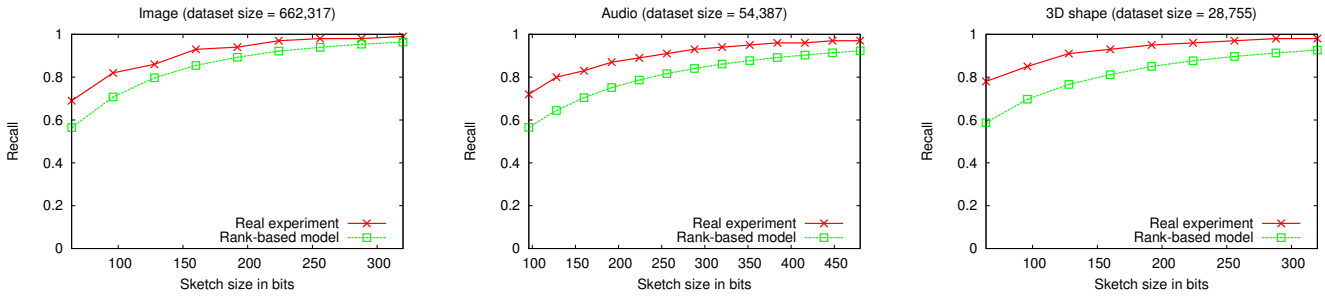

Figure 4: Filter Quality vs Sketch Size (H = 3): Model gives conservative estimate, and the estimate is closer when recall value is high

5.2 Sizing Sketches

The goal of the analytical model is to help systems designers choose design parameters properly. The following reports our experimental results to see how well the model can help systems designers choose sketch size B and the related parameter XOR Bits H properly. In our experiments, we set the result set size k to be 100 and filter ratio t to be 10.

Choosing sketch size B. Sketch size in bits B is perhaps the most crucial parameter in a sketch-based filtering mechanism for similarity search. Finding a good sketch size for a real system with a given expected dataset size will deliver high filter quality with minimal storage requirement.

Figure 4 shows the trend of filtering quality for different sketch sizes. The recall at a filter ratio of 10 is used to compare the experimental result with our analytical model. As expected, using more bits in a sketch leads to higher filtering quality.

Suppose we wish to build a system that achieves a recall of 0.9. Our results show that the sketch sizes must be about 160 bits, 256 bits and 128 bits for the image, audio, and 3D shape datasets, respectively. The amount of storage required for sketch storage are substantially smaller than the feature vector storage requirement. We use these sketch sizes in the following experiments.

Notice that our analytical model conservatively predicts the average recall in all cases and the predicted trend is consistent with the experimental results. This implies that the model is appropriate for conservative estimates for real systems designs. We will discuss the reasons for the consistent underestimation in Section 5.4.

Choosing XOR Bits H. Choosing the H judiciously is important because a good choice of H may give better filtering quality without increasing the sketch size and thus the storage requirement. Although the best H value is data type dependent, our analytical model can help choose the value without experimenting with full original dataset. We found that the best H value is relatively stable when the sketch size B changes, so in practice we will choose the best H value first.

Figure 5 shows that our analytical model predicts similar H values to the experimental results. For the image dataset, both predicted and experimental results indicate the the best H value is 3. For audio dataset, the model predicts that the best H value is 3 and the experimental results show that the best is 2. For 3D shape data both indicate that the best H value is 4.

5.3 Extrapolating to Larger Dataset Size

When building a real system, it is common not to have the full dataset available at the initial deployment. It is important to be able to choose system parameters with only a small sample dataset, and have some performance and quality guarantees as the dataset size grows. Our analytical model is useful in this scenario since the system designer cannot conduct full-scale experiments to figure out the parameters to be used.

In order to validate our model's prediction, we conducted an experiment that simulates dataset growth. For each dataset, we used a small subset of the full dataset to configure our model: that is, we only use one tenth of the total data objects to model the distance distributions and then use the model parameters derived from small dataset to predict the filter quality when dataset size grows. The result is compared with the experimental results, where more and more data points in the dataset are included in each experiment to simulate the growing dataset.

Figure 6 shows the filtering quality with different dataset sizes. In each plot, the first data point corresponds to the small sample dataset that we use to derive our model parameters; the following data points labeled as "rank-based model" are the projected results using the model. The experimental results are also shown in the same plot.

