Similarity Image Retrieval with Significance-Sensitive Nearest-Neighbor Search

Norio Katayama and Shin'ichi Satoh

National Institute of Informatics
2-1-2 Hitotsubashi, Chiyoda-ku, Tokyo 101-8430, Japan
E-mail: {katayama,satoh}@nii.ac.jp
Phone: +81-3-4212-2570 Fax: +81-3-3556-1916

Abstract

Nearest-neighbor (NN) search in high dimensional space is widely used for the similarity retrieval of images. Recent research results in the literature reveal that NN-search might return insignificant NNs in high dimensional space because points could be so scattered that every distance between them might yield no significant difference. Insignificant NNs are troublesome with respect to the efficiency of the similarity retrieval. They have less meaning for users and also degrades the performance of NN-search. However, in real applications, we can expect that the data distribution is so skewed that we might have insignificant NNs in one region but could have significant ones in another. This implies that we could enhance the efficiency of the retrieval if we distinguish significant NNs from insignificant ones. Hence, we devised a way to estimate the significance of NNs based on the local intrinsic dimensionality. Then, with applying it, we developed a new NN-search algorithm: the significance-sensitive nearest-neighbor search. This algorithm not only enables us to distinguish more significant NNs from less significant ones but also enables us to cut down the search cost compared with the conventional NN-search algorithm. The experimental evaluation with a photo image database demonstrates the advantages of the proposed algorithm.

1 Introduction

Nearest-neighbor search (NN-search) in the feature space is widely used for the similarity retrieval of multimedia information. Each piece of multimedia information is mapped to a vector in a multi-dimensional space where the distance between two vectors (typically, Euclidean distance between the heads of vectors) corresponds to the similarity of multimedia information. These vectors and the space are called feature vectors and feature space respectively. Once the feature space is obtained, the similarity retrieval of multimedia information is reduced to NN-search in the feature space. One of the typical examples of the feature space is the color histogram of an image which indicates how much portion of an image has a specific color (e.g., red, blue, green, etc.). One of the important properties of the feature space is high dimensionality. It is very common to use 10 or higher dimensional space for the feature space. For example, colors are often classified into 16 or more colors for color histograms. This generates 16 or more dimensional feature vectors.

Recent results in the literature reveal that a curious problem happens in high dimensional space, which is not imagined in low dimensional space. Since high dimensional space has high degree of freedom, points could be so scattered that every distance between them might yield no significant difference. A typical example appears in the uniformly distributed points, i.e., points distributed uniformly in a unit hypercube. For example, Berchtold et al.[1] reported that most of the points are located near the surface of the cube, and Katayama et al.[2] reported that the distances among the points are very similar for any combination of them. Recently, Beyer et al.[3] showed that under a broad set of conditions (broader than uniform distribution) the distance to the nearest data point approaches the distance to the farthest data point as dimensionality increases.

This problem in high dimensional space, i.e., no significant difference in distances, is troublesome with respect to the efficiency of the similarity retrieval. Insignificant nearest neighbors have less meaning to users and also degrades the performance of NN-search since NN-search operations are forced to examine many points having similar distances. However, in real applications, we can expect that the data distribution is so skewed that we might have insignificant nearest neighbors in one region but could have significant ones in another. This implies that we could enhance the efficiency of the retrieval if we distinguish significant nearest neighbors from insignificant ones. Hence, in our previous paper [4], we proposed a new NN-search algorithm: the significance-sensitive nearest-neighbor search. This algorithm not only enables us to distinguish more significant nearest neighbors from less significant ones but also enables us to cut down the search cost compared with the conventional NN-search algorithm.

In this paper, we evaluate the impact of the signifi-

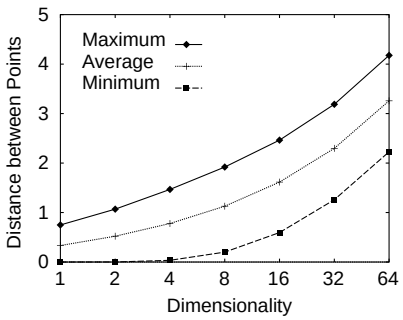


Figure 1: The maximum, the average, and the minimum of the distances between 100,000 points generated at random in a unit hypercube (cited from [2]).

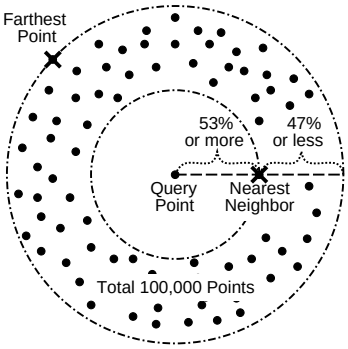


Figure 2: In the case of 64 dimensional space of Figure 1, the distance to the nearest neighbor is 53% or more of the distance to the farthest point.

cance-sensitive nearest-neighbor search with applying it to the similarity image retrieval under practical setting. The test database is Corel Photo Collection which is one of the popular photo collections and contains more than 60,000 photo images. By conducting similarity retrieval with choosing every image as a query, we evaluated the characteristics of search results from various viewpoints. The evaluation result clearly demonstrates the advantages of the significance-sensitive nearest-neighbor search in the similarity retrieval of images.

This paper is organized as follows. Section 2 shows how insignificant nearest neighbors affect the similarity retrieval of multimedia information. Section 3 describes how to estimate the significance of nearest neighbors. Section 4 describes the significance-sensitive NN-search algorithm briefly. In Section 5, we apply the significance-sensitive nearest-neighbor search to the similarity retrieval of photo images. Section 6 contains conclusions.

