Keyword Extraction Using Support Vector Machine

Kuo Zhang, Hui Xu, Jie Tang, and Juanzi Li

Department of Computer Science and Technology, Tsinghua University
Beijing, P.R.China, 100084
{zkuo99, xuhui99, j-tang02}@mails.tsinghua.edu.cn,
ljz@keg.cs.tsinghua.edu.cn

Abstract. This paper is concerned with keyword extraction. By keyword extraction, we mean extracting a subset of words/phrases from a document that can describe the 'meaning' of the document. Keywords are of benefit to many text mining applications. However, a large number of documents do not have keywords and thus it is necessary to assign keywords before enjoying the benefit from it. Several research efforts have been done on keyword extraction. These methods make use of the 'global context information', which makes the performance of extraction restricted. A thorough and systematic investigation on the issue is thus needed. In this paper, we propose to make use of not only 'global context information', but also 'local context information' for extracting keywords from documents. As far as we know, utilizing both 'global context information' and 'local context information' in keyword extraction has not been sufficiently investigated previously. Methods for performing the tasks on the basis of Support Vector Machines have also been proposed in this paper. Features in the model have been defined. Experimental results indicate that the proposed SVM based method can significantly outperform the baseline methods for keyword extraction. The proposed method has been applied to document classification, a typical text mining processing. Experimental results show that the accuracy of document classification can be significantly improved by using the keyword extraction method.

1 Introduction

Keywords summarize a document concisely and give a high-level description of the document's content. Keywords provide rich semantic information for many text mining applications, for example: document classification, document clustering, document retrieval, topic search, and document analysis.

Unfortunately, a large portion of documents (on internet and intranet) still do not have keywords assigned. In order to enjoy the benefit of keywords, it is necessary to extract keywords from the documents. This is exactly the problem addressed in this paper.

Existing methods on keyword extraction have been done mainly by using a predefined controlled-vocabulary, which cannot process the unknown words/phrases. In natural language processing, base Noun Phrase (baseNP) finding and key-term recognition have been studied. But the extracted baseNPs are not necessary to be keywords

J.X. Yu, M. Kitsuregawa, and H.V. Leong (Eds.): WAIM 2006, LNCS 4016, pp. 85 – 96, 2006.

of the document. Recently, several methods are proposed to extract keywords by utilizing the 'global context information'. The global information includes term frequency, term inverted document frequency, etc. This kind of global information ignores the term's local context information and makes the extraction performance limited. In the research community, no previous study has so far sufficiently investigated the problem by making use of not only the 'global context information', but also the 'local context information', to the best of our knowledge.

Three questions arise for keyword extraction: (1) how to formalize the problem (since it involves many different understandings of the 'keyword'), (2) how to solve the problem in a principled approach, and (3) how to make an implementation.

(1) We formalize keyword extraction as a classification problem, in which the words/phrases in a document can be classified into three groups: 'good keyword', 'indifferent keyword', and 'bad keyword'. We give a specification of keyword in this paper.

(2) We propose to conduct keyword extraction by a classification approach. In the approach, we select the candidate keywords by tri-grams, and then define the features by both 'global context information' and 'local context information'. We then accomplish the keyword extraction by a classification model that is trained in advance.

(3) We propose a unified statistical learning approach to the tasks, based on Support Vector Machines (SVMs).

We tried to collect data from as many sources as possible for experiments. Our experimental results indicate that the proposed SVM based method performs significantly better than the baseline method for keyword extraction. We also applied our method to document classification. Experimental results indicate that our method can indeed enhance the accuracy of document classification. We observed improvements ranging from 7.79% to 22.12% on document classification in terms of F1-Measure.

The rest of the paper is organized as follows. In section 2, we introduce related work. In section 3, we formalize the problem of keyword extraction. In section 4, we describe our approach to the problem and in section 5, we explain one possible implementation. Section 6 gives our experimental results. We make concluding remarks in section 7.

2 Related Work

2.1 Keyword Extraction

Keyword extraction is the task of selecting a small set of words/phrases from a document that can describe the meaning of the document. It is an important area in text mining [Hulth2004].

