A Citation-based approach to automatic topical indexing of scientific literature

Abdulhussain E. Mahdi[1] **and Arash Joorabchi**

Department of Electronic and Computer Engineering, University of Limerick, Limerick, Republic of Ireland

Abstract

Topical indexing of documents with keyphrases is a common method used for revealing the subject of scientific and research documents to both human readers and information retrieval tools, such as search engines. However, scientific documents that are manually indexed with keyphrases are still in the minority. This article describes a new unsupervised method for automatic keyphrase extraction from scientific documents which yields a performance on a par with human indexers. The method is based on identifying references cited in the document to be indexed and, using the keyphrases assigned to those references for generating a set of high-likelihood keyphrases for the document. We have evaluated the performance of the proposed method by using it to automatically index a third-party testset of research documents. Reported experimental results show that the performance of our method, measured in terms of consistency with human indexers, is competitive with that achieved by state-of-the-art supervised methods.

Keywords: keyphrase extraction; topical indexing; data mining; citation based indexing

1. Introduction

Scientific literature comprises scientific publications in form of journal articles, conference papers, technical reports, theses and dissertations, book chapters, and other materials about the theory, practice, and results of scientific inquiry. These materials are produced by individuals or groups in academic and research-centric organizations such as universities, national research labs, and research and development companies. A great majority of such publications are published in electronic form on the Internet and are available through institutional repositories, digital libraries, publishers' websites, authors' webpages, etc. The sheer volume of scientific literature published on the internet makes finding relevant materials a challenging task, for example, reportedly the number of new papers published in the field of biomedical science exceeds 1800 a day [1]. Topical indexing of documents with keyphrases is a common method used for revealing the subject of scientific and research documents to both human readers and information retrieval tools such as search engines. However,

[1] *Correspondence to:* Abdulhussain E. Mahdi, Department of Electronic and Computer Engineering, University of Limerick, Limerick, Republic of Ireland. Email: Hussain.Mahdi@ul.ie

JIS-1205-v3 Received: 25th June 2010 Revised: 15th September 2010

despite the fact that the authors of such materials, especially those published in scientific journals and conference proceedings, are encouraged and often required by the publishers to provide a list of keyphrases for their papers, the documents that are manually indexed by either the author or professional indexers are still in the minority. To tackle this problem, researchers working in the field of Information Retrieval (IR) and Machine Learning (ML) have developed a wide range of automated keyphrase indexing methods (e.g., see [2-6]). In general, these methods can be divided into two main categories:

1. Extraction indexing: where the keyphrases are extracted from the content of the document itself based on their statistical properties, such as frequency of occurrence. The main weakness of this approach is that it does not take into account the semantic relations between the phrases and their contextual meanings. This weakness leads to a host of defects such as inability to cope with homographs, synonyms, inconsistently worded concepts, burstiness phenomenon [7], etc.
2. Assignment indexing: where the keyphrases assigned to the document come from a controlled vocabulary such as the Medical Subject Heading (MeSH) and are not confined to the terms that appear in the document (e.g., see [8]). In this approach, indexing is treated as a multi-label text classification problem. ML algorithms such as, Support Vector Machines (SVMs), Naïve Bayes (NB), and Decision Trees (DTs) are used to learn a model for each term in the controlled vocabulary from a set of training documents which are manually indexed. These models are then compared against the new documents to assign a set of the most likely keyphrases to each document. In general, this supervised learning approach to keyphrase indexing yields a better performance compared to the extraction indexing approach. It is more resilient to the negative effects of homographs and synonyms, and can cope with cases where a concept is discussed but not identified in the text by an indexable keyphrase. However, the drawback of this approach is its dependence on a large volume of training data. Also, the learnt models would require periodical updating to include new concepts and cope with the concept drift phenomenon [9].

In this article, we introduce a new unsupervised keyphrase indexing method for scientific literature which falls into the extraction indexing category and leverages the links among the scientific publications, represented in form of citations, to achieve a high indexing performance competitive to those achieved by the supervised assignment indexing methods.

The rest of the article is organised as follows: Section 2 provides a rationale for the proposed method in view of recently reported research work in the field. Section 3 describes the proposed method in details and discusses its implementation aspects. Section 4 describes the evaluation process and presents its results. This is followed by Section 5 which analyses presented results and highlights some of the main factors affecting the performance of our method. Section 6 provides a conclusion along with a summary account of planned future work.

2. Rationale and related work

A significant portion of electronic documents published on the Internet become part of a large chain of networks via some form of linkage that they have to other documents. For example, it is a common practice for scientific papers to cite other papers, documented law cases to refer to other cases, patents to cite other patents, and webpages to have links to other webpages. Networked characteristic of such documents have been successfully used to improve the performance of ML-based subject indexing methods, using an approach known as collective classification [10].

In relation to scientific literature which is the subject of our work, the citation networks among scientific documents have been successfully used to improve the search and retrieval methods for scholarly publications. Aljaber et al. [11] show that using citation contexts can provide relevant synonymous and related vocabulary which help increase the effectiveness of the bag-of-words representation used for clustering related scientific texts. Cao and Gao [12] show that incorporating citation links data improves the accuracy of their ML-based system for classifying scientific documents. A series of work done by Bradshaw et al. [13, 14] and Ritchie et al.

JIS-1205-v3 Received: 25th June 2010 Revised: 15th September 2010

[15-17] show that using index terms from cited documents can optimize the full-text indexing and searching of scientific literature.

The above-mentioned studies [11-17] have successfully shown that citation networks among scientific documents can be utilized to improve the performance of three major information retrieval tasks; namely, clustering, classification, and full-text indexing. In our opinion, the results of these studies indirectly suggest that the content of cited documents could also potentially be used to improve the performance of keyphrase indexing of scientific documents. In the current work, we investigate this hypothesis as a new application of citation networks by developing a new citation-based keyphrase extraction method for scientific literature and evaluating its performance. The proposed method can be outlined in three main steps:

3. Reference extraction: this comprises the process of identifying and extracting reference strings (a.k.a citation strings) in the bibliography or reference section of a given document and parsing them into their logical components such as title, author(s), publisher, etc.
4. Data mining: this is a three-fold process. In the first stage, we query the Google Book Search[2] (GBS) [18] to retrieve a list of publications which cite either the given document or one of its references. Then, in the second stage, we retrieve the metadata records of these citing publications from the GBS database. Among other metadata elements, these records contain a list of key terms extracted from the content of the citing publications. In the final stage of the data mining process, we extract these key terms along with their numerically represented degree of importance from the metadata records of the citing publications to be used as primary clues for keyphrase indexing of the given document.
5. Term weighting and selection: this process starts by searching the content of the given document for the set of key terms collected in the data mining process (step 2 above). Each matching term would be assigned a keyphraseness score which is the product function of seven statistical properties of the given term, namely: frequency among the extracted key terms, frequency inside the document, number of words, average degree of importance, first occurrence position inside the document, frequency inside reference strings, and length measured in terms of the number of characters. After computing the keyphraseness scores for all the candidate key terms, a simple selection algorithm is applied to index the document with a set of most probable keyphrases.

