Searching Documents with Semantically Related Keyphrases

Ibrahim Aygul, Nihan Cicekli
Department of Computer Engineering
Middle East Technical University
Ankara, Turkey
ibrahimaygul@gmail.com, nihan@ceng.metu.edu.tr

Ilyas Cicekli
Department of Computer Engineering
Hacettepe University
Ankara, Turkey
ilyas@cs.hacettepe.edu.tr

Abstract — **In this paper, we present a tool, called SemKPSearch, for searching documents by a query keyphrase and keyphrases that are semantically related with that query keyphrase. By relating keyphrases semantically, we aim to provide users an extended search and browsing capability over a document collection and to increase the number of related results returned for a keyphrase query. Keyphrases provide a brief summary of the content of documents, and they can be either author assigned or automatically extracted from the documents. SemKPSearch uses a set of keyphrase indexes called SemKPIndex, and they are generated from the keyphrases of documents. In addition to a keyphrase-to-document index, SemKPIndex also contains a keyphrase-to-keyphrase index which stores semantic relation scores between the keyphrases in a document collection. The semantic relation score between keyphrases is calculated using a metric which considers the similarity score between words of the keyphrases, and the semantic similarity score between two words is determined with the help of two word-to-word semantic similarity metrics based on WordNet. SemKPSearch is evaluated by human evaluators, and the evaluation results showed that the evaluators found the documents retrieved with SemKPSearch more related to query terms than the documents retrieved with a search engine.**

***Keywords:** keyphrase extraction, semantic similarity, information retrieval, digital librarie.*

I. INTRODUCTION

The number of documents available electronically has increased dramatically and the use of large document collections such as digital libraries has become widespread. Browsing a document collection and finding the documents of interest turns out to be more difficult. The full-text inverted indexes and ranking algorithms cause standard search engines often return a high number of results, and it is an overwhelming process to find whether a collection covers the useful information.

Gutwin et al. state that full-text indexing has several problems in browsing a collection [6]. First, although users can retrieve documents containing the words of user's query text, they usually use short topic phrases to explore a collection. The second problem stated by Gutwin et al. is the result set. Standard search engines return a list of documents which is too specific for browsing purposes. And lastly with the nature of browsing, the third problem is the query refinement, and standard engines do not support constituting new queries. For the solution to these problems, Gutwin et al. propose a search engine "Keyphind" which is especially designed to help browsing document collections [6]. Keyphind uses keyphrase indexes in order to allow users to interact with the document collection at the level of topics and subjects. Keyphrases provide a brief description of a document's content and can be viewed as semantic metadata that summarize documents. Keyphrases are widely used in information retrieval systems [4,5,7,9,11] and other document browsing systems [8,15]. With the help of the keyphrases of documents in the collection, the user can easily guess the coverage of documents and browse the relevant information.

In this paper, we present a keyphrase-based search engine, called SemKPSearch, using a set of keyphrase based indexes which is similar to the Keyphind index, for browsing a document collection. With the help of keyphrase indexes, the user can browse documents which have semantically related keyphrases with the query text. In this work we extend the keyphrase index with a novel keyphrase to keyphrase index which stores the evaluated semantic similarity score between the keyphrases of the documents in a collection. To calculate similarity scores between keyphrases, we use the text semantic similarity measure given in [3] which employs a word-to-word similarity measure. We use a word-to-word semantic similarity metric [12] in the calculation of keyphrase similarities.

To evaluate SemKPSearch, we used a test corpus that is collected by Krapivin et. al. [10]. The corpus has full-text articles and author assigned keyphrases. We also used the keyphrase extraction system KEA [16] to evaluate the system with automatically extracted keyphrases. We created keyphrase indexes for both author assigned and automatically extracted keyphrases. To determine the retrieval performance of SemKPSearch, we have evaluated SemKPSearch with Google Desktop search tool which uses full-text index. The evaluation is done by human testers, and evaluation results showed that SemKPSearch suggests valuable and helpful keyphrases that are semantically related with the query of the tester and the document retrieval performance is better than Google Desktop.

Section 2 describes the overall structure of SemKPSearch in addition to its index structure and generation. In Section 3, the evaluation methods and experimental results are presented. Section 4 concludes the paper and discusses the future work.

