Annales UMCS Informatica AI 5 (2006) 101-111

Annales UMCS
Informatica
Lublin-Polonia
Sectio AI

http://www.annales.umcs.lublin.pl/

Automatic Keyphrase Extraction

Danuta Zakrzewska[*], Katarzyna Matańska
Institute of Computer Science, Technical University of Łódź,
Wólczańska 215, 93-005 Łódź, Poland

Abstract

Increasing number of documents in the Web caused the growth of needs for tools supporting automatic search and classification of texts. Keywords are one of characteristic features of documents that may be used as criteria in automatic document management. In the paper we describe the technique for automatic keyphrase extraction based on the KEA algorithm [1]. The main modifications consist in changes in the stemming method and simplification of the discretization technique. Besides, in the presented algorithm the keyphrase list may contain proper names, and the candidate phrase list may contain number sequences. We describe experiments, that were done on the set of English language documents available in the Internet and that allow for optimization of extraction parameters. The comparison of the efficiency of the algorithm with the KEA technique is presented.

1. Introduction

Huge number of documents that appear in the Web together with the growing searching needs of Internet users as well as the development of Internet applications supporting document management caused the growth of interest in automatic finding of characteristic features of documents. Keyphrases, which are short phrases that describe the main topics of texts may briefly characterize document goals and topics and may decide if the document belongs to the domain of reader's interests. In the area of automatically assigning keyphrases into documents by machine learning, there are two different approaches: keyphrase assignment and keyphrase extraction [2]. In the first technique texts are categorized by using the set of training documents. This paper deals with the second approach, which consists in automatic selection of important, topical phrases from within the body of the document [3].

There are different machine learning methods for automatic keyphrase extraction, described in literature so far, the most important of them will be presented in Section 2. In our investigations we will base on using Naïve Bayes machine learning algorithm for training and keyphrase extraction, KEA, the

[*]Corresponding author: *e-mail address*: dzakrz@ics.p.lodz.pl

technique that was introduced in [4]. The algorithm together with our modifications, which allow for simplifying and improving the technique, will be described in detail in Section 3. In Section 4 we will present some tests, done on the specially prepared English language documents, we will examine optimal choice of parameter values, as well as the efficiency of the considered algorithm. We will also compare efficacy of our algorithm with the KEA technique. Finally, some conclusions concerning future research and further modifications of the algorithm will be presented.

2. Keyphrase Extraction

There are two types of methods for solving problems of keyphrase extraction: dictionary based and corpus based methods. The first approach depends significantly on a stiff lexicon and sophisticated word segmentation rules. In the second approach keyphrase extraction systems are trained using a corpus of documents with corresponding free text keyphrases. Despite KEA algorithm, which will be described in the next section in detail, several automatic keyphrase extraction techniques, based on the second approach, have been proposed in literature so far.

Tourney in [3] introduced the GenEx keyphrase extraction system, consisting of a set of parameterized heuristic rules that are tuned to the training corpus by a genetic algorithm. A document is treated as a set of phrases, which must be classified as either positive or negative examples of keyphrases based on examination of their features.

Wu and Agogino [5] proposed a non-dominated sorting multi-objective genetic algorithm, whose goal is to find the optimal set of keyphrases that can represent the corpus, with the precision, measured by average condensation clustering [6], which reflects the degree of “clumping”of candidate phrases in the context.

Wang and Peng investigated keyphrase extracted approaches based on the backpropagation algorithm [7] and the least squares support vector machine [8]. In both cases they used not only term frequency and inverted document frequency, but also the phrase structural features to determine whether a phrase is a keyphrase or not.

Barker and Cornacchia [9] proposed an algorithm, which indicates noun phrases from a document as keyphrases. A noun phrase is chosen on the basis of its length, its frequency and the frequency of its head noun. Noun phrases are extracted from a text using a base noun phrase skimmer and an off-the-shelf online dictionary.

In spite of considerations concerning English language documents, Chien [10] presented a PAT-tree-based adaptive technique for domain-specific keyphrase extraction from the documents written in Chinese and other oriental languages, taking into consideration requirements of Internet utilization.

We focus our research on the KEA algorithm, based on the Naïve Bayes classification, which is still the subject of investigations and improvements (compare [11]). In the next section we describe the KEA technique and propose some changes into the algorithm, that are to simplify the procedure and to shorten the time of its performance.

