Turkish Journal of Electrical Engineering & Computer Sciences

http://journals.tubitak.gov.tr/elektrik/

Research Article

Turk J Elec Eng & Comp Sci
(2017) 25: 1784 – 1794

doi:10.3906/elk-1511-203

Using latent semantic analysis for automated keyword extraction from large document corpora

Tuğba ÖNAL SÜZEK*
Department of Computer Engineering, Faculty of Engineering, Muğla Sıtkı Koçman University, Muğla, Turkey

Received: 22.11.2015 • **Accepted/Published Online:** 23.06.2016 • **Final Version:** 29.05.2017

Abstract: In this study, we describe a keyword extraction technique that uses latent semantic analysis (LSA) to identify semantically important single topic words or keywords. We compare our method against two other automated keyword extractors, Tf-idf (term frequency-inverse document frequency) and Metamap, using human-annotated keywords as a reference. Our results suggest that the LSA-based keyword extraction method performs comparably to the other techniques. Therefore, in an incremental update setting, the LSA-based keyword extraction method can be preferably used to extract keywords from text descriptions from big data when compared to existing keyword extraction methods.

Key words: Bioinformatics, text mining, information retrieval

1. Introduction

Biomedical document management systems have grown in popularity and it is getting difficult to index, categorize, group, filter, and explore the large amounts of documents from the diverse disciplines they contain. One way to categorize biomedical text documents is the extraction and assignment of keywords to individual documents in the system. Keywords are the shortest lists of words that represent the main topic of the documents and they can be used in many applications including automatic document indexing, clustering, filtering, and topic detection.

Keywords can be extracted from the documents either manually by curators or automatically by dedicated text-mining programs. The amount of data increase makes curator-generated keyword extraction infeasible; hence, the semiautomated or fully automated keyword extraction methods stand out as the only viable options.

Keyword extraction methods can be classified into two main categories: supervised or unsupervised. Several supervised machine learning algorithms have been proposed for classifying candidate terms as keywords [1–3]. However, due to the supervised nature of these algorithms, they all require a training phase and training is expensive in terms of computational resources.

There have been a few unsupervised keyword extraction algorithms published in the last decade. One well-known biomedical term extraction program is Metamap, a domain-specific knowledge base [4]. Metamap is an initiative launched and maintained by the National Library of Medicine. It provides automatic term recommendations to human curators via Metamap to assist in the curation process. This vocabulary of headings in the biomedical text processing domain is called Medical Subject Headings (MeSH). MeSH vocabulary is used as keywords in some keyword extraction studies [5]. Based on the MeSH terms, the topical similarity

*Correspondence: tsuzek@gmail.com

of biomedical documents can be computed [6]. MeSH terms are assigned manually by human curators to each PubMed article using a controlled, hierarchical vocabulary. The manual nature of MeSH term extraction requires a highly expensive and time-consuming indexing pipeline [7]. Metamap binary is freely available for download and can be used as a standalone application to extract keywords for any given text document in an unsupervised manner on Linux, Mac OS/X, or Windows machines.

Tf-idf is an unsupervised method [8] used in keyword extraction studies as a baseline for performance analysis. It is very effective and efficient at identifying the keywords distinguishing a document from the rest of the corpus simply by assigning Tf-idf weights to all words and then sorting these words in descending order according to their weight. Top weighted terms are picked according to a predetermined threshold and identified as keywords. This unsupervised method requires a collection of documents due to the formula requiring the computation of the document frequency (DF) and term frequency (TF). Document frequency is the number of documents in the collection that contain that particular term. Term frequency, in its simplest form, is the number of times that particular term occurs in a particular document. The Tf-idf thresholding method has been previously shown to be a reliable, simple, and effective technique for vocabulary reduction with easy scalability to large corpora [9].

Many unsupervised keyword extraction studies use Tf-idf as a baseline for comparing keyword extraction performance. For example, one study [10] proposed a chi-square measure that picked the keywords based on their cooccurrence with frequent terms and measured the performance of this method against Tf-idf. Similarly, another recent unsupervised keyword extraction method used Tf-idf as a baseline to demonstrate that combining the lexical class (i.e. part-of-speech information) or sentence ranking score improved simple Tf-idf based keyword extraction [11].

