Distinct Nearest Neighbors Queries for Similarity Search in Very Large Multimedia Databases *

Tomáš Skopal
Department of Software Engineering, FMP,
Charles University in Prague, Malostranské
nám. 25, 118 00 Prague, Czech Republic
skopal@ksi.mff.cuni.cz

Vlastislav Dohnal, Michal Batko,
Pavel Zezula
Faculty of Informatics, Masaryk University,
Botanická 68a, 602 00 Brno, Czech Republic
{dohnal,batko,zezula}@fi.muni.cz

ABSTRACT

As the volume of multimedia data available on internet is tremendously increasing, the content-based similarity search becomes a popular approach to multimedia retrieval. The most popular retrieval concept is the k nearest neighbor (kNN) search. For a long time, the kNN queries provided an effective retrieval in multimedia databases. However, as today's multimedia databases available on the web grow to massive volumes, the classic kNN query quickly loses its descriptive power. In this paper, we introduce a new similarity query type, the *k distinct nearest neighbors* (kDNN), which aims to generalize the classic kNN query to be more robust with respect to the database size. In addition to retrieving just objects similar to the query example, the kDNN further ensures the objects within the result have to be distinct enough, i.e. excluding near duplicates.

Categories and Subject Descriptors

H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval—*Retrieval models*

General Terms

algorithms, performance, design

Keywords

similarity search, kNN query, content-based retrieval

1. INTRODUCTION

In the last decade, the research concerning similarity search in multimedia databases underwent a long way towards effective and efficient retrieval [2, 6, 4]. The two basic similarity query types, the range query and the k nearest neighbors (kNN) query, have been adopted as a simple yet powerful model of content-based query-by-example similarity search paradigm. In particular, given a database of multimedia objects (e.g., images), the kNN query returns the k most similar objects to the query object.

Since kNN query is widely accepted as sufficient for many tasks in multimedia databases and other domains, the researchers focused rather on the *efficiency* issues (performance/speed of indexing and query processing), than on improving the *effectiveness* problem (quality of query answer, i.e., different query types). Hence, during the last years the indexing power of similarity search engines increased significantly, thus scaling quite well with the quickly growing volumes of multimedia data available. On the other hand, in this new situation we experience a problem that the classic kNN queries lose their expressive power because of increasing "density" of the indexed data space. Consequently, in such space the kNN query returns a bunch of almost identical objects, where most of which are not useful for the user.

In this paper, we introduce a new type of similarity query – the k distinct nearest neighbors (kDNN) query, generalizing the classic kNN query. Unlike kNN query, the kDNN query not only returns k objects similar to the query object, but also eliminates indistinct objects – those very similar to another object in the answer. Such kNN query extension is more descriptive and robust with respect to the space density, while it also provides flexible parameterization (allowing also the classic kNN, as a special case).

2. METRIC SPACE MODEL OF SIMILARITY SEARCH

In the past two decades, the metric space model of similarity has been established as a suitable framework for efficient similarity search in complex and unstructured domains, like multimedia databases, time series retrieval, bioinformatics, etc. [9, 8]. The vote for *metric* space paradigm was motivated by performance needs. Specifically, the number of similarity scores computed during query processing is required to be minimized, because a single computation of such a score is assumed to be computationally expensive.

In the metric space model, the content of each object (either a database object or query object) is represented by a descriptor of features $x \in \mathbb{U}$, where $\mathbb{U}$ is the descriptor universe. Furthermore, the pair-wise similarity measure δ has to be a metric distance, i.e., $(\mathbb{U}, \delta)$ has to fulfill the metric postulates. The metric postulates are critical for the model, because they provide a mechanism of space partitioning that is further used by the *metric access methods* (MAMs) for efficient similarity search [5, 9, 8].

*This research was partially supported by GAČR projects 201/09/0683, 201/07/P240, 201/08/P507 and by GAUK project no. 18208. Hardware infrastructure was provided by MetaCentrum (meta.cesnet.cz).

WIDM'09, November 2, 2009, Hong Kong, China.

