Thesaurus-Based Index Term Extraction for Agricultural Documents

Olena Medelyan,[a] Ian H. Witten[b]

Department of Computer Science, The University of Waikato, Private Bag 3105, Hamilton, New Zealand

[a] *olena@coling.uni-.freiburg.de,* [b] *ihw@cs.waikato.ac.nz*

Abstract

This paper describes a new algorithm for automatically extracting index terms from documents relating to the domain of agriculture. The domain-specific Agrovoc thesaurus developed by the FAO is used both as a controlled vocabulary and as a knowledge base for semantic matching. The automatically assigned terms are evaluated against a manually indexed 200-item sample of the FAO's document repository, and the performance of the new algorithm is compared with a state-of-the-art system for keyphrase extraction.

Key words: automatic indexing, machine aided indexing, keyphrase extraction, keyphrase assignment.

1 Introduction

Keyphrases are widely used in both physical and digital libraries as a brief but precise summary of documents. They help organize material based on content, provide thematic access, represent search results, and assist with navigation. In some contexts keyphrases are freely chosen; in others they are restricted to terms that occur in a controlled vocabulary or thesaurus. In either case assigning high-quality keyphrases to documents manually is expensive, because for best results professional human indexers must read the full document and select appropriate descriptors according to defined cataloguing rules.

There are two approaches to automating the identification of keyphrases. In keyphrase *extraction*, the phrases that occur in the document are analyzed to identify apparently significant ones, on the basis of intrinsic properties such as frequency and length. In keyphrase *assignment*, which might more properly be called "index term assignment" because keyphrases are not constructed as free text but are chosen from a controlled vocabulary of terms, documents are classified according to their content into classes that correspond to elements of the vocabulary. One serious disadvantage of the extraction approach is that the extracted phrases are often ill formed or inappropriate. The assignment approach circumvents this problem, but for satisfactory results a very large and accurate corpus of training material is needed, which must be produced by manually classifying training documents.

This paper describes a new approach that we call *index term extraction*, which is intermediate between keyphrase extraction and term assignment. It combines the advantages of both, while avoiding their shortcomings. A domain-specific thesaurus is used as the controlled vocabulary, and the relationship between manually assigned index terms and the phrases that actually appear in the document text are explored using the semantics implicit in the thesaurus. A machine-learning model is trained based on features from this relationship, along with other characteristics of extracted phrases that are used in conventional automatic keyphrase extraction. Given a test document, vocabulary terms are assigned to it by mapping all the document's phrases to those in the thesaurus and using the learned model to decide which ones are significant descriptors of the document's content. The resulting set contains only well-formed phrases from the thesaurus that are strongly related to the given document.

2 Related work

Space permits no more than a cursory glimpse at related work. Keyphrase extraction has been investigated by Witten *et al.* (1999), who developed a system called KEA on which the present work is based. Turney (1999) developed a similar system, called GenEx, and, more recently, Hulth (2004) has explored a number of different approaches. There has been some evaluation of these systems using human judges (Jones and Paynter, 2003; Barker and Cornacchia, 2000). Keyphrase assignment uses methods of text classification (Sebastiani, 2002), and has been investigated by Dumais *et al*. (1998), who obtained the best performance from support vector machines. Markó *et al*. (2003) developed a system for keyphrase assignment that combines a sophisticated approach to orthographical, morphological and semantic term normalization with a hybrid weighting technique. Paice and Black (2003) describe an interesting combination of keyphrase indexing and information extraction; they introduced the notion of pseudo phrase that is central to the method described below.

3 The term extraction algorithm

The new term extraction algorithm, called KEA++ because it improves on the original keyphrase assignment algorithm KEA, combines the positive elements of keyphrase extraction and term assignment into a single scheme. It is based on machine learning and works in two main stages: *candidate identification*, which identifies candidate terms that relate to the document's content (including ones that appear verbatim), and *filtering*, which uses a learned model to identify the most significant terms based on certain properties or "features." The filtering operation involves two phases: learning a suitable model based on training data, and applying it to new test documents. The method of candidate identification is described first, and then features are defined that are used to characterize the phrases. Next we explain how the model is learned, based on training data, and finally show how it is applied to fresh test documents.