3. Citation-based keyphrase extraction

In this Section, we describe each of the three main steps of the proposed Citation-based Keyphrase Extraction (CKE) method in details and discuss their implementation aspects.

3.1. Reference extraction

The wide diversity of citation styles used in scientific literature makes automatic extraction and segmentation of references a non-trivial task. A variety of methods have been proposed to address this problem. These methods can be divided into two main categories: (a) rule-based methods, such as those reported in [19, 20], which rely on a set of rules designed by domain experts to extract and segment the reference strings via pattern recognition techniques such as regular expression-based string matching and template mining; and (b) ML-based methods such as those reported in [21, 22], which deploy learning algorithms such as Hidden Markov Models (HMMs) and Conditional Random Fields (CRFs) to derive the pattern models of reference strings in the training documents and use the learnt models to extract and segment the reference strings in the test documents. In this work, we use an open-source reference extraction software system called ParsCit [23], which is ML-based and uses Conditional

[2] http://books.google.com/

JIS-1205-v3 Received: 25th June 2010 Revised: 15th September 2010

Random Fields as its learning mechanism. ParsCit has been successfully deployed within CiteSeerX[3] project, a well-known digital library of computer science publications, to extract and segment millions of reference strings. ParsCit is also able to extract and segment the header metadata of publications and, therefore, satisfies our algorithm's requirement to know the title of the given document as well as the title of publications cited in it. The reported F1 score of ParsCit for parsing the titles of references in a CiteSeerX dataset is 0.93.

3.2. Data mining

The task of the data mining process is to mine a large-scale database of publications in order to discover and extract the key terms of published materials which either cite the document to be indexed or one of its references. The database that we use for this purpose is that of the Google Book Search (GBS) project. GBS enables the full-text search of books, magazines, and other materials that Google and its library and publisher partners scan, OCR, and index. In October 2009, Google announced that they had over 10 million items searchable through GBS [24]. Google does not provide public access to the full content of the majority of these items due to copyright restrictions. However, the metadata record of each item includes a so called "word cloud" which contains a list of key terms that have been identified as statistically significant within the full textual content of the archived item. Figure 1 shows the Google Word Cloud (GWC) for a book titled: "Data mining: practical machine learning tools and techniques". As can be seen in Figure 1, the significance of each key term in relation to others in the word cloud, is reflected by its font size. GBS deploys the simple TF-IDF weighting scheme combined with some heuristic rules, which detect and emphasize proper nouns, to identify and extract the key terms from the items' content [25]. However, despite the simplicity of the approach taken, a considerable number of extracted key terms are domain-specific, semantically rich, and directly related to the core subject of the items. This, in our opinion, could be attributed to the enormous size of the GBS collection resulting in accurate IDF values.

Figure 1. A sample GWC from GBS database

In the first stage of our data mining process, the GBS database is queried to retrieve a list of publications that cite either the given document or one of its references. This is done by submitting a number of URL queries to the GBS engine in the following format:

http://www.google.com/books/feeds/volumes?**max-results=20**&q=%22**[*title*]**%2C%22

For the first query, the variable *title* in above format is set to the title of the document itself. For the subsequent queries, this variable is set to the titles of each of the references in the document consecutively. The parameter *max-results* is set to 20 in order to limit the number of returned results to a maximum of 20 items. This limitation is set heuristically, to reduce the negative effect of noisy and biased data resulting from one of the following situations: (a) when the reference extraction unit has failed to segment the reference string properly and, therefore, the extracted title is either erroneous or incomplete. For example, consider a case when the actual title is "Data mining: practical machine learning tools and techniques" and the extracted title is "Data mining"; (b) when the title is either too generic (e.g. "computer programming") or heavily cited. in both (a) and (b) cases, the number of returned results could be in the magnitude of thousands with little or no semantic relation to the subject of the document to be indexed.

[3] http://citeseerx.ist.psu.edu

JIS-1205-v3 Received: 25th June 2010 Revised: 15th September 2010

The returned result for each query is an XML file containing metadata records of matching publications. Each record contains metadata elements such as title, author(s), ISBN/ISSN, etc. The ISBN/ISSN of each item is extracted to be used as its key identifier. In the second stage of the data mining process these key identifiers are used to retrieve the HTML pages containing the key terms extracted from their respective items. This is done by submitting URL queries to the GBS engine in one of the following formats depending on the type of the key identifier used for the item, i.e., ISSN or ISBN:

http://books.google.com/books?vid=ISBN[***ISBN***]

http://books.google.com/books?vid=ISSN[***ISSN***]

In the third and final stage of the data mining process a simple regular expression-based HTML parsing method is used to extract the key terms and their corresponding significance value (i.e., an integer number between 1 and 10), from the retrieved HTML pages. Figure 2 illustrates the data mining process described above.

