Similarity Searching in the CORDIS Text Database*

Euripides G.M. Petrakis†
Department of Electronic and Computer Engineering
Technical Univ. of Crete, Chania, Greece
e-mail: petrakis@ced.tuc.gr

Kostas Tzeras‡
GMD-IPSI
Darmstadt, Germany
e-mail: tzeras@darmstadt.gmd.de

January 31, 2001

Abstract

Similarity searching in text databases with multiple field types is still an open problem. We focus our attention on the "COmmunity Research and Development Information Service" (CORDIS) database of the European Union and we evaluate the effectiveness of many text retrieval methods in terms of *precision*, *recall* and *ranking quality*. Our experiments indicate that different field types should be handled by different retrieval methods.

Index Terms: text database, text retrieval, similarity searching.

1 Introduction

The ever-increasing on-line availability of text databases on the Internet has changed the way that end users access such databases. Modern text retrieval systems are mainly targeted to ordinary end users and can be characterized as "similarity retrieval systems". They encourage the use of unstructured "natural language" queries by example text (e.g., find all documents with a title like "*modeling and reuse of information over time"*) and return lists of texts ranked according to their similarity with the query. Older text retrieval systems are referred to as "Boolean retrieval systems" and support Boolean queries (e.g., find all documents with titles containing the terms *"modeling"*, *"reuse"*, *"information"* and *"time"*). They return texts ordered by date, author etc. but, often, overstrain users with unsatisfactory results. However, neither mechanism guarantees superior performance for every query [1]. Therefore, systems that combine Boolean with Similarity retrieval methods may yield better results.

We focus our attention on text databases organized into various field types differing in content type and character length. Examples of such databases are scientific and bibliographic databases which hold information on scientific publications, authors, citations etc. The CORDIS database of the European Union (EU) is a characteristic example. CORDIS is a very popular EU database widely available to the

*Software Practice and Experience, No. 13, Vol. 30, pp. 1447-1464, Nov. 2000.

†This author was supported by an ERCIM fellowship, Contract Nr. 97-02. The work was carried out while E. Petrakis was visiting GMD-IPSI.

‡Current address of K. Tzeras is Metropolis Worknet SA., Stratigou Kallari 7, GR-54622, Thessaloniki, Greece, e-mail: tze@metropolis.com.gr.

public in CD-ROM and on the Internet[1]. Currently, CORDIS offers simple search capabilities based on Boolean queries and exact string searching.

We need to extend the search mechanism of CORDIS with similarity retrieval capabilities. In this work, we study the performance of well known similarity retrieval methods on CORDIS and we propose the most *effective* access method for each field type. The user is the *novice searcher* who is not familiar with complex retrieval strategies and heuristics and is not willing to engage in lengthy interactions with the system. The queries are "by example": The user specifies an example text (typically a name or sentence that describes his or her information need). We have to retrieve documents that contain fields with similar text with the query, ranked by descending similarity with respect to the query.

This investigation is a follow-up of our earlier work for project ProCORDIS[2] (Multi-lingual indexing and search in the CORDIS databases), an EU project (contract No. CSA95C) that was running at GMD-IPSI in collaboration with the European Commission from May 1996 until July 1997. ProCORDIS explored the feasibility of applying modern text indexing and retrieval methods in a WWW-based environment that would improve user's access in CORDIS. In this project, we experimented with probabilistic retrieval methods, methods based on abductive logic, sophisticated proper name recognition methods etc. We also experimented with a multi-lingual indexing method allowing non-English language access in the CORDIS's English language databases. We developed a prototype system which includes an improved search engine with thesaurus-aided free-text search. It also supports multi-lingual classification of documents, automatic indexing using a controlled vocabulary and translation of non-English language queries.

The rest of this paper is organized as follows: A review of related work on text retrieval and a short presentation of the underlying theory is presented in Section 2. The CORDIS database is described in short in Section 3. Our evaluation methodology is presented in Section 4. Our experiments on proper names and long text fields (e.g., titles, abstracts) are discussed in Section 5 and in Section 6 respectively. Finally, conclusions and issues for future research are discussed in Section 7.

2 Related Work

A wide arsenal of text retrieval methods has been proposed in the literature (e.g., [2]) and some have been integrated into commercial systems (e.g., Yahoo[3], AltaVista[4], INQUERY [3] etc.) and experimental prototypes (e.g., SCAM [4], COPS [5], GLIMPSE [6], SMART [7], LSI [8] etc.). The effectiveness of retrieval varies depending on the application domain, the query type and on the characteristics of the stored text. For example, GLIMPSE is effective in retrieving unstructured text on queries with relatively small length (i.e., words and small phrases) that contain (possibly) misspelled terms while, SCAM and COPS have been developed for plagiarism detection in Digital Libraries and work more effectively with structured large text documents like conference papers, user manuals, technical descriptions, etc. INQUERY, LSI and SMART detect general content similarity between texts.

Below we present a short review of the similarity retrieval methods we considered in this work. This is not an exhaustive review. The methods below were selected based on their availability and as representative of a much broader class of methods.

2.1 Error Counting Methods

These are methods which can be used to retrieve similar words (e.g., names, keywords) or phrases. The query or the stored text may be mispelt or have similar pronunciation. The existing methods fall into

[1] `http://www.apollo.codrdis.lu`
[2] `http://www.darmstadt.gmd.de/delite/Projects/ProCordis/index.html`
[3] `http://www.yahoo.com`
[4] `http://altavista.telia.com`

two categories:

String similarity methods. They compute a numerical estimate of the similarity between two strings. They count the number of characters (or syllables) the two strings have in common or the number of steps required to transform the one string to the other. *Edit distance* and *n-grams* are the most characteristic methods of this category. Methods such as "agrep"[5], retrieve words with up to k errors (k is user defined) with respect to the query. Matching is binary in this case and ranking of the retrieved words is not possible.

Phonetic coding methods can be used to search for words with similar pronunciation. *Soundex* and *Phonix* [9] are the most characteristic examples of this category. Both these methods were developed for the English language and they have to be modified for other languages.