3.1 Candidate identification

Each document in the collection is segmented into individual tokens on the basis of white space and punctuation. Lexical clues to syntactic boundaries such as punctuation marks, digits and paragraph separators are retained. Words that contain numeral characters are discarded, since none of the descriptors in the controlled vocabulary contain digits. Next all concatenations of one, two and three words—that is, word n-grams—that do not cross phrase boundaries are extracted, and the number of occurrences of each n-gram in the document is counted. The n-grams are restricted to three words because that is the maximum length of index terms in the controlled vocabulary,

Most extracted n-grams are ungrammatical or meaningless in isolation, and some selection is essential. Unlike other keyphrase extraction algorithms, which use rough heuristics based on punctuation and stopwords, or syntactic processing and phrase identification, KEA++ selects n-grams with reference to a controlled vocabulary. To achieve the best possible matching and also to attain a high degree of conflation, each n-gram is transformed into a *pseudo phrase* in three steps:

- remove all stopwords from the n-gram
- stem the remaining terms to their grammatical roots
- sort them into alphabetical order.

This matches similar phrases such as "algorithm efficiency", "the algorithms' efficiency", "an efficient algorithm" and even "these algorithms are very efficient" to the same pseudo phrase "algorithm effici", where "algorithm" and "effici" are the stemmed versions for the corresponding full forms. We experimented with stemming algorithms developed by Lovins (1968) and Porter (1980). The technique of alphabetic sorting was proposed by Paice and Black (2004).

In the next step each pseudo phrase is matched against vocabulary terms, also represented as pseudo phrases. If they are the same, the n-gram is identified with the corresponding vocabulary term. Each vocabulary term receives an occurrence count which is the sum of the occurrence counts of its associated n-grams.

For semantic term conflation, non-descriptors are replaced by their equivalent descriptors using the links in the thesaurus (these are called USE-FOR links in the Agrovoc thesaurus employed in this work). The occurrence counter of the corresponding descriptor is increased by the sum of the counts of all its associated non-descriptors. This operation recognizes terms whose meaning is equivalent, and greatly extends the usual approach of conflation based on word-stem matching.

The result is a set of candidate index terms for a document, and their occurrence counts. As an optional extension, the set is enriched with all terms that are related to the candidate terms, even though they may not correspond to pseudo-phrases that appear in the document. For each candidate its one-path related terms, i.e. its hierarchical neighbors (BT and NT in Agrovoc), and associatively related terms (RT), are included. If a term is related to an existing candidate, its occurrence count is increased by that candidate's count. For example, suppose a term appears in the document 10 times and is one-path related to 6 terms that appear once each in the document and to 5 that do not appear at all. Then its final frequency is 16, the frequency of the other terms that occur is 11 (since the relations are bidirectional), and the frequency of each non-occurring term is 10. This technique helps to cover the entire semantic scope of the document, and boosts the frequency of the original candidate phrases based on their relations with other candidates.

In both cases—with and without related terms—the resulting candidate descriptors are all grammatical terms that relate to the document's content, and each has an occurrence count. The next step is to identify a subset containing the most important of these candidates.

3.2 Feature definition

A simple and robust machine-learning scheme is used to determine the final set of index terms for a document. It uses as input a set of attributes, or "features," defined for each candidate term. The following features have been considered for inclusion.

The **TF×IDF** score compares the frequency of a phrase's use in a particular document with the frequency of that phrase in general use. General usage is represented by *document frequency*—the number of documents containing the phrase in the document collection. KEA++ creates a document frequency file that stores each pseudo-phrase and a count of the number of documents in which it appears. With this file in hand, the TF×IDF for phrase P in document D is:

$$\text{TF}\times\text{IDF} = \frac{\text{freq}(P, D)}{\text{size}(D)} \times -\log_2 \frac{\text{df}(P)}{N},$$

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 (negated because the probability is less than one). This score is high for rarer phrases that are more likely to be significant.

The **position of the first occurrence** of a term is calculated as the distance of a phrase from the beginning of a document in words, normalized by the total number of words in the document. The result represents the proportion of the document preceding the phrase's first appearance.

Number of words in a candidate phrase is another feature we want to consider. Our experiments revealed that indexers prefer to assign descriptors consisting of two words, whereas one word terms are the majority in Agrovoc. Using phrase length in words as a feature boosts the probability of two-word candidates being keyphrases.

Node degree reflects how richly the term is connected in the thesaurus graph structure. The "degree" of a thesaurus term is the number of semantic links that connect it to other terms—for example, a term with one broader term and four related terms has degree 5. We will consider three different variants:

- the number of links that connect the term to other thesaurus terms
- the number of links that connect the term to other candidate phrases
- the ratio of the two.