In 1990, Deerwester et al. [12] published a technique called latent semantic analysis (LSA) to analyze term-by-document matrices. LSA uses the singular value decomposition (SVD) algorithm to organize terms and documents into their underlying semantic space based, in part, on term cooccurrence. LSA has been shown to reveal the underlying semantic interrelations between sentences and words of a document when applied on the SVD of the document matrix. The SVD models relationships between words and sentences while filtering out the noisy information, leading to an improvement in retrieval accuracy [13]. LSA [14] along with the nonnegative matrix factorization [15], semidiscrete matrix decomposition [16], probabilistic latent semantic indexing [17], and latent Dirichlet allocation [18] algorithms can be generally classified as linear algebraic reduction methods. Linear algebraic reduction methods have been widely used for the document summarization.

LSA-based summarization methods have been demonstrated to capture outstanding and recurring word patterns corresponding to the 'semantic' topic represented in one of the columns of U, i.e. by one of the vectors of the left singular matrix [19]. The actual magnitude of the singular value corresponding to the vector represents the relative degree of importance of the pattern in the document. This method has been shown to correlate surprisingly well with a human being's judgment while classifying a document collection [20].

As a first step in using SVD as a part of LSA, a sparse input matrix is created in which each column of the matrix represents a sentence , text passage, or other context, and each row represents a unique word. The cells are populated with weights depicting the importance of the word. Weight scoring schemes vary, but they can be as simple as noting in the appropriate cell the frequency with which the word appears in the text. Some common local weighting functions are defined in Table 1 [21].

Using the predetermined weighting scheme, input matrix A is created as in Figure 1. The strategy used for the creation of the input matrix strongly affects the resulting matrices of the SVD [22].

Table1. Common local weighting functions.

Weighting function	Weighting formula for cells of matrix A = (a_{ij})
Binary	a_{ij} = 1 if term i exists in document j, or else 0
Term frequency (tf)	$a_{ij} = tf_{ij}$, the number of occurrences of term i in document j
Log	$a_{ij} = \log(tf_{ij} + 1)$
Augnorm	$a_{ij} = ((tf_{ij} / \max_i(tf_{ij})) + 1) / 2$

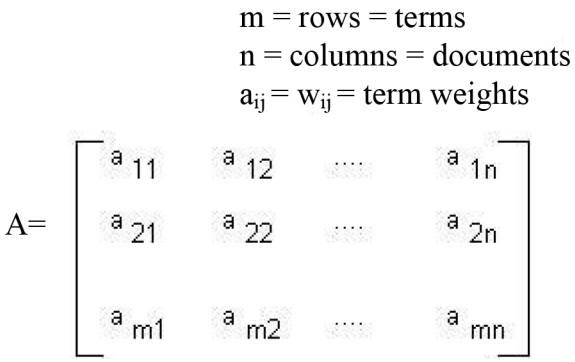

Figure 1. Sparse matrix A for entire corpora.

As the second step of LSA, the SVD decomposes the input matrix A into three matrices:

$$A=U\sum V^t, \tag{1}$$

where A is the input matrix with dimensions m $\times$ n, U is an m $\times$ n matrix that represents the description of the original rows of the input matrix as a vector of extracted concepts, $\sum$ is an n $\times$ n diagonal matrix containing scalar singular values sorted in descending order, and V is an m $\times$ n orthonormal matrix that represents the description of the original columns of the input matrix as a vector of the extracted concepts. Matrices can be summarized as follows:

A: Input matrix (m $\times$ n)

U: Words $\times$ extracted concepts (m $\times$ m)

Σ : Scaling values, diagonal descending matrix (m $\times$ n)

V: Sentences $\times$ extracted concepts (n $\times$ n)

Algorithms that use LSA for text summarization differ after the single value decomposition step. Gong and Liu's algorithm [23], for instance, uses V^T (Figure 1) for sentence selection and places the most important concept in the top row. This top-placed sentence has the highest cell value of the row. Steinberger and Jezek [24] proposed a modification to the algorithm to compensate for varying sentence lengths. To do this, the length of each sentence (i) is computed using the formula $\text{length}_i = \sqrt{\sum_{j-1}^{n} \Sigma_{jj} \times V_{ij}}$ and this length is represented by a row in the V matrix. After all length calculations the longest sentences are chosen as the summary. Another modification to Gong and Liu's algorithm was published by Murray et al. [25] where, instead of picking the best sentence for each concept after the first two steps, the n best sentences were extracted with n determined by the corresponding singular values from matrix Σ. The number of sentences selected from each topic row is determined by the ratio of the largest singular value to the sum of all singular values.

