RC 21740 (97931) 4 May 2000 ComputerScience/Mathematics

IBM Research Report

What is the Nearest Neighbor in High Dimensional Space

Alexander Hinneburg, Daneil A. Keim
Institute of Computer Science
University of Halle
Kurt-Mothes-Str.1
06120 Halle (Saale), Germany

Charu Aggarwal
IBM T. J. Watson Research Center
P. O. Box 218
Yorktown Heights, NY 10598

Research Division
Almaden - Austin - Beijing - Haifa - T. J. Watson - Tokyo - Zurich

What is the nearest neighbor in high dimensional spaces?

Alexander Hinneburg
Institute of Computer Science
University of Halle
Kurt-Mothes-Str.1
06120 Halle (Saale), Germany
hinneburg@informatik.uni-halle.de,

Charu Aggarwal
IBM T. J. Watson Research Center
Yorktown Heights, NY 10598
USA
charu@watson.ibm.com

Daniel A. Keim
Institute of Computer Science
University of Halle
Kurt-Mothes-Str.1
06120 Halle (Saale), Germany
keim@informatik.uni-halle.de

Abstract

Nearest neighbor search in high dimensional spaces is an interesting and important problem which is relevant for a wide variety of novel database applications. As recent results show, however, the problem is a very difficult one, not only with regards to the *performance issue* but also to the *quality issue*. In this paper, we discuss the *quality issue* and identify a new generalized notion of nearest neighbor search as the relevant problem in high dimensional space. In contrast to previous approaches, our new notion of nearest neighbor search does not treat all dimensions equally but uses a quality criterion to select relevant dimensions (projections) with respect to the given query. As an example for a useful quality criterion, we rate how well the data is clustered around the query point within the selected projection. We then propose an efficient and effective algorithm to solve the generalized nearest neighbor problem. Our experiments based on a number of real and synthetic data sets show that our new approach provides new insights into the nature of nearest neighbor search on high dimensional data.

Keywords: Nearest Neighbor Search, High-dimensional Data, Similarity Search

1 Introduction

Nearest neighbor search in high dimensional spaces is an interesting and important, but difficult problem. The traditional nearest neighbor problem of finding the nearest neighbor x_{NN} of a given query point $q \in \mathcal{R}^d$ in the database $D \subset \mathcal{R}^d$ is defined as

$$x_{NN} = \{x' \in D | \forall x \in D, x \neq x' : dist(x', q) \leq dist(x, q)\}. \quad (1)$$

Finding the closest matching object is important for a number of applications. Examples include similarity search in geometric databases [25, 22], multimedia databases [14, 30], and data mining applications such as fraud detection [18, 12], information retrieval [3, 28] among numerous other domains. Many of these domains contain applications in which the dimensionality of the representation is very high. For example, a typical feature extraction operation on an image will result in hundreds of dimensions.

Nearest neighbor problems are reasonably well solved for low dimensional applications for which efficient index structures have been proposed. Starting with the work on the R-Tree [16], a wide variety of multidimensional indexes have been proposed which work well for low dimensional data (see [15] for a comprehensive overview). These structures can support a wide range of queries such as point queries, range queries, or similarity queries to a predefined target. Many empirical studies have shown that traditional indexing methods fail in high dimensional spaces [11, 36, 10]. In such cases, almost the entire index is accessed by a single query. In fact, most indexes are handily beaten by the sequential scan [33] because of the simplicity of the latter.

However, as recent theoretical results [11] show, questions arise as to whether the problem is actually meaningful for a wide range of data distributions and distance functions. This is an even more fundamental problem, since it deals with the *quality issue* of nearest neighbor search, as opposed to the *performance issue*. If the nearest neighbor problem is not meaningful to begin with, then the importance of designing efficient data structures to do it is secondary. This paper is positioned to deal with the quality issue of nearest neighbor search, and examines several theoretical and practical aspects of performing nearest neighbor queries in high dimensional space.

There can be several reasons for the meaninglessness of nearest neighbor search in high dimensional space. One of it is the sparsity of the data objects in the space, which is unavoidable. Based on that observation it has been shown in [11] that in high dimensional space all pairs of points are almost equidistant from one another for a wide range of data distributions and distance functions. In such cases, a nearest neighbor query is said to be *unstable*. However, the proposition of [11] is not that the difference between the distance of the nearest and the farthest data point to a given query point approaches zero with increasing dimensionality, but they proved that this difference does not increase as fast as the distance from the query point to the nearest points when the dimensionality goes to infinity. It is still an open question whether and when nearest neighbor search in high dimensional spaces is meaningful. One objective of this paper is to qualify the results reported in [11].

It is useful to understand that high-dimensional nearest neighbor problems often arise in the context of data mining or other applications, in which the notion of similarity is not firmly pre-decided by the use of any particular distance function. Currently often used is an instance of the L_p metric ($p = 1$, manhatten; $p = 2$, euclidian) based on all dimensions. In this context, many interesting questions arise as to whether the current notion of NN search solves the right problem in high dimensions. If not, then what is the nearest neighbor in high dimensions? What is the meaning of

the distance metric used? One of the problems of the current notion of nearest neighbor search is that it tends to give equal treatment to all features (dimensions), which are however not of equal importance. Furthermore, the importance of a given dimension may not even be independent of the query point itself.

In this paper, we report some interesting experiments on the impact of different distance functions on the difference between the nearest and farthest neighbor. As we will see, our findings do not contradict the findings of [11] but provide interesting new insights. We discuss why the concept of nearest neighbor search in high dimensional feature spaces may fail to produce meaningful results. For that purpose, we classify the high dimensional data by their meaning. Based on our discussion and experiments, we introduce a new generalized notion of nearest neighbor search which does not treat all dimensions equally but uses a quality criterion to assess the importance of the dimensions with respect to a given query. We show that this generalized notion of nearest neighbor search, which we call *projected nearest neighbor search*, is the actually relevant one for a class of high dimensional data and develop an efficient and effective algorithm which solves the problem.

The projected nearest neighbor problem is a much more difficult problem than the traditional nearest neighbor problem because it needs to examine the proximity of the points in the database with respect to an a-priori unknown combination of dimensions. Interesting combinations of dimensions can be determined based on the inherent properties of the data and the query point which together provide some specific notion of locality. Note that the projected nearest neighbor problem is closely related to the problem of projected clustering [1, 2] which determines clusters in the database by examining points and dimensions which also define some specific notion of data locality.

