A Domain Based Approach to Information Retrieval in Digital Libraries

F. Rotella[1], S. Ferilli[1,2], and F. Leuzzi[1]

[1] Dipartimento di Informatica – Università di Bari
ferilli@di.uniba.it {fabio.leuzzi, fulvio.rotella}@uniba.it
[2] Centro Interdipartimentale per la Logica e sue Applicazioni – Università di Bari

Abstract. The current abundance of electronic documents requires automatic techniques that support the users in understanding their content and extracting useful information. To this aim, improving the retrieval performance must necessarily go beyond simple lexical interpretation of the user queries, and pass through an understanding of their semantic content and aims. It goes without saying that any digital library would take enormous advantage from the availability of effective Information Retrieval techniques to provide to their users. This paper proposes an approach to Information Retrieval based on a correspondence of the domain of discourse between the query and the documents in the repository. Such an association is based on standard general-purpose linguistic resources (WordNet and WordNet Domains) and on a novel similarity assessment technique. Although the work is at a preliminary stage, interesting initial results suggest to go on extending and improving the approach.

1 Introduction

The easy and cheap production of documents using computer technologies, plus the extensive digitization of legacy documents, have caused a significant flourishing of documents in electronic format, and the spread of Digital Libraries (DLs) aimed at collecting and making them available to the public, removing time and space barriers to distribution and fruition that are typical of paper material. In turn, the fact that anybody can produce and distribute documents (without even the cost of printing them) may negatively affect the average quality of their content. Although, as a particular kind of library, a DL has the mission of gathering a collection of documents which meets the quality standards chosen by the institution that maintains it, some repositories may adopt looser quality enforcing policies, and leave this responsibility to the authors, also due to the difficulty in manually checking and validating such a huge amount of material. In these cases, the effectiveness of document retrieval might be significantly tampered, affecting the fruition of the material in the repository as a consequence. Under both these attacks, anyone who is searching for information about a given topic is often overwhelmed by documents that only apparently are suitable for satisfying his information needs. In fact, most information in these documents is redundant, partial, sometimes even wrong or just unsuitable for the user's aims.

A possible way out consists in automatic instruments that (efficiently) return significant documents as an answer to user queries, that is the branch of interest of Information Retrieval (IR).

IR aims at providing the users with techniques for finding interesting documents in a repository, based on some kind of query. Although multimedia digital libraries are starting to gain more and more attention, the vast majority of the content of current digital document repositories is still in textual form. Accordingly, user queries are typically expressed in the form of natural language sentences, or sets of terms, based on which the documents are retrieved and ranked. This is clearly a tricky setting, due to the inherent ambiguity of natural language. Numerical/statistical manipulation of (key)words has been widely explored in the literature, but in its several variants seems unable to fully solve the problem. Achieving better retrieval performance requires to go beyond simple lexical interpretation of the user queries, and pass through an understanding of their semantic content and aims.

This work focuses on improving fruition of a DL content, by means of advanced techniques for document retrieval that try to overcome the aforementioned ambiguity of natural language. For this reason, we looked at the typical behavior of humans, when they take into account the possible meanings underlying the most prominent words that make up a text, and select the most appropriate one according to the context of the discourse. To carry out this approach, we used a well-known lexical taxonomy, and its extension to deal with domain categories, as a background knowledge.

The rest of this paper is organized as follows. After a brief recall of previous work on Information Retrieval, with a particular attention to techniques aimed at overcoming lexical limitations, toward semantic aspects, Section 3 introduces a new proposal for semantic information retrieval based on taxonomic information. Then, Section 4 proposes an experimental evaluation of the proposed technique, with associated discussion and evaluation. Lastly, Section 5 concludes the paper and outlines open issues and future work directions.