II. Searching with Semantically Related Keyphrases

The searching and browsing interface of SemKPSearch is developed for querying documents in a digital library using their keyphrases. A keyphrase based index, SemKPIndex, is created for a document collection and SemKPSearch uses SemKPIndex for querying and browsing the collection in a user friendly interface. In SemKPSearch, browsing is also aided by suggesting keyphrases that are semantically related with the given query. As the documents in the collection are indexed by their keyphrases, semantically related keyphrases are indexed with a score which is calculated by employing a semantic similarity metric. We use two semantic similarity metrics to calculate a semantic similarity score between keyphrases.

The overall structure of SemKPSearch system is shown in Figure 1. A document collection with their keyphrases is the main input to SemKPSearch. If the documents in the collection do not have author assigned keyphrases, KEA [16] is employed to extract keyphrases. In addition to indexes between keyphrases and documents in SemKPIndex, each indexed keyphrase is compared to all other keyphrases and a similarity score is calculated, and then semantically related keyphrases are also stored in SemKPIndex. Using SemKPIndex on the SemKPSearch interface, the users query the document collection with topic like keyphrases, and the interface returns a set of document results that contains query term among their keyphrases. Besides the documents that contain query term in their keyphrases, SemKPSearch suggests semantically related keyphrases using SemKPIndex, and the users can expand search results by using these suggested keyphrases.

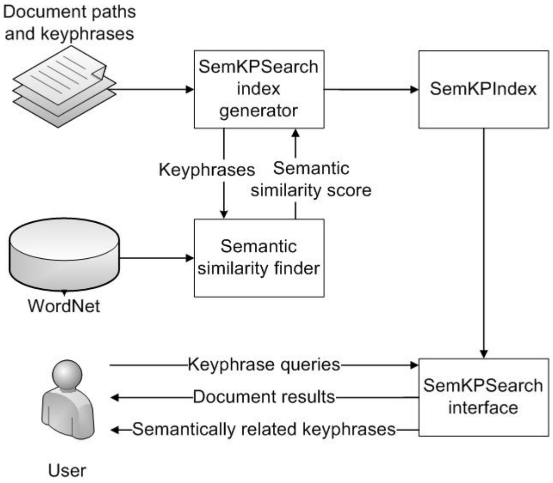
Document paths and keyphrases
SemKPSearch index generator
SemKPIndex
Keyphrases
Semantic similarity score
Semantic similarity finder
WordNet
Keyphrase queries
Document results
Semantically related keyphrases
SemKPSearch interface
User

Figure 1. Overall structure of SemKPSearch system.

A. SemKPIndex Structure

SemKPSearch uses a set of indexes, called as SemKPIndex, and it is composed of five indexes: keyphrase list, document to keyphrase index, keyphrase to document index, word to keyphrase index and keyphrase to keyphrase index. The first four indexes are very similar to the structure of Keyphind index [6], and the fifth one is our new novel index structure. The last index is a keyphrase to keyphrase index which holds semantically related keyphrases.

Keyphrase list is a list of all keyphrases that are given with the documents in the collection. This index is used as a suggestion list that guides the user with possible keyphrases as the user enters the query terms.

Document to keyphrase index contains information for each document in the collection. Each keyphrase is kept with a relation score that shows the importance of the keyphrase to the owner document. If no relation score is given for the keyphrase, it is automatically calculated during index generation. Document to keyphrase index is used to improve the search results by showing each document with its keyphrases and to order the documents in the search result.

Keyphrase to document index is a mapping from all keyphrases to the paths of the owner documents. It is somehow the inverse of the document to keyphrase index. This index is used to retrieve the documents that have a given keyphrase among its keyphrases.

Word to keyphrase index contains all words in all of the keyphrases, and each entry corresponds to the keyphrases containing the entry word. This index is needed to show the user more results and more keyphrases to extend the search. For example, when the user searches “similarity”, in addition to the documents that contain the keyphrase “similarity”, the documents containing the keyphrases “semantic similarity”, “similarity measurement”, “similarity retrieval” will be retrieved by the help of this index.

Keyphrase to keyphrase index provides the main contribution in the study, and the aim of this index is to aid users in their searches by suggesting semantically related keyphrases with query terms. The index keeps semantic relations between keyphrases in the keyphrase list. During the index generation, a semantic relation score is calculated for each pair of keyphrases in the system, and the relations that exceed a predefined threshold value are stored in this index. Each entry is a mapping from a keyphrase to its semantically related keyphrase list. For example, the index entry for the keyphrase “face recognition” in the test collection contains its semantically related keyphrases such as “face recognition algorithm”, “shape recognition”, and “identification system” together with their semantic relation scores.