2 Nearest Neighbor Search in high-dimensional Spaces

The results of [11] show that the relative contrast of the distances between the different points in the data set decreases with increasing dimensionality. In this section we first present some interesting theoretical and practical results which extend the results presented in [11]. The results are very interesting since – despite the pessimistic results of [11] – the results show that meaningful nearest-neighbor search in high dimensions may be possible under certain circumstances.

2.1 Theoretical Considerations

Let us first recall the important result discussed in Beyer et. al. [11] which shows that in high dimensions nearest neighbor queries become unstable. Let $Dmin_d$ be the distance of the query point to the nearest neighbor and $Dmax_d$ the distance of the query point to the farthest neighbor in d-dimensional space (see table 1 for formal definitions).

The theorem by Beyer et. al. states that under certain rather general preconditions the difference between the distances of the nearest and farthest points ($Dmax_d - Dmin_d$) does not increase with dimensionality as fast as $Dmin_d$. In other words, the ratio of $Dmax_d - Dmin_d$ to $Dmin_d$ converges to zero with increasing dimensionality. Using the definitions given in table 1, the theorem by Beyer et al. can be formally stated as follows.

d	Dimensionality of the data space
N	Number of data points
$\mathcal{F}$	1-dimensional data distribution in $(0,1)$
$\mu_{\mathcal{F}}$	Mean of $\mathcal{F}$
X_d	Data point from $\mathcal{F}^d$, each coordinate follows $\mathcal{F}$
$dist_d(\cdot,\cdot)$	Symmetrical distance function in $[0,1]^d$, with $dist_d(\cdot,\cdot) \geq 0$ and triangle inequality
$\|\cdot\|$	Distance of a vector to the origin $(0,\ldots,0)$
$Dmax_d = \max\{\|X_d\|\}$	maximum distance from the origin
$Dmin_d = \min\{\|X_d\|\}$	minimum distance from the origin
$P[e]$	Probability of event e
$E[X]$, $var[X]$	Expected value and variance of a random variable X
$Y_d \rightarrow_p c$	A sequences of vector $Y_1,\ldots$ converges in probability to a constant vector c if: $\forall \epsilon > 0\ lim_{d\rightarrow\infty} P[dist_d(Y_d,c) \leq \epsilon] = 1$

Table 1: Notations and Basic Definitions

Theorem 2.1 *If* $lim_{d\rightarrow\infty}\ var\left(\frac{\|X_d\|}{E[\|X_d\|]}\right) = 0$ *, then*

$$\frac{Dmax_d - Dmin_d}{Dmin_d} \rightarrow_p 0.$$

Proof: See [11]. ■

The theorem shows that in high dimensional space the difference of the distances of farthest and nearest points to some query point[1] does not increase as fast as the minimum of the two. This is obviously a problem since it indicates poor discrimination of the nearest and farthest points with respect to the query point.

It is interesting however to observe that the difference between nearest and farthest neighbor $(Dmax_d - Dmin_d)$ does not necessarily go to zero. In contrast, the development of $(Dmax_d - Dmin_d)$ with d largely depends on the distance metric used and may actually grow with the dimensionality for certain distance metrics. The following theorem summarizes this new insight and formally states the dependency between $(Dmax_d - Dmin_d)$ and the distance metric used. It allows to draw conclusions for specific metrics such as the Manhattan distance (L_1), Euclidean metric (L_2), and the general k-norm L_k.

Theorem 2.2 *Let* $\mathcal{F}$ *be an arbitrary distribution of two points and the distance function* $\|\cdot\|$ *be an* L_k *metric. Then,*

$$lim_{d\rightarrow\infty} E\left[\frac{Dmax_d^k - Dmin_d^k}{d^{1/k-1/2}}\right] = C_k,$$

where C_k *is some constant dependent on* k*.*

[1] For our theoretical considerations, we consistently use the origin as the query point. This choice does not affect the generality of our results, though it simplifies our algebra considerably.

Metric	$Dmax - Dmin$ converges against
L_1	$C_1 * \sqrt{(d)}$
L_2	C_2
$L_k, k \geq 3$	0

Table 2: Behavior of L_k norm

Proof: see Appendix. ■

We can easily generalize the result for a database of N uniformly distributed points. The following theorem provides the result.

Theorem 2.3 *Let $\mathcal{F}$ be an arbitrary distribution of n points and the distance function $\|\cdot\|$ be an L_k metric. Then,*

$$C_k \leq lim_{d\to\infty} E\left[\frac{Dmax_d^k - Dmin_d^k}{d^{1/k-1/2}}\right] \leq (n-1)\cdot C_k,$$

where C_k is some constant dependent on k.

Proof: If C is the expected difference between the maximum and minimum of two randomly drawn points, then the same value for n points drawn from the same distribution must be in the range $[C,(n-1)\cdot C]$. ■

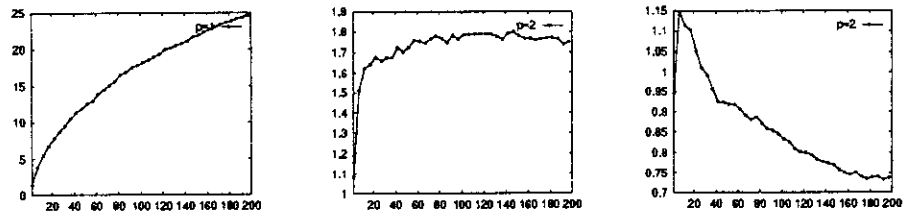
A surprising consequence of theorem 2.2 is that the value of $Dmax_d - Dmin_d$ grows (in absolute terms) as $d^{1/k-1/2}$. As a result, $Dmax_d - Dmin_d$ increases with dimensionality as $\sqrt{d}$ for the Manhattan metric (L_1 *metric*). The L_1 metric is the only metric for which the absolute difference between nearest and farthest neighbor increases with the dimensionality. It is also surprising that for the Euclidean metric (L_2 *metric*), $Dmax_d - Dmin_d$ converges to a constant, and for distance metrics L_k for $k \geq 3$, $Dmax_d - Dmin_d$ converges to zero with increasing d. These consequences of theorem 2.2 are summarized in table 2.