For example, Turney has developed a system, called GenEx, for keyword extraction based on a set of parameterized heuristic rules that can be tuned by using a genetic algorithm [Turney2000]. The system optimizes the rules' parameters from the training documents.

Eibe Frank et al propose a key phrase extraction algorithm, called KEA, based on Naïve Bayes machine learning approach [Frank1999, Witten1999]. They have

employed the extracted keywords to learn a model and have applied the model for finding keywords from new documents. They have defined the features in the model by utilizing the global context information, i.e. term frequency and first occurrence of the word/phrase.

Tang et al also apply Bayesian decision theory to the task of keyword extraction [Tang2004]. They make use of word linkage information and define two 'local context' features. See also [Azcarraga2002, Hulth2004, Matsuo2004, Zhu2003].

Our method exploits both global context and local context information. We translate the problem of keyword extraction to a binary classification problem, and use SVM algorithm as the classifier.

2.2 Keyword Assignment

Keyword assignment is aimed to assign keywords from a predefined controlled-vocabulary to documents.

For example, Dumaisn et al propose a method for finding a mapping from documents to the categories that are defined as keywords in a controlled-vocabulary [Dumais1998]. They make use of machine learning method to learn classifiers from a set of training documents. A new document then is processed by each of the classifiers and is assigned to those categories with higher probability. Keyword assignment has been limited by the predefined controlled-vocabulary, which cannot process the unknown words/phrases.

2.3 Key-Term Recognition

Term extraction is a task in which base noun phrases (base NP) are extracted from documents [Xun2000]. The extracted terms can be used as features of documents for document mining applications such as document categorization and document clustering. It is different from keyword extraction in two aspects. First of all, the goals are different. Term extraction is aimed to extract base noun phrases that can be used as features of documents for other mining applications, while keyword extraction is aimed to extract the most meaningful words/phrases that can be used to describe the documents. In this way, a keyword can be a base NP; however, a base NP is not necessary a keyword. Secondly, the number of extracted words/phrases can be significantly different. Keywords extracted from a document should be as few as possible if only they can make the user easily distinguish the document from the others, while term extraction can extract many words/phrases. See also [Brill1999].

2.4 Text Summarization

Text summarization is another type of similar work to keyword extraction. Informally, the goal of text summarization is to take a textual document as input, extract content from it, and present the most 'meaningful' content to users in a condensed form and in a manner sensitive to the user's or application's needs [Mani2001]. Extracted content by text summarization can be paragraph(s) or sentences(s). Therefore, text summarization differs in nature from keyword extraction.

Zha, for instance, proposes a summarization method by first clustering sentences of a document (or a set of documents) into topical groups and then, within each topical

group, selecting the key phrases and sentences by their saliency scores [Zha2002]. See also [Berger2000, DUC, Mani1999].

3 Keyword Extraction

Keywords extracted from a document play an important role in describing the meaning of that document. For example, in a simple application, a user wants to find an expertise report in a certain field from a document collection. He can quickly judge whether or not a document is what he wants by using only keywords of that document. In more sophisticated application, the user can group documents by using keywords. Keywords can be also useful for text mining applications, for example: document classification and document clustering. In section 6, we will demonstrate that keywords indeed improve the accuracy of document classification.

Now we formally define the keyword extraction problem that we are solving. Given a document D with a bag of words $w_1, w_2, \ldots, w_N$, we need to find a small set of keywords. Here, the word is the smallest language unit in our problem, and a keyword can be either a word or a word sequence (i.e. phrase). This also means that the keywords extracted should occur in the document.

Judging whether a word/phrase is keyword in an objective way is hard. However, we can still provide relatively objective guidelines for judgment. We call it the specification in this paper. It is indispensable for development and evaluation of keyword extraction.

In the specification, we create three categories for keywords which represent their goodness as keywords: 'good keyword', 'indifferent keyword' and 'bad keyword'.

A good keyword must contain the general notion of the document, and several important properties of the document. From a good keyword, one can understand the basic meaning of the document. Furthermore, a good keyword should be easily searchable and understandable by humans and it should be specific enough to allow the user to distinguish between documents with similar contents.