2.1.1 Edit Distance

The edit distance between two strings (referred to as *query* and *reference* respectively) is defined as the cost to transform the query to the reference string (or vice versa) following a sequence of *edit* operations. Usually, three operation are allowed, namely, *insert, delete* and *substitute*. We also adopt the *Damerau-Lavenstein* version of the edit distance, which, in addition to insertion, deletion and substitution, allows for *transposition* of characters with the same cost as the previous basic operations (1 in this work). Figure 1 shows the basic recurrence relation for the Damerau-Lavenstein algorithm we used:

$$
\begin{aligned}
&edit(0,0) = 0 \\
&edit(i,0) = i; \\
&edit(0,j) = j; \\
&edit(i,j) = \quad \min\{edit(i-1,j)+1, edit(i,j-1)+1, \\
&\qquad\qquad edit(i-1,j-1)+d(s_i,t_j), \\
&\qquad\qquad edit(i-2,j-2)+d(s_i,t_{j-1})+d(s_{i-1},t_j)+1\}
\end{aligned}
$$

Figure 1: *Recurrence relation of the Damerau-Levenstein edit distance.*

$d(s_i, t_j)$ is a function that counts the distance between letters s_i and t_j. In our case, $d(s_i, t_j) = 1$ if $s_i \neq t_j$ and 0 otherwise. The last term in the min formulae corresponds to transposition of letters and its missing from the raw (original) edit distance. For example, $edit(co\mathbf{r}dis, codis) = 1$ (**r** is deleted), $edit(ordis, \mathbf{c}odis) = 1$ (**c** is inserted), $edit(cor\mathbf{d}is, cor\mathbf{t}is) = 1$ (**d** substitutes **t**) and $edit(co\mathbf{rd}is, co\mathbf{dr}is) = 1$ (transposition of **rd**).

2.1.2 n-Grams

An n-gram of a string s is a substring of s of length n. A simple measure of the similarity between strings s and t counts the number of n-grams that s and t have in common. The Ukkonen version of n-gram distance [10] takes string lengths into account:

$$n - gram - distance(s,t) = \sum_{g \in G_s \cup G_t} |s[g] - t[g]|, \tag{1}$$

where, G_s, G_t are the sets of n-grams in s, t respectively and $s[g]$, $t[g]$ denote the number of occurrences of n-gram g in strings s, t respectively. For example, if $n = 2$, $G_{cordis} = \{co, or, rd, di, is\}$,

[5]GLIMPSE is based on "agrep".

$G_{codis} = \{co, od, di, is\}$ and $n - gram - distance(cordis, codis) = 3$ since G_{cordis} and G_{codis} have 3 different digrams, the "or", "rd" and "od".

Assuming that neither s nor t contain repeated n-grams, the above distance function is expressed as

$$n - gram - distance'(s,t) = |G_s| + |G_t| - 2|G_s \cap G_t|. \quad (2)$$

Equation 2 is an approximation to the original distance but computes faster than Equation 1. In our test data, less than 2% of strings contain a repeated 2-gram while, repeated 3-grams are even more rare. Even if a string contains a repeated n-gram, the error introduced by Equation 2 is small.

2.2 Text Similarity Methods

These methods can be used to measure content similarity on longer text fields with (possibly) many words (e.g., titles, abstracts etc.). Notice that, these methods measure similarity instead of distance. Below we describe the methods which we used in this work.

2.2.1 Cosine Similarity Measure: The SMART System

The *cosine similarity measure* relies on the Vector Space Model (VSM), one of the most well-known text retrieval methods. SMART [11] is probably the most popular implementation of VSM. SMART has been applied on many Information Retrieval collections like TREC, CACM and CISI with promising results [12].

A text (or query) T is represented by a multidimensional vector $F(T) = (F_1(T), F_2(T), \ldots F_k(T))$, called "occurrence vector". The dimensionality k of the vector equals the number of distinct terms occurring in the database. A term is usually defined as a stemmed non stop word. $F_i(T)$ is a function of the frequency of the i-th term in T and can be defined in many different ways. In our implementation, $F_i(T)$ is defined as

$$F_i(T) = \frac{1}{2}\left(1 + \frac{tf_i^T}{\max tf_i^T}\right) \log \frac{N}{n_i}, \quad (3)$$

where tf_i^T is the frequency of the i-th term in T, $\max tf_i^T$ is the number of database documents where the most frequent term of T occurs, N is the number of database entries and n_i is the number of entries where the i-th term occurs.

The cosine similarity measure between a query (A) and a stored document (B) is defined as

$$Cos_{similarity}(A,B) = \frac{\sum_{i=1}^{N} \alpha_i F_i(A) F_i(B)}{\sqrt{\sum_{i=1}^{N} \alpha_i^2 F_i(A)^2 \sum_{i=1}^{N} \alpha_i^2 F_i(B)^2}}, \quad (4)$$

where $F(A)$, $F(B)$ are the occurrence vectors of A, B respectively and α_i are user-determined parameters (weights). In this work we set $\alpha_i = 1$.

2.2.2 Latent semantic indexing (LSI)

Each text (or query) is represented by an occurrence vector as in VSM. All database documents (actually their occurrence vectors) form the so called "term-document matrix". LSI [13] uses Singular Value Decomposition (SVD) to transform the term-document matrix into a set of k (typically 100 to 300) orthogonal factors (or axes) from which the original matrix can be approximated by linear combination. Both documents and terms are represented in this space. The location of a term in this space reflects the correlation of its usage across documents.

Given a query, it is mapped to a k-dimensional vector in the above space. All documents can then be ranked according to their proximity (similarity) with the query vector. The similarity between the

query and a stored document can be computed using measures applicable to vectors such as that of Equation 4. LSI has been applied on many Information Retrieval collections like MED and CISI with promising results [14].

2.2.3 Bayesian Networks: The INQUERY System

This method is also known as *"Inference Net Model"* [15]. Both, database documents and queries are preprocessed, their information content (e.g., usually in the form of stemmed non stop words) is extracted and represented by a Bayesian network. This network consists of four kinds of nodes: (a) The *"document nodes"* representing all database documents, (b) The *"document concept nodes"* representing concepts derived from the stored documents, (c) The *"query nodes"* representing concepts derived from the queries and (d) The *"query concept nodes"* representing specific queries. A query or document concept is usually defined as a stemmed non stop word.

