Using Explicit Word Co-occurrences to Improve Term-Based Text Retrieval

Stefano Ferilli[1], Marenglen Biba[2], Teresa M.A. Basile[1], and Floriana Esposito[1]

[1] Dipartimento di Informatica
Università di Bari
via E. Orabona, 4 - 70125 Bari, Italia
{ferilli,basile,esposito}@di.uniba.it
[2] Computer Science Department
University of New York, Tirana
Rr. "Komuna e Parisit", Tirana, Albania
marenglenbiba@unyt.edu.al

Abstract. Reaching high precision and recall rates in the results of term-based queries on text collections is becoming more and more crucial, as long as the amount of available documents increases and their quality tends to decrease. In particular, retrieval techniques based on the strict correspondence between terms in the query and terms in the documents miss important and relevant documents where it just happens that the terms selected by their authors are slightly different than those used by the final user that issues the query. Our proposal is to explicitly consider term co-occurrences when building the vector space. Indeed, the presence in a document of different but related terms to those in the query should strengthen the confidence that the document is relevant as well. Missing a query term in a document, but finding several terms strictly related to it, should equally support the hypothesis that the document is actually relevant. The computational perspective that embeds such a relatedness consists in matrix operations that capture direct or indirect term co-occurrence in the collection. We propose two different approaches to enforce such a perspective, and run preliminary experiments on a prototypical implementation, suggesting that this technique is potentially profitable.

1 Introduction

The retrieval of interesting documents in digital libraries and repositories is today a hot problem, that is becoming harder and harder as long as the amount of available documents dramatically increases and their content quality tends to be of lower and lower quality. When a user issues a query, on one hand, there is the need for a stricter selection of the returned document, in order to filter out irrelevant ones; on the other hand, the retrieved documents should satisfy his specific interests from a semantic viewpoint. This is reflected in the classical evaluation measures used in Information Retrieval, precision and recall.

M. Agosti, F. Esposito, and C. Thanos (Eds.): IRCDL 2010, CCIS 91, pp. 125–136, 2010.

Almost all searches for documents are currently based on their textual content. The problem is that user interaction in information retrieval typically takes place at the purely lexical level, but term-based matching is clearly insufficient to even approximately catch the intended semantics of the query, because the syntactic aspects of sentences embed significant information that is missed by the bag-of-word approach. The exact matching of query terms with terms of documents in the repository implies a number of practical problems in retrieving the desired information, such that trying to solve them usually results in oscillations between the two extremes of high recall with very low precision or high precision with very low recall, without being able to find a suitable trade-off and balance that is useful to the user. In addition to being excessively simplistic in itself, the term matching-based approach suffers also from problems and tricks that are intrinsic to Natural Language, such as word synonymy (different words having the same meaning) and polysemy (single words having several meanings in different contexts). However, the problem we want to face in this work is yet more advanced, and can be summarized in the following example. We clearly would like the following text:

> *The locomotive machine will leave from Platform C of Penn Station after letting all passengers getting down from the carriages and sleeping-cars.*

to be retrieved as an answer to a query made up of just one term 'trains'. In fact, in the classical setting, this document would not be returned in the result set, because nor the exact word, nor its stem, appear in the text. Still worse, not even the concept of 'train' itself is present in it, although a large number of terms that are specific to the train domain are pervasive in the sentence. This complies with the *one domain per sentence* assumption [6].

In the following section, a brief overview of the techniques proposed in the literature for attacking the shortcomings of pure term-based text retrieval is provided. Then, the proposal of explicit exploitation of term co-occurrence related information is described in Section 3. Specifically, two different techniques are proposed to obtain such a result: SuMMa and LSE, described in two separate subsections. Subsequently, Section 4 discusses some of the pros and cons of the two approaches, showing that they have complementary advantages and disadvantages as regards time and space requirements for computation. Some preliminary but encouraging results obtained on a toy problem are also presented, suggesting that co-occurrences play a relevant role in improving the quality of term-based search result based only on the lexical level. Lastly, Section 5 concludes the paper and outlines current and future work that is planned to better assess the effectiveness of the proposed approach.