Appearance is a binary attribute that reflects whether the pseudo-phrase corresponding to a term actually appears in the document. Using the optional extension of candidate terms mentioned above, some candidate terms may not appear in the document.

Of these features, three (TF×IDF score, position of first occurrence, and node degree) are numeric, and the KEA++ algorithm discretizes these into nominal form for use by the machine-learning scheme. During the training process, a discretization table for each feature is derived from the training data. This table gives a set of numeric ranges for each feature, whose values are replaced by the range into which they fall. Discretization is accomplished using the supervised method of Fayyad and Irani (1993).

3.3 Training: Building the model

In order to build the model, a set of documents are used for which the author's keyphrases are known. For each training document, candidate pseudo-phrases are identified and their feature values are calculated as described above. To reduce the size of the training set, pseudo-phrases that occur only once in the document are discarded. Each phrase is then marked as an index term or not, using the actual index terms that have been assigned to that document by a professional indexer. This binary feature is the *class feature* used by the machine-learning scheme.

The scheme then generates a model that predicts the class using the values of the other included features. We have experimented with a number of different machine learning schemes; KEA++ uses the Naïve Bayes technique (e.g. Domingos and Pazzani, 1997) because it is simple and yields good results. This scheme learns two sets of numeric weights from the discretized feature values, one set applying to positive ("is an index term") examples and the other to negative ("not an index term") instances.

3.4 Extracting index terms from new documents

To select index terms from a new document, KEA++ determines candidate pseudo phrases and their feature values, and then applies the model built during training. The model determines the overall probability that each candidate is an index term, and then a post-processing operation selects the best set of index terms.

Suppose just the two features TF×IDF and position of first occurrence are being used. When the Naïve Bayes model is used on a candidate pseudo phrase with feature values t and f respectively, two quantities are computed:

$$\mathrm{P}[yes] = \frac{Y}{Y+N}\,\mathrm{P}_{TF\times IDF}[t \mid yes]\,\mathrm{P}_{distance}[f \mid yes]$$

and a similar expression for P[*no*], where Y is the number of positive instances in the training files—that is, author-identified keyphrases—and N is the number of negative instances—that is, candidate phrases that are not keyphrases. (The Laplace estimator is used to avoid zero probabilities. This simply replaces Y and N by Y+1 and N+1.)

The overall probability that the candidate phrase is a keyphrase can then be calculated:

$$p = \mathrm{P}[yes] \,/\, (\mathrm{P}[yes]+\mathrm{P}[no])$$

Candidate phrases are ranked according to this value, and two post-process steps are carried out. First, TF×IDF (in its pre-discretized form) is used as a tiebreaker if two phrases have equal probability (common because of the discretization). Second, any phrase that is a subphrase of a higher-ranking phrase is removed from the list. From the remaining ranked list, the first r phrases are returned, where r is the number of keyphrases requested. KEA++ can be used for both automatic and semi-automatic indexing, with a smaller value in first case (e.g. $r = 5$) and a longer list of ranked keyphrases, from which a human indexers selects the most appropriate ones.

4 Evaluation

The experiments in this project were conducted using an indexed document collection and thesaurus maintained by the UN Food and Agriculture Organization (FAO); these are described in the following section. Then the evaluation strategy is defined, and results for the candidate identification and filtering techniques implemented in KEA++ are presented.

4.1 Experimental Data

The FAO has a mandate to increase agricultural productivity and improve the conditions of rural populations worldwide. It collects, analyses, and disseminates information, and maintains an online document repository (www.fao.org/documents/) that is large and well used (1M hits/month). Documents are manually indexed with terms from the Agrovoc thesaurus (www.fao.org/agrovoc).

Agrovoc (1995) is domain-specific and contains 16,600 descriptors and 10,600 non-descriptors. It defines three semantic relations between descriptors: links between related terms (RT), which are bi-directional; and links between broader terms (BT) and narrower ones (NT), which are inverse. Figure 1 shows a typical entry in the printed edition of Agrovoc.

The training and evaluation material comprises 200 full-text documents that were downloaded randomly from the FAO's document repository. Each document contains around 17,000 words on average, ranging from 95 words to over 145,000, for a total of 3.45 million words in the collection. Each document was manually assigned an average of 5.4 Agrovoc descriptors, ranging from 2 and 15, for a total of 1080 term assignments—which includes just 495 different terms.