A bad keyword neither describes the general notion nor properties of the document. It can be difficult to be understood by human. One cannot get the meaning of the document by reading a bad keyword.

An indifferent keyword is one that between good and bad keyword.

4 Our Approach

We formalize keyword extraction as a classification problem. We take 'good keyword', 'indifferent keyword', and 'bad keyword' as classes, words/phrases manually labeled with the three classes in the 'training documents' as training examples, words/phrases in the 'test documents' as test examples, so that keyword extraction can be automatically accomplished by predicting the class of each test example.

We perform keyword extraction in two passes of processing: learning, and keyword extraction.

In learning, we construct the classification model that can predict the class of each word/phrase. In the classification model, we view each word/phrase as an example. For each example, we define a set of features and assign a label. The label represents

whether the word/phrase is a 'good keyword', 'indifferent keyword', or 'bad keyword'. We use the labeled data to train the classification models in advance.

In keyword extraction, the input is a document. We extract a bag of words/phrases from that document. Then, for each word/phrase in that document, we employ the learned classification models to predict whether it is 'good keyword', 'indifferent keyword', or 'bad keyword'. We next view the words/phrases that are predicted as 'good keywords' as keywords.

In this paper, we will focus our implementation on how to classify a word/phrase as 'keyword' or not, and re-formalize the problem as a two-classification problem. However, to determine whether a word/phrase is 'good keyword', 'indifferent keyword', or 'bad keyword' is still one of our near future work.

5 Implementation

We consider one implementation of our approach. We make use of SVM (Support Vector Machines) as the classification model [Vapnik1995].

Let us first consider a two class classification problem. Let $\{(x_1, y_1), \ldots, (x_N, y_N)\}$ be a training data set, in which x_i denotes an example (a feature vector) and $y_i \in \{-1,+1\}$ denotes a classification label. In learning, one attempts to find an optimal separating hyper-plane that maximally separates the two classes of training examples (more precisely, maximizes the margin between the two classes of examples). The hyper-plane corresponds to a classifier (linear SVM). It is theoretically guaranteed that the linear classifier obtained in this way has small generalization errors. Linear SVM can be further extended into non-linear SVMs by using kernel functions such as Gaussian and polynomial kernels.

We use SVM-light, which is available at http://svmlight.joachims.org/. We choose polynomial kernel, because our preliminary experimental results show that it works best for our current task. We use the default values for the parameters in SVM-light. When there are more than two classes, we adopt the "one class versus all others" approach, i.e., take one class as positive and the other classes as negative.

5.1 Process

The input is a document. The implementation carries out extraction in the following steps.

(1) Preprocessing. For a document, we conduct the sentence split, word tokenization and POS (part-of-speech) tagging by using GATE [Cunningham2002]. We next employ Linker [Sleator1991] to analyze the dependency relationships between words in each sentence. After that, we employ tri-gram to create candidate phrases, and then filter the phrases whose frequencies are below a predefined threshold. We also exclude the common words in the stop-words list. We conduct a gentle stemming by using WordNet [Miller1990]. Specifically, we only stem the plural noun, gerund, and passive infinitive by their dictionary form. Finally, we obtain a set of 'keyword candidate' for the later processing.

(2) Feature extraction. The input is a bag of words/phrases in a document. We make use of both local context information and global context information of a word/phrase in a document to define its features (see following section for a detailed

definition of the features). The output is the feature vectors, and each vector corresponds to a word/phrase.

(3) Learning. The input is a collection of feature vector by step (2). We construct a SVM model that can identify the keyword. In the SVM model, we view a word/phrase as an example, the words/phrases labeled with 'keyword' as positive examples, and the other words/phrases as negative examples. We use the labeled data to train the SVM model in advance.

(4) Extraction. The input is a document. We employ the preprocessing and the feature extraction on it, and obtain a set of feature vectors. We then predict whether or not a word/phrase is a keyword by using the SVM model from step (3). Finally, the output is the extracted keywords for that document.

The key issue here is how to define *features* for effectively performing the extraction task.

5.2 Features in the Model

We make use of both 'global context information' and 'local context information' features to represent our data. Each word/phrase is represented by a set of its local context features and the document global context features.