2 Insignificant NNs in Feature Spaces

2.1 Insignificance of Nearest Neighbors

When we use NN-search, we expect that found neighbors are much closer than the others. However, this intuition is sometimes incorrect in high dimensional space. For example, when points are uniformly distributed in a unit hypercube, the distance between two points is almost the same for any combination of two points. Figure 1 shows the minimum, the average, and the maximum of the distances among 100,000 points which are randomly generated in a unit hypercube. As shown in the figure, the minimum of the distances grows drastically as the dimensionality increases and the ratio of the minimum to the maximum increases up to 24% in 16 dimensions, 40% in 32 dimensions, and 53% in 64 dimensions. Thus, the distance to the nearest neighbor is only 53% of the distance to the farthest point in 64 dimensional space (Figure 2). In this case, we can consider the nearest neighbor to be insignificant, because the difference between the nearest neighbor and the others is negligible, i.e., the other points are as close to the query point as the nearest neighbor is. From the perspective of the similarity retrieval, when the nearest neighbor is insignificant, the nearest neighbor has almost the same similarity with the others and does not have significant similarity to the query point.

As Figure 1 shows, insignificant nearest neighbors are more likely to occur as dimensionality increases. This characteristics can be verified with estimating the distance to k-th nearest neighbor. When N points are distributed uniformly within the hypersphere whose center is the query point, the expected distance to k-th nearest neighbor d_{kNN} is obtained as follows[5]:

$$E\{d_{kNN}\} \approx \frac{\Gamma(k+1/n)}{\Gamma(k)} \frac{\Gamma(N+1)}{\Gamma(N+1+1/n)} r, \qquad (1)$$

where n is the dimensionality of the space and r is the radius of the hypersphere. Then, the ratio of the $(k+1)$-NN distance to the k-NN distance is obtained as follows[5]:

$$\frac{E\{d_{(k+1)NN}\}}{E\{d_{kNN}\}} \approx 1 + \frac{1}{kn}. \qquad (2)$$

Thus, when points are distributed uniformly around the query point, it is expected that the difference between the k-th and $(k+1)$-th nearest neighbors decreases as the dimensionality increases. This implies that insignificant nearest neighbors are more likely to occur in high dimensional space than in low dimensional space.

2.2 Harmful Influence of Insignificant NNs on Similarity Retrieval

Insignificant nearest neighbors have harmful influence on the similarity retrieval with the following respects:

- NN-search performance is degraded.

  When the nearest neighbor is insignificant, there exist many points that have almost the same similarity with the nearest neighbor. Since these points are very strong candidates for the nearest neighbor, NN-search operations are forced to examine many points before determining the true nearest neighbor. This degrades the performance of NN-search operations.

- Result has less meaning.

  When the nearest neighbor is insignificant, NN-search operations return the closest point among many strong candidates that have almost the same similarity with the nearest neighbor. This means that all of the candidates are either similar or dissimilar. Therefore, it is meaningless to choose the nearest neighbor from plenty of similar candidates.

These effects are extremely harmful to the retrieval systems with human-computer interaction. When the nearest neighbor is insignificant, the system forces users to wait until meaningless nearest neighbors are returned with very slow response. Thus, it is necessary to handle insignificant nearest neighbors appropriately in order to achieve efficient similarity retrieval of multimedia information.

3 Estimation of NN's Significance based on the Local Intrinsic Dimensionality

3.1 Definition of Insignificant Nearest Neighbors

As stated above, insignificant nearest neighbors are more likely to occur in high dimensional space and troublesome to the similarity retrieval systems. However, this never means that high dimensional feature space is useless to the similarity retrieval of multimedia information. In real applications, we can expect that the data distribution is so skewed that the intrinsic dimensionality (or effective dimensionality) should be smaller than the dimensionality of the feature space. For example, when the data distribution is governed by a number of dominant dimensions, the intrinsic dimensionality is given by the number of such dominant dimensions. In addition, the intrinsic dimensionality should not be consistent over the data set but vary from one local region to another. Therefore, we might have insignificant nearest neighbors in one region but could have significant ones in another. This implies that we could enhance the efficiency of the similarity retrieval if we can distinguish significant nearest neighbors from insignificant ones. With this aim, we devised a way to estimate the significance of nearest neighbors based on the local intrinsic dimensionality.

In the first place, we have coined a definition of insignificant nearest neighbors as follows:

Definition 1 *Let d_{NN} be the distance from the query point to a nearest neighbor. Then, the nearest neighbor is insignificant if more than or equal to N_c points exist within the range of d_{NN} to $R_p \times d_{NN}$ from the query point.*

Here, $R_p(>1)$ and $N_c(>1)$ are controllable parameters (Figure 3). R_p determines the proximity of the query point and N_c determines the congestion of the proximity. For example, we set 1.84 to R_p and 48 to N_c at the experiment described in Section 5. The way to find appropriate values for R_p and N_c will be discussed in the remainder of this section.

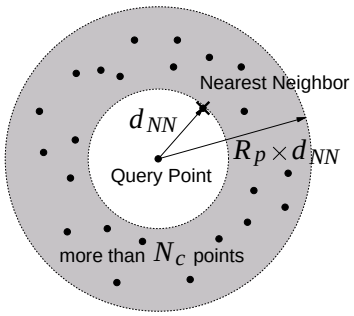


Figure 3: Definition of insignificant NNs.

3.2 Optimal Parameter Setting based on the Local Intrinsic Dimensionality

As shown above, the parameters R_p and N_c take an essential role in the definition of the insignificant nearest neighbors. We present here a way to find the optimal parameter setting based on the local intrinsic dimensionality. The parameter setting presented below consists of two steps. In the first step, the local intrinsic dimensionality is related to the insignificance of nearest neighbors, then two control-points, (ν_c, ρ_c) and (ν_r, ρ_r), are determined in order to control what local intrinsic dimensionality to be regarded as insignificant: ν_c is the cut-off dimensionality, ν_r the rejection dimensionality, ρ_c the rejection probability at ν_c, and ρ_r the rejection probability at ν_r. Then, in the second step, the parameters R_p and N_c are determined from these four parameters.