The results show that the filtering quality degrades gradually as the dataset grows larger. The model can give a good prediction on the degree of quality degradation as the dataset size grows. The prediction works better when the sample dataset size is reasonably large as seen in the image dataset. For other datasets, the degradation prediction is more conservative, but conservative estimates are more acceptable than optimistic in real systems designs.

5.4 Discussion

For all the figures showing the recall value, we noticed a consistent underestimate of the model result compared with the experimental result. In fact the underestimate of the model is largely due to the simplified independence assumption of the model – *i.e.* the assumption the sketch distances of objects are independent of the k-th nearest neighbor's sketch distance.

In section 3.3, we assumed that the sketch distances for

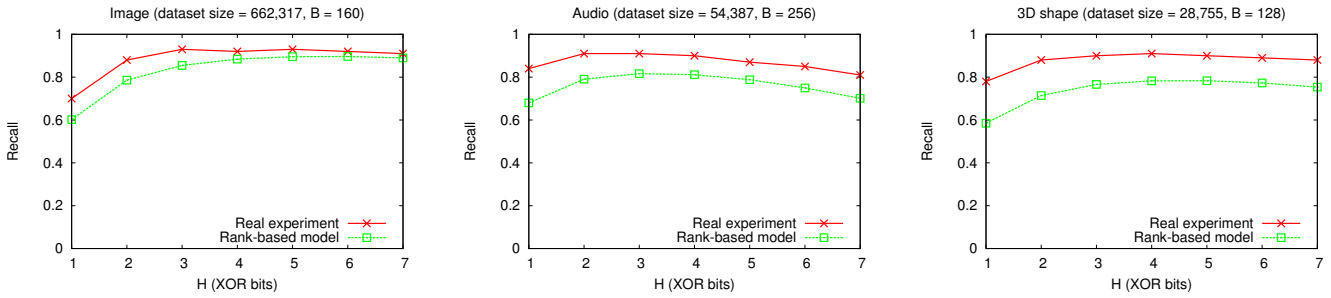

Figure 5: Filter Quality vs XOR Block Size H: Model shows same trend as real experiments, quality is good around H=3 and is not sensitive beyond that

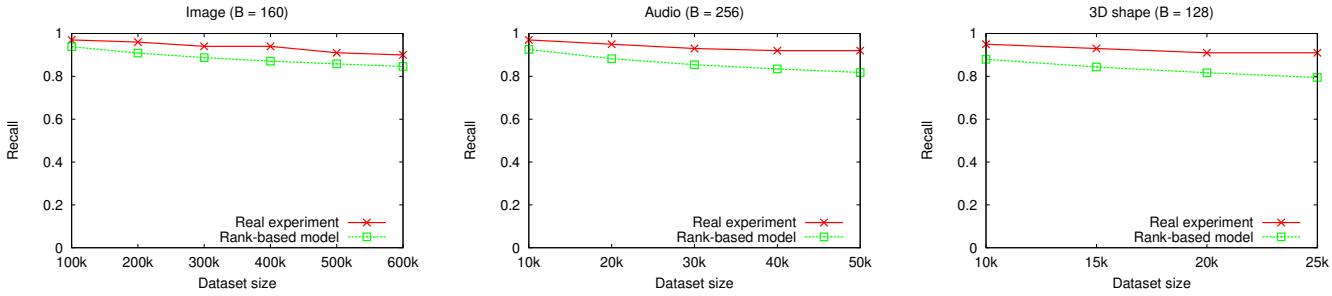

Figure 6: Filter Quality vs Dataset Size (H = 3): Model gives consistent and conservative prediction of quality as dataset grows

different objects $r \in S$ are independent This assumption simplifies the model, but in reality, there is some dependency that the model ignores. In fact, when r's sketch distance is large, the other sketch distances are also likely to be large and vice versa. In other words, there is a small positive correlation between sketch distances. In order to understand how such a correlation arises, it is instructive to consider the 1-dimensional case. Consider objects r and r' such that $d(q,r) \leq d(q,r')$. The algorithm to generate the bit vector sketch picks random thresholds for a dimension and checks if the coordinate in that dimension is above or below the threshold. The bit thus generated for q and r is different if the random threshold separates q and r. If this happens, it also likely to separate q and r'. If the sketch distance of r is large, then q and r must have been separated by several such randomly picked thresholds. But then it is likely that q and r' are also separated by these thresholds. Thus, the sketch distance of r' is likely to be large.