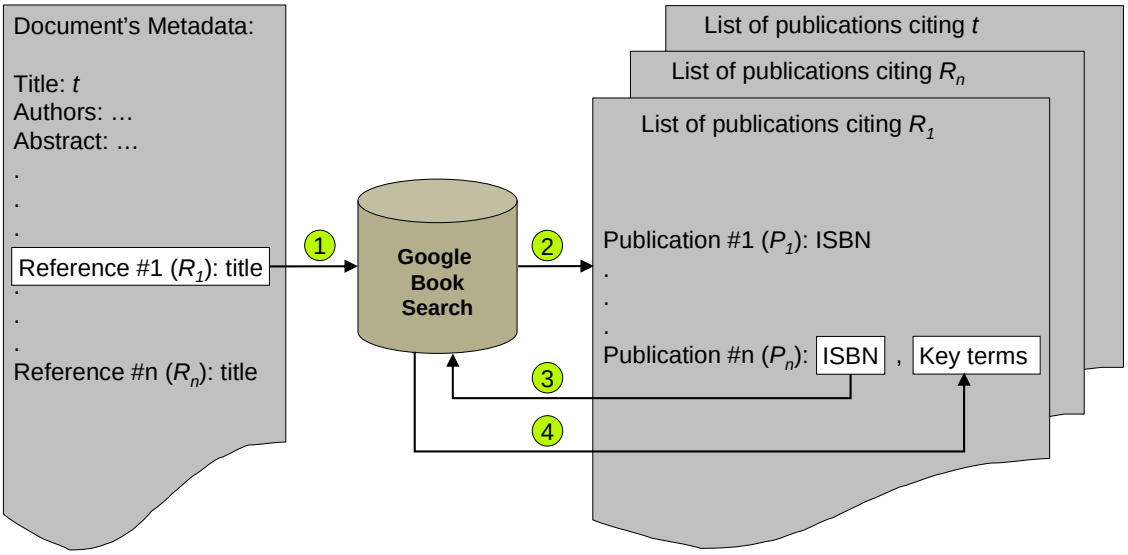

Figure 2. Data mining process

3.3. Term weighting and selection

The data mining process described above results in a pool of key terms extracted from the publications which cite either the document to be indexed or one of its references. At this point, the terms which are stopwords are excluded from the pool. Then the remaining key terms along with the content of the document go through a pre-processing text normalization step comprising punctuation removal, abbreviation expansion, case-folding, and stemming using an improved version of Porter stemmer [26] called the English (Porter2) stemming algorithm[4]. The normalization step also includes the replacement of all non-letter and non-digit characters with space characters followed by removal of redundant whitespaces.

After pre-processing, the content of the document to be indexed is searched for the key terms in the pool and each matching term t is assigned a keyphraseness score, $K(t)$, using the following formula:

$$K(t) = \log_2(GF(t)+1) \times \log_2(LF(t)+1) \times 2^{RF(t)} \times \log_2(FO(t)+1) \times 2^{NW(t)} \times \log_2 NC(t) \times 2^{ADI(t)} \quad (1)$$

[4] http://snowball.tartarus.org/algorithms/english/stemmer.html

JIS-1205-v3 Received: 25th June 2010 Revised: 15th September 2010

where,

- *GF*(*t*) is the Global Frequency of a given term *t*, and represents the occurrence frequency of the term in the pool of collected key terms.
- *LF*(*t*) is the Local Frequency of a given term *t*, and represents the occurrence frequency of the term in the document to be indexed.
- *RF*(*t*) is the Reference Frequency of a given term *t*, and is assigned the value 1 if the term occurs inside any of the reference strings extracted from the document one or more times, and 0 otherwise.
- *FO*(*t*) is the First Occurrence of a given term *t*. It represents the relative distance of the term, where it occurs for the first time in the document, from the beginning of the document. It is computed by dividing the document's length (measured in terms of the number of characters) by the first occurrence position of the term. The closer the term's first occurrence position is to the beginning of the document, the bigger is its first occurrence value.
- *NW*(*t*) is the Number of Words in a given term *t*, and is computed by segmenting the term into its constituent words and then adding up the number of segmented words such that the words which consist of more than two characters and are not stopwords are counted as 1 and the words which do not pass these two criteria are counted as 0.5. For example, for *t*="new bi-directional converter", *NW*(*t*)=0.5+0.5+1+1.
- *NC*(*t*) is the Number of Characters, including spaces, in a given term *t*.
- *ADI*(*t*) is the Average Degree of Importance of a given term *t*, and is computed by dividing the sum of the term's corresponding significance values by its occurrence frequency in the pool of key terms.

Each of the parameters defined above is designed to measure the keyphrase likelihood of a given term from a unique aspect based on a number of assumptions as follows:

- *GF*(*t*) and *LF*(*t*) reflect the assumption that the higher the occurrence frequency of a given term in the pool of collected key terms and in the document, the higher its keyphrase likelihood.
- *FO*(*t*) and *RF*(*t*) reflect the assumption that terms which appear at the beginning of the document (e.g., in the abstract section) or at the end of the document (e.g., in the references section) have a higher keyphrase likelihood.
- *NW*(*t*) and *NC*(*t*) reflect the assumption that, in general, terms which have more words and words which consist of more characters have higher keyphrase likelihood.
- *ADI*(*t*) reflects the assumption that terms in the pool of collected key terms with larger averaged degree of importance values have higher keyphrase likelihood.

The defined parameters have been characterized and treated as "weak" or "strong" according to their perceived evidential strength. This is reflected in Equation 1, where the weak parameters are set to vary logarithmically and the strong parameters to vary exponentially. Also, the weak parameters with the possibility of becoming zero are added by one to avoid taking the logarithm of zero.

The formula given in Equation 1 for computing the keyphraseness scores of the candidate key terms has been derived empirically by trial and error. To do this, we employed a dataset comprising 1000 research documents (e.g., conference papers, journal articles, technical reports, etc.) extracted from the well-known CiteSeer digital library. This dataset was compiled as part of an on-going investigation by our group into the application of citation networks for automatic classification of scientific documents according to a standard library classification schemes, such as the Dewey Decimal Classification (DDC) scheme. A small portion of the documents in this dataset (<10%) are manually indexed by their authors with keyphrases, and we used these documents to develop the formula given in Equation 1.

After computing the keyphraseness scores, the candidate terms are sorted according to their corresponding score values and those with the highest likelihood scores are selected. The number of selected keyphrases for the document could be static or dynamic. In the static mode all the documents are indexed with the same number of keyphrases (pre-defined by user). In the dynamic mode, however, a thresholding mechanism is used to determine

JIS-1205-v3 Received: 25th June 2010 Revised: 15th September 2010

the number of keyphrases to be assigned to each document individually. For example, the threshold could be defined as one tenth of the keyphraseness score value achieved by the term that has the highest keyphraseness likelihood. In the dynamic mode all the terms with a keyphraseness score above the specified threshold will be selected.