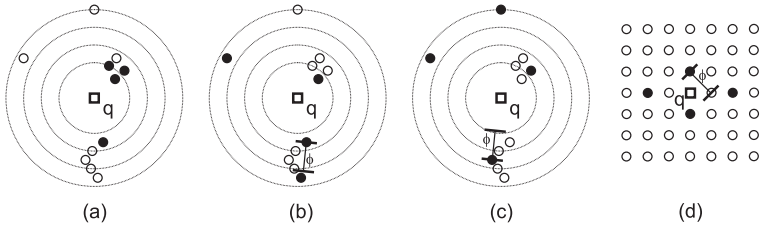

Figure 1: (a) Result of classic 4NN query. (b,c,d) Possible results of a 4DNN query.

2.1 Classic k Nearest Neighbors

The k nearest neighbors (kNN) query is the most popular query type in the metric similarity search. Formally, let (q, k) be a query, $\mathbb{S} \subset \mathbb{U}$ be the database, and $\mathbb{R} \subset \mathbb{S}$ be the query result set, such that $|\mathbb{R}| = k$.

Definition 1: Classic kNN query is defined by restricting $\mathbb{R}$ by the condition: $\forall x \in \mathbb{R}, \forall y \in \mathbb{S} - \mathbb{R} : \delta(x, q) \leq \delta(y, q)$. If several different sets $\mathbb{R}$ satisfy the condition, one of them is chosen arbitrarily.

For an example of 4NN query, see Figure 1a, where the dots represent multimedia objects (e.g., images) in the metric space. The Euclidean distance between the dots illustrates the similarity between individual objects. The black dots are those returned as the query result.

3. K DISTINCT NEAREST NEIGHBORS

Since the beginning of the new millennium, we have experienced a phenomenon of information explosion, where the volume of produced digital data grows exponentially in time. Nowadays, more than 95% of web space is considered to store multimedia content, and more multimedia data is stored in corporate and scientific databases, personal archives, and digital libraries. In particular, 80 billion of photographs were expected to be snapped in 2007 according to Enterprise strategy group (`www.enterprisestrategygroup.com`). At the same time, the complexity of the data grew as well.

Such new extreme conditions not only impose strong requirements on the efficiency of multimedia retrieval systems, but also induce problems related to the *quality* (or *effectiveness*) of retrieval, which nowadays begins to appear not generally scalable with respect to the database size. We call this phenomenon as the increasing "density" of data space. Although the data volumes grow rapidly, the underlying similarity search model, i.e., the design of particular metric space for a given domain and similarity queries, remains the same. Hence, the space becomes more filled by the data (thus, becoming dense), while if we pose a query anywhere in the space, we often encounter many almost identical database objects in the response.

As an example, Figure 2 depicts results of 10NN queries on differently sized real-world databases, where the database sizes ranged from 1 million to 100 million images. The query issued on a small database returns very different images. When the database size increases, the number of nearly duplicate images grows. In the largest database, we get all of the images visually almost identical.

To overcome the "dense space" problem, we propose the kDNN query type. It builds on the classic kNN query, but, at the same time, kDNN excludes all objects that are too similar to any of the already reported objects. Such an approach is quite robust with respect to the database size, while the robustness can be tuned by setting what still is and what already is not understood as "too similar".

3.1 Formal Definition

Let (q, k) be a query, $\mathbb{S} \subset \mathbb{U}$ be the database, and $\mathbb{R} \subset \mathbb{S}$ be the query result set, such that $|\mathbb{R}| = k$.

Definition 2: The *k distinct nearest neighbors* (kDNN) query is defined by restricting $\mathbb{R}$ as $\forall x, z \in \mathbb{R}, \forall y \in \mathbb{S} - \mathbb{R} : \delta(x, z) \geq \phi \wedge (\delta(x, q) \leq \delta(y, q) \vee \exists w \in \mathbb{R} : \delta(y, w) < \phi)$, where ϕ is user-defined *separation distance*.

It follows from the definition that there could be multiple solutions of a single kDNN query obtained, as shown in Figures 1b,c,d (compare with Figure 1a). Note that for $\phi = 0$ we get the classic kNN query.

Since the definition of kDNN query is quite loose (allowing several possible results for the same query), we introduce two kDNN algorithms complying with the definition above.