Firstly, we relate the insignificance of nearest neighbors with the local intrinsic dimensionality by Equation (2). It should be noted that this equation holds when the local intrinsic dimensionality is n (in other words, n in Equation (2) does not mean the dimensionality of the data space but means the intrinsic dimensionality)[5]. Equation (2) indicates that the ratio of $d_{(k+1)NN}$ to d_{kNN} decreases monotonically as k increases. Therefore, the maximum of the ratio between two nearest neighbors is expected as follows:

$$\max_k \frac{E\{d_{(k+1)NN}\}}{E\{d_{kNN}\}} = \frac{E\{d_{2NN}\}}{E\{d_{1NN}\}} \approx 1 + \frac{1}{n}. \qquad (3)$$

This equation shows that the maximum expected value of the relative difference between the k-th and the $(k+1)$-th nearest neighbor decreases monotonically as the local dimensionality increases (Figure 4). Only 20% or less difference is expected at the 5 dimensions, and 10% or less at 10 dimensions. Thus, the local dimensionality allows us to estimate the relative significance of nearest neighbors. We use this property to decide control-points, (ν_c, ρ_c) and (ν_r, ρ_r) as is shown below.

Secondly, we relate the local intrinsic dimensionality n with the parameters R_p and N_c. When points are distributed uniformly in a local region with the local intrin-

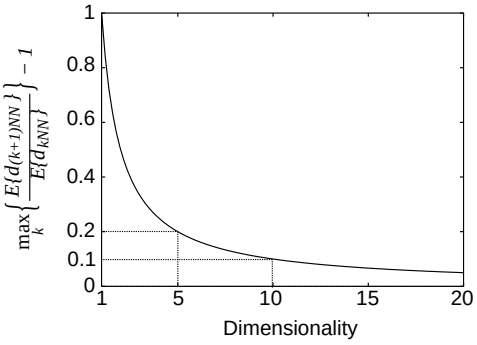


Figure 4: Maximum expected value of the relative difference between the k-th and the $(k+1)$-th nearest neighbor.

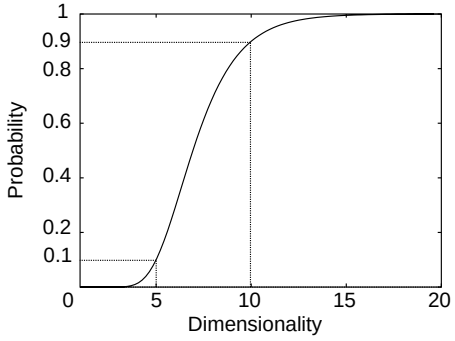


Figure 5: The rejection probability when R_p is 1.84471 and N_c is 48.0277.

sic dimensionality n, the probability that N_c points exist in the proximity specified by R_p is obtained as follows:

$$Pr\{N_c \text{ points in } R_p\} = (1 - (1/R_p)^n)^{N_c} \quad (4)$$

According to the proposed definition, a nearest neighbor is insignificant when N_c points exist in the proximity specified by R_p. Therefore, Equation 4 corresponds to the probability that a nearest neighbor is regarded as being insignificant when the local intrinsic dimensionality is n. Hence, we call this probability the *rejection probability*. The rejection probability increases monotonically as the local intrinsic dimensionality increases and can be controlled easily by two control-points as follows. Suppose that we want to set the rejection probability to ρ_1 at the dimensionality ν_1, and ρ_2 at the dimensionality ν_2 ($0 < \rho 1 < \rho 2 < 1$ and $1 < \nu_1 < \nu_2$). Then, we can determine the parameters R_p and N_c by solving the following simultaneous equations:

$$(1 - (1/R_p)^{\nu_1})^{N_c} = \rho_1 \quad (5)$$
$$(1 - (1/R_p)^{\nu_2})^{N_c} = \rho_2 \quad (6)$$

By the elimination of N_c the following equation is obtained:

$$\frac{\log(1 - (1/R_p)^{\nu_1})}{\log(1 - (1/R_p)^{\nu_2})} = \frac{\log \rho_1}{\log \rho_2} \quad (7)$$

This equation cannot be arithmetically solved. However, since the left side of the equation increases monotonically as R_p increases, it can be easily solved with numerical methods, e.g., Newton's method. Once R_p is determined, N_c is obtained from R_p as follows:

$$N_c = \frac{\log \rho_1}{\log(1 - (1/R_p)^{\nu_1})}. \quad (8)$$

Now, by Equation (7) and (8), we can determine R_p and N_c from the two control-points (ν_1, ρ_1) and (ν_2, ρ_2).

We suggest to set up the two control-points as the pair of the cut-off point (ν_c, ρ_c) and the rejection point (ν_r, ρ_r) by analogy with the low pass filter. The range of $n < \nu_c$ is the significance band where the point distribution with the local intrinsic dimensionality n should be regarded as being significant. The range of $n > \nu_r$ is the insignificant band where the point distribution with the local intrinsic dimensionality n should be regarded as being insignificant. The range of $\nu_c < n < \nu_r$ is the transition band. We also propose to determine the value of ν_c and ν_r from the maximum expected value of the relative difference between the k-th and the $(k+1)$-th nearest neighbor (Equation (3) and Figure 4). For example, it should be reasonable to set the cut-off dimensionality to 5 and the rejection dimensionality to 10 since only 20% or less difference is expected at the 5 dimensions, and 10% or less at 10 dimensions. When we choose 0.1 as the probability at the cut-off dimensionality and 0.9 as the probability at the rejection dimensionality, two control-points are obtained as (5, 0.1) and (10, 0.9). Then, R_p and N_c are computed from Equation (7) and (8) as 1.84471 and 48.0277 respectively. Figure 5 shows the rejection probability with these parameters computed from Equation (4).

As shown above, the method proposed here enables us to control the rejection probability as if we were designing such a low pass filter that filters the local intrinsic dimensionality (so to speak, a *dimensionality filter*). Although the rejection probability is estimated based on the uniform distribution, the uniformity is assumed only in each local region, i.e., the proximity specified by R_p. Because of these merits, the method described in this section should be an effective tool for estimating the significance of nearest neighbors.

