Approximate Similarity Queries: A Survey

Paolo Ciaccia
DEIS - CSITE-CNR
University of Bologna, Italy
`pciaccia@deis.unibo.it`

Marco Patella
DEIS - CSITE-CNR
University of Bologna, Italy
`mpatella@deis.unibo.it`

Abstract

We review the major paradigms for similarity queries, in particular those that allow approximate results. We propose an original classification schema which easily allows existing approaches to be compared along several independent coordinates, such as quality of results, error metrics, and user interaction.

1 Introduction

Similarity queries are a search paradigm which is profitably used when dealing with mining of data. In its essence, the problem is to find objects which are similar, up to a given degree, to a given query object. In this way, for example, we are able to cluster together similar objects by executing several similarity queries [EKSX96]. In order to assess the similarity between pair of objects, usually a notion of distance is used, being understood that low values of distance correspond to high degrees of similarity. More formally, we are faced with the following problem: Given a set of objects $\mathcal{O} \subset \mathcal{U}$ drawn from a generic metric space $\mathcal{M} = (\mathcal{U}, d)$, where $\mathcal{U}$ is a domain (the *feature space*) and $d : \mathcal{U} \times \mathcal{U} \to \Re_0^+$ is a non-negative and symmetric binary function that also satisfies the triangle inequality, retrieve the object(s) which are closest (i.e. that lead to the lowest values of d) to a user-specified query object $q \in \mathcal{U}$. Typical similarity queries include *range* queries (where all the objects in $\mathcal{O}$ whose distance to q does not exceed a user-specified threshold α are requested) and k-nearest neighbor (k-NN) queries (where the k objects in $\mathcal{O}$ which are closest to q are requested).

Several access structures have been proposed to speed up the resolution of (exact) similarity queries: They can be broadly classified (depending on their field of applicability) as multi-dimensional (or spatial) and metric access methods (the former only apply when the feature space is a vector space). Recent studies, however, pointed out the fact that using such access structures is sometimes not very efficient (e.g. when the feature space is a high-dimensional vector space [WSB98]): In such cases, the most efficient way to exactly solve similarity queries is to sequentially scan the entire data-set, comparing each object against the query object q. Obviously, such solution is not viable for very large data-sets.

To speed-up the search it is common to offer to the user a quality/time trade-off: If the user is willing to save search time, he/she has to accept a degradation in the quality of the result, i.e. an error with respect to the exact case. Approximate similarity search, therefore, has the goal to reduce search times for similarity queries by introducing an error in the result. Since k-NN queries represent the most used type of similarity queries (because the user can control the query selectivity, i.e. the cardinality of the result set), in the following we will concentrate on this kind of queries.

In this work we review the problem of approximate similarity search, proposing a classification schema able to characterize existing approaches with respect to the kind of used approximation and on how the quality of the result is measured. The goal is to present an unified view over the different approaches proposed in literature. This paper is articulated as follows: In Section 2 we present some applicability scenarios for approximate similarity search. Section 3 presents the proposed classification schema. In Section 4, some of the approaches recently proposed in literature for approximate similarity search are classified according to the presented schema. Finally, Section 5 discusses about classification results, proposing possible extensions to the schema, and concludes.

2 Applicability Scenarios

The field of applicability of approximate similarity search is very wide. As an example, consider a multimedia database, where the user can query the system to search for images similar to a given one: Usually, such kind of search consists of several steps, each refining the query by changing, through relevance feedback techniques [BCW00], both the query object and the distance function used to compare objects. In this light it is clear that the first steps of the search do not require that an exact result is found, but only that a result can be quickly obtained that is approximately similar to the exact result, in order to proceed with further steps as soon as possible.

As another example, consider modern Decision Support Systems (DSSs), where complex queries are posed to the underlying database system over Gigabytes (or even Terabytes) of data. Such queries are expected to be computationally very expensive even if the exploratory nature of many DSS applications often do not require an exact answer [CGRS00]. In this scenario, users are frequently ready to accept an approximate result if the query solution time is reduced by some orders of magnitude.

Another common case when approximate queries arise is that of cluster-based similarity search: In this context, the data-set is first divided in clusters, i.e. sets of objects sharing common characteristics (e.g. which are similar to each other), by means of data mining techniques [JD88, JMF99]; then, at query time, the query object is compared against clusters' representatives and only those cluster that are closest to the query are accessed to retrieve objects similar to the query [WYM97, BFG99, LCGMW]. It is clear that such techniques cannot guarantee that the exact result is found.

3 A Classification Schema

Being understood that exact similarity search is sometimes difficult, several approaches have been recently proposed to solve the approximate problem. Virtually every approach proposes a new technique and new kind of metrics to evaluate the quality of the approximate result, yet all of them fail in relating the presented approach with other techniques presented in literature. The goal of this Section is to present a schema able to classify existing approaches by means of the following coordinates:

1. The type of data to which the approach can be applied.
2. The metrics introduced to quantify errors produced by approximation.
3. The guarantees in the quality of the result offered by the approach.