3.2 Algorithm Variants

Our implementation of the kDNN algorithm is an extension of the classic kNN algorithm. Given a query object q, suppose a virtual ordering of database objects with respect to the distance to q. The algorithm incrementally reveals as large part of the ordering as needed. In the following, we propose two variants of kDNN processing. A common skeleton of both variants is the incremental database traversal, i.e., we retrieve objects in ascending order with respect to the distance to q.

First-Match kDNN algorithm. The first variant of the kDNN algorithm is simple. Whenever a distinct object is retrieved from the ordering it is added to the query result, i.e., the objects already reported to the query result have to be far enough from each other (see Figure 1b). A real-world example in Figure 3 shows the result of a 20DNN query on 100M database, using the same query image as in Figure 2. When compared with kNN result (third line in Figure 2), the result of kDNN query reveals much more interesting images than just duplicates. Here we can see a perfect example for the kDNN motivation. The image of the Sun was collapsed to the first DNN, while the other images are just similar to the Sun, not duplicates. Moreover, the images in result are distinct, that is, they are not much similar to each other, but all are similar to the query (the Sun).

Centroid-Match kDNN algorithm. The second variant is more complex. Besides reporting just the distinct neighbors (as in the first-match algorithm), the centroid-match algorithm tries additionally to report centers of potential clusters in the classic kNN query result (see Figure 1c).

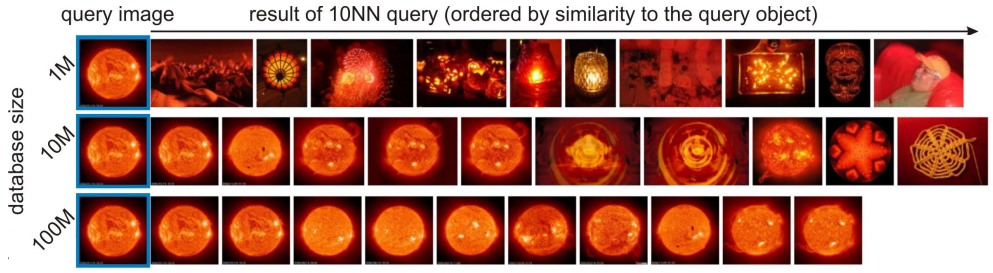

Figure 2: Limited power of classic kNN in very large database. Results of the same 10NN query in: small DB – 1M images, large DB – 10M images, and very large DB – 100M images.

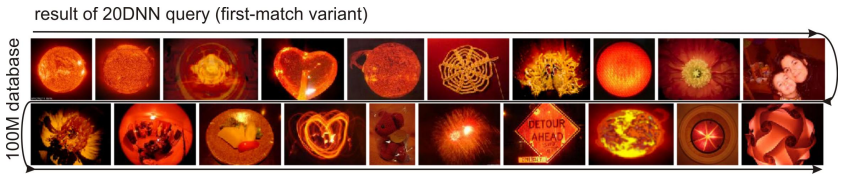

Figure 3: Result of a 20DNN query on very large database.

4. EXPERIMENTAL EVALUATION

In this section, we provide results of our experimentation with the kDNN queries. A prototype implementation was evaluated by humans to confirm their subjective satisfaction with results, and by various statistics to reveal computational pros and cons.

4.1 Testbed

For all experiments, we use the CoPhIR [3] collection of 100 million images which primarily consists of real-world photos. The similarity of the images is measured by a metric function that combines five MPEG-7 visual descriptors, deeply explained in [3]. Multi-Feature Indexing Network (MUFIN) [7] provides the indexing and query evaluation of the underlying incremental kNN queries. MUFIN is a Java-based highly-scalable information retrieval system that allows indexing pure metric space objects. A demonstration of MUFIN as an image search engine is publicly available at `http://mufin.fi.muni.cz`. The kDNN algorithms were implemented using MUFIN's programming interface.

4.2 Subjective Evaluation

Since kDNN is a new query concept, we need to asses its usefulness for real users. Therefore, we have exploited the user-satisfaction survey framework [1] offered by the MUFIN system to reveal if users prefer the results of classic kNN or our new kDNN queries.