2.2 Experimental Confirmation

We performed a series of experiments to confirm these theoretical results. For the experiments we used synthetic (uniform and clustered) as well as real data sets. In figure 1, we show the average $Dmax - Dmin$ of a number of query points plotted over d for different data distributions. Note that the resulting curves depend on the number of data points in the data set.

Note that these experimental results are no contradiction to the results of [11]. The reason that even for the L_1 and L_2 metrics $\frac{Dmax_d - Dmin_d}{Dmin_d} \rightarrow_p 0$ is that $Dmin_d$ grows faster with d than $Dmax_d - Dmin_d$. In case of the L_1 metric, $Dmin_d$ grows linearly with d and in case of the L_2 metric, $Dmin_d$ grows as $\sqrt{d}$ with d. As a result, for the L_1 metric $lim_{d\to\infty}\frac{\sqrt{d}}{d} = 0$ and for the L_2 metric $lim_{d\to\infty}\frac{C_2}{\sqrt{d}} = 0$.

The theoretical and experimental results of this section show that for L_k metrics with $k \geq 3$, nearest neighbor search in high dimensional spaces is meaningless while for the L_1 and L_2 metrics the distances may reveal important properties of the data.



(a) Manhattan (L_1) (b) Euclid (L_2) (c) $L_3 Metric$

Figure 1: $|Dmax - Dmin|$ depending on d for different L_k metrics (uniform data)

3 Problems of high dimensional data and meaningful nearest neighbor

In one- or two-dimensional spaces, it is usually relatively easy to understand the properties of the data and identify the data distribution. It is safe to assume that all dimensions are equally relevant and that a standard (Euclidean) metrics provides meaningful results. In general, this is not true in the high-dimensional case.

To get a deeper understanding of the nature of high dimensional data, it is important to uncover the meaning of the dimensions. High dimensional data points or feature vectors are typically derived from complex real world objects like products, images, CAD data, etc. There are three main methods to derive a high dimensional feature vector from complex real world objects, namely

- enumerating some properties of the objects (irreversible transformation),
- determining histograms which describe some statistical properties of the objects (irreversible transformation) or
- transforming the full description of the objects into a feature vector (reversible transformation).

In the following, we examine the impact of the three potential sources of high dimensional data to the meaningfulness of the nearest neighbor problem.

1. Enumeration of Properties: We use an example in order to elucidate this case. For our example we assume that we want to compare cars. Comparing cars is often done by deriving various properties of the cars such as motor power, equipment, design and so on. Each measurement forms a dimension which is only related to the other measurements of the same object. When users query the car data base, they can select or weight the importance of the different properties, and in that way each user is able to form his own meaningful distance metric. The reason why a user can easily perform a meaningful nearest neighbor search is that the dimensions are directly interpretable by the user. By omitting some of the dimensions and by weighting them the user can control the degree of abstraction for the nearest neighbor search. In our experience, the dimensionality of such data is in the medium range (10 to 50).

2. Determination of Histograms: Histograms are often used to produce high dimensional data because they allow a flexible description of complex properties of real world objects. Examples are color histograms [34], word counts for document retrieval and text mining [23, 28] and census data [26]. Each bin of the histogram is taken as a single dimension. The information transformation

from the real world object into the histogram is an irreversible process which means that some information about the object is lost. The user of a histogram data base has to be aware of this. The goal of the query has to match the reduced information of the transformed object. On the other hand the histogram may contain information about aspects (for instance the background in an image) the user wants to abstract from. In that case, the information in the histogram must be reduced to the relevant portion. However, in contrast to the enumeration method the users are generally not able to specify the reduction because they usually do not know the underlying transformation. Another difference to the previous method is that it is not useful to group the dimensions independent from the users and the query points. In general all possible groupings are potentially meaningful. First approaches to deal with that problem of query specification are reported in [14, 31]. In general the connection between the information in the histograms and the semantic information of the objects is weak. The dimensionality of such data can vary from the medium to large range (10 to 1000).

3. Full Feature Description: The third method is to use the description of complex objects directly as a feature vector. The advantage is that all information about the object is stored in the feature vector and that the object is reconstructible from the vector. However, often the real world objects do not allow a representation as a feature vector with fixed length. Examples for data which allow such a representation are molecular biology data [13]. Like the histogram data, it is also not meaningful to group the dimensions to sensible units independently from the query point and/or the user. Due to the possibility of reconstruction, the semantic aspects are strongly connected to the information stored in the feature vectors.

The three types of high dimensional data relate to different aspects of *meaningfulness*. In general there is not a single meaningful nearest neighbor for a query, but the user has to select the desired aspects. For the first category of high dimensional data, the user is able to specify his/her notion of 'meaningfulness' (the actual relevant aspects) by his knowledge about the real world objects. To deal with the second and third types of data, the user needs help from the data creator or the database system to specify the 'meaningful' aspects. But how does a specification assistance for the relevant aspects may look like? For certain applications, there exist data dependent methods which use interaction in the selection process [14]. In this paper, we focus on a data independent method which selects the relevant dimensions automatically by extracting and rating additional information about the data distributions.

As a second question we investigate how far a single metric can serve as a similarity measure for the second and third type of data. We already stated that for those types of data the relevant dimensions (attributes) depend on the query point and the intention of the user. If the meaningfulness of a metric depends on the query point, than a metric can not serve as a measure of similarity between the query object and all other objects. In other words, a metric which is only based on the relevant attributes (which are assumed to be a subset of all attributes) can only serve as a criterion for similarity in a local environment of the query point. Objects (or data points) outside of this environment are incomparable to the query object, because they may have other relevant attributes. In summary one can say that for the second and third type of data, the relationship between the metric and the intended similarity measure becomes weaker with increasing distance to the query point. As a consequence, meaningful metrics for high dimensional data spaces have to be varied according to the considered query point and data objects under consideration. Our generalized notion of nearest neighbor search which is presented in the next section provides an automatic adaptation of the similarity measure in order to allow a meaningful nearest neighbor search in high dimensional space.