All text summarization methods, either extractive (sentences come from the original document) or abstractive (brand new sentences are generated), aim to pick the several best 'sentences' from a document [26]. In our study, the LSA-based summarization method is going to be adapted to derive summary keywords instead of summary sentences. Most of the sentence structures are not regarded as important by human curators; thus, they do not exist in the author-picked reference set. Deriving keywords instead of sentences not only saves space by storing only the most important words for each document in large document sets, but it also complements the reference set. Evaluation of the summarization systems is a critical component of summarization systems. The evaluation method needs to complement the chosen summarization technique [27]. The reference method used for evaluation of our proposed algorithm consists of the keywords generated by PubMed abstract authors and thus complements our method in scope.

2. Materials

2.1. Tf-idf-based keywords

During the clustering phase described in a previous study [28], augmented Tf-idf scores [29] of each word in each PubChem BioAssay [30] description were generated. All words of the individual text had a score in the full ranked output and therefore keywords generated by thresholding the Tf-idf scores form a comprehensive set.

2.2. Metamap-generated keywords

Although PubChem Compound and Substance records have been annotated with MeSH terms [31], Pubchem BioAssay text descriptions are not readily annotated with them. To generate the MeSH terms for each BioAssay text description, we ran the locally installed Metamap 2010 binary with default options using the entire Unified Medical Language System (UMLS) metathesaurus. The original text description was used as input without any preprocessing. Unlike LSA or Tf-idf keywords that require all syntactic features to be preserved for optimal performance, all the stop words and words with less than 3 characters were retained in the input of this system.

The automatically generated output returned multiple lines of Metamap scores, phrases, and their unique concepts inside the square brackets. We discarded the unique concepts, split the phrases into the keywords, and sorted them by Metamap score. Only 63% of Metamap's output (BioAssay, word) pairs had an exact match in the original text (16,395/25,676). The remaining 37% of the output pairs originated from the UMLS thesaurus and did not exist in the original text. Such usage of vocabulary outside of the input text makes Metamap akin to a human paraphraser.

2.3. Author keywords as the reference set

During the submission of an abstract to the PubMed database, an author might subjectively decide on any word or word combination that the author deems relevant to the document. However, these words do not necessarily exist in the abstract or even in the original document. Unfortunately, only about 15% of PubMed articles are furnished with author-generated keywords [32]. Therefore, for most PubMed articles, either manual curation or automated methods are required for keyword extraction.

The reference dataset used in this study represented a very small subset of PubChem BioAssays submitted by the Chemical Database at the European Molecular Biology Laboratory (ChEMBL) [33], directly derived from PubMed articles. The PubChem BioAssay database contains 430 BioAssay descriptions containing author-generated keywords. Each description can be downloaded in XML format from the PubChem FTP site. The

submitting authors provided an average of 8 keywords per BioAssay for these 430 BioAssays. We derived the author keywords from the keywords field of each PubMed Central abstract. Each word of the phrase is treated as a separate word.

Although the author keywords are reliable due to their creation via human judgment, we recognize that they are not the most ideal in terms of coverage. An author keyword does not necessarily exist in the abstract. In fact, 35% of the 3419 author-generated (BioAssay, keyword) pairs in the reference set do not appear anywhere in there reference set.

3. Method

In this paper, we present an alternative unsupervised keyword extraction method based on LSA. All LSA-based summarization techniques create a term-content matrix as the first step. When creating a summary for each document in corpora, a single term-document matrix is generated representing the entire corpus where rows correspond to the terms and columns correspond to the documents (Figure 1). We instead propose generating a separate matrix for each assay document (Figure 2). This decision is of practical use particularly with big datasets, where statistics can be computed by only processing the newly added descriptions incrementally. Since our method handled each document separately in the memory, the memory requirement was minimized, which can be a major drawback for LSA applications that process whole corpora.