3. Algorithm

The idea of KEA algorithm consists in generating candidate phrases by searching for sequences of consecutive words in the input document. Candidate phrases are cleaned and stemmed, then characteristic features are assigned to each of them. These features are basic elements of a classification model, which is used for determining if the candidate phrase may belong into the set of keyphrases or not. There are two features used in KEA: TFxIDF, which means a measure of a phrase frequency in a document compared to its rarity in general use; and the first occurrence, which signifies the distance into the document of the phrase first appearance. There are three main sets of documents used by KEA: training documents (with assigned keyphrases), global corpus, used for calculating frequency of phrases, and test documents [4].

KEA has two stages:

1) training, during which a model is built by using a training set of documents,
2) extraction, when keyphrases are chosen from a new document by applying the model created in the first stage.

Training (stage 1) can be divided into three main phases:

– identification of candidate phrases,
– calculation of features, and
– building the model.

3.1. Candidate phrases

Candidate phrases are obtained by three steps:

– cleaning (deleting punctuation marks, apostrophes, and any non-token characters) and normalizing in a form of a set of lines, each sequence of tokens containing at least one letter,
– stemming, and
– identification of phrases, by applying simple and effective rules, such as limiting candidate phrases, to a certain maximum length, eliminating those which begin or end with a stopword.

Stemming which means reducing a word to its stem or root form, allows to treat different variations on a phrase as the same thing. Witten et al. [4] in their algorithm, applied the iterated Lovins' method which consists in using the classic Lovins' stemmer [12] to discard any suffix and repeating the process on the stem that remains until there is no further change. Such approach allows for obtaining stems of high quality, but only in the case of using large rule sets which, in turn, results in long duration of the process. In our implementation, we base on the idea introduced in [13], by using the file with an explicit list of suffixes and removing them from the words to leave a valid stem. This approach as very simple enables a stemming process to be very fast and gives unexpectedly good results which will be seen in the next section.

3.2. Feature calculation

TFxIDF compares the frequency of phrase use in a document with the frequency of that phrase in general use, where general usage is represented by the number of documents containing the phrase in the corpus. This feature has bigger value for phrases which are frequent in the document and rare in the corpus. TFxIDF for phrase P in document D is:

$$TFxIDF = \frac{freq\left(P,D\right)}{size\left(D\right)} \times -\log_2 \frac{df\left(P\right)}{N},$$

where *freq(P,D)* is the number of times P occurs in D, *size(D)* is the number of words in D, *df(P)* is the number of documents containing P in the global corpus, and *N* is the size of the global corpus. The second term in the equation is the log of the probability that this phrase appears in any document of the corpus.

The second feature, first occurrence is the number of words that precede the phrase appearance, divided by the number of words in the document. The result has a value between 0 and 1 and informs how much of the document precedes the first phrase appearance.

As both features are real numbers, before creating the classification model it is necessary to convert them into the nominal data. To this effect discretization of obtained values is necessary. Witten et al. [4] applied multi-interval supervised discretization [14], which is rather complicated and time consuming, and is not recommended in the case of large training data sets [15]. In our approach the fixed k-interval discretization technique, which consists in dividing sorted values of numeric attributes into k intervals, is used. The proposed method, however, is very simple, works surprisingly well for Naïve Bayes classifiers [15] which will be easily in the analysis of test results in the next section.

3.3. Building the model

During this phase classification the schema is created, on the basis of candidate phrases for the training set of documents as well as on the basis of their feature values. To reduce the size of the training set, phrases that occurred only once in the document are discarded.

The extraction model is determined by the probability of being the keyphrase by a certain candidate phrase. Taking into account TFxIDF and the first occurrence value, Naïve Bayes classifier decides if the phrase is a keyphrase or a non-keyphrase on the basis of probability values, that may be presented in the normalized form:

$$P\left[Yes/T,d\right]=\frac{P\left[T/Yes\right]\cdot P\left[d/Yes\right]\cdot P\left[Yes\right]}{P\left[T,d\right]},$$

where T means the discrete value of TFxIDF, d signifies the first discrete occurrence value, *P*[*T*/*Yes*] is the probability that the candidate phrase with TFxIDF = T, is a keyphrase; similarly *P*[*d*/*Yes*] is the probability that the candidate phrase with first occurrence d, is a keyphrase.