4.2 Measuring Indexing Quality

Automatic keyphrase extraction and assignment systems are usually evaluated by comparing the algorithm's keyphrase sets with manually assigned keyphrases. The number of matching ("correct") keyphrases is then expressed as a proportion of the number of all extracted phrases (*Precision P*) and of the number of manual assigned phrases (*Recall R*) for each document separately; the *F-measure* is a balanced combination of the two (van Rijsbergen, 1979). In other words,

$$P = \frac{\#\ correct\ extracted\ keyphrases}{\#\ all\ extracted\ keyphrases} \qquad R = \frac{\#\ correct\ extracted\ keyphrases}{\#\ manually\ assigned\ keyphrases} \qquad F\text{–}measure = \frac{2PR}{P+R}$$

A phrase is usually considered "correct" if its stemmed version equals any stemmed manually assigned keyphrase. However, we extend this by introducing two further levels of match, defined with respect to a given thesaurus. At one level, an automatically extracted keyphrase is considered "correct" if it is related by the thesaurus with a manually assigned term by any one-path relation (BT, NT or RT in Agrovoc). At the next, a keyphrase is considered "correct" if it is related to a manually assigned term by a two-path

En. Descriptor:	**Epidermis**
Scope Note:	*Of plants; for the epidermis of animals use SKIN*
En. BT:	**BT1** Plant tissues
	BT2 Plant anatomy
En. NT:	NT1 Plant cuticle
	NT2 Plant hairs
	NT3 Root hairs
	NT2 Stomata
En. RT:	RT Peel
Fr. Descriptor	Épiderme
Sp. Descriptor	Epidermis

Figure 1. Example entry in the Agrovoc thesaurus (Agrovoc 1995)

connection that involves the same relation (BT and BT, or NT and NT, or RT and RT). In other words, the three levels of "correctness" are:

- Level I: terms have equal pseudo-phrases, e.g. *epidermis* and *epidermal*.[1]
- Level II: terms have equal pseudo-phrases or are one-path related, e.g. *epidermis* and *peel*, or *plant hairs* and *root hairs*.
- Level III: terms have equal pseudo-phrases or are one- or two-path related, e.g. *plant cuticles* and *root hairs*.

Traditionally, the overlap between keyphrase sets is measured at level I: this represents *terminological consistency*. Levels II and III represent semantic similarity between keyphrases and corresponds to *conceptual consistency* (Iivonen, 1995). To evaluate KEA++, precision, recall and F-measure are computed on all three levels, and the extended version of the algorithm is expected to achieve higher conceptual than terminological consistency.

4.3 Evaluation of candidate identification

The results are presented separately for the candidate identification process, which uses pseudo-phrase normalization, and the filtering process, which uses the learned model to select index terms from the candidates. The candidates extracted from each of the 200 documents are compared with manually assigned index terms.

Table 1 summarizes the results of candidate identification as implemented by KEA, representing the current state of the art, and two versions of the new algorithm KEA++. We first compared the effect of the iterated Lovins (1968) and Porter (1980) stemmers in all three cases. For KEA the best results were achieved with iterated Lovins, while KEA++ worked better with Porter, and in each case. Table 1 shows the results using the best stemmer. The third row shows the results obtained when the candidate sets were extended with related terms from Agrovoc, as described in Section 3.1. The numbers in bold give the best (i.e., highest) results in each column.

KEA extracts over 14 times more candidate phrases than KEA++ (unextended version). However, the number of "correct"[2] terms is almost the same in both cases. Thus KEA++'s precision is almost a tenfold improvement on KEA's, although recall is slightly lower. The extended version of KEA++, which includes one-path related terms, increased recall dramatically. Although its precision of this technique is lower than KEA++'s, it is more than twice as high as KEA's. The recall values give a baseline for the best possible results that could possibly be achieved by the filtering stage.

4.4 Evaluation of the filtering technique

To evaluate the filtering techniques 10-fold cross-validation (Witten and Frank, 2000) is used, which partitions the document collection randomly into 10 sets and performs 10 separate evaluation runs, each one training on 180 documents and testing on the remaining 20 documents. The results are averaged over all documents and all runs.

