Creating a Testbed for the Evaluation of Automatically Generated Back-of-the-book Indexes

Andras Csomai and Rada Mihalcea

University of North Texas
Computer Science Department
csomaia@unt.edu, rada@cs.unt.edu

Abstract. The automatic generation of back-of-the book indexes seems to be out of sight of the Information Retrieval and Natural Language Processing communities, although the increasingly large number of books available in electronic format, as well as recent advances in keyphrase extraction, should motivate an increased interest in this topic. In this paper, we describe the background relevant to the process of creating back-of-the-book indexes, namely (1) a short overview of the origin and structure of back-of-the-book indexes, and (2) the correspondence that can be established between techniques for automatic index construction and keyphrase extraction. Since the development of any automatic system requires in the first place an evaluation testbed, we describe our work in building a gold standard collection of books and indexes, and we present several metrics that can be used for the evaluation of automatically generated indexes against the gold standard. Finally, we investigate the properties of the gold standard index, such as index size, length of index entries, and upper bounds on coverage as indicated by the presence of index entries in the document.

1 Introduction

> *"Knowledge is of two kinds. We know a subject ourselves, or we know where we can find information on it." (Samuel Johnson)*

The automatic construction of back-of-the-book indexes is one of the few tasks related to publishing that still requires extensive human labor. While there is a certain degree of computer assistance, mainly consisting of tools that help the professional indexer organize and edit the index, there are however no methods or tools that would allow for a complete or nearly-complete automation. Despite the lack of automation in this task, there is however another closely related natural language processing task – kepyphrase extraction – where in recent years we have witnessed considerable improvements.

In this paper, we argue that the task of automatic index construction should be reconsidered in the light of the progress made in the task of keyphrase extraction. We show how, following methodologies used for the evaluation of keyphrase extraction systems, we can devise an evaluation methodology for back-of-the-book indexes, including a gold standard dataset and a set of evaluation metrics.

We hope that this evaluation testbed will boost the research in the field of automatic index construction, similar to the progress made in other NLP areas following the deployment of an evaluation framework[1].

Specifically, in this paper: (1) We shortly overview the origin and typical structure of back-of-the-book indexes; (2) We show that a close correspondence can be established between techniques for automatic index construction and keyphrase extraction, and consequently we briefly review the state-of-the-art for the latter problem; and finally (3) We describe our work in creating a testbed for the evaluation of automatic indexing systems, including a dataset of books and indexes, and a proposed set of evaluation metrics. We also discuss the properties of the gold standard, such as index size, length of index entries, and upper bounds on coverage as indicated by the presence of index entries in the document.

2 Definition, Origins, and Structure

The history of indexing dates back to ancient times. Several authors link the name *index* to the little papyrus slips (also called index) attached to papyrus scrolls in Roman libraries, containing the name of the author and the title of the document, and often also a small extract, which would allow the identification of the scroll without opening it. Several examples of index uses can be found throughout the following ages, but the real boost came in the nineteenth century, when the basic structure of the back-of-the-book indexes was defined.

While there are several definitions of what a back-of-the-book index is, the most complete and recent definition is perhaps the one provided in [7]. According to this definition, the index should enumerate all the words or phrases that refer to information that will most probably be sought by a reader. Specifically:

1. an index is a guide to names, places, items, concepts in a document or collection of documents;
2. the items or concepts in the index are arranged systematically, generally in alphabetical order; and
3. there are references to where each of these items are located in the document or documents.

The style of a back-of-the-book index has undergone several changes during its long history, arriving to the current more or less standard appearance, where each index entry contains the following two components [5]: (1) a *heading* including the indexing word or phrase, and (2) one or more *page reference numbers* and/or *cross references*. The page reference number shows the page or pages where the information relevant to the entry is located, while the cross-reference points to related entries, generally consisting of a synonym (marked by *"see ..."*), or other topically related terms (marked by *"see also..."*). When there are several entries referring to the same concept, they are usually ordered hierarchically under the heading that describes the shared concept.

[1] See for instance the progress in machine translation following the release of the Bleu evaluation metric, or the large number of publications on the problem of semantic entailment following the availability of the Pascal entailment dataset.