The above criterion is applied many times during the extraction process. Good quality of the obtained results may be achieved by using training documents from the same domain as the examined texts [2].

3.4. Extraction stage

In this stage candidate phrases are ranked depending on the Naïve Bayes classifier value, according to the following rules:

– if two phrases have the same probability value, the phrase with the greater TFxIDF value is chosen,
– every phrase, that is a subphrase of a higher-ranking phrase is removed,
– required number of k first phrases from the ranking is regarded as keyphrases.

The performance and efficiency of the algorithm were examined during the experiments, the results of which, together with the comparison of the effects of KEA, will be presented in the next section.

4. Experiments

The evaluation of the algorithm was done by experimenting. Similarly to the assessment of KEA technique efficiency, the quality of the obtained results was measured by comparing keyphrases found by the implemented system and the keyphrases indicated by documents authors. Experiments, for both algorithms, were done on the basis of the same documents collections to enable reliable comparison. The following data sets were used during the tests:

– global corpus of 50 documents was created from the texts available at http://www.gutenberg.org/,
– the set of 80 training documents, with keyphrases assigned, created from the documents available at http://www.nzdl.org/Kea/,
– the set of 80 test documents available at http://www.nzdl.org/Kea/,

– the files of "stop lists", with exceptions (194 elements), word endings (44 elements) chosen on the basis of Oxford Word Power dictionary and stop words (69 elements). The last file was extended during experiments.

The experiments were divided into two stages. In the first one, optimal values of parameters were chosen, in the second step the efficiency of the algorithm and the comparison with KEA performance were investigated.

4.1. Optimal choice of parameters

Number of intervals in a discretization technique

The K-interval discretization technique, requires the number k to be determined *a priori*. In the first stage of our investigations, we examine the influence of the choice of the number k on the obtained results. Tests were done for different maximal numbers of words in keyphrases and the number of keyphrases sought by the algorithm. Qualitative (expert) analysis of the results showed that, taking into account the values from 3 to 10, the last one guaranteed the best and satisfactory effects. Further increasing of the parameter caused the significant growth of the run time of the algorithm without substantial improvements of the results. Table 1 shows the effects for different numbers of intervals with the maximal number of words in keyphrases equal to two and the number of keyphrases equal to 5 for the randomly chosen document.

Table 1. Results of kephrases searching for different numbers of intervals in the discretization algorithm

Number of intervals = 3

Obtained Keyphrases	Probability value	TFxIDF
hierarchi	0.00059	0.32
loc propag	0.00009	0.25
algorithm	0.00009	0.25
constraint hierarchi	0.00009	0.19
multiwa equalit	0.00009	0.13

Number of intervals = 5

hierarchi	0.00056	0.32
constraint hierarchi	0.00026	0.19
algorithm	0.00024	0.26
loc propag	0.00024	0.26
generaliz loc	0.00007	0.13

Number of intervals = 7

hierarchi	0.000767	0.32
constraint hierarchi	0.000254	0.19
generaliz loc	0.000231	0.13

algorithm	0.000230	0.26
multiwa equalit	0.000162	0.13

Number of intervals=10

hierarchi	0.000631	0.32
constraint hierarchi	0.000350	0.19
loc	0.000222	0.21
algorithm	0.000216	0.26
loc propag	0.000137	0.26

Keyphrases presented in the first column have the stem form. It can be easily seen that probability values are of comparable range for the parameter values equal to 7 and 10, however, the last choice guaranteed consistence with the expert's opinion.

Number of words in keyphrases

Next research consists in examining the efficiency of the algorithm for different maximal numbers of words in keyphrases. We considered two- and three-words phrases, omitting single words. Table 2 presents the effects of the algorithm, on the randomly chosen text, for the number of discretization intervals equal to 10, and the number of searched keyphrases equal to 5.