The positive correlation between sketch distances results in the rank of r_k being lower than that predicted by the independent model. In order to understand this, consider the extreme situation where the positive correlation is of the following form: for i randomly chosen in $[0, 100]$, each sketch distance is equal to the value at the i-th percentile of its individual distribution. In this case, objects with higher feature distance than the kth nearest neighbor r_k will never have their sketch distance lower than the sketch distance of r_k. The effect of positive correlation is similar, but less extreme than the situation described above. In other words, the positive correlation lowers the probability that objects further than the kth nearest neighbor will have their sketch distance lower than the sketch distance of r_k.

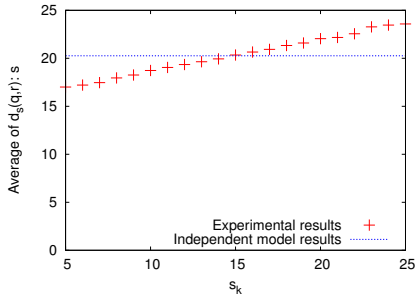

Figure 7: Dependence of s to s_k

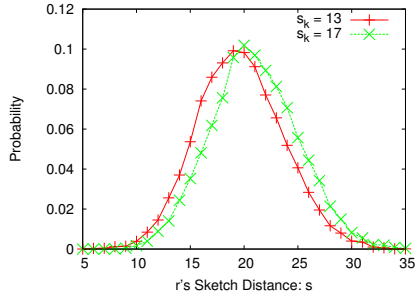

Figure 8: Probability Distribution of s with Different s_k

We conducted an experiment with the real dataset which clearly demonstrates such a dependence. In the experiment, we repeated the sketch construction 100,000 times and observed the relationship between sketch distance s of a particular data point r and the sketch distance s_k of the kth nearest neighbor r_k. Figure 7 shows the result. The points show the average value of s when s_k takes different values and the dashed line shows the constant s value expected with independent model. We can see that there is a small positive correlation between r's sketch distance s and r_k's sketch distance s_k. Figure 8 further shows the experimental result where the (empirically observed) probability distributions of r's sketch distance is plotted for two different values of s_k.

This data dependence affects the expected probability of r overtaking r_k. The experimental result shows that the probability of that particular data point r overtaking r_k is 0.125 while our independent model's prediction is 0.167 according to Equation 4. The higher rank prediction of r_k's sketch distance of our model will generate lower recall value in the quality score at filter ratio t and cause a consistent underestimate to the experimental results.

In order to accurately model the dependence of data object r's sketch distance r on r_k's sketch distance r_k, much more information about the data set is needed: value distributions on each dimension, data value dependence between different objects on each dimensions, *etc.* While this might give more accurate predictions, it is much harder to obtain reliable estimates of such fine grained information about the data set. Also it is unclear how well a model that incorporates such detailed information can be extrapolated to larger data set sizes. We have decided to adopt a simpler model in this paper that captures the essence of the experiment. Although our model gives a consistently low estimate of recall, it matches the general trend of the experimental results very well.

6. CASE STUDY

In this section, we use the 3D shape dataset as an example to illustrate how to use the rank-based analytical model to decide the parameters for sketching.

The sample dataset consists of 10,000 3D shape models, each represented by a 544-dimensional feature vector. The first step is to compute the feature vector distance distribution by randomly picking a number of query vectors, for instance 100 of them, and computing their ℓ_1 distances to the feature vectors in the sample dataset. As shown in Figure 1, the feature distance distribution of the sample dataset is very similar to that of the larger dataset.

Next, we use nonlinear least-squares curve fitting to find the parameters of the lognormal distribution that best approximate the feature distance distribution we have computed. The results are shown in Figure 2 and Figure 3. This gives us a closed form distance distribution which we can then use in the rank-based filtering model.

Next, using the analytical model of Section 3, we can compute the estimated filtering quality for a target dataset size – in this case 28,755 objects – using different sketch sizes B (Figure 4) and different XOR block size H (Figure 5).

Based on the estimations of the analytical model, we can then decide the sketching parameters. For example, suppose we want to design a system with a target recall value of around 0.8. Then the model predicts a sketch size of 128 bits and a XOR block size of 3 bits. Given that the analytical model's predictions are conservative, we know that a similarity search system using $B = 128$ and $H = 3$ will be able to achieve recall value greater than 0.8 in practice and the sketches are $544 \times 32/128 = 136$ times smaller than the original shape descriptors[3]. If we want to design a system with recall value around 0.9, we can also consult the model to pick a sketch size of 256 bits and XOR block size of 3 to achieve the desired quality. The sketches are $544 \times 32/256 = 68$ times smaller than the original shape descriptors.