2 Related Work

Most techniques for Information Retrieval are based on some variation of the Vector Space [15], a geometrical interpretation of the problem in which each document is represented as a point in a multi-dimensional space, whose dimensions

are the terms, and where the coordinates are given by (some function of) the number of occurrences of that term in that document. Since pure occurrences would be affected by terms that are very frequent just because of the kind of collection, and hence are not significant for the specific documents, weighting functions are used that merge a local factor (saying how important is a term for a specific document) smoothed by a global factor (evaluating the spread of that term in the whole collection). The best-known example is TF*IDF (Term Frequency * Inverse Document Frequency) [14]. Practically, the space associated to a document collection consists of a very large matrix, called Term-Document Matrix, whose rows corresponds to the terms appearing at least once in the collection, and whose columns correspond to documents in the collection. The user query can be considered as a document as well, and hence represented as a vector expressed in the same dimensions as the space, which allows to easily compare it to each document in the collection (Euclidean distance is a straightforward way for doing this), and then to rank the results by increasing distance.

Another problem is how to interpret the query terms. The usual default is considering them as a conjunction, i.e. as connected by an AND logical operator. Thus, only documents including all the terms expressed in the query are retrieved. This is intended to improve precision, but negatively affects recall. The opposite perspective, of considering the terms as a disjunction, i.e. as connected by an OR logical operator, is indeed exploited for widening the research (an option often reported under the specification 'Find similar items' in the search engines). It returns result sets having much higher recall but very difficult to handle by the user because of precision. An intermediate solution is allowing the user to enter complex queries, with any combination of the NOT, AND and OR logical operators and any nesting of parentheses, in order to better specify his intended target, but this requires to set up quite long and complex logic expressions even for simple queries, and very few users are acquainted with boolean logics and able to properly exploit its operators. Thus, several techniques have been proposed to automatically improve the quality of the search results without charging inexperienced users with the task of better specifying their needs, but trying to better exploit classical queries made up of just a sequence of terms.

One approach is called *query expansion*, and consists in extending the query actually entered by the user with additional terms computed in such a way to hopefully result in more hits than those produced by the original query. A strategy for doing this consists in expanding the query with terms that co-occur in some document with those in the original query. Indeed, independently of the weighting function exploited for defining the Vector Space, terms not appearing in a document will have a null weight in the corresponding matrix cell for that document. Conversely, a significant presence of terms related to those in the query, although not exactly corresponding to them, should make up for such an absence, raising the degree of similarity. However, other studies have proved that this approach yields worse results than the original query, since the additional co-occurring terms are usually also very frequent in the collection independently of the relatedness to the query [12].

Another very famous approach is the Latent Semantic Indexing (LSI) [8, 11], where the underlying assumption is that the actual meaning of a text is often obscured by the specific words chosen to express it, and hence usual term matching approaches are not suitable to catch such a meaning. The proposed solution is based on a mathematical technique called Singular Value Decomposition (SVD for short), that splits a given matrix into three matrices such that their matrix product yields again the original matrix. In the case of a Term-Document matrix representing the vector space of a collection of documents, the three matrices obtained by SVD represent, respectively, the relationships of the terms with a number of abstract items that can be considered as the concepts underlying the document collection, the concepts themselves (that correspond to the 'latent' semantics underlying the collection) and the relationships between these concepts and the documents. By ignoring the concepts having less weight, and hence less relevant, and focussing on the k most important ones only, the matrix product of the resulting truncated matrices represents an approximation of the original vector space, where the most relevant relationships have been amplified, and hence have emerged, while the less relevant ones have been stripped off. Thus, the new weight reflects the hidden semantics, at the expenses of the predominance of lexical information. The LSI approach has been widely appreciated in the literature [7], but is not free of problems. First of all, computing the SVD is computationally heavy even for medium-sized datasets (thousands of documents). Second, the choice of the amount of relevant concepts to be considered is still debated among researchers: Indeed, being the semantic 'latent', these concepts are not 'labelled', and hence one actually does not know what he is keeping in and what he is keeping out when computing the truncated product. Thus, there are only indirect ways for validating such a technique. The interesting point in the LSI approach is that a document can be retrieved according to a query even if the terms in the query are not present in that document.