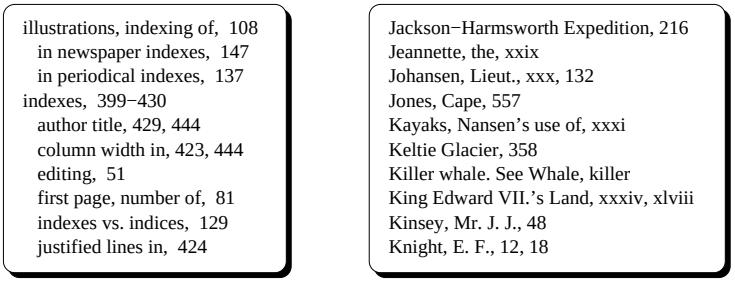

Fig. 1. Examples of entries in a back-of-the-book index.

Figure 1 shows two sample snapshots from a back-of-the-book index, illustrating the structure of index entries (headings, references, cross-references), the various types of index entries (names of persons, locations, terminology, important concepts in the text), and the hierarchical organization.

Index entries are often composed of more than one word, which results in *compound headings*. Typically, for such compound headings, indexing guidelines indicate that the head word has to be listed first, mainly for the purpose of an alphabetical ordering, which leads to the so-called *inversion*. As an example, consider the *indexing of illustrations* entry shown in Figure 1, which was changed to *illustrations, indexing of* through the process of inversion. The inversion can sometimes lead to hard-to-read headings like *children, disabled, hospitals, for* for the phrase *hospitals for disabled children*, and consequently recent work on manual indexing has discouraged the overuse of inversion.

Another important aspect of the index is the length. The literature usually defines the length of an index as a ratio between the number of pages of the index and the number of pages of the text. The length is typically affected by several factors, including the topic and specificity of the text. Less domain-specific texts such as children books or elementary school textbooks require indexes with a length accounting for about 1–3% of the length of the book, while highly specialized monographs on scientific topics may require indexes with a length of up to 15% of the text. History, biography and undergraduate textbook indexes are usually within the 5–8% range.

Finally, the content of the index, just like the length, also depends on the topic of the text. For instance, biographies tend to contain a larger number of names of persons and locations, while scientific books contain more entries referring to technical concepts and terminology.

3 Back-of-the-book Indexing and Keyphrase Extraction

As mentioned before, an index is typically composed of names of persons or locations, terminology, and important concepts. Some of these index entries can be easily identified with a name entity recognizer as for instance the one described

in [6], which automatically labels all entities that represent persons, locations, or organizations. The most difficult part of the index is however represented by the so-called *important concepts*, which consist of words or phrases that are neither person names, nor locations, and yet represent important elements in the text. This is the part that is typically handled by the keyphrase extraction methods which target the automatic identification of important concepts in a text.

The task of automatic generation of back-of-the-book indexes can be therefore defined as a compound task consisting of (1) identifying named entities and (2) extracting keyphrases, followed by a post-processing stage that combines the output of the two tasks in a way that follows the traditional indexing guidelines. Consequently, the indexing task can be accomplished by using a named-entity recognizer coupled with a keyphrase extraction system. Since in recent years the named-entity recognition task has achieved relatively high levels of performance[2], for the remainder of this section we concentrate on the state-of-the-art in keyphrase extraction, as this represents the most difficult aspect of index construction. Note that we focus on keyphrase *extraction*, as opposed to keyphrase *generation*, since the former is a more feasible goal for current automatic systems.

The main approaches to keyphrase extraction can be divided into supervised methods that require the availability of a (sometimes large) training corpus, and unsupervised approaches that require only unlabeled data and eventually a very small set of annotated seeds.

Supervised keyword extraction. All the supervised keyword extraction methods that were developed so far appear to share a common framework: they start with a preprocessing step that handles the extraction and filtering of candidate phrases, followed by the actual ranking of the keywords using a set of contextual features and a standard machine learning algorithm.

In some cases the preprocessing stage also performs several transformations on the input data set, such as stemming or lemmatisation, changing the capital letters into lower case, etc. Next, candidate phrases are extracted, typically using one of the following three methods:

1. *n-grams*: all n-grams extracted from the document, usually covering unigrams, bigrams, and trigrams [1], since they account for approximately 90% of the keyphrases.
2. *np-chunks*: a syntactic parser is employed to find np chunks in the document; this usually leads to increased precision at the cost of lower recall.
3. *syntactic patterns*: a part-of-speech tagger is used to label all the words in the document, and candidate phrases are extracted according to a predefined set of part-of-speech patterns.