4 Significance-Sensitive NN-Search Algorithm for Multidimensional Index Structures

4.1 Significance-Sensitive NN-Search Algorithm

In order to circumvent the harmful influence of insignificant nearest neighbors, we developed a new nearest-neighbor search algorithm for multidimensional index structures (e.g., SS-tree[6], VAMSplit R-tree [7], X-tree[8], SR-tree [2], LSD^h-tree[9], Hybrid tree[10], IQ-

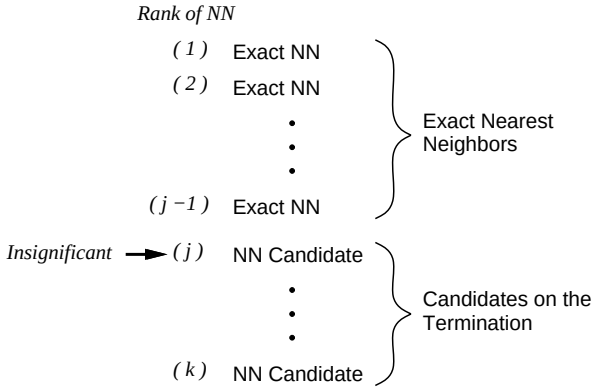


Figure 6: Search result of the significance-sensitive NN-search.

tree[11], etc.) with applying the significance estimation method described in the previous section. The algorithm tests the significance of a nearest-neighbor in the course of a search operation. Then, when it finds the nearest neighbor to be insignificant, it quits the search and returns the partial result as shown in Figure 6. When j-th nearest neighbor is found to be insignificant during k-nearest neighbor search, the search is terminated and candidates on the termination are returned for j-th to k-th nearest neighbors. Thus, the mixture of exact nearest neighbors and nearest neighbor candidates is returned when an insignificant nearest neighbor is detected during the search operation. We call this NN-search as the *Significance-Sensitive Nearest-Neighbor Search* since it is sensitive to the significance of nearest neighbors. This algorithm not only enables us to determine the significance of nearest neighbors but also enables us to cut down the search cost since this algorithm avoids pursuing exact answers for insignificant nearest neighbors.

Due to the limitation of the space, we do not show the details of the algorithm here. The full description and the program code of the algorithm can be found in [4]. The algorithm is an extension of the basic NN-search algorithm proposed in [12]. The main idea of the extension is counting the number of points that are located within the proximity of the query point in the course of search operation. The proximity is specified by the parameter R_p in terms of the distance to the nearest neighbor (d_{NN}) as in Definition 1. Therefore, the exact range of the proximity is determined only after the search is completed. However, it is possible to determine the subpart of the proximity from the lower and the upper bound of d_{NN}. The lower bound of d_{NN} ($LB\{d_{NN}\}$) is given by the distance to the closest node that is not visited so far, while the upper bound of d_{NN} ($UB\{d_{NN}\}$) is given by the distance to the current candidate of the nearest neighbor. From $LB\{d_{NN}\}$ and $UB\{d_{NN}\}$, the subpart of the proximity can be determined as the range from $UB\{d_{NN}\}$ to $LB\{d_{NN}\}\times R_p$. The proposed algorithm counts the number of points in this subpart of the proximity and detects the insignificance of the nearest neighbor even before the search is completed.

It should be noted that the significance-sensitive NN-search is different from the approximate NN-search algorithms [13, 14]. The approximate NN-search [13] terminates when the ratio of the upper bound to the lower bound of d_{NN} is reduced to less than or equal to $(1+\epsilon)$. Here ϵ is the controllable parameter of an error bound. At this point, the candidate is an approximate answer whose error in the distance from the query point is less than or equal to ϵ. Thus ϵ determines the precision of an approximate answer. The PAC NN-search (probably approximately correct NN-search)[14] also finds an approximate answer but estimates the lower bound of d_{NN} from the distance distribution of the dataset under the specified probability. Thus, the PAC NN-search returns an approximate answer under the given precision and the given probability. While these algorithms focus on the precision of an answer, the significance-sensitive NN-search algorithm focuses on the significance of the nearest neighbor. As shown in Figure 6, the proposed algorithm returns an inexact answer only when the nearest neighbor is determined to be insignificant. As long as the nearest neighbor is significant, the algorithm returns the exact answer. As its name implies, the salient feature of the proposed algorithm is the sensitivity to the significance of the nearest neighbor.

4.2 Evaluation with Synthetic Datasets

In order to evaluate the characteristics of the significance-sensitive NN-search, we synthesized datasets having various intrinsic dimensionalities. A dataset having the intrinsic dimensionality ν is synthesized by generating n-dimensional points $(x_1, \ldots, x_n)$ in accordance with the following rule:

$$x_i = \begin{cases} U(0,1) & (1 \le i < \nu) \\ \dfrac{1}{\sqrt{n-\nu+1}} U(0,1) & (i = \nu) \\ x_\nu & (\nu < i \le n), \end{cases} \tag{9}$$

where $U(0,1)$ is the uniform distribution in the range of 0 to 1. We synthesized datasets having the intrinsic dimensionality $1 \sim 20$. The dimensionality of the data space, i.e., n in Equation (9), is 20. Each dataset contains 1,000,000 points.