During query processing, probabilities are assigned to network nodes, the corresponding document nodes are instantiated, the probabilities are propagated through the network, and finally, a probability is associated with the query nodes.

INQUERY [3] is probably the most successful implementation of this method. INQUERY has been applied on many Information Retrieval collections with promising results. The results of the TREC-2 conference [12] have established INQUERY as one of the most effective retrieval systems.

3 CORDIS Database

CORDIS holds information on EU programs and funded projects, persons and organizations involved, project results, publications, commission documents and news. Typical end users are researchers, engineers, managerial or administrative stuff searching for state-of-the-art publications, planned and running projects, potential project partners, technical documentation etc.

CORDIS consists of a multitude of fields (over 250) providing mostly textual information. In our experiments we used the fields shown in Table 1. These contain information on person surnames (SURNAME), project titles (TITLE), publication abstracts (ABSTRACT) and general information about projects (GEN_INFORMATION). The reasons for choosing these fields are:

- They are the most commonly used fields for searching in CORDIS and are representative of many more additional fields such as project names, addresses, project results etc.
- They are important for the type of service CORDIS aims to offer, that is, search of information on EU research activities and on potential partners for EU projects.
- They typically occur in text databases with similar content such as in scientific - technological databases.

Field name	Number of records	Mean length (words)	Mean length (characters)
SURNAME	18,971	1.08	8.5
TITLE	28,051	10.14	86.64
ABSTRACT	18,940	33.69	256.37
GEN_INFORMATION	18,832	132.98	1024.21

Table 1: *CORDIS fields used in this work.*

We decided to exclude a number of CORDIS fields from our experiments, either because they are used infrequently or because they are not yet maintained in CORDIS. An additional problem with fields such as keywords and organization names is the that their format is not yet standard and they contain abbreviations together with full text names.

4 Evaluation Method

In the following, we present the evaluation criteria and the evaluation measures we used in this work.

4.1 Evaluation Criteria

We used human relevance judgments to compute the effectiveness of each method. These relevance judgments have been suggested by Mrs. Karin Mölle, employee at GMD-IPSI, a professional documentalist with over 20 years experience with text retrieval, real user's needs on European project databases such as CORDIS and evaluation methodologies. The same person carried out the relevance judgments for our experiments.

4.1.1 Similarity of Names

Even if we know the correct spelling of the name we are looking for, the search will not be always successful. This is because the database may contain the same name in several forms and even worst, we may not be certain about the correct spelling of the name. In particular, we have to deal with the following two sources of uncertainty:

Database errors: The same name can be entered into the database in several forms (e.g., Kourtis, Courtis), together with middle names or prefixes (e.g., Fouqet, De Fouqet), or mis-spelled (e.g., Raymon, Raymond) or with typing errors (e.g., Curtis, Cvertis). Moreover, the exact spelling of a name may be unknown or the spelling of a given name may vary or there may exist no single commonly accepted spelling. For example, in CORDIS, there are at least 4 correct spellings for the name Kourtis (i.e., Kourtis, Kurtis, Curtis, Courtice) not including the feminine versions of these spellings (e.g., Kourti, Curti).

User uncertainty: A name may be known by its pronunciation (e.g., when one has heard the name) rather than its spelling. There may exist several forms of the same name with similar pronunciation (e.g., Walter, Valter sound very similarly in German, Raymon, Ramon etc.).

The search for names in CORDIS is approximate: A search method (e.g., the edit distance) returns all names with up to a number of errors (e.g., 4 or 5) or the best k (e.g., 50) answers. However, not all these answers are always correct. For example, the edit distance between the names Becker and Bucher is 2 but they are not similar. At the same time, the edit distance between the names Curtis and Courtice is 3 but they must be considered similar.

In our evaluations, a name is considered similar to a query if it is either identical or a spelling variant that sounds about the same with the query (e.g., Matei, Mattei, Mathei). If a name contains an obvious typing mistake with respect to the query, it is also judged similar (e.g., Curtis, Cvrtis). To cope with foreign language users (CORDIS is a multilingual database), surnames are also considered similar if they differ in consonants or syllables but their pronunciation is not affected significantly (e.g., Walter, Valter and Kourtis, Curtis, Curtice and Fouqart, De Fouqart).

4.1.2 Similarity of Long Text Fields

Similarly, a search method on long text fields (i.e., title, abstract or general information) will return all entries with up to a prespecified degree of similarity (tolerance), or the k (e.g., 50) best answers. In our evaluations, a database entry other than a surname is considered similar to a query, if

- It is originated from the same domain (e.g., medicine, economics, agriculture etc.) and if it is dealing with one or more essential aspects formulated within the query (e.g., *query:* "epidemiology of drug resistance in plasmodium falciparum", *aspect 1:* "epidemiology of drug resistance", *aspect 2*: "epidemiology of plasmodium falciparum - parasite of malaria").

- For queries and database entries that could not be uniquely classified under a single domain, special effort has been made to identify the different domains addressed in order to consider also interdisciplinary searches in the evaluation. For example, a document is considered similar to the query if any piece of it corresponds to one of the domains addressed in the query (i.e. it is at least partially relevant to the query).

Both domains and aspects have been determined by the information expert after analyzing the informational content of the CORDIS database. The systematic of domains and aspects has been developed based on our earlier work for project ProCORDIS and relies on the combination of the OECD Thesaurus systematic, the categorization used by the current CORDIS retrieval system and extensions of these based on our indexing work for ProCORDIS.

4.2 Evaluation Measures

To measure effectiveness, for each candidate method we computed:

Precision that is, the percentage of relevant entries retrieved with respect to the total number of retrieved entries.

Recall that is, the percentage of relevant entries retrieved with respect to the total number of relevant entries in the database. Because we don't have the resources to compare every query with each database entry, for each query, we merged the answers obtained by all candidate methods and we considered this as the database which is manually inspected for relevant entries. This is a valid sampling method known as *"pooling method"* [16]. This method does not allow for absolute judgments such as "method A misses 10% of the total relevant answers in the database". It provides, however, a fair basis for comparisons between methods allowing judgments such as "method A returns 5% fewer correct answers than method B".