Perhaps the most important step in supervised keyword extraction is the ranking of candidate phrases, usually performed using a machine learning algorithm, which can range from Naive Bayes [1, 2, 10], to rule induction [3] and

[2] See for instance the state-of-the-art systems from the recent CoNLL 2002 and CoNLL 2003 shared tasks.

genetic algorithms [9]. In terms of features, several have been proposed so far, including:

1. *tf.idf*: A weighting factor based on term frequency inverse document frequency feature, as defined in information retrieval.
2. *tf* and *idf*: Sometimes term frequency and inverse document frequency are not combined, thus allowing the learning algorithm to eventually improve on the *tf.idf* combination of the two features.
3. *distance*: The distance of the phrase from the beginning of the document, usually measured by the number of individual words preceding the candidate phrase.
4. *POS pattern*: The part-of-speech pattern of the candidate phrase
5. *length*: The length of the candidate phrase. The distribution of the length of human expert assigned keywords, as reported by [3], shows that 13.7% of the human assigned keyphrases contain a single term, 51.8% contain two terms, and 25.4% contain three terms.
6. *stemmed forms*: The frequency of the stemmed word forms
7. *syntactic elements*: Binary features showing the presence of an adjective at the end of the phrase, or the presence of a common verb anywhere in the phrase [9]
8. *domain specific features*: Using a domain-specific hierarchical thesaurus and features indicating the presence of semantically related terms, an almost spectacular jump in recall was reported in [4], from 64% to 94%.
9. *coherence feature*: A new feature based on the hypothesis that candidates that are semantically related to one another tend to be better keyphrases is introduced in [10]. The semantic relatedness of the candidate terms is estimated by a measure of mutual information (pointwise mutual information), with the help of a search engine.

In terms of performance, supervised keyword extraction systems usually exceed by a large margin the simple frequency-based baselines. The best system was reported in [3] with an F-measure of 41.4%, comparing the automatically generated keyphrase set against human expert assigned keywords on a corpus containing scientific article abstracts. [2] reports an F-measure of 23%, also calculated based on author assigned keywords, but on a collection of full length computer science technical reports, which is a more difficult task than extracting keywords from abstracts. Finally, [9] reports a precision of around 25% over a large and varied collection. They also performed a manual evaluation of acceptability, and reported an 80% acceptability rate.

Unsupervised methods Unsupervised methods generally rely on variations of *tf.idf* or other similar measures, in order to score the candidate phrases. The method proposed in [11] extracts a set of candidate terms (only nouns), and ranks them according to their *relative frequency ratio*, which is in fact similar to *tf.idf*. First, only the terms with scores higher than a given threshold are kept, and all these terms are associated with their WordNet synsets. A pairwise semantic similarity score is calculated between all the terms, and a single link

clustering is performed on the candidate set, using the similarity scores. Next, a cluster score and concept score are calculated, reflecting the "coherence" of the cluster (the sum of all pairwise similarities) and the overall "importance" of the cluster. The ranking of the candidate phrases is then performed with respect to the cluster and concept scores. The results of the method show clear improvement with respect to a baseline method that performs only *tf.idf* score ranking.

Another method is presented in [8], where keyword extraction is performed using language models. The method is intended to extract keyphrases not from a single document, but from a collection of documents. They make use of two document collections, called "background" and "foreground", with the later being the target set. They build n-gram language models for both the foreground and background corpora, with the goal of measuring the *informativeness* and *phraseness* of the phrases of the foreground corpus. The phraseness is defined as the Kullback-Liebler divergence of the foreground n-gram language model (see article), which represents the "information loss" by assuming the independence of the component terms. The "informativeness" is calculated by applying the same statistical measure to the foreground and background models. Once the informativeness and phraseness of the candidate phrases is defined, they can be combined into an *unified score* that can be used to order the candidate phrases.

4 Building an Evaluation TestBed for Back-of-the-Book Indexing

The construction of a gold standard benchmark that can be used for the evaluation of automatically generated back-of-the-book indexes requires a collection of books in electronic format, each of them with their corresponding index. We had therefore to: (1) identify a collection of books in electronic format, and (2) devise a method to extract their index entries in a format that can be used for automatic evaluations.

4.1 Collecting Books in Electronic Format

Currently one of the largest available on-line collection of electronic books is the Gutenberg project[3], built as a result of volunteer contributors, and containing the electronic version of books that are in the public domain.