4. System Evaluation & Experimental Results

In this section we evaluate the performance of our CKE method and compare its results with those achieved by competitive state-of-the-art methods. For this purpose, we use a dataset called wiki-20[5] developed by Medelyan et al. [3, 4]. The wiki-20 collection consists of 20 Computer Science (CS) related technical research reports each manually indexed by fifteen different human teams independently. Each team consists of two undergraduate and/or graduate CS students. The teams were instructed to assign about 5 keyphrases to each document from a controlled vocabulary of over 4 million terms which serve as article titles in Wikipedia[6]. The teams have assigned an average of 5.7 keyphrases to each document and each document has received an average of 35.5 different keyphrases.

We follow the evaluation approach from [3, 4] and use the inter-indexer consistency formula proposed by Rolling [28] to measure the quality of keyphrases assigned to the wiki-20 documents by our algorithm, as compared to those assigned by human indexers:

$$\text{Inter-indexer consistency} = \frac{2C}{A+B} \quad (2)$$

where C is the number of terms two indexers have in common, and A and B represent the number of terms assigned by each indexer. As shown in [6], Rolling's inter-indexer consistency formula is equivalent to the well-known F1 measure, which is a widely used metric in information retrieval [29]. The inter-consistency scores achieved by the wiki-20 human indexing teams range from 21.1% to 37.1% with an average value of 30.5%. This clearly demonstrates the fact that there is usually a considerable disagreement amongst the human indexers as to the appropriate keyphrases for the documents. Nevertheless, it is an established practice in studies of automatic keyphrase indexing to regard manually assigned keyphrases as of the highest quality (a.k.a gold standard), and evaluate the quality of automatically generated keyphrases by measuring their agreement with those assigned by human indexers.

In the evaluation process, we consider a given term to be common between a human indexing team and our CKE algorithm if the normalised version of the term (i.e. case-folded, abbreviation-expanded, and stemmed) assigned by the human indexing team appears in the set of keyphrases extracted by our algorithm from a given document. Table 1 shows the performance of our algorithm in terms of averaged inter-consistency with the 15 human indexing teams under several experimental conditions, and compares it with the performance of three competitive algorithms: the well-known KEA [2], an enhanced version of KEA known as Maui [3], and the work of Grineva et al. [5]. Both KEA and Maui use ML-based supervised learning techniques (assignment indexing), whereas the method proposed by Grineva and her co-workers takes an unsupervised approach (extraction indexing). We have evaluated the performance of the CKE algorithm under three different experimental conditions. In condition A, the system is set to assign each document 5 keyphrases which is the same number of keyphrases that the human indexers were instructed to assign to each document. In condition B, each document is assigned 6 keyphrases which is closest to the average number of keyphrases that the human indexers have assigned to each document (i.e., 5.7). In condition C, the documents are assigned the same number of keyphrases as assigned by the human indexing teams. For example, if team T1 has assigned only 2 keyphrases to a document, then when computing the inter-consistency of our method with T1 for the given document, the two keyphrases

[5] http://code.google.com/p/maui-indexer/downloads/detail?name=wiki20.tar.gz&can=2&q=

[6] http://www.wikipedia.org

JIS-1205-v3 Received: 25th June 2010 Revised: 15th September 2010

with the highest keyphraseness scores resulted from our algorithm are compared against the 2 keyphrases assigned by T1.

As can be seen from Table 1, our CKE algorithm clearly outperforms its unsupervised rival, Grineva et al. algorithm. In comparison to its supervised rivals, the CKE algorithm significantly outperforms KEA under all conditions. However, it yields a slightly lower averaged inter-consistency score (≤1.1%) compared to Maui depending on the condition. It should be noted that the inter-consistency scores achieved by the CKE algorithm under condition A are the most appropriate set of results to compare with rival algorithms reported here. This is due to the fact that under this condition the CKE algorithm assigns exactly 5 keyphrases to each document, which is the same experimental condition under which the KEA, Maui, and Grineva et al. algorithm have been evaluated.

Table 1. Performance of the CKE algorithm compared to human indexers and competitive methods.

Method		No. of keyphrases assigned to each document	Inter-consistency (%)		
			Min.	Avg.	Max.
Manual	Human indexing (gold standard)	Varied	21.4	30.5	37.1
Supervised	KEA (Naïve Bayes)	Static - 5	15.5	22.6	27.3
	Maui (Bagged Decision Trees & best features)	Static - 5	23.6	31.6	37.9
Unsupervised	Grineva et al.	Static - 5	18.2	27.3	33.0
	CKE (condition A)	Static - 5	22.7	30.6	38.3
	CKE (condition B)	Static - 6	26.0	31.1	39.3
	CKE (condition C)	Varied - the same as assigned by human indexers	22.0	30.5	38.7

Table 2 shows the inter-consistency of each human indexing team with the other teams as well as with the Maui and the CKE algorithm under A, B, and C conditions described above. The teams are sorted in ascending order according to their average inter-consistency scores with the other teams (column 4). As can be seen from Table 2, the inter-consistency of our algorithm with the teams, which have the highest inter-consistency with the other teams (i.e., teams 13, 14, and 15), is considerably higher than that of the Maui. The average inter-consistency of these three teams with the other teams is 35.5%. The average inter-consistency of our algorithm (under both conditions A and B) with these teams is 35.3%, whereas the average inter-consistency of the Maui with them is 31.8%.

Table 2. The inter-consistency of the CKE algorithm with each human team compared to that of the Maui.

JIS-1205-v3 Received: 25th June 2010 Revised: 15th September 2010

Team ID	Native English speakers	Average study year	Inter-consistency (%) with other teams	Inter-consistency (%) with the Maui (Bagged Decision Trees & best features)	Inter-consistency (%) with CKE algorithm under three conditions:		
					A	B	C
1	No	4.5	21.4	23.6	38.35	39.31	38.66
2	No	1	24.1	35.5	24.96	26.00	24.32
3	No	4	26.2	26.7	27.84	27.97	28.58
4	No	2.5	28.7	34.7	30.56	32.45	30.25
5	Yes	4	30.2	29.2	30.22	29.27	30.50
6	Mixed	4	30.8	30.2	27.39	28.21	28.76
7	Yes	3	31.0	34.4	29.94	30.13	28.26
8	No	3	31.2	33.7	27.82	27.12	27.67
9	Yes	4	31.6	31.4	29.35	29.99	31.14
10	Yes	3.5	31.6	31.3	32.30	32.06	32.83
11	Yes	4	31.6	29.4	22.69	26.02	22.00
12	Mixed	3	32.4	37.9	31.73	31.59	31.50
13	Yes	4	33.8	28.9	31.13	30.94	31.27
14	Mixed	4	35.5	33.6	38.23	37.60	36.93
15	Yes	4	37.1	32.9	36.70	37.36	35.33
Average			**30.5**	**31.6**	**30.61**	**31.07**	**30.53**