Table 2. Results of the algorithm for different number of words in keyphrases

Maximal number of words = 2

Keyphrases	Probability	TFxIDF
attribut	0.00060	0.34
continuousvalu attribut	0.00055	0.42
discretiz continuousvalu	0.00051	0.42
aha	0.00012	0.27
learn	0.00003	0.03

Maximal number of words=3

continuousvalu attribut	0.00052	0.25
discretiz continuousvalu attribut	0.00040	0.25
aha	0.00026	0.25
learn	0.00003	0.0

The increasing number of words does not change significantly the quality of effects of the algorithm, however, it enables finding three-words keyphrases in the case of their presence. In the example presented in Table 2, the algorithm extracted only four keyphrases, because all the subphrases with lower ranking were omitted and that situation must have also happened to the phrases: "continuousvalu attribut" and "attribut".

Number of keyphrase extracted

The aim of this investigation was to check how the number of extracted keyphrases influence the performance of the algorithm. Table 3 shows the results of extracting 5 and 10 keyphrases, in the case of number of intervals for discretization equal to 10, and maximum number of words equal to 3.

Table 3. Results for different numbers of keyphrases

Number of keyphrases extracted = 5

Keyphrase	Probability value	TFxIDF
certificate	0.00039	0.14
public key	0.00033	0.17
atm	0.00033	0.17
certificate path	0.00016	0.10
authenticat	0.00016	0.07

Number of keyphrases extracted = 10

certificate	0.000388	0.143149
public key	0.00033	0.167007
atm	0.00033	0.167007
certificate path	0.000161	0.095433
authenticat	0.000158	0.071574
signal mess	0.000158	0.071574
atm network	0.000158	0.071574
call part	0.000139	0.095433
public key certificat	2.83E-05	0.047716
two parti	2.83E-05	0.04582

The analysis of the obtained results showed that the number five to seven keyphrases is the best for characterizing the document. In the case presented in Table 3, seven keyphrases seem to be optimal according to expert's opinion, "atm network" for example better represents the text subject than single word "atm", however, the probability value is significantly bigger for the eight keyphrases.

4.2. Efficiency of the algorithm

The efficiency of the algorithm was tested for the maximal number of words equal to 3, number of extracted keyphrases equal to 10 and number of intervals in the discretization process equal to 10. Evaluation of the obtained results was done by comparing the effects of the algorithm with the keyphrases assigned by experts. Table 4 presents the comparison of the results for the exemplary document.

Table 4. Comparison of effects obtained by expert's assigning and by algorithm

Expert's choice	Extracted phrase
multicast algorithms	multicast
causal multicast algorithms	multicast algorithm
multicasting	network
ATM network	caus ord multicast
partition method	caus multicast algorithm
	call caus ord
	atm network
	multimedia application
	cost
	messag

The satisfactory results are obtained by choosing 5-7 keyphrases. 80% of keyphrases allocated by expert can be found in the second column. Only the phrase "partition method" was omitted by the algorithm, but it is worth mentioning that this phrase occurred in the text only once.

Taking into account the results of the algorithm on the whole set of the documents, the average number of matches with expert's opinion was about 50 %. The quality of the obtained results improves if the domain of the training documents is the same as for those tested. Similarly to KEA, the increasing number of texts included in global corpus as well as the increasing number of training documents have no significant influence on the quality of results (see [4]).

Table 5. Comparison of KEA and the modified algorithm effects on the sample document

KEA	Modified algorithm
checkpoints	checkpoint
global checkpoints	glob checkpoint
consistent global checkpoints	consist glob checkpoint
finding consistent global	find consist glob
local checkpoints	loc checkpoint
	zigzag path
	pres
	s
	introduc

Finaly, comparison of efficiency of the considered algorithm with KEA, showed that the obtained results for both techniques are of similar quality. Table 5 presents the exemplary results for the same document, with all the phrases extracted by KEA, which are complemented with suffixes.

The detailed analysis of results indicates that both algorithms perform quite well. KEA can on average match between one and two of the five keyphrases chosen by the authors [4], while our modified algorithm, displays average match of two keyphrases, which is really a very good result taking into account simplifications of techniques used in the algorithm.

The authors of KEA enhanced their algorithm in adding the suffixes into stemmed words, which helps in displaying results in more understandable form and may give the impression of better performance, however this feature does not influence the efficiency of the algorithm.

Conclusions

In the paper, the modifications of Automatic Keyphrase Extraction (KEA) algorithm, whose aim was to simplify and shorten the procedure, were described. The improvements are connected with two techniques: stemming and discretization. In the first case a very simple method based on using the file with an explicit list of suffixes is applied. The second modification consists in using the k-fixed interval discretization method recommended for the Naïve Bayes classification [16], which is much simpler than the Fayyad & Irani's entropy minimization discretization algorithm used by KEA. Both of the modifications simplified the algorithm without loss of the quality of results, what's more in several cases we obtained more matches with the author's choices than by using KEA.