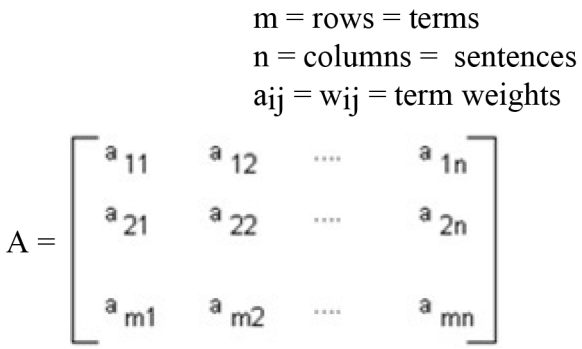

Figure 2. Sparse matrix A for each document.

For the first step of the LSA-based method, each assay document was first represented as matrix A, where rows correspond to words and columns to the sentences (Figure 2). After matrix A was decomposed via SVD, $A = U \sum V^t$, we made the assumption that columns of U would correspond to the topics of the document with the first column being the most important topic of the document. Based on this main assumption that the first column of U represents the main topic with words sorted in order of importance, a threshold was set and the top N highest values are collected. We denoted this word set as the LSA-predicted keyword set.

For our LSA-based keywords, instead of generating summary sentences, the most important keywords representing the document are picked by selecting the top N values of the first column of U ($U[,1]$ vector) and the corresponding words in the token list.

Term weight assignments and SVD decomposition steps were implemented using a Perl script with embedded R functions. Punctuation, words shorter than 3 letters, and stop words such as the function words *and*, *the*, and *of* were filtered from the analysis during the previous study [27]. Stemming, which is usually applied to reduce the inflected words to their stem in information retrieval systems, was not done as it disrupts the chemical formulas and abbreviations commonly found in PubChem BioAssay descriptions.

A weighted score for each term was calculated in two alternative ways: the binary score and sentence-based Tf-isf (term frequency-inverse sentence frequency). For the binary score, each cell value of matrix A was set to 1 if the corresponding word (row) existed in the given sentence (column). For sentence based Tf-isf weighted scores, each cell carries the Tf-isf value of the term in the corresponding sentence. Term frequency (TF) and inverse sentence frequency (ISF) formulas are used as in Eqs. (2) and (3), respectively.

Term frequency score for each word i in sentence j:

$$Tfij = \frac{\text{Number of times word } i \text{ in sentence } j}{\text{Total number of words in sentence } j} \tag{2}$$

Inverse sentence frequency score for each word i:

$$Isfi = \frac{\text{Total number of sentences in the document}}{\text{Number of sentences with word } i} \tag{3}$$

Based on these two alternative weights, two sets of LSA-based keywords were prepared, one with binary scores for a_{ij} in A and the second with sentence-based Tf-isf weighted scores for a_{ij} in A as shown in Eqs. (4) and (5). Document-wide counts were replaced with the sentence-wide counts as each matrix A represents a single document.

Binary weight assignment of $A = (a_{ij})$:

$$a_{ij} = \begin{cases} 1 \text{ if word } i \text{ exists in the given sentence (column)} \\ 1 \text{ otherwise} \end{cases} \tag{4}$$

Sentence-based Tf-isf assignment of $A = (a_{ij})$:

$$a_{ij} = Tfij \times Isfi \tag{5}$$

For the SVD step of LSA, R's svd software module was used to decompose the input matrix A into three matrices as in Eq. (1). Unlike the usual LSA applications, the complete U matrix is utilized from the full SVD. That is, we did not reduce the dimension of SVD, but instead used the full U matrix after the full unitary decomposition of the null-space of the matrix A. Sizes of individual assay descriptions are small so there is no technical need for such a reduction.