Project Gutenberg contains approximately 16,000 titles, however only very few of them include a back-of-the-book index, either because they never had one, or because the person who contributed the book decided not to include it. In order to find the books that contained their back-of-the-book index we used a search engine to identify those books in the Gutenburg collection that contained keywords such as *index of content*. Using an external search engine ensured a certain degree of topical randomness as well.

A problem that we noticed with the results obtained in this way was that many documents covered topics in the humanities, while very few books were

[3] http://www.gutenberg.org

Category	# books
Humanities	
History & Art	7
Literature & Linguistics	7
Psychology & Philosophy	7
Science	
Agriculture	1
Botany	4
Geography	2
Geology	2
Natural history	9
Zoology	6
Technology	
Electrical and nuclear	2
Manufacturing	1
Ocean engineering	1
Misc	7
TOTAL	56

Table 1. Distribution of books across topics

from the scientific/technical domain. To ensure the presence of technical documents, we used the Gutenberg Project search engine to identify all the documents classified as *science* or *technology* according to the Library Of Congress Classification (LOCC) system, and manually extracted only those books that contained an index. As a result, we retrieved a total of 56 documents, out of which 26 have an LOCC classification. Table 1 shows the distribution of the books across different topics.

4.2 Extracting Index Entries

Once we obtain a collection of books in electronic format, the next step is to extract the index in a format that can be used for the evaluation of automatically constructed back-of-the-book indexes.

First, we separate the index from the main body of the document. Next, since our long term goal is to devise methods for automatic discovery of index entries, and not referencing, all page numbers and cross references are removed, as well as special marks used by the transcriber, such as e.g. the symbol "_" used to emphasize a text as in _*Institution name*_.

Once we have a candidate list of index entries, the next step is to clean them up and convert them into a format suitable for automatic evaluation. The first problem that we faced in this process was the *inversion* applied to compound headings. As mentioned before, indexing guidelines suggest that the head word of an index phrase has to be listed first, to facilitate the search by readers within the alphabetically ordered index. However, in order to measure the performance of an automatic system for index generation, the index entries have to be reconstructed in the form they are most likely to appear in the document,

1	Acetate, of Ammonium Solution, Uses of	uses of Acetate of Ammonium Solution
2	Goldfinch, American	American goldfinch
3	Goose, Domestic	domestic goose
4	Cainozoic, term defined	cainozoic term defined
5	France, history of the use of subsidies in, the navigation laws of, commercial treaty between England and, the Merchant Marine Act of,	history of the subsidies in France the navigation laws of France commercial treaty between England and France the Merchant Marine Act of France

Table 2. Examples of index entries and their reconstructions

if they appear at all. Starting with an index entry whose structure follows the standard indexing guidelines, we therefore try to create an English phrase that is likely to be found in the document. This reconstruction is sometimes very difficult, since the human indexers do not strive to create grammatically correct phrases. In some cases, even if we manage to correctly reorder the index entry (e.g. list of modifiers followed by head word), the resulting phrase may not be always proper English, and therefore it is very likely that it will not be identified by any indexing algorithm.

The reconstruction algorithm is based on the presence of prepositions. As shown in table 2, the sequences of the original scrambled index entry are delimited by commas into smaller units. We devised several heuristics that can be used to recover the original order of these components. In the case of prepositional phrase units, the preposition is a strong clue about the placement of the phrase relative to the head-phrase, e.g. the preposition *to, in, for* at the beginning of the phrase suggests that it should follow the head-phrase, whereas the preposition *as, from, of, among* at the end of the phrase suggests that it should precede the head; see for instance example 1 in table 2. Similarly, the position of the conjunction *and* determines the placement of the phrase that contains it.

When there are no prepositions or conjunctions, the reconstruction becomes more complicated. We were able however to identify several patterns that occur fairly often, and use these patterns in the reconstruction process: (1) If the second component is a modifier of the head-phrase (adjective or adverb, or corresponding phrase) then it should be placed before the head; see for instance examples 2 and 3 in table 2. (2) If the second phrase contains an explanation referring to the head, or some additional information, then it should be placed after the phrase head. Note that the structures corresponding to the second pattern can sometime lead to ungrammatical phrases, as for example the phrase 4 in table 2. In such cases, the phrase will be post-processed using the filtering step described below. Reconstructing the index entries based on the mentioned patterns is only a back-off solution for the cases where no prepositions were found in the entry. We attempt to determine which is the most frequent pattern at the document level (based on the number of the index entries reconstructed using the selected pattern and found in the document), and use it throughout the index. This pattern selection is individually carried out for every document.