Future research will consist in further work on the stemming technique by using the Porter's algorithm, which enhances the procedure by using a set of rules for suffixes [17]. It is also worth considering to add more features that will be used in the keyphrase classification. Some research in this area has already been done, there were used such factors as frequency and coherence features and their combination [11].

The presented approach for keyphrases extraction may be used not only for documents management purposes, but also for automatic summarizing, indexing, labeling, categorizing, clustering, browsing and searching of web sites, but this application requires more research to be done in this area of expertise.

References

[1] Witten I.H., Paynter G.W., Frank E.T., Gatwin C.A., Nevill-Manning C.G., *KEA: practical automatic keyphrase extraction,* Proceedings of Fourth ACM Conference on Digital Libraries (DL' 99), (Eds: Fox E.A., & Rowe N.), Berkley, CA, ACM Press, New York, (1999) 254.

[2] Frank E., Paynter G.W., Witten I.H., Gatwin C.A., Nevill-Manning C.G., *Domain specific keyphrase extraction,* Proceedings of the Sixteenth International Joint Conference on Artificial Intelligence (IJCAI-99), Morgan Kaufmann, California, (1999) 668.
[3] Turney P.D., *Learning algorithms for keyphrase extraction,* Information Retreaval, 2 (2000) 303.
[4] Witten I.H., Paynter G.W., Frank E.T., Gatwin C.A., Nevill-Manning C.G., *KEA: practical automatic keyphrase extraction,* Working Paper 00/5, Department of Computer Science, The University of Waikato, (2000).
[5] Wu J.-L., Agogino A., *Automatic keyphrase extraction with multi-objective genetic algorithms,* Proceedings of the 37th Hawaii International Conference on System Sciences, IEEE, (2004).
[6] Bookstein A., Klein S.T., Raita T., *Clumping properties of content-bearing words,* Journal of the American Society for Information Science, 49(2) (1998) 102.
[7] Wang J., Peng H., Hu J.-S., *Automatic keyphrases extraction from document using backpropagation,* Proceedings of the Fourth International Conference on Machine Learning and Cybernetics, Guangzhou, China, (2005) 3770.
[8] Wang J., Peng H., *Keyphrases extraction from web document by the least squares support vector machine,* Proceedings of the 2005 IEEE/WIC/ACM International Conference on Web Intelligence (WI'05), (2005).
[9] Barker K., Cornacchia N., *Using noun phrase heads to extract document keyphrases,* Proceedings of the Thirteenth Canadian Conference on Artificial Intelligence 1822, Springer-Verlag, Berlin, (2000) 40.
[10] Chien L.F., *PAT-tree based adaptive keyphrase extraction for intelligent Chinese information retrieval*, Information Processing and Management, 35 (1999) 501.
[11] Turney P., *Coherent keyphrase extraction via Web Mining,* Proceedings of the Eighteenth International Joint Conference on Artificial Intelligence, Acapulco, Mexico, (2003) 434.
[12] Lovins J.B., *Development of a stemming algorithm,* Mechanical Translation and Computational Linguistics, 11 (1968) 22.
[13] Petrarca A.E., Lay W.M., *Use of an automatically generated authority list to eliminate scattering caused by some singular and plural main index terms,* Proceedings of the American Society for Information Science, 6 (1969) 277.
[14] Fayyad U.M., Irani K.B., *Multi-interval discretization of continuous-valued attributes for classification learning,* Proceedings of the 13th Joint Conference on Artificial Intelligence, Morgan Kaufmann, (1993) 1022.
[15] Dougherty J., Kohavi R., Sahami M., *Supervised and unsupervised discretization of continuous features,* Proceedings of the Twelfth International Conference on Machine Learning, (1995) 194.
[16] Yang Y., Webb G.I., *Weighted proportional k-interval discretization for Naïve- Bayes classifiers,* Proceedings of PAKDD'03, (2003) 501.
[17] Porter M.F., *An algorithm for suffix stripping,* Program, 14(3) (1980) 130.