Another semantic approach, that relies on an explicit representation of concepts, proposes to switch from the specific terms to their underlying concepts according to some standard. The idea is that, in this case, the semantics is directly plugged into the mechanism and hence will be considered during the retrieval phase. In other words, the space is not any more Terms by Documents, but rather it is Concepts by Documents. The first issue is what reference standard for concepts is to be used. Much of the literature agreed to exploit WordNet [10], a famous lexical database that identifies the concepts as the underlying set of synonymous words that express them. Each such set, called a *synset* (synonymous set), is given a unique identifier, and several syntactic and semantic relationships are expressed among concepts and between concepts and words. Thus, WordNet is in many respects a perfect resource for bridging the gap between the lexical level (terms) and the semantic one (concepts). The problem in this case is that, due to polysemy, several different concepts can correspond to a same word in the query or in a document. Including all of them in the vector space representation of the document would often yield a much larger and less precise space than the one based on terms, which in turn would make more complex the proper retrieval of

documents. Conversely, trying to identify the only correct concept for each word requires an additional step of Word Sense Disambiguation (WSD) [2], another hot topic in the research because no trivial and highly reliable technique still exist for accomplishing such a task.

An approach that mixes the latent semantics with clustering of documents is Concept Indexing [3], where documents are collected, according to some underlying semantics, into clusters such that documents in the same cluster are similar to each other, while documents belonging to different clusters are very different. Such a grouping can be carried out either in a supervised manner, or in an unsupervised one, and exploits very efficient algorithms. Then, each cluster is considered as corresponding to a concept, and exploited for dimensionality reduction purposes in the perspective of the retrieval step. Specifically, the dimensionality reduction step takes place by considering as a dimension a representative for each cluster found. It can be an actual document in the cluster, or just a surrogate thereof, computed as a kind of average of the cluster components.

3 Exploitation of Co-occurrences

Our proposal for attacking the problem represented by the examples in the Introduction and improving the search results in a text collection is to exploit term co-occurrence to compute a modified version of the Vector Space in which related terms are considered significant for a document even if they do not explicitly appear in it. Indeed, studies in [13] have confirmed that statistics on word co-occurrence can closely simulate some human behaviours concerning Natural Language, and in other works some interest has been also put on term co-occurrence as a way for improving the access to digital libraries [1]. Our idea is similar to (and inspired by) the LSI approach, but our aim is explicitly introducing the co-occurrence factor in the version space. Indeed, many papers state that term co-occurrence is taken into account by the LSI, but no specific demonstration of how much influence co-occurrences have in the overall result seems available in the literature. Moreover, the approach we propose differs also from that in [12], because here the term co-occurrence is plugged directly into the vector space, and not used just for query expansion.

More precisely, the way in which we propose to discover such a relationship is by finding sequences of documents in the collection that pairwise have at least one term in common. In this setting, different levels of co-occurrences can be defined. The most straightforward, called co-occurrence of order 1, refers to pairs of terms appearing in a same document. However, it is quite intuitive that the co-occurrence relation fulfils in some 'semantic' sense, if not in the mathematical sense, a kind of transitive property: If terms a and b appear in the same document, and hence can be considered as related, and terms b and c co-occur in another document, and hence can be considered as related as well, then also a and c, even in case they never co-occur together in the same document, can be considered in some sense related, due to their common relationship to

the intermediate term b. This is called a co-occurrence of order 2. Taking further this approach, a co-occurrence of order k between two terms t_1 and t_2 can be defined as the fact that there is a chain made up of k documents, $< d_1, \ldots, d_k >$ such that $t_1 \in d_1$, $t_2 \in d_k$, and any two adjacent documents d_i and d_{i+1} in the chain have at least one term in common. Of course, the longer the chain, the less strict the relation, and hence the lower the value for that co-occurrence should be, up to the value 0 that should be reserved to the case of absolutely no (direct or indirect) connection between the two terms in the document collection.