We recruited 30 users, mainly specialized in computer science, that evaluated results of 10NN and 10DNN queries for 45 query images. Such query images were selected automatically by the framework for which kNN and kDNN provided significantly different results, and thus users could actually assess the difference. The kDNN queries used the first-match algorithm and three different values of ϕ.

The table in Figure 4 reports on the percentage of results where users preferred one query type over the others (the classic kNN or the new kDNN). Observe that kDNN query improves user satisfaction in 66% cases, while the classic kNN gives subjectively better results for only 26% answers. For 8% answers, the "winner" could not be clearly decided

Query	ϕ	Perc.
cannot decide		8%
classic 10NN		26%
10DNN	0.8	30%
10DNN	1.0	14%
10DNN	1.2	22%

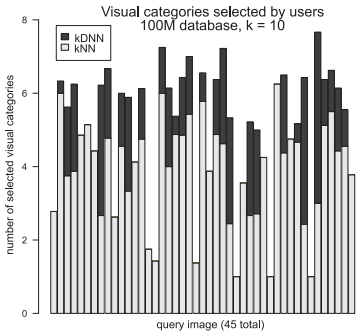

Figure 4: Subjective comparison of kNN and kDNN (left). Selected distinct images by users (right).

either because the visual similarity of images was different from the human perception or because all the results were considered good. Moreover, the kDNN queries were assigned different values of ϕ. They were selected with respect to distance distribution of the dataset.

We have also asked the volunteers to select distinct images (we call them visual categories). Figure 4 (right) summarizes average number of visual categories for each 10NN query (labeled 1–45 on x axis). The gray bars correspond to distinct images selected within a kNN result, while the black bars represent an improvement of kDNN (the values for kDNN were never less than the values for kNN). The average is computed over all users that evaluated that query. A perfect query result corresponds to 10 distinct images. We can see that kDNN was quite successful in selecting distinct objects, however, we can observe that kDNN results have about six objects selected as correct, thus users did perceive the other four objects as indistinct or completely wrong (not similar to the query image).

4.3 Statistical Evaluation

The similarity as defined by the metric function and the subjective human perception of similarity heavily influence the result of user satisfaction survey. In this section, we

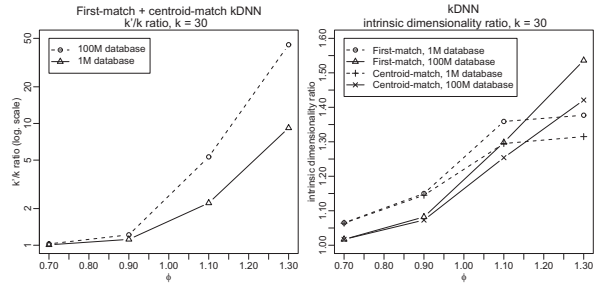

Figure 5: Extra nearest neighbors ratio for kDNN (left) and improvement of intrinsic dimensionality (right) for varying ϕ.

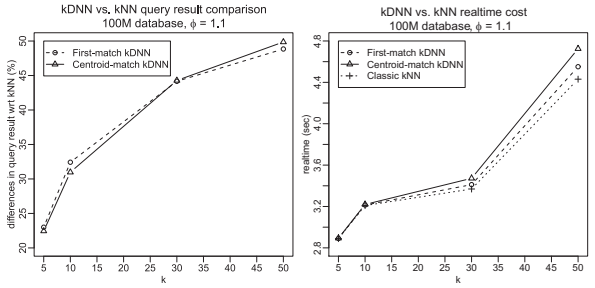

Figure 6: Differences of kNN and kDNN results (left) and processing times (right) for varying k.

provide a rigorous evaluation based on statistics collected directly from the metric space and thus independent of the actual representation of similarity. The same set of 400 images randomly selected from the database were used and average values over them are presented to minimize the statistical deviations. In addition, the experiments were computed on the whole 100 million collection (denoted 100M in the graphs) and on a 1 million subset (denoted 1M) to show the difference in density of the metric space.