The hierarchic structures (see example 5 in table 2) are reconstructed by attaching the head-phrase of the entry from the higher level to all its descendants. This results in a set of compound entries, usually inverted, which can be reconstructed using the heuristics described before.

4.3 Index Granularities

Following the example of other NLP tasks that allow for different levels of granularity in their evaluation[4], we decided to build gold standard indexes of different granularities. This decision was also supported by the fact that compound index terms are sometimes hard to reconstruct, and thus different reconstruction strategies pose different levels of difficulty.

We decided to extract two different indexes for every text: (1) a short index, consisting only of head phrases, which allows us to evaluate a system's ability to extract a coarse-grained set of index entries; and (2) a long index, containing the full reconstructed index entries, which corresponds to a more fine-grained indexing strategy.

As pointed out earlier, the reconstruction of the inverted compound entries is fairly difficult, therefore the fine grained index will sometimes contain ungrammatical phrases that could never be found by any extraction algorithm. Consequently, we decided to also create a third, filtered index, that excludes these ungrammatical phrases and allows us to measure a system performance with a higher upper bound, meaning that a larger number of index entries are present in the text and could be potentially found by an automatic indexing system. We use a simple filtering method that measures the frequency of each index entry on the Web, as measured using the AltaVista search engine. If the number of occurrences is higher than a given threshold n, we consider the phrase plausible. If the frequency is below the threshold, the entry is discarded. Finding a good value for the threshold n may be a difficult issue, since a large value will allow for the inclusion of longer phrases with small occurrence probability, while a small value may let many incorrect phrases slip through. In our experiment we use a value of $n = 2$, which was empirically determined, and resulted in the elimination of roughly 50% of the fine grained entries.

4.4 Properties of the Gold-standard Collection

Starting with the gold standard collection described in the previous section, we measured several properties of back-of-the-book indexes, such as length of the index entries, index size, and upper bounds on coverage as indicated by the presence of index entries in the document.

The length of the index entries can influence the accuracy of the index extraction algorithm and the choice of methods used for candidate phrase extraction. For instance, in the case of coarse-grained indexes, most of the index entries consist of four words or less, and therefore a four-gram model would probably

[4] For instance, word sense disambiguation gold standards allow for the evaluation of systems that can perform either coarse-grained or fine-grained disambiguation

Index type	Length of index entry										
	1	2	3	4	5	6	7	8	9	10	>10
Coarse grained	31312	9068	2232	1268	642	311	162	56	36	13	8
Fine grained	8250	9352	7627	7057	5787	4500	3188	1906	1241	727	862
Filtered	7500	8112	4590	2191	1151	657	437	303	186	138	171

Table 3. Distribution of index entries by length (defined as number of tokens)

Length of index entry	index style		
	coarse grained	fine grained	filtered index
1	92.95%	92.20%	93.22%
2	72.89%	48.80%	52.59%
3	48.68%	23.75%	36.75%
4	28.36%	10.16%	27.06%
5	16.73%	4.15%	16.95%
6	9.93%	1.99%	8.53%
7	8.66%	0.85%	3.45%
8	12.77%	0.79%	3.97%
9	3.33%	0.32%	1.08%
10	8.33%	0.55%	0.74%
Total	81.29%	30.34%	54.78%

Table 4. Presence of index entries in the original text

be sufficient. On the other side, when fine grained entries are used, larger phrases are also possible, and thus longer n-grams should be considered. Table 3 shows the distribution by length of the gold-standard index entries.

Another important aspect of the gold-standard index is whether it includes entries that can be found in the text, which impacts the value of the recall upper bound that can be achieved on the given index. This aspect is of particular interest for methods that create indexes by extracting candidate phrases from the text, rather than generating them. To determine the average value for this upper bound, we determined the number of index entries that appeared in the text, for each of the three index types (fine-grained, coarse-grained, filtered index). The results of this evaluation are shown in table 4. Not surprisingly, the smallest coverage is observed in the case of the fine-grained indexes, followed by the filtered indexes and the coarse-grained indexes. It is also worth noting that the Web-based filtering process increases the quality of the index significantly, from a coverage of 30.34% to 54.78%.