Table 3 presents and compares the inter-consistency of the human indexing teams, the Maui, and the CKE algorithm on a per document basis. It also shows how much each of the parameters defined in Section 3.3 contribute to the overall performance of our algorithm. Column 4 of Table 3 shows the average inter-consistency of our algorithm with the human indexers for each document. In this column, the scores achieved by the CKE algorithm which are higher than that achieved by the human indexers for the corresponding document are shown in bold and those higher than that achieved by the Maui are highlighted. The CKE algorithm outperforms human indexers in case of 10 test documents and the Maui in case of 9.

Table 3. The inter-consistency of the CKE algorithm with human indexers compared to that of the Maui on a per document basis.

Doc.	Inter-consistency (%)

JIS-1205-v3 Received: 25th June 2010 Revised: 15th September 2010

ID	Human indexers	Maui (Bagged Decision Trees & best features)	CKE algorithm parameters (under condition B)						
			All	LF + NW (Baseline)	+ GF	+ ADI	+ FO	+ RF	+ NC
12049	41.1	52.1	26.26	11.07	11.07	12.29	25.31	25.31	26.43
7183	46.8	48.5	46.29	38.55	39.76	40.97	46.93	46.93	46.93
43032	28.4	43.9	14.42	13.63	07.88	08.14	11.59	8.14	14.84
7502	20.4	41.4	**32.49**	02.42	25.64	21.37	24.49	32.65	32.65
20782	37.6	41.3	**41.11**	27.17	26.39	41.11	41.11	40.00	41.11
18209	39.0	40.6	**50.54**	40.05	42.64	50.55	45.81	50.55	50.55
39955	31.6	39.2	29.37	17.28	22.54	33.60	33.60	33.60	29.04
287	41.4	39.1	26.65	20.15	20.15	21.38	21.60	26.17	26.17
39172	29.1	35.7	27.28	10.73	13.17	20.06	23.69	27.23	27.23
19970	31.3	34.6	**38.80**	17.15	17.41	26.22	26.22	38.78	38.78
40879	31.1	31.7	22.46	06.18	28.83	27.49	22.65	22.65	22.65
10894	52.8	29.7	**65.57**	46.97	55.25	56.36	65.76	65.76	65.76
9307	26.6	29.0	25.92	16.34	16.02	16.24	26.40	25.19	25.19
23267	27.7	27.7	**30.96**	01.11	18.28	22.69	30.20	30.20	30.20
23507	28.9	24.4	28.55	04.07	16.15	18.00	28.86	28.86	28.86
23596	22.6	23.5	**24.32**	04.83	10.91	21.35	21.35	24.23	24.23
37632	24.1	22.9	**25.87**	18.66	20.90	08.04	25.49	25.49	25.49
13259	17.3	17.5	09.09	00.00	09.41	03.06	09.41	09.41	09.41
25473	15.3	10.1	**16.09**	02.55	09.80	15.41	15.41	16.18	16.71
16393	37.5	9.4	**39.84**	20.85	14.48	15.47	22.62	29.33	39.18
Avg.	**31.5**	**32.1**	**31.07**	**15.99**	**21.33**	**23.99**	**28.42**	**30.33**	**31.07**
Average inter-consistency (%) contribution				**+ 15.99**	**+ 5.34**	**+ 2.66**	**+ 4.43**	**+ 1.91**	**+ 0.74**

5. Discussion of Results

As can be seen from Table 3, the inter-consistency results of the CKE algorithm per document ranges widely between 9.09% and 65.57%. Since the documents' references play an essential role in our method, we believe that the higher the number of successfully extracted references from the documents, the better the quality of keyphrases assigned to the documents by our CKE algorithm. The chart shown in Figure 3 demonstrates this trend where, in majority of cases, as the number of successfully extracted references increases, the inter-consistency score improves. Generally, the larger the number of references extracted from a document, the larger would be the number of retrieved Google Word Clouds (GWCs) from the publications which cite either the document or one of its references. Figure 3 also shows the correlation between the number of extracted references and retrieved GWCs per document and the effect of the number of GWCs on the inter-consistency results. As can be seen, the trend lines for the effect of the number of extracted references and the effect of the number of retrieved GWCs per document on the inter-consistency results are very similar. The number of extracted references per document range between 10 to 79 with an average value of 25.9 references per document. The number of retrieved GWCs per document ranges between 62 to 766 with an average value of 271 GWCs per document. In total, the data mining unit has retrieved the metadata records of 5576 publications from GBS, which either cite one of the documents in the wiki-20 collection or one of their references, and almost all of these records (5421, 97.14%) contain a word cloud.