Thus, our proposal is to explicitly take into account co-occurrences when building the vector space for document retrieval. Consider the initial (possibly weighted) Term-Document Matrix $A[n \times m]$, where n is the number of terms and m is the number of documents in the collection. A matrix reporting the co-occurrence between two any terms in the collection will be a symmetric matrix sized $n \times n$: Let us call it $\overline{T}$, and ignore for the moment the order of co-occurrences expressed by such a matrix. By multiplying this matrix and the original Term-Document Matrix, we obtain a new Term-Document Matrix

$$A' = \overline{T} \times A \tag{1}$$

sized $n \times m$, that has re-weighted the importance of each term in each document by explicitly taking into account also the term co-occurrences in the collection, so that its elements can be different than 0 even when a term does not appear in a document, but is in some way related to terms actually appearing in that document. Specifically, the value should be larger according to the closeness of such relationships and the amount of terms in the document to which that term is related by some order of co-occurrence. As shown in the following, there are two different ways for computing the matrix $\overline{T}$.

3.1 The Straightforward Approach: SuMMa

Co-occurrences can be introduced into a vector space by simply following the geometrical definitions of matrices. Indeed, the term co-occurrence of order 1 can be straightforwardly computed by multiplying A by its transposed matrix:

$$T = A \times A^T \tag{2}$$

Now, $T[n \times n]$ is a Term-Term Matrix whose elements are 0 if and only if the two corresponding terms never appear in the same document, or a value other than 0 otherwise. Now, co-occurrences of order 2 can be computed by multiplying T by itself and, in general, co-occurrences of order k can be obtained by multiplying T by itself k times:

$$T_k = T^k = T^{k-1} \times T \quad \text{for } k > 1 \tag{3}$$

Each matrix T_k has size $[n \times n]$, and its elements are 0 if and only if there is no chain of documents of length k in the document collection such that the two corresponding terms have a co-occurrence of order smaller than or equal

to k. If weights in the original matrix A are greater than or equal to 1, then the larger the value of a matrix item, the closer the co-occurrence between the two corresponding terms, and hence the stronger the relationship between them. More specifically, each value will be increased by all possible co-occurrences of order at most k that can be found in the document collection. We call the indexing scheme that follows this approach SuMMa (acronym of Successive Multiplication of MAtrices).

In order to catch even the slightest relatedness between terms, the power to be computed should be $\overline{T} = T^{\infty}$. Clearly, the longest chain possible in a collection including m documents will have length m, which represents a practical upper bound to the power to be computed, but it is likely to be considerably high anyway. However, in more realistic situations, there will be no need of actually computing T^m: Applying progressive multiplications, it is possible to stop the procedure at the first k such that T^{k+1} does not change any 0 item with respect to T^k. Although this can significantly reduce the computation required, a different option can be defining in advance the desired approximation, by specifying the largest order k that is considered significant, and hence that must be taken into account, and carrying out the multiplications up to just that power: $\overline{T} = T^k$.

3.2 The Theoretical Approach: LSE

A very interesting insight into Latent Semantic Analysis, on which LSI is based, has been provided in [4, 5]. There, the Authors provide the proof of theoretical results according to which, in their opinion, co-occurrence of terms is demonstrated to underlie the LSI technique. Actually, to be more precise, they prove how the co-occurrence of terms has an important connection to the SVD, but this does not prove as a straightforward consequence that, or to which extent, the very same connection is in some way expressed by the vector space resulting from the application of the LSI.

Let us first present the result in question. Given a Term-Document Matrix $A[n \times m]$, we already pointed out that, by applying SVD, it can be split into three distinct matrices:

- $U[n \times r]$, that represents the connection between terms in the collection and underlying latent concepts
- $W[r \times r]$, a diagonal matrix whose diagonal elements represent the latent concepts and their weight/importance in the collection
- $V[m \times r]$, that represents the connection between the documents in the collection and the underlying latent concepts

where: n represents the number of terms in the collection, r represents the number of latent concepts underlying the collection and m represents the number of documents in the collection, such that:

$$A = U \times W \times V^T \tag{4}$$

By choosing the number $k < r$ of relevant concepts to be considered, and stripping off the $r - k$ elements of the diagonal of W having lower values, and the

corresponding columns in U and V, an approximation of the original vector space A more centered on the selected concepts can be obtained again as above:

$$A_k = U_k \times W_k \times V_k^T \tag{5}$$

Now, [4] demonstrated that, if instead of performing the above product, one performs the following:

$$\overline{T} = U \times W \times W^T \times U^T \tag{6}$$

or, equivalently, its truncated version:

$$\overline{T}_k = U_k \times W_k \times W_k^T \times U_k^T \tag{7}$$

the resulting matrix has value 0 in all elements for which no co-occurrence of any order exists in the given document collection, and a value different than 0 in all other cases, and that the smaller the order of co-occurrence between two words, the higher such a value. Thus, this matrix can be straightforwardly applied in our approach. We named the approach that exploits this kind of computation LSE (acronym of Latent Semantic with Explicit co-occurrences).