The keyphrase to keyphrase index gives the user a chance to see the semantically related keyphrases with the search terms. It also helps to extend search results with the suggested semantically related keyphrases. If the search term is a keyphrase in the index, the suggested related keyphrases are obtained from the index entry of that keyphrase. On the other hand, the suggested semantically related keyphrases are produced on the fly by comparing the search term with the keyphrases in the index when the search term is not available in the index.

B. Generating SemKPIndex

SemKPSearch accepts a collection of documents and their keyphrases as inputs to the index generation process.

The keyphrases can be assigned by the authors or automatically extracted from the documents using a keyphrase extraction algorithm. The documents with their keyphrases are indexed one by one during index generation. For each document, the keyphrases of the document are added to keyphrase list. Then by using these keyphrases, other indexes are created.

The keyphrases of a document are added to document to keyphrase index together with their relation scores. If the keyphrases are found by the keyphrase extraction algorithm, their relation scores are also found. For the author assigned keyphrases, their relation scores are found relative to their positions in the keyphrase list of the document. The relation score of the i^{th} keyphrase of a document with n keyphrases is equal to 1-(i/n). Using this formula we assume that the author assigned keyphrases are given by the relevance order and the last keyphrases in the list are much less related with the document than the first keyphrases.

After creating document to keyphrase index, the keyphrases of the document are added to keyphrase to document index, and each entry in this index points to a document list sorted with relation scores. Index generation continues by adding each word of the keyphrases to the word to keyphrase index, and each word entry in the index points to a keyphrase list that gives a reference to keyphrases in which the word occurs.

After keyphrase list is created, a list of related keyphrases is created for each keyphrase in order to create keyphrase to keyphrase index. A semantic relation score is calculated for each pair of keyphrases, and top keyphrases which passes a predefined threshold semantic relation score are kept as a list of related keyphrases for each keyphrase. Each related keyphrase list is sorted with respect to the relation scores.

The semantic relatedness of two keyphrases can be calculated the same as the semantic similarity between two texts are calculated, and several methods to find the semantic similarity between two texts are discussed in the literature [3,12,13,14]). The similarity between two keyphrases is based on the similarity of their words, and Corley and Mihalcea introduce a metric that combines word-to-word similarity metrics into a text-to-text semantic similarity metric [8]. In this approach, the value of the semantic similarity between two texts is calculated using the semantic similarities of words and inverse document frequencies of words. In our study, we use Corley and Mihalcea approach to calculate the semantic similarity between two keyphrases together with the WordNet based word-to-word similarity metric proposed by Li et al. [12].

In order to find the semantic similarity between two keyphrases using the discussed similarity metrics, first we create a similarity matrix for the words of the keyphrases. All words of one keyphrase are compared to each word of the other keyphrases, and a similarity score for two words is found. Since keyphrases are short texts, it is not feasible to detect part of speech tags of a bunch of words. Besides, keyphrases of documents generally consist of nouns or verbs. Thus, for word comparisons, words are compared using their noun and verb senses in WordNet and whichever sense pair produces higher similarity score, it is chosen as the similarity score of those words.

III. Evaluation

In order to evaluate the retrieval performance and the related keyphrase suggestions of SemKPSearch, we used a test corpus that is collected by Krapivin et. al. [10]. The corpus contains 2304 papers from Computer Science domain, which were published by ACM between 2003 and 2005. It has full-text of articles and author assigned keyphrases.

We created two SemKPSearch indexes for the test corpus. The first index was created with author assigned keyphrases and the other index was created with KEA extracted keyphrases. In order to extract keyphrases automatically using KEA, 30 documents were randomly selected from the corpus and their author assigned keyphrases were given to KEA to build its training model. Then for each document in the corpus, KEA extracted 5 keyphrases which were up to 2 to 5 words. These keyphrases were selected to be used in the creation of the index. Since a one word length keyphrase may be too general, we chose keyphrases with at least 2 words in order to be able to obtain more precise keyphrases. In addition to these two SemKPIndexes, a full text index over the same corpus was created by Google Desktop [1] in order to compare SemKPSearch with Google Desktop.

We used two different word-to-word semantic similarity metrics in the calculation of the semantic relatedness of keyphrases. The first one was Wu & Palmer [17] word-to-word similarity metric, and the other one was the word similarity measure introduced by Li et al. [12]. We have tested our system with these to word-to-word similarity metrics. Since the performance of the system was better when Li et al. semantic similarity was used, here we only give the performance results of the system with this metric. We called the two created SemKPIndexes as KEA_Sim$_{Li}$ in which KEA extracted keyphrases and Li et al. similarity metric were used, and Author_SimLi in which author assigned keyphrases and Li et al. similarity metric were used.