JIS-1205-v3 Received: 25th June 2010 Revised: 15th September 2010

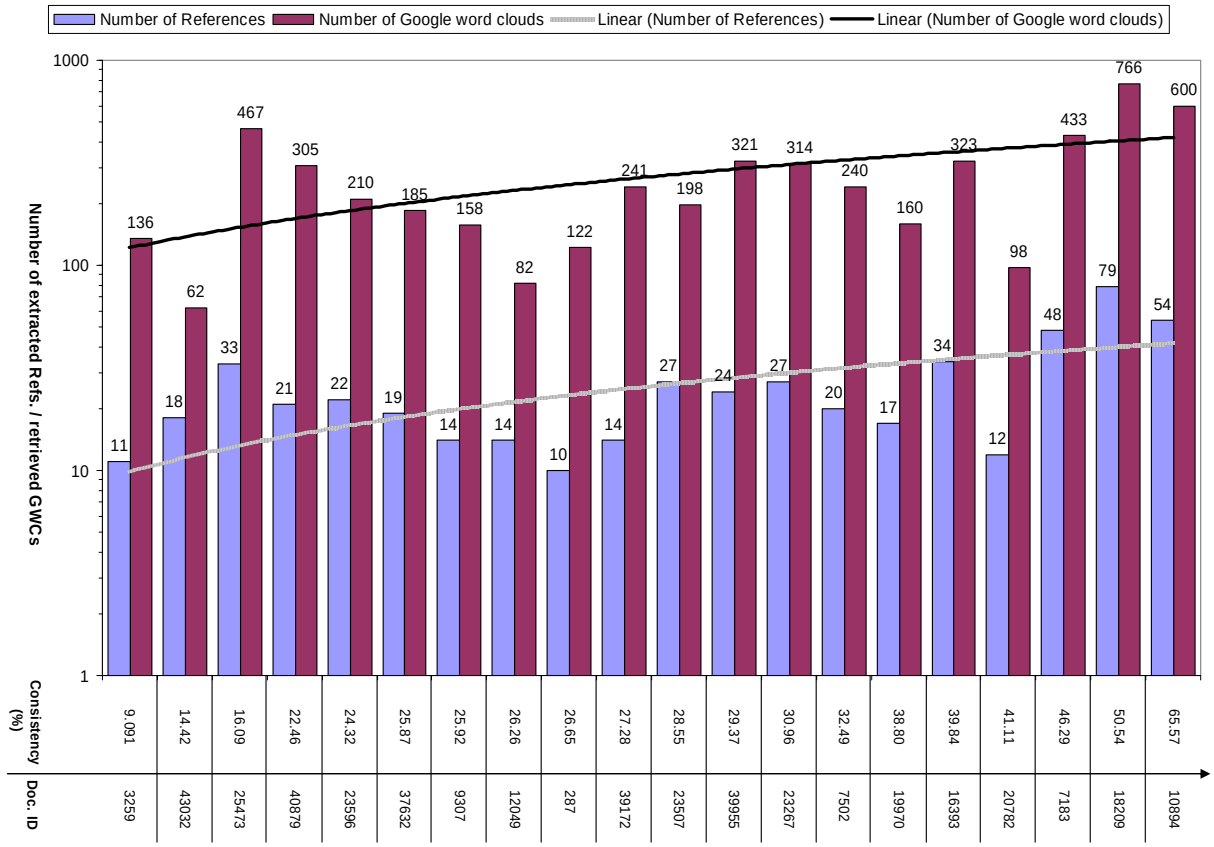

Figure 3. The correlation between the number of extracted references and GWCs per document and the inter-consistency results.

Appendix A[7] lists the five most frequent keyphrases assigned by human indexers per document and compares them with the top five keyphrases assigned by the Maui (Bagged Decision Trees & best features) and the CKE algorithm (under condition A).Table 4 shows three examples from Appendix A. The number in front of each keyphrase in parentheses represents the number of human indexing teams who have assigned the keyphrase to the document. Also, the keyphrases are sorted in a descending order according to their assignment frequency by human indexing teams or keyphraseness scores by the Maui and our algorithm.

The first example is related to document 10894 from the wiki-20 collection, for which our algorithm has achieved the highest inter-consistency score (65.6%) and outperformed both human teams and the Maui by a large margin. The only difference between the list of top five keyphrases assigned by human teams and the keyphrases assigned by our algorithm to this document is that human teams have ranked the term "algorithm" 5th, where our system has ranked it 6th. The second example is related to document 7183, for which the human indexing teams have achieved the highest inter-consistency score (46.8%). The inter-consistency score achieved by our algorithm for this document is slightly lower than achieved by human teams (46.3%). This is due to the two keyphrases "model" and "abstraction" which have been ranked 3rd and 4th by human indexers, whereas our algorithm have ranked them 10th and 29th because of their generality. The final example is related to document 12049, for which the Maui algorithm has achieved the highest inter-consistency score (52.1%).The inter-consistency score achieved by our algorithm for this document is 26.3%, which is considerably lower than that achieved by human teams and the Maui. This is mainly due to the fact that the two keyphrases "compiler-compiler" and "Backus Naur form", which are each assigned by 9 human teams to the document, do not appear in any of the GWCs retrieved for this

[7] Available online at http://www.csn.ul.ie/~arash/PDFs/Appendix_A.pdf

JIS-1205-v3 Received: 25th June 2010 Revised: 15th September 2010

document by the data mining unit of our system. This could be attributed to the relatively low number of references which have been successfully extracted from this document.

Table 4. Comparison of the top five keyphrases assigned to three sample documents from wiki-20 collection by the human indexers, Maui, and our algorithm.

Document ID. Title	**The 5 Most frequent keyphrases assigned by 15 human indexing teams**	**The top 5 keyphrases assigned by Maui (Bagged Decision Trees & best features)**	**The top 5 keyphrases assigned by CKE algorithm (under condition A)**
10894. A safe, efficient regression test selection technique	Regression testing (15) Software maintenance (13) Control flow graph (10) Software testing (9) Algorithm (7)	Algorithm (7) Control flow (0) Software maintenance (13) Computer software (1) Test suite (2)	regression testing (15) software maintenance (13) control flow graph (10) software testing (9) test suite (2)
7183. The challenge of deep models, inference structures, and abstract tasks	Expert system (17) Artificial intelligence (12) Model (abstract) (6) Abstraction (5) Knowledge base (5)	Abstraction (computer science) (3) Expert system (17) Artificial intelligence (12) Scientific modeling (0) Medical diagnosis (0)	expert systems (17) artificial intelligence (12) knowledge based systems (3) knowledge acquisition (0) knowledge base (5)
12049. Occam's razor: the cutting edge for parser technology	Yacc (13) Parsing (12) Compiler-compiler (9) Backus Naur form (9) Compiler (6)	Compiler-compiler (9) Yacc (13) Programming language (4) Parsing (12) Compiler (6)	occam (6) programming languages (4) yacc (13) software engineering (0) obj3 (0)

A limitation of the CKE algorithm is that the quality of automatically generated keyphrases depends to a large extent on the quality of candidate keyphrases extracted from the GBS database by the data mining unit. The case of the sample document 12049, discussed above, is a good example of this dependency. Consequently, a question might be raised concerning whether the CKE algorithm could be effectively replaced by directly querying the GBS database for the metadata corresponding to the document to be indexed. This would not be viable due to the following:

1. GBS database contains only the records of materials which have been published and indexed in the GBS database.
2. GBS database does not contain metadata records of individual research documents (e.g., conference papers, journal articles). However, it may contain metadata records (including GWCs) of books, conference proceedings, journals, etc., in which research documents appear.
3. A single GWC contains a limited number of keyphrases (≈100) and, therefore, is statistically insignificant.