Ranking quality R_{norm} which computes the differences between the ranking of the results obtained by a method and by a human expert [17]. The higher the value of R_{norm} the better the ranking quality of a method, that is, the method assigns higher ranks to the relevant entries. To compute R_{norm} we proceed as follows:

1. The answers of the candidate method are evaluated by an expert (i.e., each answer is judged as relevant or not relevant).
2. Each answer is assigned a "rank" which equals its order within the answer set (i.e., the first answer has rank 1, the second answer has rank 2 and so on).
3. We take these answers in pairs such that (a) Only pairs with one relevant and one irrelevant answer are taken and (b) In each pair, the relevant entry is first and the irrelevant entry is second.

4. We examine the relative ranks of the answers in each pair and we compute R_{norm} as follows:

$$R_{norm} = \begin{cases} \frac{1}{2}(1 + \frac{S^+ - S^-}{S^+_{max}}) & \text{if } S^+_{max} > 0; \\ 1 & \text{otherwise.} \end{cases} \quad (5)$$

S^+ is the number of correctly ranked pairs (i.e., the relevant entry has higher rank), S^- is the number of the erroneously ranked pairs (i.e., the method assigned higher rank to the irrelevant entry) and S^+_{max} is the total number of ranked pairs.

Below, we present a *precision-recall plot* for each field type. The horizontal axis in such a plot corresponds to the measured recall while, the vertical axis corresponds to precision. Each method in such a plot is represented by a curve. The top-left point of a precision/recall curve corresponds to the precision/recall values for the best answer or best match (which has rank 1) while, the bottom right point corresponds to the precision/recall values for the entire answer set. Each point in our plots is the average over 20 queries.

The queries are randomly selected from the database and are characteristic of the contents of CORDIS: For proper names we have selected 20 surnames out of 18,971 and for long text fields 20 titles out of 28,051 contained in the CORDIS database respectively. We also present average retrieval response times for each method and each field. We run our experiments on a dedicated SUN Ultra 1 running Solaris^{TM}.

5 Searching on Proper Names

In the following, we examine the performance of similarity searching methods on field SURNAME. All stored surnames (and the queries) are converted to lower-case characters; spaces and punctuation characters are stripped out. A typical query would ask for projects in which the specified name (e.g., a contact person or scientist) is involved. A query has the spelling which the user believes is correct and we are looking for all names which are similar to it.

Phonetic methods, such as *Soundex* and *Phonix*, are not appropriate for mixed language databases such as CORDIS. Therefore, the competing methods for this field are:

- Ukkonen n-gram distance with $n = 2$ and $n = 3$.
- Raw Edit distance and Damerau-Levenstein edit distance that handles transposition of characters in strings.

Each query retrieves the best 50 answers (best matches). There is no query with empty answer set, since the queries are randomly selected from the database. Figure 2 illustrates the precision-recall diagrams for this set of experiments. Typically, precision and recall values are computed from each answer set after each rank, but this method would compute only a few points for each query. We obtained more points in our plots by interpolation (i.e., by computing precision and recall after each relevant entry is retrieved).

Edit distance methods perform better than n-grams achieving 10-30% better recall and up to 10% better precision. Raw edit and Damerau-Levenstein distances are about equally effective: Damerau-Levenstein edit performs slightly better for small answer sets. However, this situation is reversed for large answer sets.

We observed cases where n-grams retrieved correct answers that edit distance methods failed to retrieve (e.g., answer Curtis for query Kourtis). A solution to this problem is to combine methods: We define a new distance as the minimum of the above 4 distances (i.e., Raw-Edit, Damerau-Levenstein, 2-grams and 3-grams). Before we take the minimum, all distances are normalized by dividing edit distances by $length$ (mean edit distance) and n-gram distances by $2length - 2n + 2$ (mean n-gram

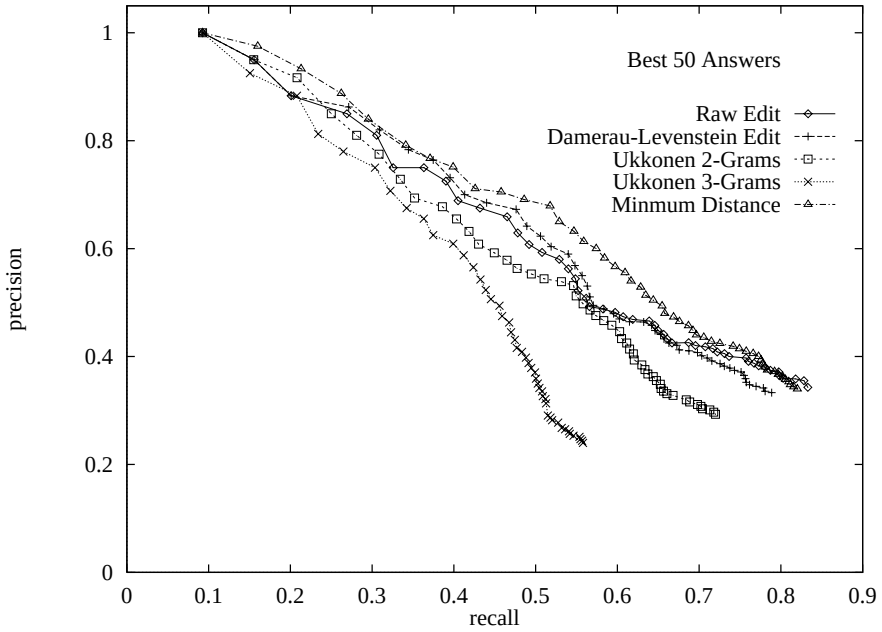

Figure 2: *Precision-Recall diagram of Minimum Distance, Damerau-Levenstein edit, raw edit and Ukkonen n-gram ($n = 2, 3$) methods for retrievals on field SURNAME.*

distance) where, $length = 8.5$ (mean character length) and $n = 2, 3$. In practice, we don't apply the raw-edit distance since it is always greater than the Damerau-Levenstein distance. As observed in Figure 2 the new method (referred henceforth as *Minimum Distance* method) achieved better precision and recall for most answers.