(1) Global Context Features

TFIDF Feature. The feature is calculated by TF*IDF, where TF is the Term Frequency in the document and IDF is the Inverted Document Frequency, i.e. $log((N+1)/(n+1))$. N is the total number of the documents in the document collection. n is the number of documents in which the current word/phrase occurs.

First Occurrence Feature. The feature represents the percentage of words/phrases occurring before the current word/phrase to the total number of words in the document. Its value ranges between 0 and 1.

Position Features. Three features respectively represent whether or not the current word/phrase occurs in document title, abstract (if there exists), and section title (if there exists). The words/phrases occurring in the document title, abstract and section title have higher probabilities of being keywords.

(2) Local Context Features

Local context features have not been investigated previously for the task of keyword extraction.

POS Feature. The feature represents the POS of the current word. A keyword usually is a noun word/phrase. For phrase, we define a unified POS: "PHRASE".

Linkage Features. In the preprocessing step, the dependency relationships between words are recognized. We give two linkage definitions: Linkage Authority and Linkage Hub. Linkage Authority denotes how many words that modify the word. The more the word is modified by, the higher authority it has. Linkage Hub denotes how many words that are modified by the word. The more the word modifies, the higher hub it has.

The two linkages are defined as follows:

$$wl_h = \frac{freq(w_i, \forall)}{count(\forall, \forall)} \times -\log \frac{df(w_i, \forall)}{N}$$

$$wl_a = \frac{freq(\forall, w_i)}{count(\forall, \forall)} \times -\log \frac{df(\forall, w_i)}{N}$$

Where:

wl_h and wl_a represent the modifying relationship and the modified relationship respectively.

$freq(w_i, \forall)$ is the number of words that w_i is modifying in a given document.

$df(w_i, \forall)$ is the number of documents containing the modifier relation $freq(w_i, \forall)$ in the global corpora.

$count(\forall, \forall)$ is the number of total modifier relationships.

N is the size of the global corpora.

The value wl_h and wl_a are located between 0 and 1. $\log(df(\forall, w_i)/N)$ is the log of probability that this word appears in any document of the corpora.

Contextual TFIDF Feature. Contextual TFIDF is the sum of the TFIDF of words in the 'context' of the current word/phrase, which represents the contextual TFIDF. We view words in the same sentence as the current word as its contextual words.

6 Experimental Results

Data Sets and Evaluation Measures

Data Sets. We tried to collect documents for experiments from as many sources as possible. We randomly chose in total 350 documents from four sources and created four data sets. ACM is from the proceedings of conference from 2002 to 2004 in ACM digital library (http://portal.acm.org/portal.cfm). CiteSeer is from CiteSeer website (http://citeseer.ist.psu.edu/cs). Reuter is from the distribution 1.0 of Reuter text collection (http://www.research.att.com/~lewis/ reuters21578.html). Web Doc is downloaded arbitrarily from the Internet.

Table 1 shows the statistics on the data sets. The columns present respectively the data set, its description and the number of documents in it.

Among the four datasets, some research papers already have author assigned keywords, but some of the keywords do not conform to our specification (defined in

Table1. Statistics of the datasets in experiments

Dataset	Description	Total Number
ACM	From ACM Digital Library, proceedings of conferences hold during 2002~2004	200
CiteSeer	From CiteSeer	65
Reuter	Distribution 1.0 of Reuter text collection	50
Web Doc	Downloaded arbitrarily from internet	35

section 3). Moreover, the other documents do not have keywords. Human annotators conducted annotation on all the documents. Keywords of research papers were updated according to the specification, and keywords in the other documents were labeled. Specifically, five graduates in our laboratory were asked to conduct the annotation for all the documents. The number of annotated keywords ranges from 4 to 10. The average of annotated keywords is 9.69 per document. Finally, for each document, intersection of keywords assigned by different annotators is taken as the 'correct' keywords. In this way, each document has six 'correct' keywords averagely.

Evaluation Measures. In all the experiments on extraction, we conducted evaluations in terms of precision, recall and F1-Measure. The evaluation measures are defined as follows:

Precision: $P = A / (A + B)$

Recall: $R = A / (A + C)$

F1-Measure: $F1 = 2PR / (P + R)$

where A, B, C and D denote number of instances.