7. RELATED WORK

Similarity search is typically formulated as a k-nearest neighbor (k-NN) search problem. The exact form of this problem suffers from the "curse of dimensionality" – either the search time or the search space is exponential in dimension d [8, 21]. Previous study [25] has shown that when the dimensionality exceeds around 10, space partition based k-NN search methods (e.g. R-tree [13], K-D tree [2], SR-tree [15]) perform worse than simple linear scan. As a result, researchers have instead focused on finding approximate nearest neighbors whose distances from the query point are at most $1 + \epsilon$ times the exact nearest neighbor's distance.

Recent theoretical and experimental studies have shown that sketches constructed based on random projections can effectively use Hamming distance to approximate ℓ_1 distance for several feature-rich datasets [5, 18, 20]. A recent work shows that such sketches can be used as an efficient filter to create candidate sets for content-based similarity search [19], which focused on efficient filtering methods of data objects each represented by one or multiple feature vectors, and not on the rank-based analytical model and experimental results.

Although recent theoretical and experimental research has made substantial progress on sketch constructions for building large-scale search engines and data analysis tools [10], not much work has been done on modeling sketches. Broder did an excellent analytical analysis for sketches constructed based on min-wise permutations for near-duplicate detection [4]. Since the application is for near-duplicate detection, his method is based on probabilistic analysis for random distribution of data.

Several approximation-based filtering techniques for k-NN search have been proposed in the literature. For example, the Vector Approximation file (VA-file) [25] method represents each vector by a compact, geometric approximation where each dimension is represented by l bits. Other approximation techniques such as A-tree method [23] and Active Vertice tree (AV-tree) method [1] were also proposed. Although experimental results using different approximation sizes were reported for these methods, no formal analysis on how to choose the approximation parameters were given.

Most previous work on content-based similarity search of feature-rich data has focused on segmentation and feature extraction methods. We are not aware of prior work on modeling the Lp distance distributions of feature-rich data such as image, audio and 3D-shape datasets. Previous work either assume uniform distribution of feature vectors in high dimensional spaces, or present end results using

[3]We assume each of the 544-dimensions of a shape descriptor is represented by a 32-bit floating point number.

the whole dataset. The notions of doubling dimension and intrinsic dimensionality (see [17, 3]) have been used previously to capture the inherent complexity of data sets from the point of view of several algorithmic problems including nearest neighbor search. However these notions do not provide a fine-grained model for distance distributions and do not have enough information to accurately estimate the performance of filtering algorithms for nearest neighbor search. By modeling distance distributions of a dataset, our analytical model can be easily adapted to different data types, and only a small sample dataset is needed for the analytical model to give good predictions for larger datasets.

8. CONCLUSIONS

This paper reports the results of modeling the parameters of using sketches to filter data for similarity search. The goal of our study is to help systems designers choose key design parameters such as sketch size. We validated our model with three feature-rich datasets including images, audio recordings, and 3D shape models. Our study shows two main results for sketches that use Hamming distance to approximate ℓ_1 norm distance:

- We have proposed a rank-based filtering model for the sketch construction to use Hamming distance to approximate ℓ_1 distance. We have shown, by experimenting with image, audio, and 3D shape datasets, that this model can conservatively predict the required sketch size for required recall, given the dataset size and its filtering candidate set size.

- Using the distance distribution with its parameters derived from a small sample dataset, we show that the rank-based filtering model can be used to perform good predictions of sketch sizes for a large dataset.

Our experimental studies show that the rank-based filtering model predicts results close to experimental results. Although there are noticeable gaps between the predicted and experimental results for certain systems parameters, the predicted trends are consistent with the experimental results. Furthermore, the predictions from the model are consistently conservative in all cases.

9. REFERENCES

[1] S. Balko, I. Schmitt, and G. Saake. The active vertice method: A performance filtering approach to high-dimensional indexing. *Elsevier Data and Knowledge Engineering (DKE)*, 51(3):369–397, 2004.

[2] J. L. Bentley. K-D trees for semi-dynamic point sets. In *Proc. of the 6th Annual ACM Symposium on Computational Geometry (SCG)*, pages 187–197, 1990.