The first column of the U matrix is assumed to represent the most important topic; therefore, words corresponding to each element of vector $U[,1]$ from the token list are considered the LSA-based keywords. The LSA-based keyword list is similar to the Tf-idf-based keyword list in the sense that both lists are ranked lists of all the words in the original text. If there is no cut-off threshold, 100% of all words overlap with the original text, so the choice of the top N threshold is a crucial decision during the performance evaluation. We assumed that words corresponding to the first column of the matrix U represent the most important topic of the document. By a heuristic method, the top N percent of the words of this column were picked as the predicted keyword set.

4. Results

The criterion for thresholding of ranked lists is crucial for the evaluation of the proposed LSA-based keyword extraction method. For the evaluation of this new method, keywords that authors declared as relevant from 430 BioAssays were used as the reference set. We utilized receiver operating characteristic (ROC) curve analysis and Student's t-test to evaluate the statistical significance of the three methods to discriminate between keywords and non-keywords.

4.1. Comparison of the Tf-idf, LSA, and Metamap keyword extraction methods using the ROC curve

Each of the three methods splits the predictions of each method into two classes, as "keyword" and "non-keyword", based on a ranked list and a threshold, offering a solution to the same question: a binary classification problem. For defining the performance of all of these three classifiers, a common thresholding criterion needs to be determined among them to split the data into two classes. The criterion for the threshold or cut-off of the binary split needed to be determined in a fair way so that the same splitting criterion was applied across the three methods even though the size and coverage of the prediction lists of each method were quite different.

To generate this binary discrimination threshold of the ROC curve for each method, the top 5% of the ranked list was considered to be positive/true, i.e. "keywords," and the bottom 95% was considered negative or "non-keywords". Then N, the percentage of true "keywords", was increased in 5% increments. For example, for the Tf-idf method, the top 5% of the Tf-idf-ranked list was accepted as true predictions and the true predictions that overlapped with the author keyword set and the original text were assumed to be true positive (TP). For the next point, the top 10% of the list was accepted as a true prediction. For all 20 points between 0 and 100, the true positive rate (TPR) and false positive rate (FPR) were computed in the same way for all four methods. The ROC curve was plotted using a combination of Perl's plot and lines function dumped in JPEG format (Figure 3). Eqs. (6) and (7) depict the formulas for each N percent increment of FPR on the x-axis and TPR on the y-axis.

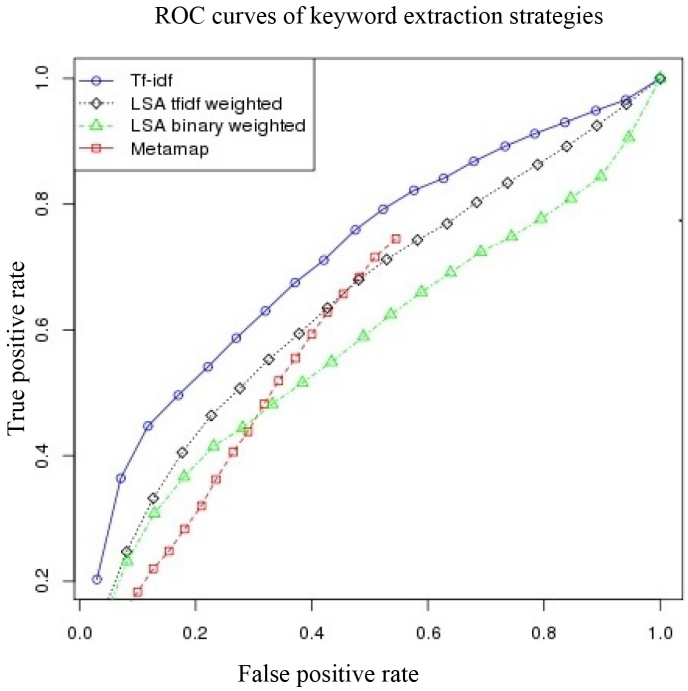

Figure 3. Receiver operating curve analysis for methods using author keywords as reference. The true positive rate (TP / (TP + FN)) is plotted against the false positive rate (FP / (FP + TN)).