Table 2. Contingence table on results of detection and extraction

	Is Target	**Is Not Target**
Detected	A	B
Non Detected	C	D

In all evaluations, we view a keywords assigned by humans as a 'target'. If a method can extract the target, we say that it makes a correct decision; otherwise, we say that it makes a mistake. Precision, recall, and F1-Measure are calculated on the basis of the result.

Baseline Method. We use TFIDF as the baseline to extract the keyword (many text mining applications use this method for bag-of-word feature selection). Specifically, for a document, we selected six words/phrases with the higher TFIDF as keywords.

We also carried out the comparison of our method and the KEA algorithm [Frank1999, Witten1999]. KEA algorithm has been proposed by Eibe Frank et al. It uses TFIDF and first occurrence as features and uses Naïve Bayes for learning and extraction. We downloaded the KEA algorithm from http://www.nzdl.org/Kea/index.html#download, and applied it to the four datasets.

Finally, we conducted the experiments of using only global features and using all the features we defined in our model (including global and local features). We carried out the comparison of the results by our methods with the two kinds of features.

Keyword Extraction

Experiment. We evaluated the performances of our keyword extraction methods on the four data sets.

We performed comparisons with the baseline methods and the KEA algorithm. We also evaluated our method with only the global context features and with all the features we defined.

Because the average number of keywords annotated manually is six, we select six keywords in the baseline method and KEA method. In our approach, we select the words as keywords that are classified as positive examples by the SVM model.

Table 3 shows the five cross-validation results on the four data sets. In the table, Global, Local respectively denotes the global context features and local context features. The second column indicates the average number of keywords extracted.

Table 3. Evaluation of Keyword Extraction (%)

	Average Number	Precision	Recall	F1-Measure
Baseline Method	6	12.70	13.11	12.90
KEA	6	32.39(+19.69)	29.09(+15.98)	30.65(+17.75)
Our Method (Global)	5.88	70.75(+58.05)	42.87(+29.76)	53.39(+40.49)
Our Method (Global)+(Local)	7.75	67.43(+54.73)	53.87(+40.76)	59.90(+47.00)

We see that our method can significantly outperform the baseline method and the KEA algorithm. Our method by using global context features and local context features also outperforms that by using only global context features. We conducted sign tests on the results. The *p* values are significantly smaller than 0.01, indicating that the improvements are statistically significant.

Discussions

(1) Improvements over baseline method. The TFIDF based extraction method results in a poor performance (only 12.90% in terms of **F1**-Measure). When using only global context features, we can obtain greatly improvement +40.09% in terms of F1-Measure. Furthermore, by combining the global context features and local context features, we can again obtain improvement +47.00% in terms of F1-Measure.

(2) Improvements over KEA algorithm. When using only global context features, we can obtain greatly improvement +22.47% in terms of F1-Measure over KEA algorithm. Furthermore, by combining the global context features and local context features, we can again obtain improvement +29.25% in terms of F1-Measure.

(3) Effectiveness of local context features. Our method using both global context features and local context feature outperforms that using only global context features (+6.51% in terms of F1-Measure). In the latter method more keywords were assigned as keywords than that in our method with only global features. With the local features, we observed significant improvement (+11.00%) in terms of recall. We also note that the precision drops a little. This is because of inevitable mistaken assignments.

(4) Error analysis. We conducted error analysis on the results.

For the method using global context features only, 80% of the errors were from those extracted words that have high scores in global context features but are not 'keyword'. 20% of the errors were due to the 'ambiguity' of the extracted keywords. Those 'ambiguity' words/phrases represent some kind of the meaning of the given document. But it is difficult to classify them to 'keyword' or not. Maybe they should correspond to the 'indifferent keyword' in our specification.

For the method combining both global and local context features, 25% of the errors were from those words having high scores in global context features but not 'key-

word'. This was due to the effect of local context features. 75% of the errors were due to the 'ambiguity' of the extracted keywords.
(5) No free lunch. As the proverb says, "Every coin has its two sides". Although the local context features can improve the performance of the keyword extraction, their computation costs are heavy, especially on linker analysis. The linker analysis on the four data sets cost nearly 22 hours.
(6) Difficult task. It is difficult to accurately extract keywords from documents. That is because in some cases judging whether a word/phrase is keyword or not is difficult, even for human.