2 Related Work

Many works, aimed at building systems that tackle the Information Retrieval problem, exist in the literature. Most of such works are based on the ideas in [15], a milestone in this field. This approach, called *Vector Space Model* (VSM), represents a corpus of documents D, and the set of terms T appearing in those documents, as a $T \times D$ matrix, in which the (i, j)-th cell contains a weight representing the importance of the i-th term in the j-th document (usually computed according to the number and distribution of its occurrences both in that document and in the whole collection). This allows to compute the degree of similarity of a user query to any document in the collection, simply using any geometrical distance measure on that space. Much research has been spent on developing effective similarity measures and weighting schemes, and on variations of their implementations to enhance retrieval performance. Most similarity approaches [8, 17, 16] and weighting schemes [14, 13, 18] are based on inner product and cosine measure. Motivations came, on one hand, from the growth of the Web, and, on the other, from the success of some implementations in Web search engines. One limitation of these approaches is their considering a document only from a lexical point of view, which is typically affected by several kinds of linguistic tricks: e.g., synonymy (different words having similar meaning), and polysemy (words having many different meanings).

More recently, techniques based on dimensionality reduction have been explored for capturing the concepts present in the collection. The main idea behind these techniques is mapping both the documents in the corpus and the queries into a lower dimensional space that explicitly takes into account the dependencies between terms. Then, the associations provided by the low-dimensional representation can be used to improve the retrieval or categorization performance. Among these techniques, Latent Semantic Indexing (LSI) [3] and Concept Indexing (CI) [9] can be considered relevant. The former is a statistical method that is capable of retrieving texts based on the concepts they contain, not just by matching specific keywords, as in previous approaches. It starts from a classical VSM approach, and applies Singular Value Decomposition (SVD) to identify latent concepts underlying the collection, and the relationships between these concepts and the terms/documents. Since the concepts are weighted by relevance, dimensionality reduction can be carried out by filtering out less relevant concepts, and the associated relationships. In this way new associations emerge between terms that occur in similar contexts, and hence query results may include documents that are conceptually similar in meaning to the query even if they don't contain the same words as the query. The latter approach, CI, carries out an indexing of terms using concept decomposition (CD) [4] instead of SVD (as in the LSI). It represents a collection of documents in k-dimensions by first clustering the documents in k groups using a variant of the k-means algorithm [11], and considering each group as potentially representing a different concept in the collection. Then, the cluster centroids are taken as the axes of the reduced k-dimensional space. Although LSI and CI have had much success (e.g., LSI was implemented by Google) for their ability to reduce noise, redundancy, and ambiguity, they still pose some questions. First of all, their high computational requirements prevent exploitation in many digital libraries. Moreover, since they rely on purely numerical and automatic procedures, the noisy and redundant semantic information must be associated with a numerical quantity that must be reduced or minimized by the algorithms. Last but not least, a central issue is the choice of the matrix dimension [2].

3 A Domain-based Approach

This section describes a proposal for a domain-based approach to information retrieval in digital libraries. In order to get rid of the constraints imposed by the syntactic level, we switch from the terms in the collection to their meaning by choosing a semantic surrogate for each word, relying on the support of external resources. At the moment, we exploit WordNet [5], and its extension WordNet Domains [12], as readily available general-purpose resources, although the proposed technique applies to any other taxonomy.

The first step consists in off-line preprocessing the digital library in order to obtain, for each document, a list of representative keywords, to each of which the corresponding meaning will be associated later on. Using a system based on the DOMINUS framework [7], each document in the digital library is progressively split into paragraphs, sentences, and single words. In particular, the Stanford Parser [10] is used to obtain the syntactic structure of sentences, and the lemmas of the involved words. In this proposal, only nouns are considered and used to build a classical VSM weighted according to the TF*IDF scheme. In addition to stopwords, typically filtered out by all term-based approaches, we ignore adverbs, verbs and adjectives as well, because their representation in WordNet is different than that of nouns (e.g., verbs are organized in a separate taxonomy), and so different strategies must be defined for exploiting these lexical categories, which will be the subject of future work. More specifically, only those nouns that are identified as keywords for the given documents, according to the techniques embedded in DOMINUS, are considered. In order to be noise-tolerant and to limit the possibility of including non-discriminative and very general words (i.e., common words that are present in all domains) in the semantic representation of a document, it can be useful to rank each document keyword list by decreasing TF*IDF weight and to keep only the top items (say, 15) of each list.