4 Generalized Nearest Neighbor Search

From the previous sections we have seen, that the problem of finding a meaningful nearest neighbor in high dimensional spaces consists of the following two steps: First, an appropriate metric has to be determined, and second, the nearest neighbor with respect to this metric has to be determined. The first step deals with selecting and weighting the relevant dimensions according to the users intention and the given query point. This step is obviously rather difficult since it is difficult to select and weight the relevant dimensions among hundreds of dimensions. The basic idea of our approach is to automatically determine the relevant dimensions for a given query point based on the properties of the data distribution. Although our approach can not guess the users intention, in general the data distribution contains highly relevant information and allows a much better and more meaningful nearest neighbor search.

4.1 Definition

In this section, we propose a generalization of the nearest neighbor search problem which remains meaningful in high-dimensional spaces. The basic idea of our new notion of nearest neighbor search is to use a quality criterion to dynamically determine which dimensions are relevant for a given query point and use those dimensions to determine the nearest neighbor[2]. The space of all subsets of dimensions can also be seen as the space of orthogonal projections of the data set, and the problem can therefore be defined as an optimization problem over the space of projections. In the following, we formalize our generalized notion of nearest neighbor search. First, we formally introduce a quality criterion which is used to rate the usefulness of a certain combination of dimensions (projection).

Let $D = \{x_1, \ldots, x_n\}$, $x \in \mathcal{R}^d$ be a database of d-dimensional feature vectors, $x_q \in \mathcal{R}^d$ the query point, $p : \mathcal{R}^d \rightarrow \mathcal{R}^{d'}$, $d' \leq d$ a projection, and $dist(\cdot, \cdot)$ a distance function in the projected feature space.

Definition 4.1 (Quality Criterion) *The* **quality criterion** *is a function $C(p, x_q, D, dist) \rightarrow \mathcal{R}$, $C \geq 0$ which rates the quality of the projection with respect to the query point, database, and distance function. In other words, the quality function rates the meaningfulness of the projection p for the nearest neighbor search.*

In section 4.3 we develop a useful quality criterion based on the distance distribution of the data points to the query point within a given projection.

Let P be the space of all possible projections $p : \mathcal{R}^d \rightarrow \mathcal{R}^{d'}$, $d' \leq d$ and $\forall x \in \mathcal{R}^d : p(p(x)) = p(x)$. To find a meaningful nearest neighbor for a given query point x_q we have to optimize the quality criterion C over the space of projections P.

Definition 4.2 (Generalized Nearest Neighbor Search) *A meaningful nearest neighbor for a*

[2]Note that the nearest neighbor determined by our approach might be different from the nearest neighbor based on all dimensions.

given query point $x_q \in \mathcal{R}^d$ *is the point*[3]

$$x_{NN} = \Big\{ x' \in D | \forall x \in D, x \neq x' : dist(p_{best}(x'), p_{best}(x_q)) \leq dist(p_{best}(x), p_{best}(x_q)) \Big\}$$

where $p_{best} = \Big\{ p \in P | MAX_{p:\mathcal{R}^d \to \mathcal{R}^{d'}, d' \leq d} \{ C(p, x_q, D, dist) \} \Big\}$.

Solving the generalized nearest neighbor problem problem is a difficult and computation intensive task. The space of all general projections P is infinite and even the space of all axes-parallel projections is exponential. In addition, the quality function C is a-priory unknown and therefore, it is difficult to find a general and efficiently computable solution of the problem. In the next section, we develop an algorithm which provides a very general solution of the problem.

4.2 Generalized Nearest Neighbor Algorithm

The most important but difficult task in solving the generalized nearest neighbor problem is to find the relevant projections. As mentioned in the previous subsections, this decision is in general query and data dependent which makes the problem computationally difficult. For our following considerations, we restrict the projections to the class of axes-parallel projections, which means that we are searching for meaningful combinations of dimensions (attributes). The restricted search space has still an exponential size with respect to dimensionality, which makes enumeration impossible for higher dimensionalities.

In order to keep our algorithm generic and allow different quality criterions (cf. 4.3), our first approach was to use general optimization algorithms such as random search, genetic and greedy optimization, for which the implementations can be made largely independent of the specific problem structure. In random search, simple random combinations of dimensions are evaluated in terms of the quality criterion, and the best projection is returned. The genetic algorithm uses multiple populations which are mutated and combined based on the quality criterion, and the greedy algorithm directly uses the best one-dimensional projections which are combined in higher-dimensional ones. All three algorithms are sketched in pseudo code in the appendix.

The results of these first experiments showed that none of the three algorithms was able to find the relevant subset of dimensions. Even for synthetic data, for which the relevant subset of dimensions is known, only a subset of the relevant dimensions was found. Random search was found only useful to check whether a given quality criterion is effective on a specific data set or not. If the random search does not find any projection with good quality, both genetic and greedy algorithm are likely to fail in finding a good projection as well. However, in cases when random search does not fail, the genetic search provides much better results. The greedy algorithm assumes that the influence of a dimension on the quality is independent from other dimensions. In general, this assumption is not true for real data sets. A crucial problem is that one-dimensional projections of high dimensional data usually do not contain much information and so the greedy algorithm would pick the first dimensions randomly and is therefore not useful for picking the first dimensions. It turned out,

[3]Note that our definition can be easily generalized to solve the k-nearest neighbor problem by fixing the selected projection and determining the k nearest neighbors.

p_nn_search $(x_q, d_{tar}, D, C, dist)$
d_{tmp} := 3 to 5
no_iter := 10 to 20
p_{tmp} := genetic_search($x_q, d_{tmp}, D, C, dist, no\_iter$)
p_{best} := greedy_search($x_q, d_{tar}, D, C, dist, p_{tmp}$)
x_{NN} := p_nn_search($x_q, D, dist, p_{best}$)
return (x_{NN})

Figure 2: Generalized Nearest Neighbor Algorithm

however, that the greedy algorithm can be used effectively to refine results from random or genetic search.