The user evaluation was done by 8 human evaluators who were all computer scientists. Each evaluator evaluated the relevancy of the keyphrases suggested by SemKPSearch, and the documents retrieved by SemKPSearch and Google Desktop. They gave a relevance score between 0 and 4 (0:irrelevant, 1:poorly relevant, 2:partially relevant, 3:relevant, 4:completely relevant) to each retrieved document and to each suggested keyphrase according to their relevancy to the query term. Each evaluator created his own two sets of query terms by randomly selecting terms from the two given sets of query terms. The first set contains query terms which occur as keyphrases of the documents in the collection, and the second set contains query terms which do not occur as keyphrases in the collection. This means that there is no document which is indexed by a query term in the second set. The results reported here are the average scores of the 8 evaluators.

TABLE I. AVERAGE SCORES FOR THE FIRST K SUGGESTED KEYPHRASES

Index	Avg@1	Avg@3	Avg@5	Avg@10
KEA_Sim$_{Li}$	3,34	3,21	3,04	2,80
Author_Sim$_{Li}$	3,69	3,42	3,08	2,81

A. Keyphrase Suggestion Success

The performance of the semantically similar keyphrase suggestion of the system is discussed by calculating the average score of the evaluator scores for the first 10 suggested keyphrases. Table 1 gives the average scores for the first k keyphrase suggestions where k∈{1,3,5,10}. According to the results in Table 1, Author_Sim$_{Li}$ achieves better results than KEA_Sim$_{Li}$. This is an expected outcome, since author assigned keyphrases may be more meaningful from the automatically extracted keyphrases. Although, Author_Sim$_{Li}$ index has better suggestion results, KEA_Sim$_{Li}$ index results are still competitive. Considering that in real life applications most of the documents in a collection do not have author assigned keyphrases, we can argue that keyphrase suggestion can be done with the automatically extracted keyphrases. Of course, if author assigned keyphrases are available for a collection, they can be used for better performance. The average scores for the first 3 suggested keyphrases indicate that a big percentage of these 3 suggested keyphrases has a relevance score above 3. This means that the first three suggested keyphrases are relevant with the query term.

B. Document Retrieval Success

In order to measure document retrieval success, SemKPSearch configured with KEA_Sim$_{Li}$ index was compared to Google Desktop on the same document collection. The document retrieval performances of the two systems were compared with the relevance scores for the retrieved documents given by the evaluators. Each evaluator randomly selected query terms from a set of keyphrases appearing in the SemKPSearch index and a set of query terms not appearing in the index. During scoring SemKPSearch, if the result set contained less than 10 documents, the evaluators expanded the result set by using the suggested keyphrases until reaching 10 documents. If the query text was not indexed in SemKPIndex, then semantically related keyphrases are calculated on the fly by comparing the query text to all keyphrases. Since our evaluation results indicate that the first three suggested keyphrases are very relevant with a given query term, the evaluators first used the documents retrieved for three suggested keyphrases for expansion in the suggestion order. If they did not reach ten documents, they used a single document from other suggested keyphrases.

Table 2 presents the average relevance scores, mean reciprocal rank (MRR) values and precision values for both systems. Table 2.a shows the evaluation results for the documents returned for keyphrase queries which were indexed by the evaluated SemKPIndex. In other words there was at least one document such that the queried term is its keyphrase. Table 2.b shows the evaluation results for queries that do not occur as keyphrases. The average relevance scores are the averages of the evaluator scores for documents. The reciprocal rank of a query result list is equal to $1/rank_{fc}$ where $rank_{fc}$ is the position of the first correct answer in the result list, and we treat the retrieved documents with scores 4 and 3 (completely relevant and relevant) as correct answers. The MRR value of a query set is the average of the reciprocal ranks of the queries in the set. The precision value is the percentage of correct answers in the retrieved document set.