Method	Ranking Quality (R_{norm})
Ukkonen 2-grams	0.134
Ukkonen 3-grams	0.101
Raw Edit Distance	0.132
Damerau-Levenstein	0.120
Minimum-Distance	0.247

Table 2: *Ranking Quality (R_{norm}) for the methods tested on field SURNAME.*

Table 2 illustrates the computed values of ranking quality for the above methods. The Minimum Distance method achieved far better ranking quality than any other method. Table 3 illustrates the average (per query) retrieval response times of the above 4 methods. These are times for sequential search. Obviously, Minimum Distance is slower than any other method.

Conjecture 1 *The prevailing method for searching in field SURNAME is the Minimum-Distance method. This method performs better than any other in terms of precision, recall and ranking quality. Although, it is slower than its competitors, our selection must be based on effectiveness (i.e., quality of results) rather than on efficiency (i.e., speed).*

Method	Time SURNAME (secs)
Ukkonen 2-grams	0.63
Ukkonen 3-grams	0.55
Raw Edit Distance	0.28
Damerau-Levenstein	0.39
Minimum-Distance	1.46

Table 3: *Average retrieval response times in seconds for the methods tested on field SURNAME.*

5.1 Comparison with Other Methods

We compare the results of the current study with the most recent work on name search. This includes the work by Zobel and Dart [18] and the work by Pfeifer Poersch and Fuhr [19].

Zobel and Dart experimented with the DICT and the NAMES databases with 113,212 words and 31,763 words respectively. However, they showed results of relevant judgements (mainly precision) only on the NAMES database. They showed that edit distance techniques are far more effective the phonetic coding techniques, with Phonix being more effective than digrams and edit distance. Pfeifer Poersch and Fuhr combined several databases into a single collection with 18,000 words (not only names). They reported precision/recall results only on the Ziff-Davis Publishing database with 8,125 words. Their results showed that digrams perform better than Phonix, which in turn performs slightly better than Damerau-Levenstein edit alone. They also showed that a combination of digrams with Phonix exhibit better performance than any other method alone. In this work we showed that a combination of edit distance and n-gram methods exhibit better performance than any method alone.

The comparison with our results can be done only to a certain extend: These results refer to work on different databases, with different methods and with different query sets. All methods (including ours) considered mis-spellings and typing errors. However, the exact evaluation criteria for the cited methods are not known exactly. The results of these evaluations do not agree on the effectiveness of the methods. Nevertheless, the effectiveness of the name search methods seem to depend not only on the evaluation criteria but also on database contents.

Except the above, all methods have the following common features:

- The queries were randomly selected from the database.
- A single person carried-out all the evaluations.
- All methods used binary assignments ("relevant" and "not relevant") for the evaluation of the results.
- The relevance assessment method: Zobel and Dart merged the output from each method and they used the same pooling method with us to obtain their results. Pfeifer, Poersch and Fuhr followed a similar approach: They considered answers with up to 4 or 5 errors as the only correct answers in the database.

There are also some additional differences between our and the above cited work:

- Zobel and Dart evaluated the top 200 responses of each query and they found 6.4 correct matches per query on the average. Pfeifer Poersch and Fuhr considered answers with up to 4 or 5 errors and they found 6.6 correct matched per query on the average. In our experiments the average number of correct matches per query is 15.1.

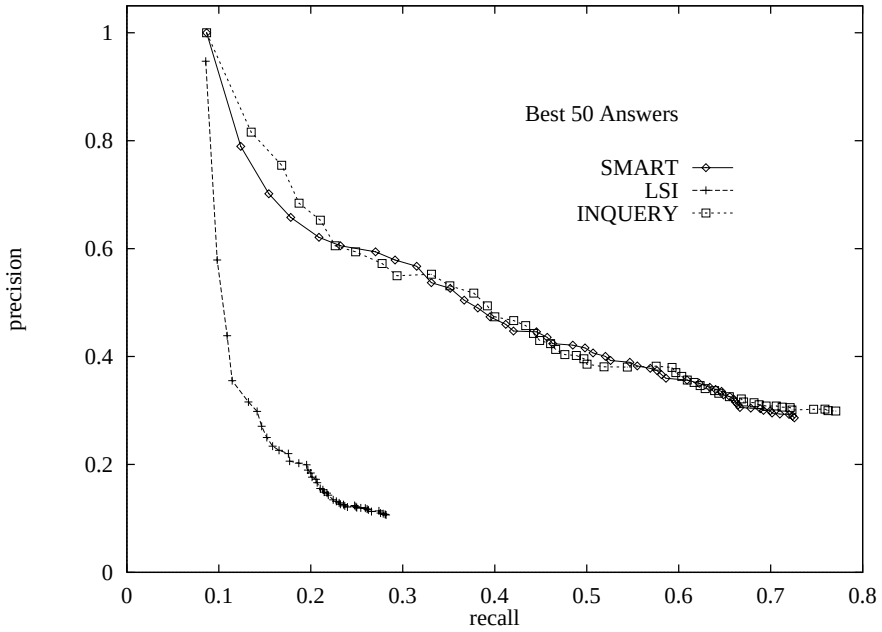

Figure 3: *Precision-Recall diagram of SMART, LSI and INQUERY for retrievals on field TITLE.*

- In CORDIS, we searched on field SURNAME which contains only names. The datasets used in the above cited work do not contain only names. To extract names, the original databases (e.g., NAMES) had to be scanned first, looking for words starting with capital letters. It is almost impossible to ensure that this method yielded datasets containing only names.
- We presented results with ranking quality in addition to precision and recall.

6 Searching on Long Text Fields

In the following, we examine the performance of similarity retrieval methods on fields TITLE, ABSTRACT and GEN_INFORMATION. Both, database text and queries are converted to lower-case characters; punctuation characters are stripped out. The queries are in the form of short text sentences and are randomly selected from the stored titles. Notice that, longer queries (e.g., full abstracts or text documents) are less likely to be formulated by a typical CORDIS user. Novice users usually issue short text queries that capture the main aspects of information they are interested in. Titles are a good approximation to such queries, since they contain much information in very compact form. In our experiments, the same query addresses each field separately and, each time, retrieves the best 50 answers from each field. A typical query would ask for all projects referring to a specific subject (e.g., *"performance evaluation of multidimensional access methods"*).