False positive rate for top N percent of the ranked list:

$$FPR(N/100) = \frac{\begin{array}{c}\text{Number of predictions in top N percentage by method and original} \\ \text{text that does not exist in the author keywords}\end{array}}{Number\ of\ words\ in\ original\ text\ that\ does\ not\ exist\ in\ author\ keywords} \tag{6}$$

Positive rate for top N percent of the ranked list:

$$TPR(N/100) = \frac{\begin{array}{c}\text{Number of common words between top } N \text{ percentage of} \\ \text{method and author keywords and original text}\end{array}}{\begin{array}{c}\text{Number of common words between} \\ \text{original text and author keywords}\end{array}} \quad (7)$$

As illustrated in the ROC curves in Figure 3, the Tf-idf thresholding method outperformed both variants of the LSA-based method and the Metamap method. Tf-idf used external information as it computed the weights using the complete set of documents. Using global information about the whole corpus gives the Tf-idf method a clear advantage. LSA uses only the local information from a single document. Although Tf-idf has access to the global information from the full corpus, LSA's ROC curve is still close to that of Tf-idf. A recent study showed that LSA-based algorithms do not perform as well as machine learning-based algorithms or other algorithms that use external information [21], so it is not surprising that Tf-idf is performing better than our LSA-based method.

The main reason for selecting Metamap for comparison was its dependence on local information only, although with a dependence on a predefined knowledge base such as the UMLS thesaurus. However, one apparent disadvantage of using the Metamap approach can be seen in the ROC curve in Figure 3. Metamap's FPR maximized at 0.545 and it could not ever reach 1.0. The reason is that a zero count of TN (true negatives) can never be achieved. Even with the most liberal threshold that keeps all of the Metamap output as positive predictions, the predictions can never overlap fully with the original document due to Metamap's word set originating from the UMLS thesaurus.

Similarly, Metamap's maximum TPR was 0.745 and it could not reach 1.0. The reason is that a zero count of FN (false negatives) can never be obtained. Even with the most liberal threshold, Metamap output does not overlap fully with the complete author set, the true positives. Another disadvantage is that it is applicable to English documents only. Both Tf-idf and the LSA-based method are language-independent.

4.2. Comparison of the Tf-idf, LSA, and Metamap methods using random keywords

We applied a statistical hypothesis test measure, a paired sample t-test with unequal variance, to determine whether there was a significant difference between the average values of the method's keywords and random keywords. For each of the 430 assays and for each of the three methods (Tf-idf, LSA, and Metamap), six main steps were followed:

1. The number of author keywords for Assay$_i$ was defined as N_i.
2. The total number of words in Assay$_i$ was defined as S_i.
3. N_i numbers were picked between (1,S_i) = k and the kth word was put into a random keyword bin for Assay$_i$ using Perl's rand function.
4. The overlap ratio of the Random keywords to the author set was computed = R_i.
5. The top N_i words of the ranked list of the method were collected as a method keyword set.
6. The overlap ratio of the Method keyword set to the author set was computed = M_i.

This procedure creates two vectors (R_i and M_i) consisting of 430 elements for each of the three methods. The sizes of these vectors are equal and comparable due to their common reference of the author keyword set.

Averages of the two distributions generated by random method R_i and method M_i were denoted by $\mu 1$ and $\mu 2$, respectively. The null hypothesis for the paired sample t-test is defined in Eq. (8) and the alternative hypothesis is defined in Eq. (9).

$\mu 1 =$ Mean overlap of Method generated keywords with the author keywords.

$\mu 2 =$ Mean overlap of Random keywords with the author keywords.

$$H0 : d = \mu 1 - \mu 2 = 0 \tag{8}$$

$$H1 : d > 0 \tag{9}$$

R's built-in t.test command was integrated into a Perl script for the t-test computations. A significance level of 0.01% was chosen to increase the accuracy of the confidence interval. The confidence interval and the significance for each vector representing the methods were reported for each vector from each method. Results are provided in Table 2.

Table 2. Comparison of paired t-test results of the methods. Results of t-tests are presented for the three methods when the alternative hypothesis was set as Method having a greater mean of overlap with the author set than the Random keyword set.

