Improvement of KEA Based on Lexical Chain

Zefeng Li[1,2], Xianghui Zhao[1], Jin Yi[1], Bin He[1,2]
[1]China Information Technology Security Evaluation Center, Beijing, China
[2]School of Information, Renmin University of China, Beijing, China
{cblizefeng111, binhe}@ruc.edu.cn, xianghuizhao@hotmail.com, yijin@itsec.gov.cn

***Abstract*—Keyphrases are very useful and significant for information retrieval, automatic summarizing, text clustering, etc. KEA is a traditional and classical algorithm in keyphrase automatic extraction. But it is mainly based on the statistical information without considering the semantic information. In this paper, We propose a method which combine semantic information with KEA by constructing lexical chain that based on Reget's thesaurus. In our method, the semantic similarity between terms is used to construct the lexical chain, and then we use the length of the chain as a feature to build the extraction model. The experiment result shows that the performance of the system has a big improvement compare with the KEA.**

Keywords-keyphrases extraction; KEA; lexcial chain;semantic similarity.

I. INTRODUCTION

With the development of society, people contact with lots of information every day. Keyphrases, as a brief summary of a document, provide a solution to help people effectively organize and retrieve documents, search and manage information. They have been widely used in many fields. For example, digital libraries system and information retrieval system use keyphrases to construct file index [1-2]，text mining system use keyphrases to extract abstract sentences [3-4], many clustering and classification algorithms also use the keyphrase to build feature vectors of the article [5-6]. Keyphrases can be regarded as a sequence of one or more words that have a main description of a document. They are often chosen manually, usually by the author and sometimes by professional indexers. Unfortunately, not all documents include author or indexer assigned keyphrases especially in collections of scientific papers or news articles those with keyphrases are in the minority. Manual keyphrase identification is boring and time-consuming, requires expertise, and may give inconsistent result, and therefore automatic methods benefit both the developers and the users of large document collections.

Most previous methods for keyphrase extraction are based on statistical properties including the phrase's TFIDF, position and other statistical information in the document. These methods have low performances as many appropriate keyphrases may not appear frequently especially for some short documents. This paper proposed a method that combine the KEA [7] (Keyphrase Extraction automatically) algorithm with lexical chain algorithm which is based on semantic similarity.

II. RELATED WORK

Krulwich and Burkey [8] use heuristic method to extract keyphrases. Sreier and Belew [10] use mutual information to discover keyphrase that consists of double words. Munoz [9] uses unsupervised learning techniques to solve the problem of automatic keyphrase extraction. The unsupervised approaches for keyphrase extraction proposed so far have involved a number of techniques, including language modeling, graph-based ranking, clustering and so on while supervised methods typically regard this problem as a binary classification situation, in which a model is trained on annotated data to determine whether a given candidate phrase is a keyphrase or not. Turney and Witten use supervised learning method to exploit the GenEx system and KEA system, which have significant meaning in the history of keyphrase extraction. They first get the model by training the corpus of labeled keyphrases, and then extract keyphrases from unlabeled documents using the trained model. This method makes a great progress compare with the predecessors on precision and recall. Turney uses genetic algorithm and C4.5 decision tree learning method to design the GenEx system while Witten chooses Naive Bayes technique to train the discrete feature value of the candidate phrases, and the most important features for classifying a candidate phrase are the frequency and location in the document. Medelyan and Witten [11] propose KEA++ that enhances automatic keyphrase extraction by using controlled vocabulary while a domain-specific thesaurus will be essential. This method created opportunity for Nguyen and Kan [12] focus on keyphrase extraction in scientific publications by using new features that capture salient morphological phenomena found in scientific keyphrases. Recent years, some researchers begin to use the relationships between words to extract keyphrases. Like natural network and social natural network, natural language can also be regarded as a network. Every word or phrase is just a node of the net and the relations among words or phrases can be called joints.

III. KEA

KEA was proposed by Witten in 1999 to extract keyphrases automatically. It uses lexical methods to identify candidate keyphrase, computes feature values for every candidate, and uses Naïve Bayes machine-learning algorithm to predict which candidates are good keyphrases [7]. Kea has two main steps: The machine learning scheme first builds a prediction model using training documents with annotated keyphrases, and then uses the established model to find keyphrases in new documents.

Published by Atlantis Press, Paris, France.

In order to get the training model, manual annotated keyphrases of the documents are needed, which would be used as a part of a training set. KEA chooses candidate phrases in three steps. It first cleans the input texts. The input stream is split into tokens (sequences of letters, digits and internal periods), and then the punctuation marks, brackets, numbers, etc. are removed. The second step is phrase identification. KEA considers all the subsequences in each line and determines which candidate phrases are suitable. Candidate phrases are of course limited to a certain maximum length (usually three words) and candidate phrases cannot begin or end with a meaningless word. The last step is to fold all words and stem them using the iterated Lovins method, which involves using the classic Lovins stemmer to deal with any suffix and repeating the process on the stem that remains until there is no further change.