TABLE II. EVALUATION RESULTS TO COMPARE DOCUMENT RETRIEVAL PERFORMANCE OF SEMKPSEARCH AND GOOGLE DESKTOP

a) Searching with keyphrases indexed in SemKPIndex

	SemKPSearch			Google Desktop		
first n docs.	Avg. Score	MRR	Pre.	Avg. Score	MRR	Pre.
1	3,95	1,00	1,00	3,05	0,70	0,70
3	3,57	1,00	0,83	2,94	0,83	0,67
5	3,32	1,00	0,78	2,74	0,83	0,56
7	3,04	1,00	0,70	2,49	0,83	0,49
10	2,74	1,00	0,62	2,15	0,83	0,40

b) Searching with phrases not indexed in SemKPIndex

	SemKPSearch			Google Desktop		
first n docs.	Avg. Score	MRR	Pre.	Avg. Score	MRR	Pre.
1	2,04	0,43	0,43	2,14	0,29	0,29
3	1,93	0,50	0,33	1,81	0,29	0,25
5	2,01	0,54	0,34	1,85	0,29	0,21
7	1,71	0,54	0,25	1,90	0,31	0,25
10	1,71	0,54	0,21	1,73	0,31	0,22

According to Table 2.a, the documents retrieved with SemKPSearch get higher average scores than the documents returned by Google Desktop. Since this table is for the evaluation of the results with the keyphrases indexed in SemKPIndex, one can argue that this is the success of the keyphrase extraction algorithm. The results in the first orders get apparently high scores because they are the directly returned documents having the search term as one of their keyphrases. With a further analysis of the raw results we see that for all queried keyphrases, the number of directly returned documents is 2,4 out of 10 on the average, and 76% of the evaluated documents are returned by assisting the query with semantically related keyphrases. The average score for the documents that are retrieved by the suggested keyphrases is 2,47. On the other hand, the average score for the last 8 documents out of 10 retrieved by Google Desktop is 1,9. MRR and Precision values on Table 2.a are similar to the average scores, and SemKPSearch beats Google Desktop. Here we see that the MRR value for SemKPSearch is 1, which means that for all queries, SemKPSearch returned a relevant document to the query term at the first place. Actually this result comes from the success of the keyphrase extraction algorithm KEA because the first document has always the query term as its keyphrase extracted by KEA. These values reasonably show us that using keyphrases of documents, the document retrieval with SemKPSearch is more successful than Google Desktop.

In Table 2.b, a slightly different result is seen for the documents returned for the phrases not indexed in SemKPIndex. The average scores are a bit lower for the SemKPSearch results. However MRR and precision values show that for the queries with phrases that are not indexed as a keyphrase of a document, related documents appear on the higher orders in SemKPSearch.

Although Keyphind system [6] is not tested with our data set, we can still compare it with the results of our system. Keyphind returns the documents if the searched keyphrase is available in its index. But it does not return any documents if the searched keyphrase is not available in its index. For this reason, Keyphind system would not have returned any documents for the searched keyphrases in Table 2.b since those keyphrases would not have been in Keyphind index. On the other hand, our SemKPSearch system returns the documents using the semantically related keyphrases. If there are enough documents associated with the searched keyphrase in a digital library, the performance of SemKPSearch configured with KEA_Sim$_{Li}$ index will be similar to the performance of Keyphind since both use KEA to extract keyphrases. When there are not enough documents associated with the searched keyphrase, Keyphind will return only associated documents while SemKPSearch returns additional documents using semantically related keyphrases in addition to the documents associated with the searched keyphrase.

In Table 2.a, the average number of returned documents that are directly associated with searched keyphrase is 2,4 out of 10 documents, the rest of the returned documents are associated with semantically related keyphrases. The average score of the documents associated with searched keyphrase is 3,78 and the average score of the documents associated with semantically related keyphrases is 2,47. With a further analysis, the average score of the first results associated with semantically related keyphrases is 3,47, and the average score for the first three results associated with semantically related keyphrases is 3,01. These results indicate that the first results associated with semantically related documents are actually related with the searched keyphrase. These results also indicate that Keyphind system would have returned only 2,4 documents on the average for the keyphrases in Table 2.a and its average score will be similar to our average score (3,78). But SemKPSearch returns 3 more related documents associated with semantically related documents with average score 3,01.

IV. CONCLUSION

In this paper, we propose SemKPSearch system which has a user friendly search and browsing interface for querying documents by their keyphrases in a digital library. SemKPSearch indexes the documents with their keyphrases in SemKPIndex. Through the user interface of SemKPSearch, the user can search documents with topic like query phrases. SemKPSearch returns keyphrases that are semantically related to the query text, as well as the documents having keyphrases containing the query text. The user can continue to browse more documents with the suggested semantically related keyphrases or with the keyphrases of the retrieved documents. In this way, it is expected that the user can reach the related documents with the query text even if the documents do not contain the query term.