TF×IDF and position of the first occurrence were the features used by the original KEA algorithm. We now compare the performance of the two systems using these features. Again, iterated Lovins stemming is used with KEA because it gives the best results, and Porter stemming is used with KEA++ for the same

Table 1. Performance of candidate identification in KEA and KEA++

	# candidates	# manual keyphrases	# correct candidates	Precision	Recall
KEA	5766.8	5.37	3.98	0.14%	76.1%
KEA++	407.6	5.38	3.96	**1.34%**	75.35%
Extended KEA++	1765	5.38	**4.99**	0.37%	**92.99%**

[1]Terms are normalized using Agrovoc, so it is unnecessary to use pseudo-phrases here. We do so to ensure fair comparison with the KEA system, which does not use a controlled vocabulary.
[2]We have evaluated only Level I matching for this experiments, i.e. by using the pseudo-phrase technique.

Table 2. Overall performance of KEA and KEA++

	Level I			Level II			Level III		
	P	R	F	P	R	F	P	R	F
KEA	13.28	12.43	11.97	17.94	16.02	15.41	21.92	20.02	19.48
KEA++	**20.53**	**19.7**	**18.71**	31.06	28.09	27.77	45.58	41.00	40.18
Extended KEA++	13.11	12.1	11.81	**34.33**	**31.40**	**30.56**	**52.99**	**47.90**	**46.89**

reason (although in this case the difference is not significant at the 5% level according to a one-tailed paired t-test ($p > 0.15$)). Table 2 summarizes the precision P, recall R, and F-measure F that were achieved. KEA has a parameter that specifies a minimum occurrence frequency for terms: this was set to 2 in both cases (the default for KEA). For both systems the 5 highest-scoring phrases were selected as the automatic keyphrase set. Again the best results are in bold in Table 2.

At Level I, where thesaurus links are not taken into account in the evaluation, there is no advantage in artificially augmenting the candidate set with related terms as the extended version of KEA++ does, and this is confirmed by the evaluation. The main result is that the basic KEA++ roundly outperforms the original KEA, achieving level of recall, precision, and F-measure that are all over 1.5 times as high. Table 2 gives overall recall figures for the candidate identification and filtering processes together: the recall of the filtering technique alone is obtained by expressing the values in the table in terms of the baseline determined in Section 4.3.1. The revised figures are a recall of 17% of the baseline for KEA and 26% for KEA++, while the extended version of KEA++ fares even worse than KEA (13%) in terms of its improvement over baseline recall.

Levels II and III take account of the conceptual similarity between keyphrase sets. Here the improvement of KEA++ over KEA is even more significant, and the extended version of KEA++ yields an even greater improvement. At Level II, which takes account of one-path relations between automatic and manual keyphrases, KEA++'s precision, recall, and F-measure values are over 1.7 as high as those of KEA, and for the extended version the figures are almost double those of KEA. At Level III, which takes into account more distant relationships between terms, the improvement is even greater, and the values achieved by KEA++ on Level III for all three performance indicators are almost 2.5 times those for KEA. In other words, only one out of five phrases extracted by KEA is related to any manually assigned index term by a two-path link, while around half of KEA++'s terms are closely related to ones assigned by professional indexers.

When professional indexers independently assign keyphrases to the same documents, their sets differ. Studies have shown that typical measures of inter-indexer consistency range from 4% to 67% (with an average of 27%) for free indexing, and from 13% to 77% (with an average of 44%) when a controlled vocabulary is used (Leininger, 2000).[3] It is normal for indexers to disagree!—and this should be considered when evaluating automatic keyphrase assignment systems. Compared to the state-of-the-art system KEA, the new algorithm KEA++ performs significantly better. Its consistency with human indexers has not yet been compared to the consistency of human indexers with each other.

5 Conclusions and future work

This paper has developed the idea of "index term extraction" for thesaurus-based indexing of documents. This new approach to keyphrase extraction uses a machine learning technique and semantic information about terms encoded in a structured controlled vocabulary. The main advantage over conventional keyphrase *extraction* is the use of a controlled vocabulary, which eliminates the occurrence of meaningless or obviously incorrect phrases, and also yields a dramatic improvement in performance, as shown above. The main advantage over conventional keyphrase *assignment*, which already use a controlled vocabulary, is a dramatically lowered requirement for training data—performance is independent of the size of the controlled vocabulary size, and all experiments have been conducted with just 180 training documents—whereas keyphrase assignment using Agrovoc's vocabulary of 16,600 descriptors would require perhaps 100,000 training documents.