Our algorithm to determine the relevant subset of dimensions is therefore based on a combination of the genetic and the greedy algorithm. For determining the first three to five dimensions, we use a genetic algorithm and for extending the result to more dimensions we use a directed greedy-based search. Figure 2 shows the pseudocode of the algorithm. For controlling the degree of abstraction and improving the efficiency, we use the target dimensionality $d' = d_{tar} \leq d$ as a parameter of the algorithm. If the genetic algorithm determines the first five of the relevant dimensions and the greedy algorithm the remaining ones, the complexity of our algorithm is

$O((5 \cdot \#(Iterations) \cdot PopulationSize + d \cdot (d_{tar} - 5)) \cdot O(\text{Quality Determination}))$.

4.3 Distance Distributions

In this section we develop a quality measure based on the distance distribution with respect to the query point. The distance distribution of a data set D with respect to a query point x_q is the distribution of distances of the data points $x \in D$ from x_q. More formally, we have to consider the probability that the distance of a query point x_q to another data point is smaller that a threshold $dist_t$:

$$\Phi(dist_t) = P[dist(x_q, x) < dist_t], x \in D, dist_t \in \mathcal{R}$$

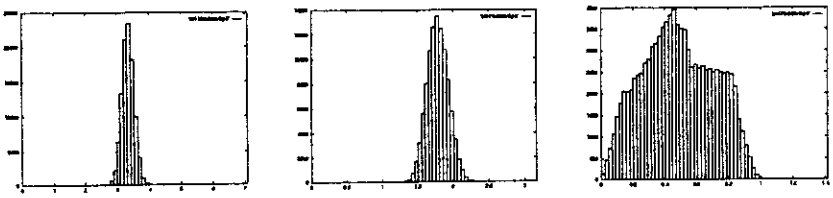
The corresponding probability density is

$$f(dist_t) = \Phi'(dist_t).$$

Note that $\Phi(dist_t)$ is not continuous and therefore we can only estimate the probability density $f(dist_t)$. In this subsection, we use simple histograms showing the distances of the data points from random query points.

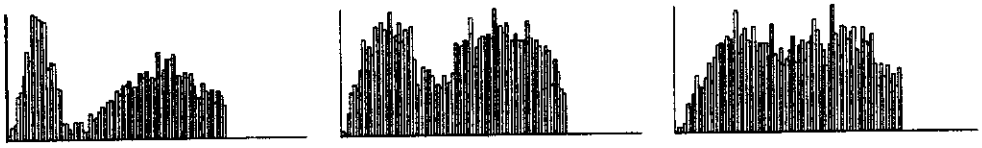
To examine how typical distance distributions look like, we examine the distance distribution in different dimensionalities. Let us first consider the case of high-dimensional uniform data. We know that in this case the distances are meaningless. Figure 3 shows typical distance distributions[4] of a 50-dimensional data set consisting of 100,000 data points uniformly distributed in $[0, 1]^d$. Figure 3 (a)-(c) show typical projections[5] onto randomly chosen 50, 10, and 2 dimensions. The distance distribution has always one peak which means that all data points are basically in one big

[4] In case of uniform data, the distance distribution is always similar independent of the chosen query point.
[5] In case of uniform data, the distance distribution always looks the same independent of the chosen projection.



(a) 50 Dimensions (b) 10 Dimensions (c) 2 Dimensions

Figure 3: Distance Distribution of Uniform Data



(a) all Dimensions (b) one non-rel. Dim. (c) two non-rel. Dim.

Figure 4: Distance Distribution of Data

distance cluster from the query point. As a consequence from the theorem in [11] the peak gets sharper as the distance to the query point grows. We neglect this effect for our quality criterion by estimating the density only in the range $[d_{min}, d_{max}]$, because this effect is common to mostly all distributions and from section 2 we conclude that this effect does not necessarily tell something about the meaningfulness of the nearest neighbor. From the discussion in section 3 we assume that a meaningful distance distribution should show two peaks. The nearer peak is formed by the points which are comparable to the query point (the metric is related to a type of similarity). The other peak – in most cases the larger one – is formed by those points which are incomparable to the query point because other attributes are relevant for those data objects. However, with respect to the currently used attributes they are assumed to behave like uniformly distributed data.

How to detect a two peak distance distribution? Our idea is to use kernel density estimation (see [35] for an introduction) to smooth the distribution and suppress random artifacts. To measure the quality we increase the kernel width (smoothing factor) until the smoothed distribution yields only two maxima. The obtained kernel width is h_1. Then we increase the kernel width further until the distance distribution yields only one maximum. This results in the kernel width h_2. We use the difference between the smoothing factor for one maximum and for two maxima $h_2 - h1$ as our quality criterion to measures the similarity of a current distance distribution with a distance distribution that yields two significant peaks. To get rid of possible disturbances in the distribution, which may also result in two maxima, we use only the k nearest percent of the data.

Figure 4 shows distance distributions of data, which contains full uniformly distributed data and a projected cluster, which means that these points follow a Gaussian distribution in some dimensions and a uniform distribution in the others. Figure 4(a) shows the distance distribution in a projection where all dimensions are relevant, which means that all selected dimensions are used in the definition of the projected cluster. In Figure 4(b), one relevant dimension is replaced by a non-relevant and in Figure 4(c) two relevant dimensions are replaced by non-relevant ones. In 4(c) the two peak structure is hard to recognize and the quality criterion gives no hint on the hidden relevant dimensions. From these observations we can conclude that the genetic algorithm can

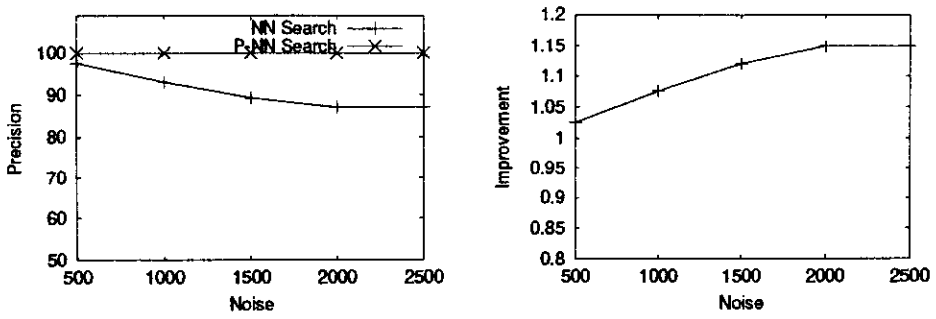


Figure 5: Generalized Nearest Neighbor Classification (Synthetic Data)

only optimize projections with a dimensionality of 3-5. If the dimensionality is higher the quality criterion degenerates to an oracle and the algorithm can only guess a good projection – and the probability to guess a good projection in high dimensional data is rather low.