4. The degree of interaction with the user, i.e. the possibility the user has to *tune* the technique parameters to adapt to his/her actual needs.

The above coordinates have been chosen in order to evaluate the field of applicability of existing techniques for approximate similarity search. In fact, it is understood that if a technique A is applicable only to a subset of the data to which another technique B is applicable, then A is less general than B. On the other hand, it could be the case that A is more efficient or leads to lower errors: We are not interested in overall efficiency or accuracy of existing techniques here, but only on how they are achieved and how they can be measured.

3.1 Data Types

Since (exact) similarity queries require a notion of distance to be defined, an approximate technique is usually applied to a set of objects $\mathcal{O} \subset \mathcal{U}$ drawn from a generic metric space $\mathcal{M}$ (see Section 1). Examples of metric spaces include the D-dimensional vector space $\Re^D$ with the Euclidean distance L_2 or the set Σ^* of (finite length) strings obtained from an alphabet of symbols Σ with the edit distance d_{edit} (i.e. the minimum number of symbols that have to be inserted, deleted or substituted in order to transform a string into another).

Since, however, in most of the cases the data-set $\mathcal{O}$ is drawn from a vector space, several techniques exploit this fact by explicitly referring to coordinates of the space. Of course, this limits the applicability of such techniques, since they cannot be used on non-vectorial objects (e.g. strings with the edit distance). Moreover, some techniques are only applicable when the distance used to measure the (dis-)similarity between objects is an L_p metric[1] or, even more restrictive, the Euclidean distance L_2.

In this light, the following classification is given, in decreasing order of applicability:

MS (metric spaces) Methods in this class are applicable to the more general case of objects drawn from a generic metric space.

VS (vector spaces) In this class fall all those techniques that explicitly use objects' coordinates, thus are only applicable to vector spaces. However, these techniques do not make any assumption on the metric used to compare vectors (thus arbitrarily chosen distance functions can be used, e.g. quadratic form functions where the distance between vectors is defined by way of a positive definite symmetric matrix [SK97]).

$\mathbf{VS}_{L_p}$ (vector spaces, L_p distance) Techniques belonging to this class can only be applied when the considered objects are vectors in a D-dimensional space and the distance used to compare them is an L_p metric (thus no correlation between coordinates is allowed). Specific classes can be obtained by instantiating p (e.g. the class $\mathbf{VS}_{L_2}$ contains techniques that only applies to Euclidean spaces, i.e. when the distance used is the L_2 Euclidean metric). If p is not instantiated, then the technique is applicable to any vector space with an L_p metric, independently of the value of p.

As examples of the above classification method, we now describe three approximations techniques, assigning each of them to the proper class.

[1]We recall that the definition of the L_p distance between two points x and y in a D-dimensional space is as follows: $L_p(x,y) = \left(\sum_{i=1}^{D} |x[i] - y[i]|^p\right)^{1/p}$, $1 \leq p < \infty$, $L_\infty(x,y) = \max_{i=1}^{D} |x[i] - y[i]|$.

Example 1
In [GIM99] the authors propose the use of Locality-Sensitive Hashing (LSH) to transform a D-dimensional vector x into a sequence of C bits (binary vector) $v(x)$. Since the L_1 distance between vectors can be approximated by the Hamming (edit) distance between the corresponding binary vectors, they propose a hashing technique to index only the binary vectors $v(x)$. Of course, both accuracy and efficiency of the technique highly depend on the number C of bits used for approximating vectors. Since the approximation for the Hamming distance only applies to the L_1 metric, this technique is of class $\mathbf{VS}_{L_1}$.

Example 2
In [WB00] approximate nearest neighbor search techniques based on the VA-file [WSB98] are presented. Such structure, in its essence, is a sequential structure containing approximations of vectors using a fixed number b of bits. Exact k-NN search is performed by first executing a sequential scan of the structure using the query distance on vectors approximations, which yields a number $M > k$ of *candidate vectors*, and then applying a refinement step, where the distance is evaluated on real vectors and only the k "best" vectors are kept. Proposed techniques suggest either to reduce the number of considered approximations by reducing the query radius (VA-BND) or to avoid the refinement phase by returning only the "best" k candidate vectors, using the approximations (VA-LOW). Since no assumption is made on the distance to be used, both techniques fall in the **VS** class.

Example 3
In [GR00] the authors propose the P-Sphere tree, a 2-level index structure for approximate 1-NN search. In order to find the nearest neighbor for the query point, the leaf node which is closest, according to the used distance function, to the query point is accessed. The query is solved through a simple linear scan of objects contained in such node. In this case, no assumption is made on the query distance to be used (which, however, should be the same used to build the tree) and no coordinates are used, thus this technique is classified as **MS**.