[3]Of course, the size of the controlled vocabulary has a significant effect. The larger it is, the lower the probability that indexers will assign same keyphrases.

This is work in progress, and much remains to be done. Only two features have been used of the five identified in Section 3.2; preliminary evaluation using the other features has suggested that they will yield even better performance. An important evaluation step would be to compare KEA++ with several professional indexers who have assigned keyphrases to the same set of documents independently. Previous studies of inter-indexer consistency have not taken account of semantic relations between assigned keyphrases, and we believe that this is necessary for a fair evaluation. Such experiments would provide useful insight into how similar professional indexers are to each other when determining a document's relevant concepts, and how well automatic indexing can perform the same task. Finally, only one controlled vocabulary, the Agrovoc thesaurus, has been used. Further work to extend the system to other structured indexing vocabularies and other domains would be an interesting extension of the project.

Even in its present form, with only a small volume of training data, KEA++ assigns index terms that may or may not appear in the document but are all strongly related to its content. The system is domain-independent and language-independent,[4] and can be used with any controlled vocabulary that contains hierarchical links and other relationships between index terms. Preliminary evaluations using a collection consisting of 200 manually indexed documents has shown even the prototype version of KEA++ significantly outperforms the state-of-the art keyphrase extraction algorithm KEA. It increases precision and recall using a similar candidate identification strategy and the same features for final classification. We conclude that automatic index term extraction from a controlled vocabulary has great advantages over pure keyphrase extraction.

References

Agrovoc. 1995. Multilingual Agricultural Thesaurus. Food and Agricultural Organization of the United Nations.

Barker, K., Cornacchia N. 2000. Using noun phrase heads to extract document keyphrases. In Proceedings of the 13th Canadian Conference on Artificial Intelligence, pp. 40–52.

Domingos, P. and Pazzani, M. 1997. On the optimality of the simple bayesian classifier under zero-one loss. Machine Learning, 29 (2/3), pp. 103-130.

Dumais, S.T., Platt, J., Heckerman, D., Sahami, M. 1998. Inductive learning algorithms and representations for text categorization. In Proceedings of ACM-CIKM98, pp. 148–155.

Fayyad, U.M. and Irani, K.B. 1993. Multi-interval discretization of continuous-valued attributes for classification learning. Proceedings of IJCAI-1993, pp. 1022-1027.

Hulth, A. 2004. Combining Machine Learning and Natural Language Processing for Automatic Keyword Extraction. Ph. D. thesis, Department of Computer and Systems Sciences, Stockholm University.

Iivonen, M. 1995. Consistency in the selection of search concepts and search terms. Information Processing and Management, Vol. 31, No. 2, pp. 173-190; March-April.

Jones, S. , Paynter, G. W. 2002. An evaluation of document keyphrase sets. Journal of Digital Information. 4(1).

Leininger, K. 2000. Inter-indexer consistency in PsycINFO. Journal of Librarianship and Information Science 32(1), pp. 4-8.

Lovins, J.B. 1968. Development of a stemming algorithm. Mechanical Translation and Computational Linguistics 11, pp. 22–31.

Markó, K., Daumke, P., Schulz, S., Hahn, U. 2003. Cross-language mesh indexing using morphosemantic normalization. Proceedings of AIMIA-2003, pp. 425–429.

Paice, C., Black, W. 2003. A three-pronged approach to the extraction of key terms and semantic roles. Proceedings of RANLP-2003, Recent Advances in Natural Language Processing.

van Rijsbergen, C.J. 1979. Information Retrieval. Butterworths, London.

Porter, M.F. 1980. An algorithm for suffix stripping. Program 14(3), pp. 130-137.

Sebastiani, F. 2002. Machine learning in automated text categorization. ACM Computing Surveys 34(1), pp. 1-47.

Turney, P. 1999. Learning to extract keyphrases from text. Technical report, National Research Council Canada.

Witten, I.H. and Frank, E. 1999. Data mining: Practical machine learning tools and techniques with Java implementations. Morgan Kaufmann, San Francisco, CA.

Witten, I.H., Paynter, G.W., Frank, E., Gutwin, C., Nevill-Manning, C.G. 1999. Kea: Practical automatic keyphrase extraction. Proceedings of the Fourth ACM Conference on Digital Libraries, pp. 254–255. Berkeley, CA: ACM Press.

[4]Only the stopword file need be adjusted.