5 Experiments

In this section we report experiments, to show the effectiveness of our quality function and the generalized notion of nearest neighbor search. Note that in real world application the quality function have to be modified due to the data dependency of the term 'meaningful'. In our experiments we focused on improving the effectiveness of the nearest neighbor search in general and omitted as far as possible dependencies of the quality function from the data.

First we compared the effectiveness of the generalized k-nearest neighbor search with the full k-nearest neighbor search. For this purpose used synthetic labeled data, consisting of two types of data. The first and relevant part follows a normal distribution in some of the dimensions, but are uniformly distributed with respect to the other dimensions. The second not relevant part is uniformly distributed in the whole feature space. In the experiments with the synthetic data we used only query points from the first part. For the effectiveness we measured the percentage of relevant data in the result of a k-nearest neighbor search(Precision). For all experiment we set $k = 20$.

Figure 5 shows the results for the comparison of the generalized nearest neighbor search with the full nearest neighbor search. The data sets consist of a projected cluster of 200 relevant points (normal distributed in 7 of 30 dimensions) and an amount of 500 up to 2500 not relevant points (uniformly distributed). The improvement over the full nearest neighbor search is up to 14%.

We applied our method to some real data sets from the UCI Machine Learning Repository. [6] We used the Ionosphere Database and the Spambase Database. The Ionosphere Database consist of 351 instances with 34 numeric attributes and contains 2 classes, which come from a classification of radar returns from the ionosphere. The Spambase Database come from a collection of spam and non-spam e-mails and consists of 4601 instances with 57 numeric attributes. In both cases we used for the generalized nearest neighbor a target dimensionality of $d_{tar} = 10$. The results are averages over 20 randomly selected queries. Our generalized nearest neighbor search shows (figure 6) an improvement up to 27%, which is significantly for a classification application.

[6] *www.ics.uci.edu/mlearn/.*

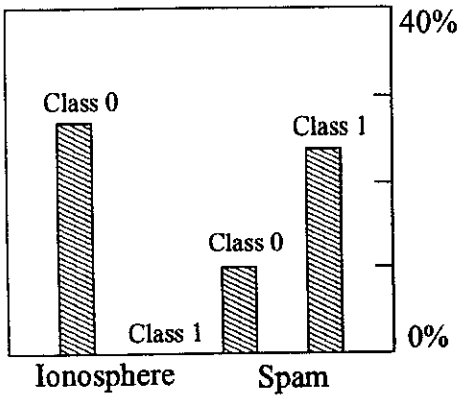


Figure 6: Improvement (Real Data)

Database	Class	NN Search	P-NN Search	Improvement
Ionosphere	0	0.52%	0.66%	27%
	1	0.95%	0.94%	0%
Spam	0	0.77%	0.85%	10%
	1	0.64%	0.79%	23%

Table 3: Generalized Nearest Neighbor Classification (Real Data)

To adopt our generalized nearest neighbor search to other applications like image retrieval or document search we suggest to use a fast k-nearest neighbor search on all dimensions with large k or a key word search as a filter step.

To show the applicability of our method we show (Figure 7) the search time depending on the number of data points. In our implementation we did not use any index structure, but used a simple linear scan to calculate our quality function and the query results. The experiments were measured on a Pentium III, 500 MHz with 200 MB RAM.

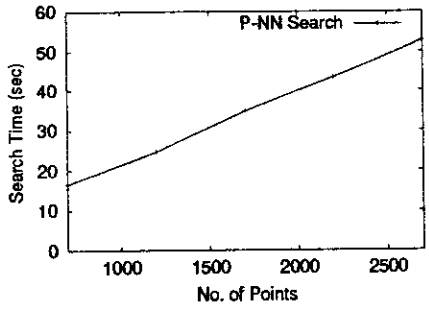


Figure 7: Search Time (Synthetic Data)

6 Conclusion

In this paper, we developed a generalized notion of nearest neighbor search in high dimensional spaces. We show that our new notion is highly relevant in practical applications and improves the *effectiveness* of the search. The basic idea is to determine a relevant subset of dimensions depending on the query point and the data distribution by an optimization process which rates the distance distribution for the selected subset of dimensions according to an elaborate quality criterion. We also discussed some interesting aspects of using different L_p-distance metrics for finding the nearest neighbor. Our new technique for solving the generalized nearest neighbor problem is not only valuable for allowing a more meaningful and effective nearest neighbor search in high dimensional spaces but it also provides a better understanding of the data and the relevant notion of proximity. The experimental results show the high potential of our new technique which is likely to extent the common full-dimensional nearest neighbor search in most applications that deal with high dimensional data.

References

[1] Aggarwal C. C. et. al. Fast Algorithms for Projected Clustering. *Proceedings of the ACM SIGMOD Conference*, 1999.

[2] Aggarwal C. C., Yu P. S. Finding Generalized Projected Clusters in High Dimensional Spaces. *Proceedings of the ACM SIGMOD Conference*, 2000.

[3] Altschul S. F., Gish W., Miller W., Myers E. W., Lipman D. J.: *A Basic Local Alignment Search Tool*, Journal of Molecular Biology, Vol. 215, No. 3, 1990, pp. 403-410.

[4] Arya S. Nearest Neighbor Searching and Applications. Ph. D. Thesis, University of Maryland, College Park, MD, 1995.

[5] Beckman, N., Kriegel, H., Schneider, R., Seeger, B. The R*-Tree An Efficient and Robust Method for Points and Rectangles. *Proceedings of the ACM SIGMOD Conference.* 322–331, 1990.

[6] Bennett K. P., Fayyad U., Geiger D. Density-Based Indexing for Approximate Nearest Neighbor Queries. *Proceedings of the ACM SIGKDD Conference*, pages 233-243, 1999.

[7] Berchtold S., Böhm C., Kriegel H.-P. The Pyramid Technique: Towards Breaking the Curse of Dimensionality. *Proceedings of the ACM SIGMOD Conference on Management of Data.* pages 142–153, June 1998.