4 Discussion and Preliminary Results

A first, immediately apparent difference between SuMMa and LSE is in the order of co-occurrence that can be taken into account. Indeed, LSE yields at once a Term-Term Matrix that accounts for all possible co-occurrences of any order, while SuMMa requires to preliminarily set a threshold k of interesting co-occurrence order to be computed, or else needs to discover on-the-fly the k such that no higher-order co-occurrences can be found in the document collection. In any case, such a value can be large, and require longer computational times. In this respect, the user can decide which approach to use according to various considerations. If he needs to exploit the full set of co-occurrence orders, he can go for the LSE. Conversely, if he wants to reduce the indexing computational time, or he wants to purposely set a bound on the order of co-occurrences to be considered (indeed, an intuitive assumption can be that co-occurrences above a given order are not significant and can be safely ignored without loss of retrieval power), he can choose the SuMMa. The SuMMa is also useful in case one has time biases, because it can be stopped at any multiplication step and still return a significant result (where only less significant co-occurrences have been ignored), whereas the LSE requires all the needed computation to be carried out at once, and no significant intermediate result is available before the process accomplishment.

Empirically, a prototypical Java language implementation of the two techniques revealed that the time and space requirements of the two approaches are complementary, and specifically space requirements are lower for the LSE approach, while SuMMa is faster. Progressively extending the cardinality of the

document dataset, the preliminary prototype has shown that SuMMa was able to stand matrices up to about 1500 terms (present in 36 documents), while LSE reached about 2600 (present in 69 documents). As to time, in the above case of a document collection made up of 36 medium-length documents including a total of 1475 terms, applying SuMMa with $k = 7$ (a threshold that can be considered sufficient to include in the outcome all significant co-occurrences between terms) took about 1 minute for indexing on an Intel Dual Core processor running at 3 GHz, while the LSE approach with truncation $k = 2$, took 10 minutes for computing the SVD, plus an additional minute for the final matrix multiplication on the same architecture.

A quick analysis of the two techniques can show the point. Since both must perform (1), the different complexity depends on the preliminary computations. **Space evaluation**. In SuMMa, for (2), matrices $T_{n\times n}$ and $A_{n\times m}$ are needed, for a total of $n^2 + nm$ values, that must also be kept for (3), where additionally T_k and T^{k-1} are needed, both of size $n \times n$, and hence yield a total memory requirement of $3n^2 + nm$. Conversely, LSE needs to store matrices $U_{n\times r}$, $W_{r\times r}$, $V_{r\times m}$ as a result of the SVD in (4), where W is diagonal (and hence can be reduced to just the vector of r diagonal values), and r (the rank of A) is usually comparable or equal to m. This results in a total of $m^2+2nm+m$ values, that can be significantly reduced after truncation of r to k in (5). Then, for computation of (7), $T_{n\times n}$, $U_{n\times k}$ and $W_{k\times k}$ are needed, i.e. (representing W as a vector of k elements) n^2+kn+k values. Hence, if $n > m$, (6) is the worst step, which makes SuMMa worst, whereas if $m > n$ the worst becomes LSE because of step (4). **Time evaluation**. (2) computes n^2 elements in T by m multiplications and $m-1$ additions, for a total of n^2m, while k repetitions of (3) compute each n^2 elements by n multiplications and $n-1$ additions, for a total of kn^3. Thus, overall SuMMA requires $2(kn^3+mn^2)$ steps (multiplications and sums). Then, the SVD has been proved to have complexity $O(\min(n^2m, m^2n))$. As to (6), it can be performed by the associative property of matrix multiplication as $(U_k \times (W_k \times W_k^T)) \times U_k^T$ $W \times W^T$ requires k^2 multiplications; the intermediate product results in a $n \times k$ matrix each of which elements is obtained by a single multiplication, for a total of kn; the external product produces a $n \times n$ matrix, each of whose elements is obtained by r multiplications and $r - 1$ summations, for a total of kn^2. Thus, overall LSE requires $O(\min(n^2m, m^2n)) + kn^2 + kn + k^2$ steps (that, considering again k comparable to m, becomes $O(\min(n^2m, m^2n)) + mn^2 + mn + m^2$).