Document Classification
Experiment. In order to evaluate the effectiveness of our keyword extraction method, we have applied it to document classification. Document classification is a task in which we aim to assign documents of a corpus to a fixed set of categories. We chose Naïve Bayes as the classification method. For each document, the extracted keywords are viewed as features and are assigned with higher weights.

Table 4. Performance of document classification (%)

Category	Method	Precision	Recall	F1-Measure
alt.atheism	**Bayes**	65.71	91.09	76.35
	+Keyword	81.51(+18.5)	96.04(+4.95)	88.18(+11.83)
comp.graphics	**Bayes**	27.17	68.32	38.88
	+Keyword	50.00(+22.83)	78.22(+9.9)	61.00(+22.12)
comp.os.ms-windows.misc	**Bayes**	28.25	86.14	42.55
	+Keyword	43.96(+15.71)	90.10(+3.96)	59.09(+16.54)
comp.sys.ibm.pc.hardware	**Bayes**	29.23	75.25	42.10
	+Keyword	39.90(+10.67)	82.18(+6.93)	53.72(+11.62)
comp.sys.mac.hardware	**Bayes**	27.65	85.15	41.74
	+Keyword	38.99(+11.34)	84.16(-0.99)	53.29(+11.55)
comp.windows.x	**Bayes**	51.72	44.55	47.87
	+Keyword	64.04(+12.32)	56.44(+11.89)	60.00(+12.13)
misc.forsale	**Bayes**	31.60	84.16	45.95
	+Keyword	40.95(+9.35)	85.15(+0.99)	55.30(+9.35)
rec.autos	**Bayes**	33.33	92.08	48.94
	+Keyword	50.00(+16.67)	96.04(+3.96)	65.76(+16.82)
rec.motorcycles	**Bayes**	39.83	93.07	55.79
	+Keyword	47.76(+7.93)	95.05(+1.98)	63.58(+7.79)
rec.sport.baseball	**Bayes**	50.54	93.07	65.51
	+Keyword	60.63(+10.09)	96.04(+2.97)	74.33(+8.82)
Average	**Bayes**	38.50	81.29	50.57
	+Keyword	51.77(+13.27)	85.94(+4.65)	63.43(+12.86)

The data set used for classification is 20 newsgroups, which is downloaded from CMU Text Learning Group Data Archives (http://www-2.cs.cmu.edu/afs/cs.cmu.edu/project/theo-20/www/data/news20.html). Among the collection, ten categories were used in the evaluation. The bag-of-word (BOW) features are extracted as features. Both stemming and stop-word removal were applied in the processing of feature extraction.

We carried out the experiments as follows. In the first experiment, we used all the BOW features to evaluate the classification performance. In the other experiment, we conducted the keyword extraction, viewed them as features, and doubled the weight of the extracted keywords.

Table 4 shows the five cross-validation results on the ten categories. In the table, the first column lists ten categories. Bayes, Keyword respectively denotes the Naïve Bayes classification on the original data and on the data with extracted keywords.

From table 4, we see that by using keyword extraction, a significant improvement can be obtained on the task of document classification (ranging from 7.79% to 22.12% in terms of F1-Measure). We observed improvements on precision (+13.27%), recall (+4.65%), and F1-Measure (+12.86%) on average. The results indicate that our method of keyword extraction is effective. The results are also consistent with the result obtained in the experiment of keyword extraction.

We also applied the extracted keywords to document classification by using SVM as the document classification method. However, no significant improvement obtained. The reason maybe lies in that SVM has already achieved the start-of-art result on the task of document classification.

7 Conclusion

In this paper, we have investigated the problem of keyword extraction. We have defined the problem as that of extracting words/phrases in the document. We have proposed a classification approach to the task. Using Support Vector Machines, we have been able to make an implementation of the approach. Experimental results show that our approach can significantly outperform baseline methods for keyword extraction. When applying it to document classification, we observed a significant improvement on extraction accuracy.