3.2 Error Metrics

In this Section we review the most relevant error measures introduced to evaluate the accuracy of approximate techniques for similarity search. From the point of view of approximation, existing techniques can be classified as follows:

CS (changing space) To this class belong approximate methods that change the metric space, either by changing the distance used to compare objects or by modifying the feature space, then solve the exact problem on the so obtained approximate space. The goal is to obtain a "simpler" exact problem. Examples of such techniques are those that approximate vectors using a fixed number of bits, or dimensionality reduction techniques [CM00].

RC (reducing comparisons) Techniques in this class use the exact distance to compare objects but reduce the number of objects to be compared against the query in order to obtain a speedup with respect to the exact search. Examples of such techniques are those that prune from the search regions of the space according to the computation of bounds.

Of course, if we use a **CS** technique, the *ranking* of objects, i.e. the ordering of objects in the data-set with respect to the distance to the query object, is changing with respect with the exact case. On the other hand, using a **RC** technique, since the distance is not changed, the ranking of objects with respect to the exact case changes because some results are missed.

Therefore, **CS** methods commonly use comparisons in ranking of objects between approximate and exact results to measure the accuracy of their approximation (*ranking* measures), whereas **RC** techniques use measures of *precision/recall* (i.e. how many exact results are returned by the approximate query)[2] or measure the difference in distance between the exact and the approximate result (*distance* measures). Other measures include the percentage of correct results, i.e. the percentage of times in which the approximate result is equal to the exact result.

In the following we will describe some error measures presented in literature, classifying them by means of above categories.

Example 4
In [AMN+98] the authors propose the BBD-tree, which is a primary memory structure able to answer to approximate k-NN queries in a time that is logarithmic in the number of objects included in the data-set $\mathcal{O}$. To reduce the number of tree cells accessed, during the search the query radius is reduced by a factor of ϵ with respect to the radius used for exact search. Therefore, this method can be classified as **RC**. The measure proposed to rate the accuracy of the proposed algorithm is the *average relative error* which is defined as the ratio between the distance of the approximate NN and the exact nearest neighbor with respect to the query minus 1: This measure is also used in [CP00], where it is called *effective error* ϵ_{eff}. More formally, if we denote the query object as q, its exact NN as $nn(q)$, and the approximate NN as $\widetilde{nn}(q)$, it is

$$\epsilon_{eff} = \frac{d(q, \widetilde{nn}(q))}{d(q, nn(q))} - 1 \tag{1}$$

Clearly, ϵ_{eff} is a *distance* measure.

Example 5
In [ZSAR98] three different algorithms are presented to solve approximate k-NN queries with M-tree [CPZ97]. All of them use heuristic conditions to prematurely stop the exact k-NN search on the tree. Thus, such methods fall in the **RC** class. To measure the accuracy of proposed heuristics on k-NN queries, the *relative distance error* $\overline{\epsilon}$ is proposed, which is defined as the average ϵ_{eff_i}, $i = 1, \ldots, k$:

$$\overline{\epsilon} = \frac{1}{k} \sum_{i=1}^{k} \left(\frac{d(q, \widetilde{nn_i}(q))}{d(q, nn_i(q))} - 1 \right) \tag{2}$$

where $nn_i(q)$ and $\widetilde{nn_i}(q)$ denote the i-th exact and approximate NN, respectively. It can be easily proven that $\overline{\epsilon} \leq \epsilon_{eff_k}$.

Example 6
The technique proposed in [FTAA01] combines clustering and dimensionality reduction to approximate k-NN search. During the search, only the clusters which are closest to the query are considered and, for all the points in such clusters, only a fraction of dimensions is used to assess the distance

[2]It has to be noted that, since we only consider k-NN queries, the number of retrieved objects (the result of the approximate query) is equal to the number k of relevant objects (the result of the exact query), so that the two notions of precision and recall used in Information Retrieval [Sal88] are coincident.

to the query. To improve accuracy, the user can increase both the number of visited clusters and the fraction of considered dimensions. This technique, therefore, combines characteristics of both classes **CS** (for using only some dimensions) and **RC** (for looking up only in some clusters).

Example 7
The VA-LOW technique discussed in Example 2 belongs to the **CS** class, since the approximate results are chosen by considering only the vector approximations. One of the metrics proposed to measure the result quality is the *normalized rank sum*, i.e. the inverse of the sum of ranks in the exact result of objects in the approximate result, computed as

$$nrs = \frac{k(k+1)}{2 \cdot \sum_{i=1}^{k} rank(\widetilde{nn_i}(q))} \tag{3}$$

where the function $rank(x)$ returns the ranking of object x in the exact result. Of course, this is a *ranking* measure.

3.3 Quality Guarantees

Having determined how approximate techniques measure the result quality, it is worth considering whether each method is able to bound its performance above a predetermined level. In other words, we are asking if an approximate technique can guarantee its error measure to be lower than a (user-specified) threshold. The classification we give is as follows:

NG (no guarantees) In this class fall those methods that only use heuristic conditions to approximate the search; thus such methods are not able to give any formal bound on the error introduced by the approximation.