To support above claims, we conducted a simple investigation to illustrate the inefficiency of using the GBS direct querying approach. Figure 3 shows the GWC of a conference proceedings, in which the sample document 12049 appears. Inspection of this figure in conjunction with the data presented in Table 4 shows that none of the top five keyphrases assigned to this document by the human indexers appear in the GWC extracted using the GBS direct querying approach. The only common keyphrase between the extracted GWC and the set of manually assigned keyphrases is "subroutine", which has been assigned to the sample document 12049 by only one of the human indexing teams. This low level of inter-consistency with the human indexers is not surprising as the

JIS-1205-v3 Received: 25th June 2010 Revised: 15th September 2010

approach effectively uses a single GWC to determine the candidate keyphrases. In contrast, our CKE algorithm uses on average 271 GWCs per document to extract the potentially relevant keyphrases and estimate their keyphraseness scores.

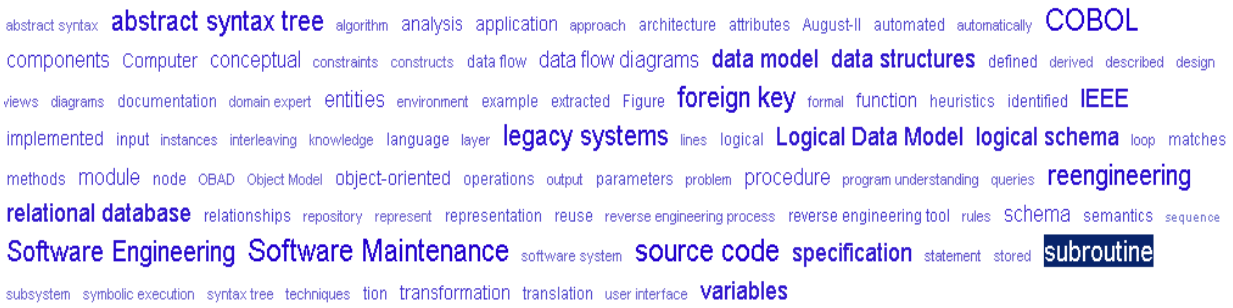

Figure 3. The GWC extracted for the Sample document 12049 using the GBS direct querying approach

6. Conclusions and Future Work

In this article, we proposed a new unsupervised method for keyphrase extraction from scientific literature with a performance that clearly outperforms existing unsupervised methods and yields similar results to those produced by human indexers and existing supervised methods. The novelty of our method lies in leveraging the networked nature of scientific documents, represented in form of citation networks, to help identifying the high-likelihood keyphrases in the documents. We briefly reviewed the work of other researchers [11-17] who have successfully used the citation networks among scientific documents to improve the automatic subject classification, clustering, and full-text indexing of scholarly publications. We then hypothesized and proved via the proposed CKE method that citation networks can also be utilized as a valid knowledge source to improve the keyphrase extraction from the scientific documents. The success of this hypothesis opens up a number of avenues and interesting possibilities for future work:

- Beside the quantity of the retrieved GWCs for a given document (discussed in Section 5), the other major factor which affects the performance of our method is their quality. As described in Section 3.2., the GWCs contain the key terms extracted from publications which cite either the document to be indexed or one of its references. The CKE algorithm is based on the assumption that the majority of citing publications are about the same or similar subject as the document to be indexed and therefore share a significant number of keyphrases with the document. However, by taking this assumption we expose our algorithm to some level of noise caused by a minor number of citing publications which discuss a different subject than the document to be indexed and therefore their GWCs do not contain key terms relevant to the document. To filter out this type of noise we plan to cluster the citing publications according to their Dewey Decimal Classification (DDC) [27] and/or Library of Congress Classification (LCC) [30] numbers and exclude the GWCs of the citing publications which belong to the minor clusters with a population below a threshold.
- The CKE algorithm as described in this article currently utilizes only the GBS database and the GWCs to mine the potential keyphrases for the documents. However, there are at least two other major sources that can be used to improve the quality of keyphrases extracted by the CKE algorithm. The first of such sources is the list of keyphrases which are manually assigned to the references of the document to be indexed by their authors. Despite the fact that the scientific documents with manually assigned keyphrases are in the minority, adding them (when available) to the pool of potential keyphrases could improve the overall performance of the method. The other source for mining high likelihood keyphrases are the terms from controlled vocabularies such as Library of Congress Subject Headings (LCSHs) which are assigned to the publications that cite either the document to be indexed or one of its references. The

JIS-1205-v3 Received: 25th June 2010 Revised: 15th September 2010

list of LCSHs assigned to the citing publications can be retrieved from library union catalogues, such as OCLC's WorldCat [31], which enables querying the catalogues of more than 70,000 libraries around the world.

– As discussed in Section 5, the number of references extracted from a document greatly affects the quality of keyphrases assigned to the document by our algorithm. Based on this, we can expect our method to yield its best performance when applied to documents which have a large number of references, such as Electronic Thesis and Dissertations (ETDs). In the next step, we plan to implement a new version of the proposed method by incorporating the enhancements described above and use it for automatic keyphrase extraction from a large collection of ETD documents archived in a digital library, such as the Networked Digital Library of Thesis and Dissertations (NDLTD) [32]. This would also address the limitation of our current study in terms of the size of the test dataset (20 documents) used for evaluating the performance of the CKE algorithm.

7. References

[1] L. Hunter and K. B. Cohen, Biomedical Language Processing: What's Beyond PubMed?, *Molecular Cell*, 21(5) (2006), 589-594.

[2] 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* (ACM, New York, 1999).

[3] O. Medelyan, *Human-competitive automatic topic indexing* (Ph.D Thesis, University of Waikato, 2009). Available at: http://adt.waikato.ac.nz/public/adt-uow20091029.160923 (accessed 11 June 2010)

[4] O. Medelyan, I. H. Witten and D. Milne, Topic Indexing with Wikipedia. In *Proceedings of the Wikipedia and AI workshop at the AAAI-2008 Conference* (AAAI Press, Chicago, USA, 2008).

[5] M. Grineva, M. Grinev and D. Lizorkin, Extracting key terms from noisy and multi-theme documents. In *Proceedings of the 18th international conference on World wide web* (ACM, New York, USA, 2009).