The next step consists in mapping each keyword in the document to a corresponding synset (i.e., its semantic representative) in WordNet. Since this task is far from being trivial, due to the typical polysemy of many words, we adopt the one-domain-per-discourse (ODD) assumption as a simple criterion for Word Sense Disambiguation (WSD): the meanings of close words in a text tend to refer to the same domain, and such a domain is probably the dominant one among the words in that portion of text. Hence, to obtain such synsets, we need to compute for each document the prevalent domain. First we take from WordNet all the synsets of each word, then, for each synset, we select all the associated domains in WordNet Domains. Then, each domain is weighted according to the density function presented in [1], depending on the number of domains to which each synset belongs, on the number of synsets associated to each word, and on the number of words that make up the sentence. Thus, each domain takes as weight the sum of all the weights of synsets associated to it, which results in a ranking of domains by decreasing weight. This allows to perform the WSD phase, that associates a single synset to each term by solving possible ambiguities using the domain of discourse (as described in Algorithm 1). Now, each document is represented by means of WordNet synsets instead of terms.

The output of the previous step, for each document, is a list of pairs, made up of keywords and their associated synsets. All these synsets are partitioned into different groups using pairwise clustering, as shown in Algorithm 2: initially each synset makes up a different singleton cluster; then, the procedure works by iteratively finding the next pair of clusters to merge according to the *complete*

Algorithm 1 Find "best synset" for a word

```
Input: word $t$, list of domains with weights.
Output: best synset for word $t$.

  $bestSynset \leftarrow empty$
  $bestDomain \leftarrow empty$
  for all $synset(s_t)$ do
    $maxWeight \leftarrow -\infty$
    $optimalDomain \leftarrow empty$
    for all $domains(d_s)$ do
      if $weight(d_s) > maxWeight$ then
        $maxWeight \leftarrow weight(d_s)$
        $optimalDomain \leftarrow d_s$
      end if
    end for
    if $maxWeight > weight(bestDomain)$ then
      $bestSynset \leftarrow s_t$
      $bestDomain \leftarrow optimalDomain$
    end if
  end for
```

link strategy (shown in Algorithm 3), based on the similarity function proposed in [6]:

$$sf(i',i'') = sf(n,l,m) = \alpha\frac{l+1}{l+n+2} + (1-\alpha)\frac{l+1}{l+m+2}$$

where:

- i' and i'' are the two items (synsets in this case) under comparison;
- n represents the information carried by i' but not by i'';
- l is the common information between i' and i'';
- m is the information carried by i'' but not by i';
- α is a weight that determines the importance of i' with respect to i'' (0.5 means equal importance).

In particular, we adopt a global approach based on all the information provided by WordNet on the two synsets, rather than on just one of their subsumers as in other measures in the literature. Indeed, we compute the distance between each pair (i',i'') by summing up three applications of this formula, using different parameters n, m and l. The first component works in depth, and obtains the parameters by counting the number of common and different hypernyms between i' and i''. The second one works in breadth, and considers all the synsets with which i' and i'' are directly connected by any relationship in WordNet, and then takes the number of common related synsets as parameter l, and the rest of synsets, related to only i' or i'', as parameters n and m. Lastly, the third component is similar to the second one, but it considers the inverse relationships (incoming links) in the computation. The considered relationships in the last two measures are:

- *member meronimy*: the latter synset is a member meronym of the former;
- *substance meronimy*: the latter synset is a substance meronym of the former;
- *part meronimy*: the latter synset is a part meronym of the former;
- *similarity*: the latter synset is similar in meaning to the former;
- *antonym*: specifies antonymous word;
- *attribute*: defines the attribute relation between noun and adjective synset pairs in which the adjective is a value of the noun;
- *additional information*: additional information about the first word can be obtained by seeing the second word;
- *part of speech based*: specifies two different relations based on the parts of speech involved;
- *participle*: the adjective first word is a participle of the verb second word;
- *hyperonymy*: the latter synset is a hypernym of the former.