DG (deterministic guarantees) Techniques in this class are able to deterministically bound from above the error introduced by approximation.

PG (probabilistic guarantees) Approximate methods following this approach are only able to give probabilistic guarantees on the quality of query result. This means that, depending on the query object, the accuracy of the result can fall below the specified threshold, but, when averaging results for several queries, the quality guarantees are met. To achieve this goal, information about distribution of data is needed. In this light, techniques belonging to this class can be further divided into two basic types according to how much it is known about objects' distribution [MCS88].

PG$_{par}$ (probabilistic guarantees, parametric) Approaches in the parametric class assume that the used data-set follows a certain distribution; the only unknown information concerns a few parameters that need to be estimated (e.g. through sampling). Of course, when the considered objects do not follow the modeled distribution, quality guarantees cannot be met.

PG$_{npar}$ (probabilistic guarantees, non-parametric) In this case, little assumptions (or no assumption at all) are made on distribution of objects, so that such information has to be estimated through sampling and stored in a suitable way (e.g. through histograms).

Example 8
The third technique proposed in [ZSAR98] (see also Example 5) stops the k-NN search whenever the improvement in the distance between the query and its k-th NN falls below a threshold κ. In this case, however, no guarantee can be given on the accuracy of the approximate result, defined, for example, through the relative distance error $\overline{\epsilon}$ defined by Equation 2. Thus, this method is in the **NG** class.

Example 9
The algorithm for approximate search proposed for BBD-trees in [AMN$^+$98] (see also Example 4), and the first technique proposed in [ZSAR98] both use a value ϵ to reduce the query radius during the search. In both cases, it is guaranteed that $\epsilon_{eff} \leq \epsilon$, thus both techniques belong to class **DG**.

Example 10
In [BFG99] the DBIN structure is proposed as a 2-level index for solving the k-NN problem. The method assume that the data-set is composed of K clusters, and that distribution of objects within each cluster can be modeled by way of a Gaussian distribution, parameterized by a mean vector and a covariance matrix. At query time, the cluster that best fits the query object is found, and the NN is computed by considering objects in that cluster. Then, remaining clusters are accessed iff the probability that the NN has not been found yet is higher than a user-specified threshold. Such probability is computed by relying on the assumption of a Gaussian model, with parameters estimated at index construction time. Since the correct NN is found only with high probability and a Gaussian distribution is assumed (where mean and covariance have to be estimated), this method is in the $\mathbf{PG}_{par}$ class.

Example 11
The PAC technique proposed in [CP00] is a paradigm for approximate 1-NN search with metric access methods, where the effective error ϵ_{eff} measure (Equation 1) is allowed to exceed the user-specified accuracy threshold ϵ with a probability limited by the user-specified confidence δ. To guarantee this, the distance between the query objects and its NN is estimated from the distance distribution [CPZ98a] of indexed objects. Since this latter information is not known at query time, such distribution is estimated (through sampling) and stored (in a histogram). By above considerations, this technique can be classified in the $\mathbf{PG}_{npar}$ class.

3.4 User Interaction

The last classification we propose relates to the possibility given to the user to specify, at query time, the parameters for the search (e.g. the maximum error allowed). Some techniques, in fact, are inherently static, in the sense that a structure is built by using a set of parameters to offer some guarantees: If the user wants to change, for example, the accuracy of the result, he/she has to modify the value of the parameters and to rebuild the structure from scratch. Other methods, on the other hand, exploit a single structure that is not bound to any parameter and can be used with different sets of parameters, according to user's needs.

SA (static approach) When using a technique in this class, the user cannot vary the set of parameters for query approximation, but is bound to those specified when building the approximate structure. Usually, to provide several quality of result profiles, different structures are built, using different sets of parameters, and the user is given the possibility to choose the structure that best fits his/her actual needs.

IA (interactive approach) Methods in this class are not bound to a specific set of parameters, but can be interactively used by varying such parameters at query time. Usually, interactive techniques are obtained as modifications of the exact similarity search method, that can be executed by requesting a maximum error of 0%.

Example 12
In the P-Sphere technique presented in [GR00] (see also Example 3), the size of leaf nodes, i.e. the number of objects in each data page, is estimated by taking into account a user-specified accuracy. Of course, if the accuracy parameter is changed, the P-Sphere tree has to be rebuilt from scratch. Therefore, this method is static and belongs to the **SA** class.

Example 13
In [HAK00] the authors propose the generalized NN search as a new approach for high-dimensional NN search. The key idea here is to find a suitable projection to reduce the space dimensionality; then, the NN search is performed on the reduced space using the original distance function and projected points. Of course, the higher the value of the dimensionality D' of the reduced space, the better accuracy is obtained by this technique. Since the user can specify, at query time, the value of D', this method can be classified as **IA**.