Firstly, we observed the number of nearest neighbors (denoted as k') needed to answer kDNN query for different values of ϕ, i.e., the increase in the costs. Figure 5 (left) shows the ratio k'/k for $k = 30$ and ad-hoc ϕ values in $[0.7, 1.3]$. Both first- and centroid-match algorithms gave practically the same results, so there is only one line per database in the graph. We can see that $\phi = 0.7$ is so small that the kNN and kDNN results are practically the same. On the other hand, the value of $\phi = 1.3$ requires more than 40 times more objects for 100M database and nearly 10 times more for 1M database (the scale in the figure is logarithmic). It means that on average, each object in the kDNN result filtered 40 other candidate neighbors within the separation distance ϕ.

Figure 5 (right) shows the variance of distances between the objects in the query result in terms of the intrinsic dimensionality [5], denoted as ρ. In particular, the ratio between the intrinsic dimensionality of the kDNN and kNN result sets (ρ_{kDNN}/ρ_{kNN}) is depicted. The higher the ratio is the more distinct the objects in kDNN are. Observe that the answers returned by centroid-match kDNN are really more clustered (higher variance) than the answers returned by first-match algorithm. Moreover, we can see that the 1M database can find more distinct object for lower values of ϕ, but since the metric space is "sparse" the improvement is slowing down faster than for 100M database.

Secondly, statistics revealing the dependence on different values of k are shown in Figure 6. The left graph shows the percentage of different objects in kNN and kDNN results with respect to size of the result. We see the expected trend – the more objects in the result the more the results differ.

Finally, we report on the query processing time. In Figure 6 (right), "classic kNN" refers to the query execution time in MUFIN. We can clearly see the increase of time caused by the additional computation needed by the kDNN query. This includes all the costs for retrieving the additional neighbors incrementally as well as the algorithms' processing. In total, the increase of time is only marginal (by 2-7%), so the query can be easily used in any online system.

5. CONCLUSIONS AND FUTURE WORK

In this paper, we have presented the concept of k distinct nearest neighbors (kDNN) as a new query type suitable for content-based similarity search in very large multimedia databases. Unlike the classic k nearest neighbors (kNN) that often return nearly-duplicate objects in the query result (given a very large multimedia database), the kDNN queries return only such objects that are distinct enough (i.e., somewhat dissimilar to each other, yet similar to the query object). We have proposed two algorithms for kDNN retrieval, based on the well-known incremental kNN algorithm and have performed experimental evaluation of qualitative properties (both subjective and statistical) of the kDNN query results with respect to classic kNN. The experiments confirmed that kDNN query exhibits better and more robust behavior in very large databases than the classic kNN.

6. REFERENCES

[1] M. Batko, P. Kohoutkova, and D. Novak. CoPhIR Image Collection under the Microscope. In *Proc. SISAP*, pages 47–54. IEEE, 2009.

[2] H. M. Blanken, A. P. de Vries, H. E. Blok, and L. Feng. *Multimedia Retrieval*. Springer, 2007.

[3] P. Bolettieri, A. Esuli, F. Falchi, C. Lucchese, R. Perego, T. Piccioli, and F. Rabitti. CoPhIR: a test collection for content-based image retrieval. *CoRR*, abs/0905.4627v2, 2009.

[4] V. Castelli and L. D. Bergman, editors. *Image Databases : Search and Retrieval of Digital Imagery*. Wiley-Inter., 2002.

[5] E. Chávez, G. Navarro, R. Baeza-Yates, and J. L. Marroquín. Searching in metric spaces. *ACM Computing Surveys*, 33(3):273–321, 2001.

[6] S. Deb. *Multimedia Systems and Content-Based Image Retrieval*. Information Science Publ., 2004.

[7] D. Novak, M. Batko, and P. Zezula. Generic similarity search engine demonstrated by an image retrieval application. In *Proc. ACM SIGIR*, page 1, 2009.

[8] H. Samet. *Foundations of Multidimensional and Metric Data Structures*. Morgan Kaufmann, 2006.

[9] P. Zezula, G. Amato, V. Dohnal, and M. Batko. *Similarity Search: The Metric Space Approach (Advances in Database Systems)*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2005.