To calculate the semantic similarity between keyphrases, we propose to use a text-to-text semantic similarity metric that is proposed by Corley and Mihalcea [3]. This metric employs a word-to-word semantic similarity measure, and we used Li et al. word-to-word similarity measure [12]. Thus, the semantic similarity of the keyphrases is formulated as a function of the similarity of the words of the keyphrases.

The evaluation of the system was done by the human evaluators. The evaluators judged the quality of the results and the effectiveness of the suggested semantically related keyphrases. In order to evaluate the document retrieval performance, SemKPSearch system was compared to Google Desktop which is a full-text index based search engine. The evaluation results showed that the evaluators found the documents retrieved with SemKPSearch more related to the query term than the documents retrieved with Google Desktop. Besides the document retrieval, the semantically related keyphrase suggestions were also evaluated by the evaluators. According to the results obtained for the related keyphrase suggestions, it is feasible to use the automatically extracted keyphrases and to relate them with the keyphrase semantic similarity that we proposed.

REFERENCES

[1] Google Desktop - Features. http://desktop.google.com/features.html.

[2] WordNet - About WordNet. http://wordnet.princeton.edu.

[3] C. Corley and R. Mihalcea. Measuring the semantic similarity of texts. In Proceedings of the ACL Workshop on Empirical Modeling of Semantic Equivalence and Entailment, pages 13–18. Association for Computational Linguistics, 2005.

[4] W.B. Croft, H.R. Turtle, and D.D. Lewis. The use of phrases and structured queries in information retrieval. In Proceedings of the 14th annual international ACM SIGIR conference on Research and development in information retrieval, pages 32–45. ACM, 1991.

[5] J.L. Fagan. The effectiveness of a nonsyntactic approach to automatic phrase indexing for document retrieval. Journal of the American Society for Information Science, 40(2):115–132, 1989.

[6] C. Gutwin, G. Paynter, I.Witten, C. Nevill-Manning, and E. Frank. Improving browsing in digital libraries with keyphrase indexes. Decision Support Systems, 27(1-2):81–104, 1999.

[7] S. Jones. Design and evaluation of phrasier, an interactive system for linking documents using keyphrases. In Proceedings of Human-Computer Interaction: INTERACT'99, pages 483–490, 1999.

[8] S. Jones and G. Paynter. Topic-based browsing within a digital library using keyphrases. In Proceedings of the fourth ACM conference on Digital libraries, page 121. ACM, 1999.

[9] S. Jones and M.S. Staveley. Phrasier: a system for interactive document retrieval using keyphrases. In Proceedings of the 22nd

annual international ACM SIGIR conference on Research and development in information retrieval, pages 160–167. ACM, 1999.

[10] M. Krapivin, A. Autaeu, and M. Marchese. Large Dataset for Keyphrases Extraction. Technical Report DISI-09-055, DISI, University of Trento, Italy, 2009.

[11] Q. Li, YB Wu, R.S. Bot, and X. Chen. Incorporating document keyphrases in search results. In Proceedings of the Americas Conference on Information Systems (AMCIS), New York, 2004.

[12] Y. Li, Z.A. Bandar, and D. McLean. An approach for measuring semantic similarity between words using multiple information sources. IEEE Transactions on knowledge and data engineering, pages 871–882, 2003.

[13] Y. Li, D. McLean, Z.A. Bandar, J.D. O'Shea, and K. Crockett. Sentence similarity based on semantic nets and corpus statistics. IEEE Transactions on Knowledge and Data Engineering, pages 1138–1150, 2006.

[14] R. Mihalcea, C. Corley, and C. Strapparava. Corpus-based and knowledge-based measures of text semantic similarity. In Proceedings of the 21st national conference on Artificial intelligence-Volume 1, pages 775–780. AAAI Press, 2006.

[15] N. Wacholder, D.K. Evans, and J.L. Klavans. Automatic identification and organization of index terms for interactive browsing. In Proceedings of the 1st ACM/IEEE-CS joint conference on Digital libraries, page 134. ACM, 2001.

[16] I.H. Witten, G.W. Paynter, E. Frank, C. Gutwin, and C.G. Nevill-Manning. KEA: Practical automatic keyphrase extraction. In Proceedings of the fourth ACM conference on Digital libraries, page 255. ACM, 1999.

[17] Z. Wu and M. Palmer. Verbs semantics and lexical selection. In Proceedings of the 32nd annual meeting on Association for Computational Linguistics, pages 133–138. Association for Computational Linguistics, 1994.