4 Some Relevant Cases

In this Section we use the schema introduced in Section 3 to classify some of the approaches for approximate similarity search presented in recent years. For each method presented in the following Sections, the classification is expressed as a 4-tuple consisting of the following "coordinates": $(<$ *data type* $>, <$ *error metric* $>, <$ *quality guarantee* $>, <$ *user interaction* $>)$.

4.1 *Fastmap* [FL95]: $(\mathbf{MS}, \mathbf{CS}, \mathbf{NG}, \mathbf{SA})$

The *Fastmap* technique [FL95] has been proposed as a tool for mining and visualization of metric data-sets. In its essence, the *Fastmap* algorithm is able to map a set of objects drawn from a generic metric space to a D'-dimensional Euclidean space, where D' is a user-specified value, such that distances between objects are preserved as much as possible. Of course, this approach can also be used for approximate searching, since performing a similarity search in the target D'-dimensional space can be viewed as an approximate search in the original metric space. Since the method applies to general metric spaces, it belongs to the **MS** class; the transformation of the space is reflected in a transformation of the distance used to compare objects, thus this technique is in the **CS** class; in the paper, the authors give no guarantee on the error introduced for distance in the target space,[3] hence the quality guarantee class is **NG**; finally, as for user interaction, the mapping in the D'-dimensional space has to be made before any index structure is built on the transformed objects, thus *Fastmap* falls in the **SA** class.

[3]The error between exact distances and distances between transformed objects can be limited, for the relevant case of vector spaces, by exploiting the Johnson-Lindenstrauss lemma. However, for the general case of metric spaces, no general rule has been proposed so far.

4.2 DBIN [BFG99]: $(\mathbf{VS}, \mathbf{RC}, \mathbf{PG}_{par}, \mathbf{IA})$

The DBIN (density based indexing) method was presented in [BFG99] as an approach to solve approximate similarity queries in high-dimensional spaces. The base assumption is that the distribution of objects in the space can be modeled as a mixture of Gaussian distributions. Each point, therefore, can be associated to a cluster, parametrized with a mean vector and a covariance matrix, by using an expectation-maximization algorithm. When searching for the NN of a query point, the clusters obtained in the building phase are ranked according to the probability that the query point belongs to them; then, each cluster is accessed (and points in that cluster compared to the query) until the probability that the NN has not been found falls below an user-specified tolerance. Since no assumption is made on the distance used to compare vectors (even if analytical results are given only in the case of quadratic form distance functions), this method falls in class **VS**; the distance used to compare vectors is the exact distance, thus the class of this technique is **RC**; as for quality guarantees, this technique assumes that indexed objects are distributed in clusters according to a Gaussian distribution, for which the mean and the covariance are estimated in the building phase, hence this method belongs to class $\mathbf{PG}_{par}$; since the user can specify the tolerance parameter, used when stopping the search, this method is of class **IA**. Finally, the metric used to measure quality of the result is the percentage of times the exact NN has been found (this is a *precision/recall* measure).

4.3 PAC [CP00]: $(\mathbf{MS}, \mathbf{RC}, \mathbf{PG}_{npar}, \mathbf{IA})$

PAC (probably approximately correct) nearest neighbor queries, introduced in [CP00], represent a probabilistic approach to approximate 1-NN search in metric spaces, where the error in the result can exceed a specified accuracy threshold ϵ with a probability that is limited by a confidence parameter δ. The PAC paradigm can be applied to *any* distance-based (either multi-dimensional or metric) index tree that is based on a recursive and conservative decomposition of the space (thus it is of **MS** class). The only information that is needed by the algorithm to prune index nodes from the search is the value of r^q_δ, the maximum value of distance from the query object q for which the probability that the exact NN of q has a distance lower than d is not greater than δ:

$$r^q_\delta = \sup\{r | \Pr\{d(q, nn(q)) \leq r\} \leq \delta\} \tag{4}$$

In the paper, this value is estimated by using the distance distribution of indexed objects with respect to the query object (estimated through sampling and stored as an histogram); it is therefore clear that this approach is probabilistic and non parametric ($\mathbf{PG}_{npar}$ class). The distance used to query the distance-based index structure is the exact one, the approximation is introduced by reducing the number of object to be compared against the query object q by means of r^q_δ and of the ϵ parameter, and quality of the result is measured by the effective error ϵ_{eff} defined by Equation 1, thus the class of this approach is **RC**; finally, since the accuracy and the confidence parameters (ϵ and δ, respectively) can be specified at query time, this technique belongs to the **IA** class.