[6] O. Medelyan and I. H. Witten, Domain-independent automatic keyphrase indexing with small training sets, *Journal of the American Society for Information Science and Technology*, 59(7) (2008), 1026-1040.

[7] K.-M. Schneider, *On Word Frequency Information and Negative Evidence in Naive Bayes Text Classification*, in: J. L. Vicedo, P. Martínez-Barco, R. Muñoz and M. S. Noeda, (eds.), *Advances in Natural Language Processing*, (Springer, Berlin / Heidelberg, 2004).

[8] K. Yi and J. Beheshti, A hidden Markov model-based text classification of medical documents, *Journal of Information Science*, 35(1) (2009), 67-81.

[9] A. Tsymbal, *The problem of concept drift: definitions and related work*, (Technical report TCD-CS-2004-15, Computer Science Department, Trinity College Dublin, 2004). Available at: http://www.scss.tcd.ie/publications/tech-reports/reports.04/TCD-CS-2004-15.pdf (accessed 11 June 2010)

[10] P. Sen, G. Namata, M. Bilgic, L. Getoor, B. Galligher and T. Eliassi-Rad, Collective Classification in Network Data, *AI Magazine*, 29(3) (2008).

[11] B. Aljaber, N. Stokes, J. Bailey and J. Pei, Document clustering of scientific texts using citation contexts, *Information Retrieval*, 13(2) (2009), 101-131.

JIS-1205-v3 Received: 25th June 2010 Revised: 15th September 2010

[12] M. D. Cao and X. Gao, *Combining Contents and Citations for Scientific Document Classification*, in: S. Zhang and R. Jarvis, (eds.), *AI 2005: Advances in Artificial Intelligence*, (Springer, Berlin / Heidelberg, 2005).

[13] S. Bradshaw, *Reference Directed Indexing: Redeeming Relevance for Subject Search in Citation Indexes*, in: w. kit and I. T. Sølvberg, (eds.), *Research and AdvancedTechnology for Digital Libraries*, (Springer, Berlin / Heidelberg, 2003).

[14] S. Bradshaw and K. Hammond, Automatically indexing documents: content vs. reference. In *Proceedings of the 7th international conference on Intelligent user interfaces* (ACM, New York, 2002).

[15] A. Ritchie, S. Robertson and S. Teufel, Comparing citation contexts for information retrieval. In *Proceedings of the 17th ACM conference on Information and knowledge management* (ACM, New York, 2008).

[16] A. Ritchie, S. Teufel and S. Robertson, How to find better index terms through citations. In *Proceedings of the Workshop on How Can Computational Linguistics Improve Information Retrieval?* (Association for Computational Linguistics, Sydney, Australia, 2006).

[17] A. Ritchie, S. Teufel and S. Robertson, *Using Terms from Citations for IR: Some First Results*, in: C. Macdonald, I. Ounis, V. Plachouras, I. Ruthven and R. W. White, (eds.), *Advances in Information Retrieval*, (Springer, Berlin / Heidelberg, 2008).

[18] L. Vincent, Google Book Search: Document Understanding on a Massive Scale. In *Proceedings of the Ninth International Conference on Document Analysis and Recognition (ICDAR 2007).* (2007).

[19] D. Bergmark, P. Phempoonpanich and S. Zhao, Scraping the ACM Digital Library, *SIGIR Forum*, 35(2) (2001), 1-7.

[20] M.-Y. Day, R. T.-H. Tsai, C.-L. Sung, C.-C. Hsieh, C.-W. Lee, S.-H. Wu, K.-P. Wu, C.-S. Ong and W.-L. Hsu, Reference metadata extraction using a hierarchical knowledge representation framework, *Decision Support Systems*, 43(1) (2007), 152-167.

[21] A. Takasu, Bibliographic attribute extraction from erroneous references based on a statistical model. In *Proceedings of the 3rd ACM/IEEE-CS joint conference on Digital libraries* (IEEE Computer Society, Washington, DC, USA, 2003).

[22] F. Peng and A. McCallum, Information extraction from research papers using conditional random fields, *Information Processing & Management*, 42(4) (2006), 963-979.

[23] I. G. Councill, C. L. Giles and M. Y. Kan, ParsCit: An open-source CRF reference string parsing package. In *Proceedings of the Language Resources and Evaluation Conference (LREC 08)* (Morrocco, Marrakesh, 2008).

[24] S. BRIN, *A Library to Last Forever*, (The New York Times, 8 October 2009). Available at: http://www.nytimes.com/2009/10/09/opinion/09brin.html?_r=1 (accessed 11 June 2010)

[25] D. Puppin, *Responses to "More Metadata Problems in Google Books?: Word Clouds"*, (Seeing the picture, 1 October 2009). Available at: http://blog.lib.uiowa.edu/hardinmd/2009/09/30/more-metadata-problems-in-google-books-word-clouds/#comment-1627 (accessed 11 June 2010)

[26] M. F. Porter, An algorithm for suffix stripping, *Program*, 14(3) (1980), 130−137.

[27] M. Dewey, *Dewey Decimal Classification (DDC)*, (Online Computer Library Center (OCLC), Dublin, Ohio, USA, 1876-2010). Available at: http://www.oclc.org/us/en/dewey (accessed 11 June 2010)

[28] L. Rolling, Indexing consistency, quality and efficiency, *Information Processing & Management*, 17(2) (1981), 69-76.

[29] C. J. V. Rijsbergen, *Information retrieval* (Butterworths, London / Boston, 1979).

JIS-1205-v3 Received: 25th June 2010 Revised: 15th September 2010

[30] H. Putnam, *Library of Congress Classification (LCC)*, (Library of Congress, Cataloging Policy and Support Office, Washington, DC, USA, 1897-2010). Available at: http://www.loc.gov/catdir/cpso/lcc.html (accessed 11 June 2010)

[31] *WorldCat*, (Online Computer Library Center (OCLC), Dublin, Ohio, USA, 2001-2010). Available at: http://www.oclc.org/worldcat/default.htm (accessed 11 June 2010)

[32] *Networked Digital Library of Thesis and Dissertations*, (NDLTD, 1996-2010). Available at: http://www.ndltd.org (accessed 11 June 2010)

JIS-1205-v3 Received: 25th June 2010 Revised: 15th September 2010