Compared keyword extraction method pairs	P-value	d = Mean of the differences	Test statistic	Degrees of freedom	99% Confidence interval
Tf-idf vs. Random	< 2.2e-16	0.19	19.76	430	0.171–∞
LSA vs. Random	< 2.2e-16	0.10	10.08	430	0.081–∞
Metamap vs. Random	< 1.08e-4	0.03	3.73	430	0.011–∞

As shown by the significantly low P-values in Table 2, all three automatically generated keywords overlapped significantly more with the author keyword list than the randomly generated keywords. However, the mean difference for the Tf-idf method was slightly higher than that for LSA. The statistical significance of LSA and Tf-idf was much higher than that of Metamap, suggesting that both of them agree with the author keywords more than with Metamap.

5. Discussion

A new keyword extraction method was proposed using the LSA method combined with a ranking scheme from a single document and its precision and recall were analyzed using author keywords as a reference. LSA has already been used for sentence extraction in previous studies. The contribution of this study was to utilize the first column of the left singular matrix of the LSA with the assumption that it represents the ranking of the most important concept of the document with the keywords sorted from most relevant to least. With any given cut-off number N, the top N most important keywords were extracted from the first column. Output keywords were compared to two popular automated keyword extraction methods, ranked Tf-idf and ranked Metamap. We evaluated our method by using the author keywords. Student t-test-based results also showed that the LSA-based extracted keywords overlapped with the author keywords significantly better than a randomly extracted keyword set with very low P-values.

The main advantage of our proposed method over Tf-idf is the mere utilization of the term weights of a single document at hand, whereas Tf-idf needs the knowledge of the whole document set in order to compute the inverse document frequency. For each word, the inverse document frequency formula needs to compute the number of documents that contain that particular word. Efficiency becomes crucial for large document sets where keywords need to be extracted automatically from daily submissions incrementally.

The LSA-based keyword extraction method is language-independent, knowledge-lean, and easy to apply in a diverse set of settings. It does not require any language-specific information or external knowledge sources other than the input text.

In summary, our proposed LSA-based keyword extraction method offers an alternative to Tf-idf-based keyword extraction with comparable statistical significance and the practical advantage of not depending on global information.

Availability: The source code of the scripts to compute the LSA method, to plot the ROC curve, and to compute t-test statistics is available to academic users 'as is' on request. The list of reference BioAssays with the author keywords and the predicted keywords is available from http://ceng.mu.edu.tr/∼tugba/keywordextractiondata/.

Acknowledgment

This work was supported in part by the TÜBİTAK 2232 Reintegration Fellowship Grant 113C030.

References

[1] Hulth A. Improved automatic keyword extraction given more linguistic knowledge. In: Conference on Empirical Methods in Natural Language Processing; 11–12 July 2003; Sapporo, Japan. pp. 216-23.

[2] Litvak M, Last M. Graph-based keyword extraction for single-document summarization. In: 2nd Workshop on Multi-Source Multilingual Information Extraction and Summarization; 2008; Morristown, NJ, USA.

[3] Turney PD. Learning algorithms for keyphrase extraction. Inform Retrieval 2000; 2: 303-336.

[4] Aronson AR, Mork JG, Gay CW, Humphrey SM, Rogers WJ. The NLM indexing initiative's medical text indexer. St Heal T 2004; 107: 268-272.

[5] Chiang JH, Liu HH, Huang YT. Condensing biomedical journal texts through paragraph ranking. Bioinformatics 2011; 27: 1143-1149.

[6] D'Souza JL, Smalheiser NR. Three journal similarity metrics and their application to biomedical journals. PLoS One 2014; 9: e115681.

[7] Huang M, Neveol A, Lu Z. Recommending Mesh terms for annotating biomedical articles. J Am Med Inform Assoc 2011; 18: 660-667.

[8] Salton G, Buckley C. Term-weighting approaches in automatic text retrieval. Inf Process Manag 1988; 24: 513-523.

[9] Yang Y, Pedersen JO. A comparative study on feature selection in text categorization. In: Fourteenth International Conference on Machine Learning; 1997. pp. 412-210.

[10] Matsuo Y, Ishizuka M. Keyword extraction from a single document using word co-occurrence statistical information. Int J Artif Intell T 2004; 13: 157-169.