4.4 VA-BND [WB00]: $(\mathbf{VS}, \mathbf{RC}, \mathbf{PG}_{npar}, \mathbf{IA})$

In [WB00] two approximate query evaluation techniques are presented for the VA-File. The VA-File structure [WSB98] approximates vectors using a fixed number of bits, and stores such approximations in a file. For exact k-NN search, the approximation file is sequentially scanned to exclude vectors that can not be in the result set through the computation of bounds on exact distances (such scan is very fast since the computation of bounds between approximations has to consider only a few bits); finally, exact vectors corresponding to approximations included in the result of the previous scan (the candidate vectors) are compared against the query point to compute the final result. Since the approximations of the VA-File are only applicable to vector spaces and any distance can be used to compare vectors (even if computation of bounds can be a difficult task if complex metrics are used), all approximate techniques developed for this structure fall in the **VS** class. In the paper, two different metrics are proposed to measure the accuracy of approximate techniques:

1. The *ratio of false dismissals* rfd, defined as

$$rfd = \frac{1}{k} \cdot \sum_{i=1}^{k} \begin{cases} 1 & \text{if } rank(\widetilde{nn_i}(q)) > k \\ 0 & \text{otherwise} \end{cases} \tag{5}$$

   which counts the number of exact results not included in the approximate result (divided by k). This can be classified as a *precision/recall* measure and it is the same measure used in [BFG99] (see Section 4.2), where it is called *discounted accuracy.*

2. The *normalized rank sum* nrs, defined by Equation 3, is computed as the inverse of the sum of rankings in the exact result of objects in the approximate result, normalized in order to obtain a value in the interval $[0, 1]$. This is a *ranking* measure and it is used to better distinguish the quality of result with respect to the above measure.

The first approach to reduce the complexity of similarity searching in the VA-File through approximation proposes to adapt the computation of distance between approximate vectors. The user is given the possibility to specify a value α to adapt computed bounds: Higher values of α correspond to higher errors in the result but the candidate set will consist in a lower number of vectors. Since the approximation is introduced in the computation of bounds and not on the exact distance, this technique can be classified as **RC**. The number of vectors missed can be computed as a function of the distance distribution between objects, thus this technique can give probabilistic guarantees on the ratio of false dismissals rfd as a function of the parameter α; therefore, the class for this method is $\mathbf{PG}_{npar}$. Finally, since the parameter α can be specified at query time, the VA-BND technique is in class **IA**.

4.5 VA-LOW [WB00]: $(\mathbf{VS}, \mathbf{CS}, \mathbf{DG}, \mathbf{SA})$

The second approximate technique for the VA-file (also presented in [WB00]) proposes to completely omit the second refinement phase and to return, as the approximate result, the k vectors corresponding to the best approximations. Of course, in this case, the approximation in the result comes from using, in computing the distance, the approximate vectors instead of the exact vectors, thus this method can be classified as **CS**. The error in computation of the distance on approximate vectors can be controlled by means of the quantity of bits used for the approximations: The more bits are used, the better the approximation but the slower the first sequential phase. Since a bound on the error between the distance on approximate vectors and the exact distance can be easily computed, this technique falls in the **DG** class. As for the interaction with the user, it is clear that the only parameter used, i.e. the number of bits used for vectors' approximation, has to be specified before the actual VA-File is built, so that the class for this technique is **SA**.

5 Comments and Extensions

We believe that the proposed classification schema can be very fruitful for the analysis of approximate techniques for similarity search. By using such schema interesting relations and similarities between techniques can be found that may not be evident at a first sight. As an example, consider the PAC approach (reviewed in Section 4.3) and the VA-BND technique (Section 4.4): Both are classified as belonging to the **RC**, $\mathbf{PG}_{npar}$, and **IA** classes, the only difference being in the fact that the VA-File only applies to vector spaces. Indeed, at a closer look, these two method share several analogies:

- In both cases the approach requests for an additional confidence parameter (δ and α, respectively) representing the quality of the result the user is willing to obtain. The lower the value of the parameter, the lower the error and the higher the search costs.
- Both methods use information about the distance distribution in order to estimate the distance between the query object and its nearest neighbor.
- In both cases the distance distribution and the confidence parameter are jointly used to derive bounds to stop the search.
- Different error metrics are proposed for the two methods, but estimates on other metrics can be easily obtained (e.g. on rfd and nrs for PAC, and on ϵ_{eff} for VA-BND).

It is clear that, by using the proposed classification schema, we are able to immediately understand the field of applicability of a particular approximate technique. In this way, we can conceive whether, for example, a method is more general, i.e. it applies to a superset of scenarios, with respect to another, or how its quality measures relate to those proposed for other techniques. In search for the "best" approximate technique for a specific scenario at hand, in fact, different aspects are to be considered, in particular the generality/efficiency trade-off: A more general method is expected to have a lower efficiency (i.e. to lead to higher search costs) than a method that applies to a lower number of cases, since it is expected that the latter is able to exploit some domain-specific information that the former cannot to take into account; this applies immediately when considering methods that apply to metric spaces or only to vector spaces. The same considerations can be made when dealing with quality guarantees: Parametric approaches usually attain better performance with respect to non-parametric ones, yet they are only applicable to particular distributions of objects. On the other hand, deterministic techniques, in some senses, are the most general ones, since guarantees are met in all possible cases and not only in a probabilistic way. Finally, it is clear that interactive approaches are more general than static ones, since the user is given the possibility to choose at query time the desired quality of the result, which is inversely related to search costs needed to obtain the approximate result.