Finally, another important property of the index is its size relative to the length of the document. We measured the ratio of the number of entries in the index and the number of tokens in the text. On average, the coarse-grained indexes contain about 0.44% of the text tokens, which corresponds roughly to one

coarse grained keyphrase for every 227 words of text. The fine-grained indexes have a ratio of 0.7%, which represents one index phrase for every 140 words in the document.

4.5 Evaluation Metrics

Finally, another important aspect that needs to be addressed in an evaluation framework is the choice of metrics to be used. Provided a gold standard collection of back-of-the-book indexes, the evaluation of an automatic indexing system will consist of a comparison of the automatically extracted set of index entries against the correct entries in the gold standard. We propose to use the traditional information retrieval metrics, *precision* and *recall*. Precision measures the accuracy of the set automatically extracted, as indicated by the ratio of the number of correctly identified entries and the total number of proposed entries. Recall is defined as the ratio of the number of correctly identified entries and the total number of correct entries in the gold standard.

$$precision = \frac{extracted\ and\ correct}{extracted}$$

$$recall = \frac{extracted\ and\ correct}{correct}$$

In addition, the F-measure combines the precision and recall metrics into a single formula:

$$F - measure = \frac{2 * precision * recall}{precision + recall}$$

Moreover, we also suggest the use of a "relative recall", which represents the ratio between the traditional recall as defined earlier, and the maximum recall that can be achieved on a given gold standard index using only entries that literally appear in the text. The relative recall is therefore defined as the fraction of correctly identified index entries and the total number of entries from the gold standard that appear in the text.

$$recall_r = \frac{extracted\ and\ correct}{correct\ and\ in\ text}$$

This measure targets the evaluation of systems that aim to extract indexes from the books, rather than generating them. Correspondingly, we can also define an F-measure that takes into account the precision and the relative recall.

5 Conclusion and Future Work

In this paper, we described our work in creating an evaluation testbed for automatic back-of-the-book indexing systems. We also overviewed the background of back-of-the-book indexing and current trends in keyphrase extraction that are relevant to this problem. The long term goal of this work is to devise an

automatic method for building back-of-the-book indexes. Since no evaluation framework is currently available for this task, we had to start our work in this project by creating a testbed that will allow for the comparative evaluation of a variety of indexing methods.

We plan to extend our collection by splitting the index entries into named entities and important concepts, which will allow for a separate evaluation of the named entity recognition and the keyphrase extraction components. We also plan to include a larger number of contemporary books, in order to eliminate the discrepancies arising from the stylistic variety due to the age of the books in our collection.

The data set described in this paper is publicly available for download from `http://www.textrank.org/data`.

Acknowledgments

This work was partially supported by an award from Google Inc.

References

1. Frank, E., Paynter, G. W., Witten, I. H., Gutwin, C., and Nevill-Manning, C. G. Domain-specific keyphrase extraction. In *Proceedings of the International Joint Conference on Artificial Intelligence* (1999).
2. Gutwin, C., Paynter, G., Witten, I., NevillManning, C., and Frank, E. Improving browsing in digital libraries with keyphrase indexes, 1998.
3. Hulth, A. *Combining Machine Learning and Natural Language Processing for Automatic Keyword Extraction*. PhD thesis, Stockholm University, 2004.
4. Hulth, A., Karlgren, J., Jonsson, A., and Asker, H. B. L. Automatic keyword extraction using domain knowledge. In *Proceedings of the Conference on Computational Linguistics and Intelligent Text Processing* (2001).
5. Knight, N. *Indexing, the art of*. Allen & Unwin, 1979.
6. Mikheev, A., Moens, M., and Grover, C. Named entity recognition without Gazetteers. In *Proceedings of the European Association for Computational Linguistics* (Bergen, Norway, 1999).
7. Prasher, R. *Index and Indexing Systems*. Medallion Press, 1989.
8. Tomokiyo, T., and Hurst, M. A language model approach to keyphrase extraction. In *Proceedings of ACL Workshop on Multiword Expressions* (2003).
9. Turney, P. Learning algorithms for keyphrase extraction. *Information Retrieval 2*, 4 (2000).
10. Turney, P. Coherent keyphrase extraction via web mining. In *Proceedings of the International Joint Conference on Artificial Intelligence* (Acapulco, Mexico, 2002).
11. van der Plas, L., Pallotta, V., Rajman, M., and Ghorbel, H. Automatic keyword extraction from spoken text. a comparison of two lexical resources: the EDR and WordNet. In *Proceedings of the Language Resources and Evaluations Conference* (Lisbon, Portugal, 2004).