As future work, we plan to make further improvement on the accuracy. We also want to apply the keyword extraction method to other text mining applications.

References

[Azcarraga2002] Azcarraga, A; Yap, T. J.; and Chua, T. S. Comparing Keyword Extraction Techniques for WEBSOM Text Archives, International Journal of Artificial Intelligence Tools, 11(2):219 - 232.

[Berger2000] Berger, A. L.; and Mittal, V. O. OCELOT: A System for Summarizing Web Pages. In Proceedings of the 23rd ACM SIGIR Conference, 144 - 151.

[Brill1999] Brill, E.; and Ngai, G. Man vs. machine: A case study in baseNP learning. In Proceedings of the 18th International Conference on Computational Linguistics, 65-72.

[Cunningham2002] Cunningham, H.; Maynard, D.; Bontcheva, K.; and Tablan, V. GATE: A Framework and Graphical Development Environment for Robust NLP Tools and Applications. In Proceedings of the 40th Anniversary Meeting of the Association for Computational Linguistics. Philadelphia.

[DUC] Document Understanding Conference. http://www-nlpir.nist.gov/projects/duc/.

[Dumais1998] Dumais, S.T.; Platt, J; Heckerman, D.; and Sahami, M. Inductive Learning Algorithms and Representations for Text Categorization. In Proceedings of the 7th International Conference on Information and Knowledge Management, 148-155.

[Frank1999] Frank, E.; Paynter, G. W.; and Witten, I. H. Domain-Specific Keyphrase Extraction. In Proceedings of the 16th International Joint Conference on Artificial Intelligence, 668-673, Stockholm, Sweden, Morgan Kaufmann.

[Hulth2004] Hulth, A. Combining Machine Learning and Natural Language Processing for Automatic Keyword Extraction. Ph.D. diss., Dept. of Computer and Systems Sciences, Stockholm University.

[Mani1999] Mani, I.; and Maybury, M. T. Advances in Automatic Text Summarization. The MIT Press.

[Mani2001] Mani, I. Automatic Summarization. John Benjamins Pub Co.

[Matsuo04] Matsuo, Y.; and Ishizuka. M. Keyword Extraction from a Single Document using Word Co-occurrence Statistical Information, Int'l Journal on Artificial Intelligence Tools, 13(1):157-169.

[Miller1990] Miller, G.; Beckwith, R.; Fellbaum, C.; Gross, D.; and Miller, K.J. Wordnet: An On-line Lexical Database. International Journal of Lexicography, 3(4):235--312.

[Sleator1991] Sleator, D.; and Temperley, D. Parsing English with a Link Grammar. Technical Report, CMU-CS-91-196, Dept. of Computer Science, Carnegie Mellon University.

[Tang2004] Tang, J.; Li, J.Z.; Wang, K.H.; and Cai, Y.R. Loss Minimization based Keyword Distillation. In Proceedings of 6th Asia-Pacific Web Conference, 572-577. Springer, LNCS 3007, ISBN 3-540-21371-6.

[Turney2000] Turney, P.D. Learning Algorithms for Keyphrase Extraction. Information Retrieval, 2(4):303–336.

[Vapnik1995] Vladimir N. Vapnik. The Nature of Statistical Learning Theory. Springer.

[Witten1999] Witten, I. H.; and Paynter, G. W. et al. 1999. KEA: Practical Automatic Keyphrase Extraction. In Proceedings of 4th ACM Conference on Digital Libraries, 254-255. Berkeley, CA.

[Xun2000] Xun, E.; Huang, C.; and Zhou, M. A Unified Statistical Model for the Identification of English baseNP. In Proceedings of the 38th Annual Meeting of the Association for Computational Linguistics. Hong Kong.

[Zha2002] Zha, H. Generic Summarization and Keyphrase Extraction Using Mutual Reinforcement Principle and Sentence Clustering. In Proceedings of the 25th ACM SIGIR Conference, 113-120.

[Zhu2003] Zhu, M.; Cai, Z.; and Cai, Q. Automatic Keywords Extraction of Chinese Document Using Small World Structure. In Proceeding of the international conference on Natural Language Processing and Knowledge Engineering.