Experiments are conducted on a Sun Microsystems workstation, Ultra 60 (CPU: UltraSPARC-II 360MHz, main memory: 512Mbytes, OS: Solaris 2.6). Programs are implemented in C++. The VAMSplit R-tree[7] is employed as the index structure, since it has an efficient static construction algorithm suitable for the NN-search in high dimensional space. We measured the average performance of 1,000 random trials of finding the first nearest neighbor. Query points are selected at random from the dataset. The parameters R_p and N_c are set to 1.84471

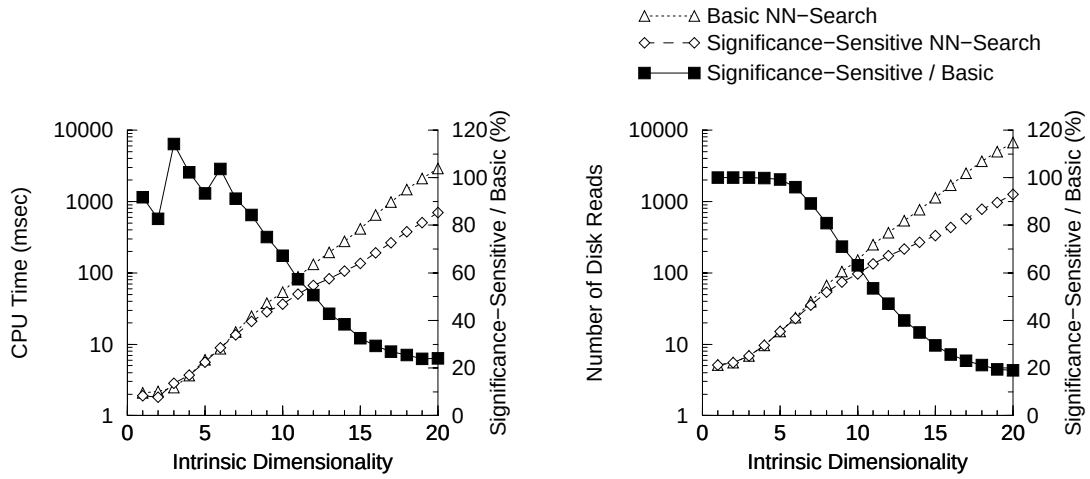


Figure 8: Performance with the synthetic datasets.

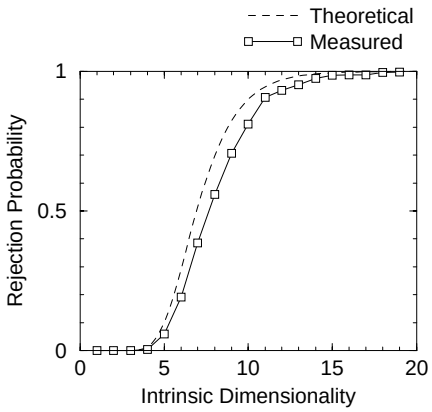


Figure 7: Rejection probability.

and 48 respectively according to the setting described in Section 3.2.

Figure 7 and 8 shows the result of the experiment. In both figures, the horizontal axis indicates the intrinsic dimensionality of the datasets. Figure 7 shows the rejection probability, i.e., the ratio of such a case that the nearest neighbor is determined to be insignificant. The measured probability is very close to the theoretical one. This proves the validity of the parameter setting presented in Section 3.2. Figure 8 shows the CPU time and the number of disk reads. When the intrinsic dimensionality is low, the difference between the basic NN-search and the significance-sensitive NN-search is negligible. However, when the intrinsic dimensionality is high, both the CPU time and the number of disk reads are reduced remarkably. When the intrinsic dimensionality is 20, the CPU time is reduced by 76% and the number of disk reads is reduced by 81% compared with the basic NN-search. This result demonstrates that the NN-search performance can be improved by the significance-sensitive NN-search.

5 Similarity Image Retrieval with Significance-Sensitive NN-Search

5.1 Experiment Configuration

We evaluated the performance of the significance-sensitive NN-search with applying it to the similarity retrieval of images. The dataset is 60,195 images of Corel Photo Collection contained in the product called Corel Gallery 1,000,000. The similarity of images is measured in terms of the color histogram. Munsell color system is used for the color space. It is divided into nine subspaces: black, gray, white, and six colors. For each image, histograms of four sub-regions, i.e., upper-left, upper-right, lower-left and lower-right, are calculated in order to take account of the composition of the image. Four histograms are concatenated to compose a 36-dimensional feature vector. Similarity of images is measured by Euclidean distance among these 36-dimensional vectors. We measured the performance of finding 100 nearest neighbors with employing every image as a query. Therefore, one of the found nearest neighbors is the query image itself.

5.2 Cost of the Significance-Sensitive NN-Search

We compared the cost of the significance-sensitive NN-search with that of the basic NN-search. As shown in Figure 9, both the CPU time and the number of disk reads of the significance-sensitive NN-search are remarkably reduced compared with those of the basic NN-search. The CPU time is reduced by 75% and the number of disk reads is reduced by 72%. This result demonstrates that the significance-sensitivity nearest-neighbor search enables us to cut down the search cost. Although the reduction rate depends on the dataset, this cost reduction capability of the significance-sensitive NN-search should be advantageous to the interactive similarity retrieval systems that need quick response to users.

Table 1: Number of significant NNs.

# of Significant NNs	# of Occurrence
100	0
90 ∼ 99	6
80 ∼ 89	40
70 ∼ 79	0
60 ∼ 69	1
50 ∼ 59	53
40 ∼ 49	13
30 ∼ 39	37
20 ∼ 29	87
10 ∼ 19	269
2 ∼ 9	13649
1	46040
Total	60195

Table 2: Categories of search results.

Observed Category	# of Significant NNs (at most)
Texture	92
Sky / Sea	62
Portrait	35
Card	23
Firework	23
Sunset	19
Kung-Fu	18
Steamship	17
Desert	16

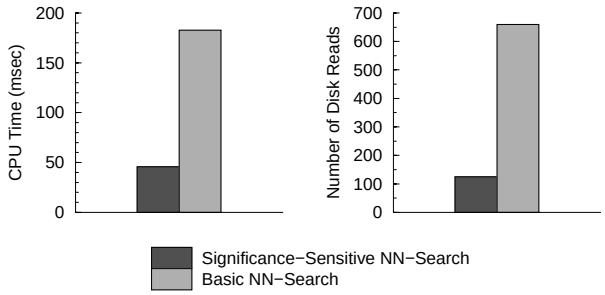


Figure 9: Performance of significance-sensitive nearest-neighbor search with the real dataset.