[8] Berchtold S., Böhm C., Keim D., Kriegel H.-P. A Cost Model for Nearest Neighbor Search in High Dimensional Space. *Proceedings of the ACM PODS Conference*, 1997.

[9] Berchtold S., Ertl B., Keim D., Kriegel H.-P., Seidl T. Fast Nearest Neighbor Search in High Dimensional Spaces. *Proceedings of the 14th International Conference on Data Engineering*, Orlando, 1998.

[10] S. Berchtold, D. A. Keim, H.-P. Kriegel: *The X-Tree: An Index Structure for High-Dimensional Data*, Proc. Int. Conf. on Very Large Databases (VLDB'96), Bombay, India, 1996, pp. 28-39.

[11] Beyer K., Goldstein J., Ramakrishnan R., Shaft U. When is Nearest Neighbors Meaningful? *Proceedings of the ICDT*, 1999.

[12] F. Bonchi, Fosca Giannotti, Gianni Mainetto, Dino Pedreschi: *Using Data Mining Techniques in Fiscal Fraud Detection*, First Int. Conf. on Data Warehousing and Knowledge Discovery, 1999, pp. 369-376.

[13] X. Daura, B. Jaun, D. Seebach, W. F. van Gunsteren, A. E. Mark.: Reversible peptide folding in solution by molecular dynamics simulation, *Journal of Molecular Biology* 280, 1998, pp. 925-932.

[14] Faloutsos C., Barber R., Flickner M., Hafner J., et al.: *Efficient and Effective Querying by Image Content*, Journal of Intelligent Information Systems, 1994, Vol. 3, pp. 231-262.

[15] Gaede V., Gnther O.: *Multidimensional Access Methods*, ACM Computing Surveys, Vol. 30, No. 2, 1998, pp. 170-231.

[16] Guttman, A. R-Trees: A Dynamic Index Structure for Spatial Searching. *Proceedings of the ACM SIGMOD Conference*, 47–57, 1984.

[17] Hinrichs, K., Nievergelt, J. The Grid File: A Data Structure to Support Proximity Queries on Spatial Objects. *Proceedings of the WG'83*. 100–113, 1983.

[18] Hongxing He, Warwick Graco, Xin Yao: *Application of Genetic Algorithm and k-Nearest Neighbour Method in Medical Fraud Detection*, Asia-Pacific Conference on Simulated Evolution and Learning, SEAL'98, 1998, pp. 74-81.

[19] Jain R., White D. A. Similarity Indexing: Algorithms and Performance. *Proceedings of SPIE Storage and Retrieval for Image and Video Databases IV*, Vol. 2670, San Jose, CA, pages 62–75, 1996.

[20] Kamel I., Faloutsos C. Hilbert R-Tree: An improved R-Tree Using Fractals. *Proceedings of the Very Large Databases*, pages 500–509, 1994.

[21] Katayama N., Satoh S. The SR-Tree: An Index Structure for High Dimensional Nearest Neighbor Queries. *Proceedings of the ACM SIGMOD International Conference on Management of Data*. pages 369–380, 1997.

[22] Korn F., Sidiropoulos N., Faloutsos C., Siegel E., Protopapas Z.: *Fast Nearest Neighbor Search in Medical Image Databases*, Proc. 22nd Int. Conf. on Very Large Data Bases, Mumbai, India, pp.215-226, 1996.

[23] Kukich K. Techniques for Automatically Correcting Words in Text, *ACM Computing Surveys*, Vol.24, No. 4, 1992, pp.377-440.

[24] Lin K.-I., Jagadish H. V., Faloutsos C. The TV-tree: An Index Structure for High Dimensional Data. *VLDB Journal*, Volume 3, Number 4, pages 517–542, 1992.

[25] Mehrotra R., Gary J.: *Feature-Index-Based Similar Shape Retrieval*, Proc. of the 3rd Working Conf. on Visual Database Systems, March 1995.

[26] Openshaw S.: Census User Handbook, *Pearson Professsional Ltd.*, Cambridge, 1995.

[27] Roussopoulos N., Kelley S., Vincent F. Nearest Neighbor Queries. *Proceedings of the ACM-SIGMOD International Conference on Management of Data*, pages 71–79, 1995.

[28] Salton G. Automatic Text Processing: The Transformation, Analysis, and Retrieval of Information by Computer. *Addison-Wesley Publishing Company.*

[29] Seidl T., Kriegel H.-P. Optimal Multi-Step k-Nearest Neighbor Search. *Proceedings of the ACM SIGMOD Conference on Management of Data.* pages 154–165, 1998.

[30] Seidl T., Kriegel H.-P. Efficient User-Adaptable Similarity Search in Large Multimedia Databases. *Proceedings of the 23rd International Conference on Very Large Databases.* Athens, Greece, 1997.

[31] Seidl TKDE98 - flexible specification of histogram-based search

[32] Sellis T., Roussopoulos N., Faloutsos C. The R+ Tree: A Dynamic Index for Multidimensional Objects. *Proceedings of the 13th International Conference on Very Large Databases,* pages 507–518, 1987.

[33] Shaft U., Goldstein J., Beyer K. Nearest Neighbor Query Performance for Unstable Distributions. Technical Report TR 1388, Department of Computer Science, University of Wisconsin at Madison.

[34] Shawney H., Hafner J. Efficient Color Histogram Indexing, *Proc. Int. Conf. on Image Processing*, 1994, pp. 66-70.

[35] Silverman B.W.: *'Density Estimation'*, Chapman & Hall 1986.

[36] Roger Weber, Hans-Jrg Schek, Stephen Blott: *A Quantitative Analysis and Performance Study for Similarity-Search Methods in High-Dimensional Spaces*, VLDB 1998, pp. 194-205.

[37] White D. A., Jain R. Similarity Indexing with the SS-Tree, *Proceedings of the 12th International Conference on Data Engineering,* New Orleans, USA, pages 516–523, February 1996.

Appendix

Theorem 2
Let $\mathcal{F}$ be an arbitrary distribution of two points and the distance function $\|\cdot\|$ be an L_k metric. Then,

$$lim_{d\to\infty} E\left[\frac{Dmax_d^k - Dmin_d^k}{d^{1/k-1/2}}\right] = C_k,$$

where C_k is some constant dependent on k.