The candidate methods are:

- SMART. We used the implementation of [7]. This is an experimental implementation. We also implemented the cosine similarity method according to the formulaes in Section 2.2.1 and we obtained similar results.
- LSI. We used the implementation of [8]. This is an experimental implementation.
- INQUERY. We used the implementation of [20]. This is a commercial implementation.

Table 4 illustrates the measured values of R_{norm} for each method and each field.

Method	R_{norm} TITLES	R_{norm} ABSTRACT	R_{norm} GEN_INFORMATION
SMART	0.827	0.696	0.779
LSI	0.699	0.681	0.636
INQUERY	0.799	0.711	0.709

Table 4: *Ranking Quality (R_{norm}) for the methods tested on fields TITLE, ABSTRACT and GEN_INFORMATION.*

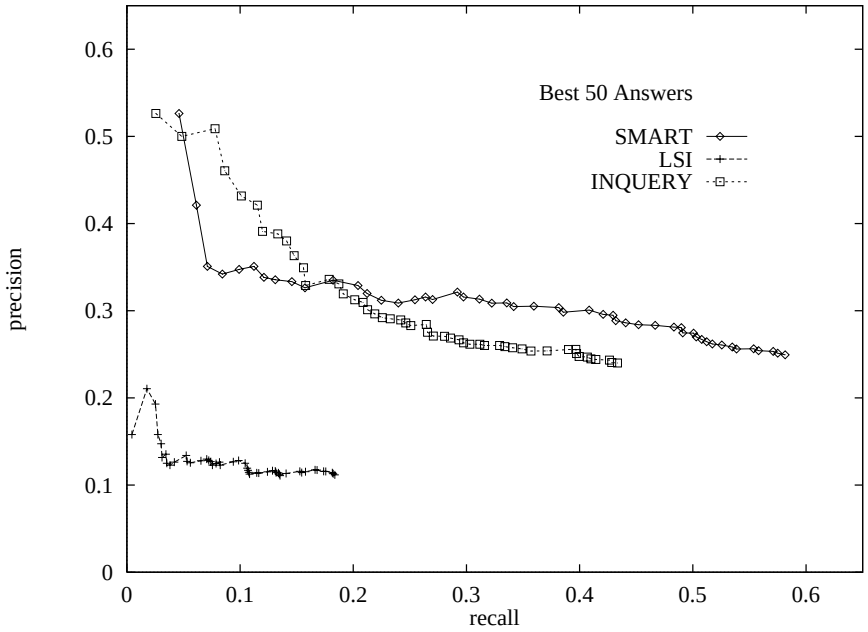

Figure 4: *Precision-Recall diagram of SMART, LSI and INQUERY for retrievals on field ABSTRACT.*

In the following, we present a separate precision-recall diagram for each field. Each point in such a diagram corresponds to a rank and it is computed based on the ranked answers obtained up to that point. The total number of points in each curve is 50, since we retrieve the best 50 answers per query and all answers have distinct ranks.

Figure 3 illustrates the precision-recall diagram on field TITLE. INQUERY is slightly more effective than SMART except for few ranks. SMART has slightly better ranking quality (i.e., it retrieves the relevant answers before the irrelevant ones). However, our selection must be based on precision and recall rather than on ranking quality. Both, INQUERY and SMART are far more effective than LSI.

Conjecture 2 *The prevailing method for searching on field TITLE is INQUERY.*

Figure 4 illustrates the precision-recall diagram for searching on field ABSTRACT. For small answer sets returning up to 11 entries, INQUERY performs better than SMART but this situation is reversed for larger answer sets. Specifically, SMART achieves up to 20% better recall for answers with 50 entries. INQUERY achieves slightly better ranking quality than SMART. Both SMART and INQUERY perform much better than LSI.

Conjecture 3 *The prevailing method for searching on field ABSTRACT is SMART. However, if one is interested in the first few best matches (i.e., up to 11) INQUERY should be preferred.*

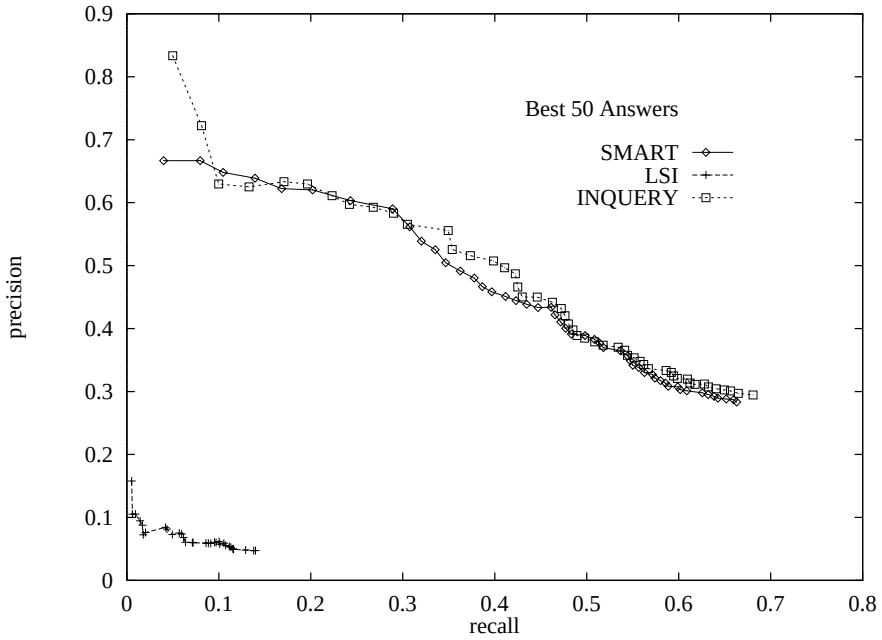

Figure 5: *Precision-Recall diagram of SMART, LSI and INQUERY for retrievals on field GEN_INFORMATION.*

Figure 5 illustrates the precision-recall diagram for searching on field GEN_INFORMATION. INQUERY is slightly more effective than SMART in terms of precision and recall for all answer sizes, but SMART has better ranking quality. Both SMART and INQUERY perform better than LSI.