5.3 Number of Significant Nearest Neighbors in Search Results

Table 1 shows the number of significant nearest neighbors found by the significance-sensitive nearest-neighbor search. Because we employed every image as a query, the total of the occurrence is equal to the number of images in the dataset. Table 2 shows what kind of images are retrieved as significant nearest neighbors. We examined the search results having relatively many significant nearest neighbors and then determined what kind of images are retrieved as significant nearest neighbors. Figure 10 shows examples of search results when the number of significant nearest neighbors is relatively large. Due to the space limitation, top 5 images are shown. The numerical value under each image is the distance from the query to the image. We can see that similar images are successfully retrieved by the significance-sensitive nearest-neighbor search.

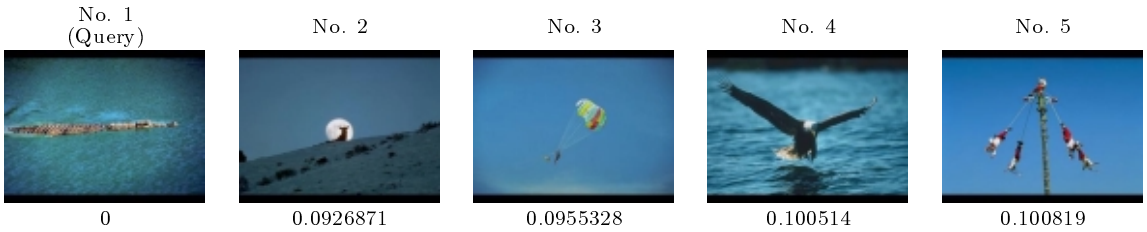
The amazing result of Table 1 is that we obtained only one significant nearest neighbor for 46,040 images. Since each query image is chosen from the images in the dataset, the obtained significant nearest neighbor is the query itself. Therefore, we obtained no significant nearest neighbor except for the query image. In this case, the query image is surrounded by plenty of neighbors that have almost the same similarity to the query. Since Corel Photo Collection collects wide variety of photos, it is not strange that an image has no similar image in the collection. In addition, the collection contains some texture photos. In this case, we have plenty of images with small difference. Figure 11 shows examples of search results when we obtained no significant nearest neighbors except for the query image. Due to the space limitation, top 5 images are shown. Figure 11 (a) is the case that the query image is surrounded by plenty of dissimilar images, while Figure 11 (b) is the case that the query image is surrounded by plenty of images with small difference. It should be noted that we do not claim that these search results are insignificant or meaningless. We only claim that *the nearest neighbor is insignificant*, i.e., the nearest neighbor is not significantly closest. The aim of the proposed algorithm is to avoid pursuing exact answers for such insignificant nearest neighbors.

The examples shown above illustrate that the significance-sensitive nearest-neighbor search allows us to see how retrieved images are significantly close to the query image. This capability should be advantageous to the interactive information retrieval systems.

5.4 Cumulative Distribution of Points around Query Points

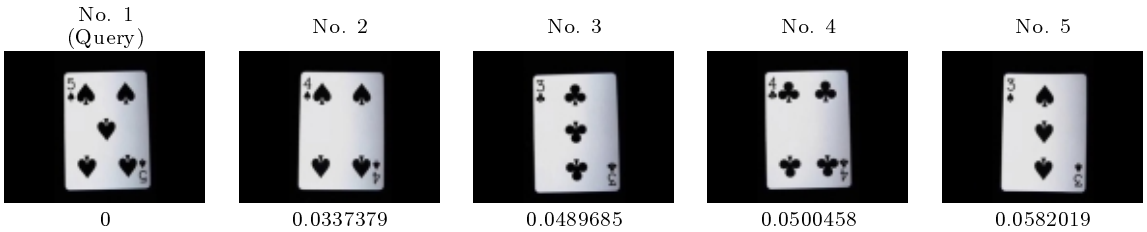
In order to validate the results of the significance-sensitive nearest neighbor search, we plotted cumulative distribution of points around the query points used in the above examples. Figure 12 shows the cumulative distribution of points around the query points used in Figure 10 (a), 10 (b), and 11 (a). The number of significant nearest neighbors obtained by the search are 92, 62, and 1, respectively. The horizontal axis indicates the distance from the query point normalized by the distance to the 1st nearest neighbor of each query point, while the vertical axis indicates the cumulative frequency of points. As the figure shows, the distribution of nearest neighbors differs distinctively from one query point to another. The broader the distribution is, the more significant nearest-

(a) Texture (Significant NN: 92)

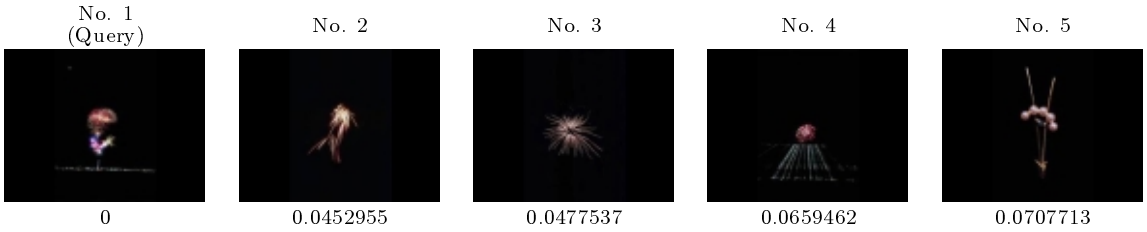


(b) Sky / Sea (Significant NN: 62)