Proof: Let A_d and B_d be the two points in a d dimensional data distribution such that each coordinate is independently drawn from the data distribution $\mathcal{F}$. Specifically $A_d = (P_1 \ldots P_d)$ and $B_d = (Q_1 \ldots Q_d)$ with P_i and Q_i being drawn from $\mathcal{F}$. Let $PA_d = \{\sum_{i=1}^d (P_i)^k\}^{1/k}$ be the distance of A_d to the origin using the L_k metric and $PB_d = \{\sum_{i=1}^d (Q_i)^k\}^{1/k}$ the distance of B_d. The difference of distances is $PA_d - PB_d = \{\sum_{i=1}^d (P_i)^k\}^{1/k} - \{\sum_{i=1}^d (Q_i)^k\}^{1/k}$.

We assume that the kth power of a random variable drawn from the distribution $\mathcal{F}$ has mean $\mu_{\mathcal{F},k}$ and standard deviation $\sigma_{\mathcal{F},k}$. This means that: $\frac{PA_d^k}{d} \to_p \mu_{\mathcal{F},k}$, $\frac{PB_d^k}{d} \to_p \mu_{\mathcal{F},k}$ and therefore

$$\frac{PA_d}{d^{1/k}} \to_p (\mu_{\mathcal{F},k})^{1/k}, \quad \frac{PB_d}{d^{1/k}} \to_p (\mu_{\mathcal{F},k})^{1/k}. \tag{2}$$

We intend to show that $\frac{|PA_d - PB_d|}{d^{1/k-1/2}} \to_p C_k$ for some constant C_k depending on k.

We express $|PA_d - PB_d|$ in the following numerator/denominator form which we will use in order to examine the convergence behavior of the numerator and denominator individually.

$$|PA_d - PB_d| = \frac{|(PA_d)^k - (PB_d)^k|}{\sum_{r=0}^{k-1} (PA_d)^{k-r-1}(PB_d)^r} \tag{3}$$

Dividing both sides by $d^{1/k-1/2}$ and regrouping on right-hand-side we get

$$\frac{|PA_d - PB_d|}{d^{1/k-1/2}} = \frac{|(PA_d)^k - (PB_d)^k|/\sqrt{d}}{\sum_{r=0}^{k-1} \left(\frac{PA_d}{d^{1/k}}\right)^{k-r-1} \left(\frac{PB_d}{d^{1/k}}\right)^r} \tag{4}$$

Consequently, using Slutsky's theorem and the results of Equation 2 we have:

$$\sum_{r=0}^{k-1} \left(\frac{PA_d}{d^{1/k}}\right)^{k-r-1} \cdot \left(\frac{PB_d}{d^{1/k}}\right)^r \to_p k \cdot (\mu_{\mathcal{F},k})^{(k-1)/k} \tag{5}$$

Having characterized the convergence behavior of the denominator of the right-hand-side of Equation 4, let us now examine the behavior of the numerator:

$$|(PA_d)^k - (PB_d)^k|/\sqrt{d} = |\sum_{i=1}^d ((P_i)^k - (Q_i)^k)|/\sqrt{d} = |\sum_{i=1}^d R_i|/\sqrt{d}.$$

Here R_i is the new random variable defined by $((P_i)^k - (Q_i)^k)\ \forall i \in \{1, \ldots d\}$. This random variable has zero mean and standard deviation which is $\sqrt{2} \cdot \sigma'$ where σ' is the standard deviation of $(P_i)^k$.

The the sum of different values of R_i over d dimensions will converge to a normal distribution with mean 0 and standard deviation $\sqrt{2} \cdot \sigma' \cdot \sqrt{d}$ because of the central limit theorem. Consequently, the mean average deviation of this distribution will be $C \cdot \sigma'$ for some constant C. Therefore, we have:

$$\lim_{d \to \infty} E\left[\frac{|(PA_d)^k - (PB_d)^k|}{\sqrt{d}}\right] = C \cdot \sigma_{\mathcal{F},k} \tag{6}$$

Since the denominator of Equation 4 shows probabilistic convergence, we can combine the results of Equations 5 and 6 to obtain:

$$\lim_{d \to \infty} E\left[\frac{|PA_d - PB_d|}{d^{1/k-1/2}}\right] = C \cdot \frac{\sigma_{\mathcal{F},k}}{k \cdot \mu_{\mathcal{F},k}^{(k-1)/k}} \tag{7}$$

The result follows. ■

Optimization Algorithms

random_search $(x_q, d_{tar}, D, C, dist, no\_iter)$
$p_{best}.quality := 0$
for $i := 0$ **to** no_iter **do**
  $p :=$ generate_random_projection(d_{tar})
  $p.quality := C(p, x_q, D, dist)$
  if $p_{best}.quality < p.quality$ **then**
    $p_{best} := p$
return (p_{best})

genetic_search $(x_q, d_{tar}, D, C, dist, no\_iter)$
$population := \emptyset$, $pop\_size = 100$, $elite := 10$, $child := 80$
for $i := 0$ **to** pop_size **do**
  $p :=$ generate_random_projection(d_{tar})
  $p.quality := C(p, x_q, D, dist)$
  $population.insert(p)$
for $i := 0$ **to** no_iter **do**
  $new\_pop := \emptyset$
  insert the $elite$ best projection into new_pop
  for $j :=$ $elite$ **to** $elite + child$ **do**
    $parent1 :=$ select a projection from old_pop, prefer high quality
    $parent2 :=$ select a projection from old_pop, prefer high quality
    $child :=$ generate a new projection by combining $parent1$ and $parent2$
    $child.quality := C(p, x_q, D, dist)$
    new_pop.insert($child$)
  qualify and insert $pop\_size - (elite + child)$ random projections into new_pop
  $population := new\_pop$
select the best projection p_{best} and return it

greedy_search $(x_q, d_{tar}, D, C, dist, p_{tmp})$
set of selected dimensions $S := \emptyset$ or from p_{tmp}

for $i := 0$ **to** dim_{tar} **do**
  pick the dimension $k_i \notin S$ such that the quality of the projection
    based on $S \cup \{k_i\}$ is maximal
  $S := S \cup \{k_i\}$
return $(p_{best}(S))$