Example 1. To give an idea of the breadth-distance between S_1 and S_2, let us consider the following hypothetical facts in WordNet:

$$\begin{array}{ccc} rel_1(S_1, S_3) & rel_2(S_1, S_4) & rel_3(S_1, S_5) \\ \multicolumn{3}{c}{rel_4(S_2, S_5) \qquad\qquad rel_5(S_2, S_6)} \end{array}$$

for the direct component, and

$$\begin{array}{ccc} rel_1(S_7, S_1) & rel_2(S_8, S_1) & rel_3(S_9, S_1) \\ rel_4(S_9, S_2) & rel_5(S_3, S_2) & rel_2(S_8, S_2) \end{array}$$

for the inverse component, where rel_i represents one of the relationships listed above. In the former list, the set of synsets linked to S_1 is $\{S_3, S_4, S_5\}$ and the set of synsets linked to S_2 is $\{S_5, S_6\}$. Their intersection is $\{S_5\}$, hence we have $n = 2$, $l = 1$, $m = 1$ as parameters for the similarity formula. In the latter list, the set of synsets linked to S_1 is $\{S_7, S_8, S_9\}$ and the set of synsets linked to S_2 is $\{S_9, S_3, S_8\}$, yielding $n = 1$, $l = 2$, $m = 1$ as parameters for the similarity formula. The depth-distance component considers only hypernyms, and collects the whole sets of ancestors of S_1 and S_2.

Now, each document is considered in turn, and each of its keywords votes for the cluster to which the associated synset has been assigned (as shown in Algorithm 4). The contribution of such a vote is equal to the TF*IDF value established in the keyword extraction phase normalized on the sum of the weights of the chosen keywords. However, associating each document to only one cluster as its descriptor would be probably too strong an assumption. To smooth this, clusters are ranked in descending order according to the votes they obtained, and the document is associated to the first three clusters in this ranking. This closes the off-line preprocessing macro-phase, aimed at suitably partitioning the whole document collection according to different sub-domains. In our opinion, the pervasive exploitation of domains in this phase justifies the claim that the proposed approach is *domain-based*. Indeed, we wanted to find sets of similar synsets that might be usefully exploited as a kind of 'glue' binding together a sub-collection of documents that are consistent with each other. In this perspective,

Algorithm 2 Pairwise clustering of all detected synsets

Input: *S*: list of all synsets detected in WSD phase applied to the keywords; *C*: an empty set of clusters.
Output: set of clusters.

```
for all $s_i \in S$ do
    $c_i \leftarrow s_i \mid c_i \in C$
end for
for all $pair(s_i, s_j) \mid i \neq j$ do
    if $completeLink(s_i, s_j)$ then
        $clustersAgglomaration(s_i, s_j)$
    end if
end for
```

Algorithm 3 Complete link between two clusters

Input: *C1*: former cluster; *C2*: latter cluster; *T*: the threshold for Ferilli at al. similarity measure.
Output: check outcome.

```
for all $c_i \in C1$ do
    for all $k_j \in C2$ do
        if $similarityScore(c_i, k_j) < T$ then
            $return \rightarrow false$
        end if
    end for
end for
$return \rightarrow true$
```

the obtained clusters can be interpreted as intensional representations of specific domains, and thus they can be exploited to retrieve the sub-collection they are associated to. Note that a cluster might correspond to an empty set of documents (when it was not in the 3 most similar clusters of any document in the collection).

The previous steps pave the way for the subsequent on-line phase, in which information retrieval is actually carried out. This phase starts with a user's query in natural language. The query undergoes the same grammatical preprocessing as in the off-line phase, yielding a set of words that are potentially useful to detect the best subset of documents to be presented as a result. For consistency with the off-line phase, only nouns are chosen among the words in the query. However, since the query is usually very short, keyword extraction is not performed, and all nouns are retained for the next operations. For each word, all corresponding synsets are taken from WordNet. Since WSD applied to the query would not be reliable (because it might be too short to identify a significant domain), we decided to keep all synsets for each word, and to derive from a single lexical query many semantic queries (one for each combination of synsets, one from each word). Specifically, given an n-term query, where the i-th term has associated