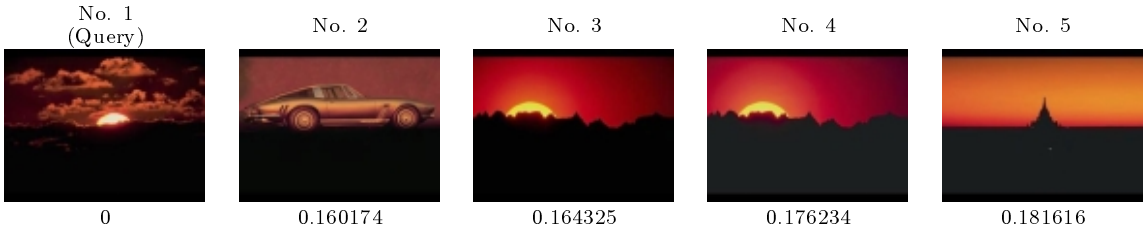
(c) Portrait (Significant NN: 35)



(d) Card (Significant NN: 23)

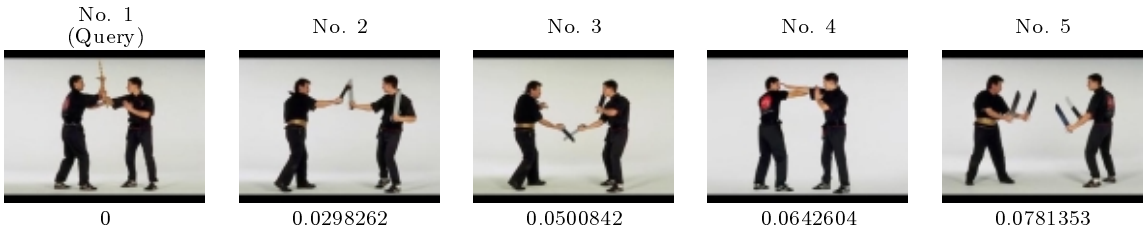


(e) Firework (Significant NN: 23)

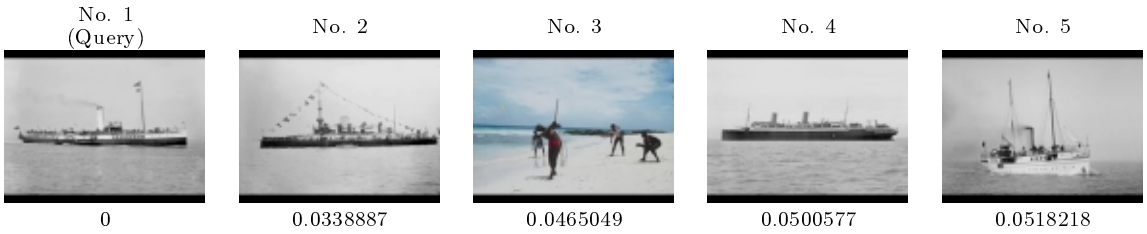


(f) Sunset (Significant NN: 19)

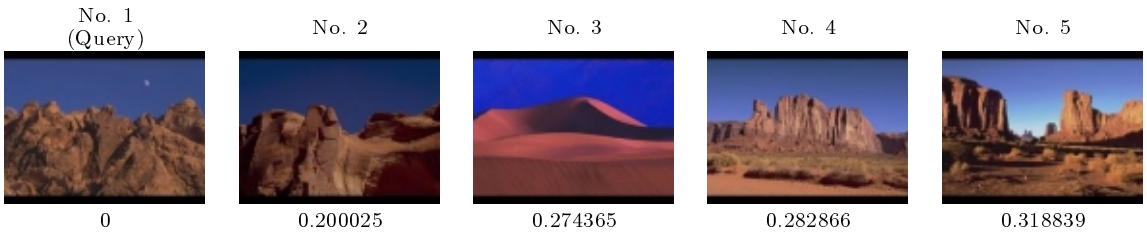
Figure 10: Examples of search results.



(g) Kung-Fu (Significant NN: 18)

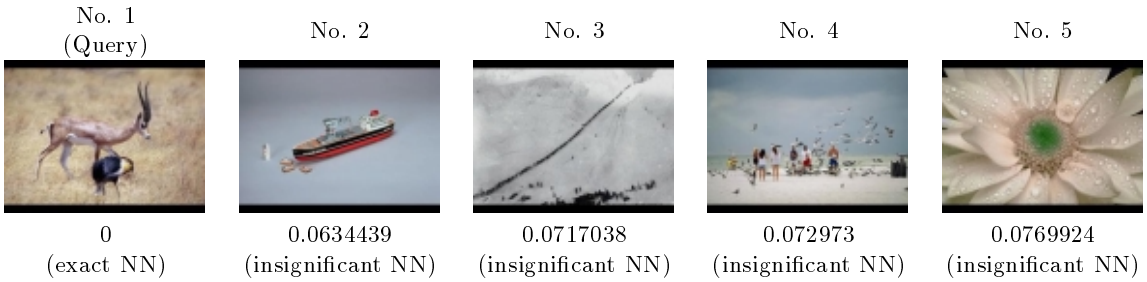


(h) Steamship (Significant NN: 17)

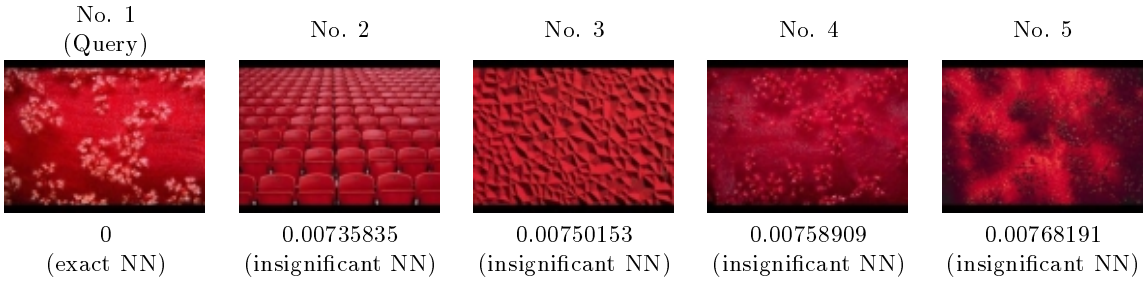


(i) Desert (Significant NN: 16)

Figure 10: Examples of search results (cont'd).