Limits of approximate techniques can also be discovered by means of the proposed classification schema. As an example, consider the PAC technique reviewed in Section 4.3: Since only the distance between the query object and its nearest neighbor is estimated, it is clear that such approach cannot deal with approximate k-NN queries when $k > 1$. In order to extend the PAC approach to generic k-NN queries, we need to estimate the distance between the query object and its k-th nearest neighbor, e.g. by using formulas from [CPZ98a].

Finally, we would like to point out some issues that are currently open for research. In our view, the most pressing one is that of extending approximate similarity techniques to deal with the case of *complex* queries, where different similarity predicates are jointly evaluated to derive the similarity between the object and the query [Fag96, CPZ98b]. In the simplest case, all the similarity predicates refer to a single *feature*, thus, for example, we could ask for the objects which are most similar to a query object q_1 *and* to a query object q_2: In this case, the overall similarity score for an object $O \in \mathcal{U}$ is obtained by computing the similarity between O and q_1, and that between O and q_2, and combining them through a scoring function [Fag96]. In order to solve complex queries, some existing approaches suggest to independently solve the two sub-queries, i.e. to consider objects that are sufficiently close to q_1 *or* to q_2, and then to combine the so-obtained results [Fag96, GBK00, WFSP00], whereas other techniques are able to consider the complex query *as a whole* and to process it with classical distance-based access methods [CPZ98b]. The former approach is usually much less efficient than the latter, since a lot of work is wasted during the first phase to consider objects that are not contained in the final result (for a comparison, see [CPZ98b]). Besides specific problems, however, it is clear that this kind of similarity search is affected by the same problems that limit (simple) similarity search, thus one could conceive approximate techniques applying to complex similarity queries. By our knowledge, no previous work dealt with such issue, probably because of the preliminary state for approximate techniques for simple similarity search. We plan to investigate this issue in the future.

References

[AMN+98] Sunil Arya, David M. Mount, Nathan S. Netanyahu, Ruth Silverman, and Angela Y. Wu. An optimal algorithm for approximate nearest neighbor searching. *Journal of the ACM*, 45(6):891–923, November 1998.

[BCW00] Ilaria Bartolini, Paolo Ciaccia, and Florian Waas. Using the wavelet transform to learn from user feedback. In *Proceedings of the First DELOS Network of Excellence Workshop on Information Seeking, Searching and Querying in Digital Libraries*, Zurich, Switzerland, December 2000. Online publication `http://www.ercim.org/publication/ws-proceedings/DelNoe01/`.

[BFG99] Kristin P. Bennett, Usama M. Fayyad, and Dan Geiger. Density-based indexing for approximate nearest-neighbor queries. In *Proceedings of the Fifth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD 1999)*, pages 233–243, San Diego, CA, August 1999.

[CGRS00] Kaushik Chakrabarti, Minos N. Garofalakis, Rajeev Rastogi, and Kyuseok Shim. Approximate query processing using wavelets. In *Proceedings of 26th International Conference on Very Large Data Bases (VLDB 2000)*, pages 111–122, Cairo, Egypt, September 2000.

[CM00] Kaushik Chakrabarti and Sharad Mehrotra. Local dimensionality reduction: A new approach to indexing high dimensional spaces. In *Proceedings of 26th International Conference on Very Large Data Bases (VLDB 2000)*, pages 89–100, Cairo, Egypt, September 2000.

[CP00] Paolo Ciaccia and Marco Patella. PAC nearest neighbor queries: Approximate and controlled search in high-dimensional and metric spaces. In *Proceedings of the 16th*

International Conference on Data Engineering (ICDE 2000), pages 244–255, San Diego, CA, March 2000.

[CPZ97] Paolo Ciaccia, Marco Patella, and Pavel Zezula. M-tree: An efficient access method for similarity search in metric spaces. In *Proceedings of the 23rd International Conference on Very Large Data Bases (VLDB'97)*, pages 426–435, Athens, Greece, August 1997.

[CPZ98a] Paolo Ciaccia, Marco Patella, and Pavel Zezula. A cost model for similarity queries in metric spaces. In *Proceedings of the 16th ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems (PODS'98)*, pages 59–68, Seattle, WA, June 1998.

[CPZ98b] Paolo Ciaccia, Marco Patella, and Pavel Zezula. Processing complex similarity queries with distance-based access methods. In *Proceedings of the 6th International Conference on Extending Database Technology (EDBT'98)*, pages 9–23, Valencia, Spain, March 1998.