Algorithm 4 Association of documents to clusters

Input: D: the list of documents; W: the list of words of each document; S: the list of synsets of each document; C: the set of clusters.
Output: set of clusters with the assigned documents.

```
$V$ : vector of votes, one for cluster. Starting value: 0.
for all $d_i \in D$ do
  for all $w_i \in W$ do
    $s \leftarrow getSynset(w_i)$
    $c \leftarrow getClusterOfSynset(s)$
    $V.getVoteOfCluster(c, s.getCluster())$
  end for
  $rankedList \leftarrow descendingOrdering(V)$
  for all $v_j \in V \mid 0 \leq j < 3$ do
    $associateDocumentToCluster(d_i, v_j.getCluster)$
  end for
end for
```

n_i synsets, $\prod_{i=1}^{n} n_i$ semantic queries are obtained, each of which represents a candidate disambiguation.

For each such query, a similarity evaluation is performed against each cluster that has at least one associated document, using the same complex similarity function as for clustering, that takes as input two sets of synsets (those in the query and those associated to the cluster), computes the distance between each possible pair of synsets taken from such sets, and then returns the maximum distance between all such pairs. This evaluation has a twofold objective: finding the combination of synsets that represents the best word sense disambiguation, and obtaining the cluster to which the involved words are most similar. The main motivation for which this phase considers only clusters that have at least one associated document is that, as already stated, clusters can be interpreted a set of descriptors for document subsets, and hence it makes sense keeping only those descriptors that are useful to identify the best set of documents according to the user's search. At this point, the best combination is used to obtain the list of clusters ranked by descending relevance, that can be used as an answer to the user's search. It should be pointed out that the ranked list is exploited, instead of taking just the best cluster, to avoid the omission of potentially useful results contained in positions following the top, this way losing information.

4 Evaluation

To understanding the contribution of each step in the overall result, we used a collection made up of 200 documents obtained by randomly drawing 50 documents from 4 Wikipedia top-categories (general science, music, politics, religion). A structured version of the Wikipedia dump was obtained exploiting the Java Wikipedia Library [19]. A selection of queries, with a corresponding performance

Table 1. Performance evaluation

#	Query	Outcomes	P	P'
1	creation of the mankind	[1 to 5] religion [6 to 10] science [+3] science	0.5	1.0
2	traditions and folks	[1 to 8] music [9 to 10] religion [+3] religion	0.8	1.0
3	ornaments and melodies	[1 to 8] music [9] science [10] religion	0.8	0.9
4	capitalism vs communism	[1 to 2] religion [3 to 10] politics [+4] politics	0.8	0.8
5	markets and new economy	[1 to 10] politics [+1] politics	1.0	1.0
6	gene structure and function	[1 to 2] science [3] religion [4] politics [5 to 10] science [+2] science	0.8	0.8

evaluation, is summarized in Table 1. For each query, the ranked list of most similar clusters was considered, and the top 10 documents were exploited for evaluating two performance measures: classical Precision P, expressing how many retrieved documents belong to the intended category of the query, and a looser version thereof P', considering as good outcomes also documents in categories that are compatible with the query, even if that was not in the user intention. A first consideration is that the decision to take several clusters (not just the top-ranked one) improved the result for all queries as regards true positives. In addition to the best 10 documents used for computing P and P', we have also reported (preceded by a '+' symbol) the number of immediately following documents that were nevertheless relevant for the query, which shows that good performance is not limited to top items only. Going beyond the purely numerical figures expressing the above measures, also a deeper insight into the specific cases reveals interesting aspects. For instance all results for query # 1 can be accepted as good, taking into account that a scientific perspective might correctly satisfy the user's search about the creation of the mankind, as well. Also for query # 2, it is quite agreeable that both traditions and folks are strictly related to religion as well as popular music. This motivated further analysis of some specific queries. In the following, for the sake of readability, when dealing with concepts both the synset code, and the set of associated terms, along with the corresponding gloss, will be reported. We will focus specifically on two sample queries purposely selected to help the reader understand the corresponding behavior.