[3] A. Beygelzimer, S. Kakade, and J. Langford. Cover trees for nearest neighbor. In *Proceedings of the 23rd International Conference on Machine Lea rning (ICML)*, pages 97–104, Pittsburgh, PA, USA, June 2006.

[4] A. Z. Broder. Identifying and filtering near-duplicate documents. In *Proceedings of the 11th Annual Symp. on Combinatorial Pattern Matching*, pages 1–10. Springer-Verlag, 2000.

[5] M. Charikar. Similarity estimation techniques from rounding algorithms. In *Proc. of the 34th Annual ACM Symp. on Theory of Computing*, pages 380–388, 2002.

[6] M. Datar, N. Immorlica, P. Indyk, and V. S. Mirrokni. Locality-sensitive hashing scheme based on p-stable distributions. In *Proceedings of the 20th Annual Symposium on Computational Geometry(SCG)*, pages 253–262, 2004.

[7] Y. Deng and B. S. Manjunath. Unsupervised segmentation of color-texture regions in images and video. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 2001.

[8] D. Dobkin and R. Lipton. Multidimensional search problems. *SIAM J. Computing*, 5:181–186, 1976.

[9] M. G. et al. Gnu scientific library.

[10] I. K. Fodor. A survey of dimension reduction techniques. Technical Report UCRL-ID-148494, Lawrence Livermore National Laboratory, 2002.

[11] J. S. Garofolo, L. F. Lamel, W. M. Fisher, J. G. Fiscus, D. S. Pallett, and N. L. Dahlgren. DARPA TIMIT acoustic-phonetic continuous speech corpus, 1993.

[12] A. Gionis, P. Indyk, and R. Motwani. Similarity search in high dimensions via hashing. In *Proc. of the 25th Int. Conf. on Very Large Data Bases (VLDB)*, pages 518–529, 1999.

[13] A. Guttman. R-trees: A dynamic index structure for spatial searching. In *Proc. of ACM SIGMOD Conf. on Management of Data*, pages 47–57, 1984.

[14] P. Indyk and R. Motwani. Approximate nearest neighbors: Towards removing the curse of dimensionality. In *Proc. of the 30th Annual ACM Symposium on Theory of Computing*, pages 604–613, 1998.

[15] N. Katayama and S. Satoh. The SR-tree: An index structure for high-dimensional nearest neighbor queries. In *Proc. of ACM SIGMOD Int. Conf. on Management of Data*, pages 369–380, 1997.

[16] M. Kazhdan, T. Funkhouser, and S. Rusinkiewicz. Rotation invariant spherical harmonic representation of 3D shape descriptors. In *Proc. of the Eurographics Symposium on Geometry Processing*, 2003.

[17] R. Krauthgamer and J. R. Lee. Navigating nets: Simple algorithms for proximity search. In *Proc. of the 15th ACM Symposium on Discrete Algorithms*, pages 798–807, 2004.

[18] Q. Lv, M. Charikar, and K. Li. Image similarity search with compact data structures. In *Proc. of the 13th ACM Conf. on Information and Knowledge Management*, pages 208–217, 2004.

[19] Q. Lv, W. Josephson, Z. Wang, M. Charikar, and K. Li. Efficient filtering with sketches in the ferret toolkit. In *Workshop on Multimedia Information Retrieval*, 2006.

[20] Q. Lv, W. Josephson, Z. Wang, M. Charikar, and K. Li. Ferret: A Toolkit for Content-Based Similarity Search of Feature-Rich Data. In *Proceedings of the ACM SIGOPS EuroSys Conf.*, 2006.

[21] S. Meiser. Point location in arrangements of hyperplanes. *Information and Computation*, 106(2):286–303, 1993.

[22] R. Panigrahy. Entropy based nearest neighbor search in high dimensions. In *Proceedings of the 17th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 1186–1195, Miami, Florida, USA, Jan 2006.

[23] Y. Sakurai, M. Yoshikawa, S. Uemura, and H. Kojima. The a-tree: An index structure for high-dimensional spaces using relative approximation. In *Proceedings of the 26th International Conference on Very Large Data Bases (VLDB)*, pages 516–526, 2000.

[24] G. Tzanetakis and P. Cook. *MARSYAS: A Framework for Audio Analysis*. Cambridge University Press, 2000.

[25] R. Weber, H.-J. Schek, and S. Blott. A quantitative analysis and performance study for similarity-search methods in high-dimensional spaces. In *Proceedings of the 24th International Conference on Very Large Data Bases (VLDB)*, pages 194–205, 1998.