Conjecture 4 *The prevailing method for searching on fields GEN_INFORMATION is INQUERY.*

Method	TITLES (secs)	ABSTRACT (secs)	GEN_INFORMATION (secs)
SMART	1.15	1.15	1.15
LSI	9.90	7.20	7.15
INQUERY	0.30	0.35	0.35

Table 5: *Average retrieval response time in seconds for the methods tested on fields TITLE, ABSTRACT and GEN_INFORMATION.*

Table 5 illustrates the average retrieval response times for each method and each field. We present these values only for completeness. Response times may not be a fair basis of comparison, since (probably) not all methods are optimized for our experimentation platform. INQUERY is the faster method followed by SMART and LSI. Notice that, INQUERY is a commercial implementation.

6.1 Discussion

SMART and INQUERY are the prevailing methods for similarity searching on long text fields. They performed better than LSI in all our experiments. LSI suffers probably from the heterogeneity of the CORDIS texts: The calculated correlation of term usage across documents from different domains leads to a significant amount of erroneous associations which affect the quality of retrieval results. This negative behavior can not be reversed even if high number of dimensions of the term-document matrix

transformation (we used up to 1,000 dimensions without any significant improvement). The above experimental results are obtained for 100 dimensions.

Each one of the above three methods could be further improved for a concrete database: SMART by using different vector similarity formulaes (we used the Cosine Similarity Measure described in Section 2.2.1), INQUERY by using different instantiation probabilities for the network nodes (we used the default value 0.6 of the INQUERY package) and LSI by both using different number of dimensions for the term-document matrix transformation and vector similarity formulaes for the ranking (we used a Cosine Similarity Measure). Furthermore, the performance of all methods can be influenced by using dedicated stemming algorithms (we used the built-in stemming algorithms of each package), by adding some heuristic phrase detection mechanisms (we have not considered phrases) or by performing relevance feedback (we have excluded relevance feedback since out target user is the novice searcher).

We have nevertheless preferred to parameterize the evaluated methods in a standard way rather than fine-tuning them in a database-dependent manner mainly because: (a) Database-dependent tuning has already been exhaustively performed in the past for each of these methods, especially in the context of the Text Retrieval Conferences [6] (TREC), (b) We expect that fine-tuning will not significantly change the recommendations made above, since the improvement potential of each method is estimated to be similar for a given database and (c) For a real-life database like CORDIS being daily extended by hundreds of new texts, a robust all-round retrieval technique is rather needed than a highly configurable toolkit.

6.2 Comparison with TREC Results

SMART and INQUERY have also been applied many times in the past and the results have been presented within the TREC conferences [21, 22, 23]. We compare the results of the current study with the most recent TREC results [24]. The comparison can be done only to a certain extend. Both TREC and our experimental set-up have the following common features:

- The task, called "ad-hoc" task [16] that is, search in a static set of documents. The queries are not known in advance and the users are free to create new queries using any method they like.
- The kind and length of the queries that is, natural language topic statements with a mean length of about 17 words (10 in our experiments).
- The relevance assessment method referred to as "pooling method" [16].
- The relevance judgement method: Both TREC and the current study use one person per query for judging the relevance of the documents retrieved. The only difference is that, in TREC, different persons are used for different queries. In TREC, each person has assessed only queries formulated by him or herself; the relevance assessments were therefore based on real user's personal interests. Because of the limited resources for the current study, it was not possible for us to apply such a judgement method. Instead, we tried to compensate the missing multitude of "real" users by the contribution of a professional documentalist who has a lot of experience with real user's needs on European project databases and who has applied a similar methodology here.
- Only binary judgments are made: A retrieved document is judged either as "relevant" or "not relevant". "A document is judged relevant if any piece of it is relevant"[7].

TREC and our experiments differ on:

[6] `http://trec.nist.gov`
[7] `http://trec.nist.gov/data/reljudge_eng.html`

- The number of documents used: TREC uses the merging of 4 document collections containing from 55,630 (Federal Register, 1991) up to 210,158 (the Financial Times, 1991-1994) documents as test data (i.e. Disk 4 and Disk5 of Table 2 in [16], excluding the Congressional Record collection). This results in a total of 528,155 documents. In our study we used up to 28,000 documents (see Table 1). This is the max number of documents in CORDIS.

- The length of the documents: The above TREC document collections have mean lengths varying from 412,7 words (the Financial Times, 1991-1994) up to 644,7 words (Federal Register, 1991) per document, resulting to a mean length of 497,19 words per document for the merging of the 4 collections. The current study investigates different text lengths varying from 10.14 words up to and 132.97 words per document (see Table 1).

- The used variants of INQUERY and SMART: TREC used sophisticated query expansion for both approaches while the current study uses more or less the basic variants [7, 20].

- The use of previous results: It is possible for TREC users to take advantage of previous TREC results (e.g., test data, queries, evaluations etc.) in developing their systems. For CORDIS, such results do not exist and the whole experimental set-up had to be designed and built from the beginning.

Nevertheless, the results from both TREC and our experiments are comparable: INQUERY and SMART perform almost equal on long texts: The so-called "automatic runs" of the TREC-7 conference yielded a precision-recall diagram (Figure 5 of [16]) which is very similar to the one we achieved for retrievals on field of GEN_INFORMATION and almost similar to those on the other two fields. Furthermore, the mean precision values for INQUERY and SMART obtained by the TREC-7 automatic runs (Table 6 of [16]), which are shown in Table 6, are almost the same as those we obtained on CORDIS, which are shown in Table 7 below.

Method	(Title, Description)	Title
U. Mass (i.e. INQUERY)	0.252	not given
Cornell/SabIR (i.e. SMART)	0.254	0.239

Table 6: *Mean precision obtained by TREC-7.*

Method	TITLE	ABSTRACT	GEN_INFORMATION
INQUERY	0.298	0.240	0.294
SMART	0.286	0.249	0.283

Table 7: *Mean precision obtained for CORDIS.*

Finally, the current study indicates a better ranking quality for SMART, a parameter that was not considered within the TREC studies.