As to the quality of the result, the following experiment was run. The set of 36 categories included by WordNet Domains [9] under the section 'Social Science' was selected: social science, folklore, ethnology, anthropology, body care, health, military, school, university, pedagogy, publishing, sociology, artisanship, commerce, industry, aviation, vehicles, nautical, railway, transport, book keeping, enterprise, banking, money, exchange, finance, insurance, tax, economy, administration, law, diplomacy, politics, tourism, fashion, sexuality. Then, the Wikipedia (`www.wikipedia.org`) Web page for each such category was downloaded, and the resulting collection was indexed using both SuMMa and LSE. Thus, issuing a query, each of the retrieved documents represents a possible category for the

query, and hence this can be seen as an approach to the Text Categorization [16] problem. Note that this is not the ideal environment on which applying the proposed technique: Indeed, being present a single document for each category, and being the categories quite disjoint (they should in principle represent a partition of the 'Social Science' section), little co-occurrences can be expected to be found.

In order to avoid the problems concerning co-occurrences of frequent terms that are not very significant to the specific query interests, we used for the original matrix A the $TF * IDF$ weighting schema [14], that should assign small values to terms that appear frequently and uniformly in the collection, and hence are not discriminative for particular queries. Of course, other weighting schemata that fulfil the same requirement can be adopted as well. The objective was retrieving all the Wikipedia pages related to a given query, but not the others. More precisely, since for any query it is always possible to assess a similarity to any document in the collection, the technique always returns the whole set of documents, ordered by decreasing similarity. Thus, the actual objective consisted in finding all the Web pages related to the query in the top ranking positions, and the others in the lower positions. The technique should work because one can assume that articles concerning the same subject share many specific technical terms for that subject, and hence the co-occurrence weight of such terms should be high. Clearly, a number of general terms related to social science can be found in all documents, and hence there is a chance of having at least a chain that can connect any article to any other, which represents an additional difficulty to stress the approach and test its robustness.

Here we report two sample queries, and the corresponding top positions of the ranking. The former is a sentence actually present in document 'school': "*A school is an institution designed to allow and encourage students to learn, under the supervision of teachers*".

LSE School > Pedagogy > Law > University > Banking > ...
SuMMa School > Sociology > University > Ethnology > Aviation > ...

The latter is a pair of words that are not specifically taken from any indexed document: "*commerce transport*".

LSE Nautical > Transport > Vehicles > Commerce > Railway > ...
SuMMa Tax > Administration > Transport > Politics > Vehicles > ...

In the former case, both techniques retrieved the correct document in first positions. Then, they retrieve several similar documents/categories in the other positions, although with different ranking. Interestingly, SuMMa returns 'Ethnology' in fourth position, that has some relatedness to the query, although no query term is present in the corresponding Web page. In the latter case, both techniques return sensible, although partly different, results in all positions. We can conclude that these preliminary results showed that the technique is able to select the relevant documents and place them in the top positions of the ranking. Although spurious documents are sometimes found, even in the first places, overall the top-ranked documents are actually related to the query, and hence

the precision is quite high in the first positions. An interesting point is that the technique seems to work even for short queries made up of a single word or very few words, while other techniques based on latent semantic require longer queries in order to better assess the net of relationships with the various documents.