[11] Liu F, Pennell D, Liu F, Liu Y. Unsupervised approaches for automatic keyword extraction using meeting transcripts. In: Human Language Technologies: The Annual Conference of the North American Chapter of the Association for Computational Linguistics; 31 May–5 June 2009; Boulder, CO, USA. pp. 620-628.

[12] Deerwester S, Dumais ST, Furnas GW, Landauer TK, Harshman R. Indexing by latent semantic analysis. J Am Soc Inform Sci 1990; 41: 391-407.

[13] Gao J, Zhang J. Clustered SVD strategies in latent semantic indexing. Inf Process Manag 2005; 41: 1051-1063.

[14] Landauer TK, Dumais ST. A solution to Plato's problem: the latent semantic analysis theory of acquisition, induction, and representation of knowledge. Psychol Rev 1997; 104: 211-240.

[15] Lee DD, Seung HS. Learning the parts of objects by non-negative matrix factorization. Nature 1999; 401: 788-791.

[16] Kolda TG, O'Leary DP. A semidiscrete matrix decomposition for latent semantic indexing information retrieval. ACM T Inform Syst 1998; 16: 322-346.

[17] Hoffman T. Probabilistic latent semantic indexing. In: 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval; 15–19 August 1999; Berkeley, CA, USA.

[18] Blei DM, Ng AY, Jordan MI. Latent Dirichlet allocation. J Mach Learn Res 2003; 3: 993-1022.

[19] Berry MW, Dumais ST, O'Brien GW. Using linear algebra for intelligent information retrieval. SIAM Rev 1995; 37: 573-595.

[20] Landauer TK, Laham D, Foltz, PW. Learning human-like knowledge by singular value decomposition: a progress report. Adv Neur In 1998; 10: 45-51.

[21] Berry MW, Browne, M. Understanding Search Engines: Mathematical Modeling and Text Retrieval. 2nd ed. Philadelphia, PA, USA: Society for Industrial and Applied Mathematics, 2005.

[22] Ozsoy MG, Alpaslan FN, Cicekli I. Text summarization using latent semantic analysis. J Inf Sci 2011; 37: 405-417.

[23] Gong Y, Liu X. Generic text summarization using relevance measure and latent semantic analysis. In: 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval; 9–12 September 2001; New York, NY, USA.

[24] Steinberger J, Jezek K. Text summarization and singular value decomposition. In: Third International Conference on Advances in Information Systems; 20–22 October 2004; Czech Republic.

[25] Murray G, Renals S, Carletta J. Extractive summarization of meeting recordings. In: Interspeech; 4–8 September 2005; Portugal.

[26] Gupta V, Lehal GS. A survey of text summarization extractive techniques. J Emerg Techol Web Intell 2010; 2: 258-268.

[27] Hahn U, Mani I. The challenges of automatic summarization. Computer 2000; 33: 29-36.

[28] Han L, Suzek TO, Wang Y, Bryant SH. The text-mining based PubChem Bioassay neighboring analysis. BMC Bioinformatics 2010; 11: 549.

[29] Salton G, Buckley C. Improving retrieval performance by relevance feedback. J Am Soc Inform Sci 1990; 41: 288-297.

[30] Wang Y, Suzek T, Zhang J, Wang J, He S, Cheng T, Shoemaker BA, Gindulyte A, Bryant SH. PubChem BioAssay: 2014 update. Nucleic Acids Res 2014; 42: D1075-1082.

[31] Kim S, Thiessen PA, Bolton EE, Chen J, Fu G, Gindulyte A, Han L, He J, He S, Shoemaker BA et al. PubChem Substance and Compound databases. Nucleic Acids Res 2016; 44: D1202-1213.

[32] Neveol A, Dogan RI, Lu Z. Author keywords in biomedical journal articles. In: AMIA Annual Symposium; 13–17 November 2010; Washington, DC, USA.

[33] Bento AP, Gaulton A, Hersey A, Bellis LJ, Chambers J, Davies M, Krüger FA, Light Y, Mak L, McGlinchey S et al. The ChEMBL bioactivity database: an update. Nucleic Acids Res 2014; 42: D1083-1090.