7 Conclusions

We aim at developing a dedicated similarity retrieval mechanism for CORDIS. Our experiments indicate that a method that combines n-grams and edit-distance (for name search) and cosine similarity methods

such as SMART (for longer text fields) provide a good basis for the design of such a mechanism. Additional future work in this direction includes, the experimentation with more methods (e.g., signatures) and fields, the development of methods for handling combined queries involving more than one fields (e.g., names, titles and abstracts), the integration of new search methods in CORDIS and the design of a new interface to support similarity queries in CORDIS.

Acknowledgments

We would like to thank Mrs. Karin Mölle, employee at GMD-IPSI, for her valuable contribution in this work. Mrs. Mölle is a professional documentalist with more than 20 years experience in text retrieval who has has provided us with an excellent methodology for the intellectual evaluation of the answer sets. She has furthermore carried out the relevance judgments for our experiments, a work of crucial importance for any well-founded evaluation of retrieval techniques. We would also like to thank Roland Jeske, student at Darmstadt University of Technology, who carried out much of the experimental work. Bruce Croft at University of Massachusetts, Joel Remde at Bellcore Communications Research Inc. and Chris Buckley at Cornell University for kindly providing us with the software for INQUERY, LSI and SMART systems respectively.

References

[1] L.A.H. Paris and H.R. Tibbo. Freestyle vs. Boolean: A Comparison of Partial and Exact Match Retrieval Systems. *Information Processing and Management*, 34(23):175–190, 1998.

[2] N.J. Belkin and W.B. Croft. Retrieval Techniques. *Annual Review of Information Science and Technology*, 22(9):110–145, 1987.

[3] J.P. Callan, W.B. Croft, and S.M. Harding. The INQUERY Retrieval System. In *DEXA 3. Intern. Conf. on Database and Expert Systems Applications*, pages 83–97, Berlin: Springer Verlag, 1995.

[4] N. Shivakumar and H. Garcia-Molina. SCAM: A Copy Detection Mechanism for Digital Libraries. In *Intern. Conf. in Theory and Practice of Digital Libraries (DL' 95)*, Austin, Texas, 1995.

[5] S. Brin, J. Davis, and H. Garcia-Molina. A Copy Detection Mechanisms for Digital Documents. In *ACM SIGMOD Intern. Conf. on Management of Data*, San Francisco, CA, 1995.

[6] U. Manber and S. Wu. GLIMPSE: A Tool to Search Through Entire File Systems. Technical Report TR 93-44, Department of Computer Science, University of Arizona, Tuscon, Arizona, 1993. (http://webglimpse.org/pubs).

[7] C. Buckley. SMART, Version 7.

[8] LSI Software (for research purposes only), 1990.

[9] T. Gadd. PHONIX: The Algorithm. *Program*, 24(4):381–402, 1990.

[10] E. Ukkonen. Approximate String Matching With q-grams and Maximal Matches. *Theoretical Computer Science*, 92:191–211, 1992.

[11] G. Salton, editor. *Automatic Text Processing: The Transformation, Analysis and Retrieval of Information by Computer*. Addison Wesley, 1988.

[12] D.K. Harman, editor. *The Second Text Retrieval Conference*. National Institute of Standards and Technology Gaithersburg, MD, 1994.

[13] G.W. Furnas, S. Deerwester, S.T. Dumais, T.K. Landauer, R. A. Harshman, L.A. Streeter, and K.E. Lochbaum. Information retrieval using a Singular Value Decomposition Model of Latent Semantic Structure. In *Intern. Conf. on Research and Development in IR*, New York, 1988.

[14] S. Deerwester, S.T. Dumais, G.W. Furnas, T.K. Landauer, and R. Harshman. Indexing by Latent Semantic Analysis. *Journal of the American Society for Information Science*, 41(6):391–407, 1990.

[15] H.R. Turtle and W.B. Croft. Inference Networks for Document Retrieval. In *Intern. Conf. on Research and Development in Information Retrieval*, pages 1–24, SIGIR, 1990.

[16] E.M. Voorhees and D.K. Harmann. Overview of the Seventh Text REtrieval Conference (TREC-7). In *NIST Special Publication 500-242: The Seventh Text REtrieval Conference (TREC-7)*, pages 1–23, 1998. http://trec.nist.gov/pubs/trec7/t7_proceedings.html.

[17] P. Bollmann and F. Jochum. Experimentelle Untersuchungen von Ranking-Verfahren. In *Deutscher Dokumentartag 1985*, K.G. Saur, Munich, New York, Paris, 1986.

[18] J. Zobel and P. Dart. Finding Approximate Matches in Large Lexicons. *Software - Practice and Experience*, 25(3):331–345, 1995.

[19] U. Pfeifer, T. Poersch, and N. Fuhr. Retrieval Effectiveness of Proper Name Search Methods. *Information Processing and Management*, 32(6):667–679, 1996.

[20] INQUERY, Release Version 3.0, 1995.

[21] J. Allan, J. Callan, M. Sanderson, and J. Xu. INQUERY and TREC-7. In *NIST Special Publication 500-242: The Seventh Text REtrieval Conference (TREC-7)*, pages 201–216, 1998. http://trec.nist.gov/pubs/trec7/t7_proceedings.html.

[22] T. Strzalkowski, G. Stein, and G. Bowden Wise. Natural Language Information Retrieval: TREC-7 Report. In *NIST Special Publication 500-242: The Seventh Text REtrieval Conference (TREC-7)*, pages 217–226, 1998. http://trec.nist.gov/pubs/trec7/t7_proceedings.html.

[23] C. Buckley and J. Walz. SMART High Precision: TREC-7. In *NIST Special Publication 500-242: The Seventh Text REtrieval Conference (TREC-7)*, pages 285–298, 1998. http://trec.nist.gov/pubs/trec7/t7_proceedings.html.

[24] E.M. Voorhees and D.K. Harmann, editors. *NIST Special Publication 500-242: The Seventh Text REtrieval Conference (TREC-7)*. Department of Commerce, National Institute of Standards and Technology, 1998. http://trec.nist.gov/pubs.html.

Contents