(a) No similar image is obtained.



(b) Images with small difference are obtained.

Figure 11: Examples of search results when no significant NN is found except for the query.

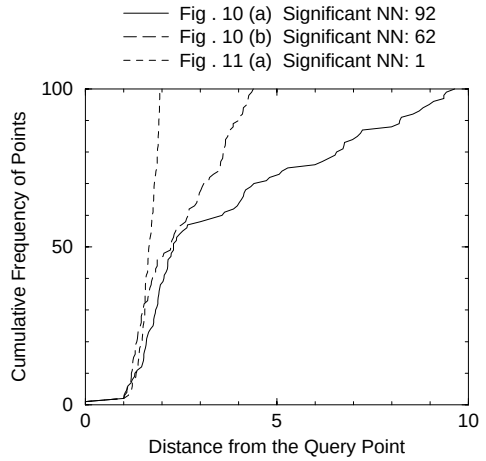


Figure 12: Cumulative distribution of points around query points.

neighbor is obtained. This tendency meets the aim of the significance sensitive NN-search, i.e., detecting the congestion of nearest neighbors.

6 Conclusion

This paper shows that the similarity image retrieval can be improved with the employment of a new NN-search algorithm, the significance-sensitive nearest-neighbor search. Insignificant nearest neighbors are likely to occur in high dimensional space. They affect NN-search performance because the search operation must examine many strong candidates to determine the true nearest neighbor. In addition, from the perspective of the similarity retrieval, it is meaningless to choose the true nearest neighbor from plenty of candidates having almost the same similarity.

The significance-sensitive nearest-neighbor search circumvents this problem by detecting insignificant nearest neighbors during the search operation. When an insignificant nearest neighbor is detected, the operation is terminated and a partial result is returned. This reduces both CPU time and disk I/O, and besides enables us to distinguish more significant neighbors from less significant ones. These capabilities are advantageous especially to the interactive information retrieval systems.

In our future work, we plan to conduct more experiments to clarify the characteristics of the proposed search algorithm in further detail. For example, we plan to employ other image features and to use other image collections.

Acknowledgment

This research is partly supported by the Ministry of Education, Science, Sports and Culture, Grant-in-Aid for Creative Basic Research (09NP1401) and Encouragement of Young Scientists (12780251).

References

[1] S. Berchtold, C. Bohm, B. Braunmuller, D. Keim, and H.-P. Kriegel, "Fast Parallel Similarity Search in Multimedia Databases," Proc. of the 1997 ACM SIGMOD, Tucson, USA (May 1997) pp. 1–12.

[2] N. Katayama and S. Satoh, "The SR-tree: An Index Structure for High-Dimensional Nearest Neighbor Queries," Proc. of the 1997 ACM SIGMOD, Tucson, USA (May 1997) pp. 369–380.

[3] K. S. Beyer, J. Goldstein, R. Ramakrishnan, and U. Shaft, "When Is "Nearest Neighbor" Meaningful?," Proc. of the 7th Int. Conf. on Database Theory, Jerusalem, Israel, pp.217–235, Jan. 1999.

[4] N. Katayama and S. Satoh, "Significance-Sensitive Nearest-Neighbour Search for Efficient Similarity Retrieval of Multimedia Information," First International Workshop on Multimedia Intelligent Storage and Retrieval Management (MISRM'99), Orlando, USA (Oct. 1999) available on-line at http://www.info.uqam.ca/~misrm/papers/katayama.ps.gz.

[5] K. Fukunaga, "Introduction to Statistical Pattern Recognition (2nd ed.)," Academic Press, 1990.

[6] D. A. White and R. Jain, "Similarity Indexing with the SS-tree," Proc. of the 12th Int. Conf. on Data Engineering, New Orleans, USA, pp.516–523, Feb. 1996.

[7] D. A. White and R. Jain, "Similarity Indexing: Algorithms and Performance," Proc. SPIE Vol.2670, San Diego, USA, pp.62–73, Jan. 1996.

[8] S. Berchtold, D. A. Keim, and H.-P. Kriegel, "The X-tree: An Index Structure for High-Dimensional Data," Proc. of the 22nd VLDB Conf., Bombay, India, pp.28–39, Sep. 1996.

[9] A. Henrich, "The LSD^h-tree: An Access Structure for Feature Vectors," Proc. of the 14th Int. Conf. on Data Engineering, Orlando, USA, pp.362–369, Feb. 1998.

[10] K. Chakrabarti and S. Mehrotra, "The Hybrid Tree: An Index Structure for High Dimensional Feature Spaces," Proc. of the 15th Int. Conf. on Data Engineering, Sydney, Australia, pp.440–447, Mar. 1999.

[11] S. Berchtold, C. Bohm, H. V. Jagadish, H.-P. Kriegel, J. Sander, "Independent Quantization: An Index Compression Technique for High-Dimensional Data Spaces," Proc. of the 16th Int. Conf. on Data Engineering, San Diego, USA, pp.577–588, Feb. 2000.

[12] G. Hjaltason and H. Samet, "Ranking in Spatial Databases," 4th Int. Symp. on Spatial Databases, SSD'95, Portland, USA, pp.83–95, Aug. 1995.

[13] S. Arya, D. M. Mount, N. S. Netanyahu, R. Silverman, and A. Y. Wu, "An Optimal Algorithm for Approximate Nearest Neighbor Searching," Proc. of the 5th Ann. ACM-SIAM Symposium on Discrete Algorithms, pp.573–582, 1994.

[14] P. Ciaccia and M. Patella, "PAC Nearest Neighbor Queries: Approximate and Controlled Search in High-Dimensional and Metric Spaces," Proc. of the 16th Int. Conf. on Data Engineering, San Diego, USA, pp.244–255, Feb. 2000.