5 Conclusions

Reaching high precision and recall rates in the results of term-based queries on text collections is becoming more and more crucial, as long as the amount of available documents increases and their quality tends to decrease. In particular, retrieval techniques based on the strict correspondence between terms in the query and terms in the documents miss important and relevant documents when it just happens that the terms selected by their authors are slightly different than those used by the final user that issues the query. Several approaches proposed in the literature try to tackle this problem by switching to an (implicit or explicit) semantic level, but this solution introduces further problems that have not been completely solved yet. Our proposal is to remain in the purely lexical level, that ensures simpler handling, but to explicitly consider term co-occurrence when building the vector space. Indeed, although the actual presence of a query term in a document is clearly a significant hint of the relevance of the document, an absence thereof must not necessarily mean that the document is irrelevant: The presence in a document of different but related terms to those in the query should strengthen the confidence that the document is relevant as well. Missing a query term in a document, but finding several terms strictly related to it, should equally support the hypothesis that the document is actually relevant. The computational perspective of such a relatedness that we proposed to adopt consists in direct or indirect term co-occurrence in the collection. We proposed two different approaches to enforce such a perspective, and run preliminary experiments on a prototypical implementation, that suggested this technique to be potentially profitable.

Currently, more extensive experiments are being run to obtain a statistically significant assessment of the performance of the proposed technique in terms of precision and recall. Moreover, a more thorough comparison to other existing techniques, and in particular to the LSI, are planned to highlight the strengths and weaknesses of each with respect to the other. Future work will also include the definition of techniques to threshold the search results avoiding that the whole list is displayed. Indeed, not being associated to the actual presence of specific terms, the query result always include in the ranking all documents in the collection. This means that also the precision/recall evaluation needs to be carried out after defining a precise strategy to select the results to be considered. Available options are using an absolute threshold for the final weight assigned to each document, or a fixed number of results to be returned independently of the actual weights, or a technique based on the difference in weight between two adjacent elements in the ranking, or a combination thereof. Other issues to be studied are related to the efficiency improvement in terms of space and time of

computation, in order to make the technique practically viable also on real-sized document collections.

References

[1] Buzydlowski, J.W., White, H.D., Lin, X.: Term co-occurrence analysis as an interface for digital libraries. In: Proceedings of the Joint Conference on Digital Libraries (2001)
[2] Ide, N., Véronis, J.: Word sense disambiguation: The state of the art. Computational Linguistics 24, 1–40 (1998)
[3] Karpys, G., Han, E.: Concept indexing: A fast dimensionality reduction algorithm with applications to document retrieval and categorization. Technical Report 00-016, Minnesota University Minneapolis (March 2000)
[4] Kontostathis, A., Pottenger, W.M.: Detecting patterns in the lsi term-term matrix. In: Proceedings of the ICDM 2002 Workshop on Foundations of Data Mining and Discovery (2002)
[5] Kontostathis, A., Pottenger, W.M.: A framework for understanding latent semantic indexing (lsi) performance. Inf. Process. Manage. 42(1), 56–73 (2006)
[6] Krovetz, R.: More than one sense per discourse. In: NEC Princeton NJ Labs., Research Memorandum (1998)
[7] Landauer, T.K., Dumais, S.T.: A solution to plato's problem: The latent semantic analysis theory of the acquisition, induction, and representation of knowledge. Psychological Review 104, 111–140 (1997)
[8] Landauer, T.K., Foltz, P.W., Laham, D.: An introduction to latent semantic analysis. Discourse Processes 25, 259–284 (1998)
[9] Magnini, B., Cavaglià, G.: Integrating subject field codes into wordnet. In: Proceedings of LREC 2000, Second International Conference on Language Resources and Evaluation, pp. 1413–1418 (2000)
[10] Miller, G.A.: Wordnet: A lexical database for English. Communications of the ACM 38(11), 39–41 (1995)
[11] O'Brien, G.W.: Information management tools for updating an svd-encoded indexing scheme. Technical Report CS-94-258, University of Tennessee, Knoxville (October 1994)
[12] Peat, H.J., Willet, P.: The limitations of term co-occurrence data for query expansion in document retrieval systems. Journal of the American Society for Information Science 42(5), 378–383 (1991)
[13] Rapp, R.: The computation of word associations: Comparing syntagmatic and paradigmatic approaches. In: Proceedings of the 29th International Conference on Computational Linguistics (2002)
[14] Salton, G., McGill, M.J.: Introduction to Modern Information Retrieval. McGraw-Hill, New York (1983)
[15] Salton, G., Wong, A., Yang, C.S.: A vector space model for automatic indexing. Communications of the ACM 18(11), 613–620 (1975)
[16] Sebastiani, F.: Machine learning in automated text categorization. ACM Computing Surveys 34, 1–47 (2002)