The former is *ornaments and melodies.* Only 2 combinations were found, among which the best one was:

- *synset*: 103169390; *lemmas*: decoration, ornament and ornamentation; *gloss*: something used to beautify;
- *synset*: 107028373; *lemmas*: air, line, melodic line, melodic phrase, melody, strain and tune; *gloss*: a succession of notes forming a distinctive sequence.

This combination was recognized by the technique to be most similar to the following cluster:

- *synset*: 107044760; *lemmas*: symphonic music, symphony; *gloss*: a long and complex sonata for symphony orchestra;
- *synset*: 107033753; *lemmas*: mass; *gloss*: a musical setting for a Mass;
- *synset*: 107026352; *lemmas*: opera; *gloss*: a drama set to music, consists of singing with orchestral accompaniment and an orchestral overture and interludes;
- *synset*: 107071942; *lemmas*: genre, music genre, musical genre and musical style; *gloss*: an expressive style of music;
- *synset*: 107064715; *lemmas*: rock, rock 'n' roll, rock and roll, rock music, rock'n'roll and rock-and-roll; *gloss*: a genre of popular music originating in the 1950s, a blend of black rhythm-and-blues with white country-and-western;
- *synset*: 107043275; *lemmas*: concerto; *gloss*: a composition for orchestra and a soloist.

It's easy to note that this cluster contains elements that are consistent with each other, a positive result that we may trace back to the decision of using a complete link pair-wise clustering, which is more restrictive in grouping items. In particular, this cluster represents an intensional description of 8 documents returned as first (or more relevant) outcomes, all talking about music. Furthermore, it is noteworthy that this query result satisfies the initial aim, of retrieving query-related documents that do not necessarily contain the terms that are present in the query. Thus, the technique is actually able to go beyond simple lexical interpretation of the user queries, retrieving documents in which no occurrence of the words forming the query are present, even in cases in which those words are not present at all in the entire collection. The latter sample is *market and new economy.* It is made up of 2 nouns, yielding a total of 20 combinations to be analyzed, of which the system recognized as the best one the following:

- *synset*: 108424951; *lemmas*: market; *gloss*: the customers for a particular product or service;
- *synset*: 100192613; *lemmas*: economy, saving; *gloss*: an act of economizing; reduction in cost.

The most similar cluster was:

- *synset*: 108166552; *lemmas*: country, land, nation; *gloss*: the people who live in a nation or country;

- *synset*: 108179689; *lemmas*: populace, public, world; *gloss*: people in general considered as a whole;
- *synset*: 107965937; *lemmas*: domain, world; *gloss*: people in general, especially a distinctive group of people with some shared interest.

Here we obtained 8 main results talking about politics. As in the former case, we can appreciate both the benefits of returning as a result the ranked list of clusters instead just the best one, and the consistency of the cluster elements. Again, it should be noted that, although very simple, the WSD technique based on the one-domain-per-discourse assumption was able to select a strongly consistent solution.

5 Conclusions

This work proposed an approach to extract information from digital libraries trying to go beyond simple lexical matching, toward the semantic content underlying the actual aims of user queries. For all the documents in the corpus, after a keyword extraction phase, all keywords are disambiguated with a simple domain-driven WSD approach. The synsets obtained in this way are clustered, and each document is assigned to the cluster which contains more synsets related to its keywords. Then, given a user query, due to the typically low number of words in a query, that would affect the reliability of the WSD technique, all possible combinations of word meanings are considered, and the one that is most similar to a cluster is chosen. The outcome of the query presents the set of retrieved documents ranked by decreasing similarity of the associated cluster with such a combination. Preliminary experiments show that the approach can be viable, although extensions and refinements are needed to improve its effectiveness. In particular, the substitution of the ODD assumption with a more elaborated strategy for WSD might produce better results. Another issue regards incrementality: the current version of the approach requires a pre-processing, due to the underlying techniques for keyword extraction and clustering; this might be limiting when new documents are progressively included in the collection, a case that is very important in some digital libraries. Moreover, it might be interesting to evaluate the inclusion of adverbs, verbs and adjectives in order to improve the quality of the semantic representatives of the documents, and to explore other approaches to choose better intensional descriptions of each document.