[EKSX96] Martin Ester, Hans-Peter Kriegel, Jörg Sander, and Xiaowei Xu. A density-based algorithm for discovering clusters in large spatial databases with noise. In *Proceedings of the 2nd International Conference on Knowledge Discovery and Data Mining (KDD-96)*, pages 226–231, Portland, OR, 1996.

[Fag96] Ronald Fagin. Combining fuzzy information from multiple systems. In *Proceedings of the 15th ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems (PODS'96)*, pages 216–226, Montreal, Canada, June 1996.

[FL95] Christos Faloutsos and King-Ip Lin. FastMap: A fast algorithm for indexing, data-mining and visualization of traditional and multimedia datasets. In *Proceedings of the 1995 ACM SIGMOD International Conference on Management of Data*, pages 163–174, San Jose, CA, June 1995.

[FTAA01] Hakan Ferhatosmanoglu, Ertem Tuncel, Divyakant Agrawal, and Amr El Abbadi. Approximate nearest neighbor searching in multimedia databases. In *Proceedings of the 17th International Conference on Data Engineering (ICDE 2001)*, pages 503–511, Heidelberg, Germany, April 2001.

[GBK00] Ulrich Güntzer, Wolf-Tilo Balke, and Werner Kießling. Optimizing multi-feature queries for image databases. In *Proceedings of the 26th International Conference on Very Large Data Bases (VLDB 2000)*, pages 419–428, Cairo, Egypt, September 2000.

[GIM99] Aristides Gionis, Piotr Indyk, and Rajeev Motwani. Similarity search in high dimensions via hashing. In *Proceedings of 25th International Conference on Very Large Data Bases (VLDB'99)*, pages 518–529, Edinburgh, Scotland, UK, September 1999.

[GR00] Jonathan Goldstein and Raghu Ramakrishnan. Contrast plots and P-Sphere trees: Space vs. time in nearest neighbor searches. In *Proceedings of 26th International Conference on Very Large Data Bases (VLDB 2000)*, pages 429–440, Cairo, Egypt, September 2000.

[HAK00] Alexander Hinneburg, Charu C. Aggarwal, and Daniel A. Keim. What is the nearest neighbor in high dimensional spaces? In *Proceedings of 26th International Conference* *on Very Large Data Bases (VLDB 2000)*, pages 506–515, Cairo, Egypt, September 2000.

[JD88] Anil K. Jain and Richard C. Dubes. *Algorithms for Clustering Data.* Prentice-Hall, 1988.

[JMF99] Anil K. Jain, M. N. Murty, and P. J. Flynn. Data clustering: A review. *ACM Computing Surveys*, 31(3):264–323, September 1999.

[LCGMW] Chen Li, Edward Y. Chang, Hector Garcia-Molina, and Gio Wiederhold. Clustering for approximate similarity search in high-dimensional spaces. *IEEE Transactions on Knowledge and Data Engineering.* To appear.

[MCS88] M. V. Mannino, P. Chu, and T. Sager. Statistical profile estimation in database systems. *ACM Computing Surveys*, 20(3):191–221, September 1988.

[Sal88] Gerard Salton. *Automatic Text Processing: The Transformational, Analysis, and Retrieval of Information by Computer.* Addison-Wesley, Reading, MA, 1988.

[SK97] Thomas Seidl and Hans-Peter Kriegel. Efficient user-adaptable similarity search in large multimedia databases. In *Proceedings of the 23rd International Conference on Very Large Data Bases (VLDB'97)*, pages 506–515, Athens, Greece, August 1997.

[WB00] Roger Weber and Klemens Böhm. Trading quality for time with nearest-neighbor search. In *Proceedings of the 7th International Conference on Extending Database Technology (EDBT2000)*, pages 21–35, Konstanz, Germany, March 2000.

[WFSP00] Leejay Wu, Christos Faloutsos, Katia P. Sycara, and Terry R. Payne. FALCON: Feedback adaptive loop for content-based retrieval. In *Proceedings of 26th International Conference on Very Large Data Bases (VLDB 2000)*, pages 297–306, Cairo, Egypt, September 2000.

[WSB98] Roger Weber, Hans-Jörg Schek, and Stephen Blott. A quantitative analysis and performance study for similarity-search methods in high-dimensional spaces. In *Proceedings of the 24th International Conference on Very Large Data Bases (VLDB'98)*, pages 194–205, New York, NY, August 1998.

[WYM97] Wei Wang, Jiong Yang, and Richard R. Muntz. STING: A statistical information grid approach to spatial data mining. In *Proceedings of the 23rd International Conference on Very Large Data Bases (VLDB'97)*, pages 186–195, Athens, Greece, August 1997.

[ZSAR98] Pavel Zezula, Pasquale Savino, Giuseppe Amato, and Fausto Rabitti. Approximate similarity retrieval with M-trees. *The VLDB Journal*, 7(4):275–293, 1998.