References

[1] M. Angioni, R. Demontis, and F. Tuveri. A semantic approach for resource cataloguing and query resolution. *Communications of SIWN. Special Issue on Distributed Agent-based Retrieval Tools*, 5:62–66, aug 2008.

[2] Roger B. Bradford. An empirical study of required dimensionality for large-scale latent semantic indexing applications. In *Proceedings of the 17th ACM conference on Information and knowledge management*, CIKM '08, pages 153–162, New York, NY, USA, 2008. ACM.

[3] Scott Deerwester. Improving Information Retrieval with Latent Semantic Indexing. In Christine L. Borgman and Edward Y. H. Pai, editors, *Proceedings of the 51st ASIS Annual Meeting (ASIS '88)*, volume 25, Atlanta, Georgia, October 1988. American Society for Information Science.

[4] Inderjit S. Dhillon and Dharmendra S. Modha. Concept decompositions for large sparse text data using clustering. In *Machine Learning*, pages 143–175, 2001.

[5] Christiane Fellbaum, editor. *WordNet: An Electronic Lexical Database.* MIT Press, Cambridge, MA, 1998.

[6] S. Ferilli, M. Biba, N. Di Mauro, T.M. Basile, and F. Esposito. Plugging taxonomic similarity in first-order logic horn clauses comparison. In *Emergent Perspectives in Artificial Intelligence*, Lecture Notes in Artificial Intelligence, pages 131–140. Springer, 2009.

[7] Stefano Ferilli. *Automatic Digital Document Processing and Management: Problems, Algorithms and Techniques.* Springer Publishing Company, Incorporated, 1st edition, 2011.

[8] William P. Jones and George W. Furnas. Pictures of relevance: A geometric analysis of similarity measures. *Journal of the American Society for Information Science*, 38(6):420–442, 1987.

[9] George Karypis and Eui-Hong (Sam) Han. Concept indexing: A fast dimensionality reduction algorithm with applications to document retrieval and categorization. Technical report, IN CIKM00, 2000.

[10] Dan Klein and Christopher D. Manning. Fast exact inference with a factored model for natural language parsing. In *Advances in Neural Information Processing Systems*, volume 15. MIT Press, 2003.

[11] J. B. MacQueen. Some methods for classification and analysis of multivariate observations. In L. M. Le Cam and J. Neyman, editors, *Proc. of the fifth Berkeley Symposium on Mathematical Statistics and Probability*, volume 1, pages 281–297. University of California Press, 1967.

[12] Bernardo Magnini and Gabriela Cavagli. Integrating subject field codes into wordnet. pages 1413–1418, 2000.

[13] Stephen E. Robertson, Steve Walker, Susan Jones, Micheline Hancock-Beaulieu, and Mike Gatford. Okapi at trec-3. In *TREC*, pages 0–, 1994.

[14] G. Salton. *The SMART Retrieval System—Experiments in Automatic Document Processing.* Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1971.

[15] G. Salton, A. Wong, and C. S. Yang. A vector space model for automatic indexing. *Commun. ACM*, 18:613–620, November 1975.

[16] Gerard Salton. Automatic term class construction using relevance–a summary of work in automatic pseudoclassification. *Inf. Process. Manage.*, 16(1):1–15, 1980.

[17] Gerard Salton and Michael McGill. *Introduction to Modern Information Retrieval.* McGraw-Hill Book Company, 1984.

[18] Amit Singhal, Chris Buckley, Mandar Mitra, and Ar Mitra. Pivoted document length normalization. pages 21–29. ACM Press, 1996.

[19] Torsten Zesch, Christof Müller, and Iryna Gurevych. Extracting lexical semantic knowledge from wikipedia and wiktionary. In *Proceedings of the 6th International Conference on Language Resources and Evaluation*, Marrakech, Morocco, May 2008. electronic proceedings.