Then KEA calculates the feature of each candidate phrase. Two features are included for training and extraction: TF-IDF, which is a measure of a phrase's frequency in a document compared to its rarity in general use. This value is a statistic used to evaluate how important a word is to a document in a collection or corpus. It increases proportionally to the number of times a word appears in the document, but is offset by the frequency of the word in the corpus, which help to control for the fact that some words are generally more common than others. Witten uses the following formula to get the TF-IDF value:

$$TFIDF = \frac{freq(P,D)}{size(D)} * \left(-\log_2 \frac{df(P)}{N} \right), \quad (1)$$

1. $freq(P,D)$ is the number of times P occurs in D.

2. $size(D)$ is the number of words in D.

3. $df(P)$ is the number of documents containing P in the global corpus.

4. N is the size of the global corpus.

If the document is not in the global corpus, $df(P)$ and N are both plus by one before they are evaluated, to simulate its appearance in the corpus.

The other feature, first occurrence, is the distance into the document from the start to the phrase's first appearance. It is measured by the percentage between the number of words that precede the phrase's first occurrence and the number of words in the document. Terms that tend to appear at the start or at the end of a document are more likely to be keyphrases [13].

The above two features will be applied to Naïve Bayes model. Two probabilities are computed:

$$P[yes] = \frac{Y}{Y+N} P_{IF-IDF}[t \mid yes] P_{distance}[d \mid yes] \quad (2)$$

$$P[no] = \frac{N}{Y+N} P_{IF-IDF}[t \mid no] P_{distance}[d \mid no] \quad (3)$$

where Y is the number of phrases that correspond to the author identified keyphrases. N is number of candidate phrase that are not keyphrases. The overall probability that a candidate phrase is a keyphrase is calculated:

$$p = P[yes] / (P[yes] + P[no]) \quad (4)$$

KEA applies the model built in training phase to select the keyphrases from a new document. The model determines the overall probability that each candidate is a keyphrase, and then rank the possibility.KEA choose the top-n candidates as the output results where n is the number of keyphrases we required.

IV. SEMANTIC SIMILARITY

A simple and available way of calculating semantic similarity in taxonomy is to measure the distance between the words. We compare the path length of each pair items: the shorter the path, the more similar the items. We take the length of the shortest one when given multiple paths. In our paper, we measure the semantic similarity using a computerized 1987 Roget's thesaurus [14] to build the lexical chain as Roget's Thesaurus hierarchy is very regular and based on a well-constructed concept classification. Roget's Thesaurus contains almost 250,000 words. It consists of eight major classes developed by Roget: Abstract Relations, Space and Matter. These three classes cover the external world while the remaining ones, Formation of ideas, Communication of ideas, Individual volition, Social volition, Emotion, Religion and Sensation deal with the internal world of human beings [14]. A path in Roget's ontology usually starts with the classes, and then it branches to sections, sub-sections, head group, head, part-of-speech, paragraph, and group. We define the distance of each two adjacent layers is 2. Thus the distance of each two words is from 0 to 16. For example, "journey's end" and "terminus" can be found in the same group and their distance is 0 while "nag" and "like greased lighting" is 16 because we cannot find these two words in any same classes. We convert the distance measure to similarity by using the following simple mathematical expression:

$$Sim(w_1, w_2) = \frac{1}{\min dis\tan ce(n_1, n_2)} \quad (5)$$

where n_1 and n_2 are the sets of references for the words or phrases w_1 and w_2.

V. LEXICAL CHAIN BUILDING

Lexical chain was first proposed by Morris and Hirst [15]. A lexical chain is a sequence of related words in writing, spanning short (adjacent words or sentences) or long distances (entire text) [16]. A chain is independent of the grammatical structure of the text and in effect it is a list of words that captures a portion of the cohesive structure of the text. It also can provide a context for the resolution of an ambiguous term and enable identification of the concept that the term represents. For example, Beijing → capital → city → inhabitant, google → resource → web.

Algorithms of lexical chains building consider one by one the words for inclusion in the chains constructed so far [17]. Important parameters to consider are the lexical resource used, which determines the lexicon and the possible relations between words in a chain. In this section, we will construct the lexical chain as the following three steps:

A. Get candidate words

The stopwords which have high frequency but no real meaning such as 'is", "of" are not considered. We remove these words that should not appear in lexical chain by using a stoplist contains of almost 1000words. Moreover, we delete all punctuation marks, brackets, and numbers as these elements are useless in the chain calculating.

B. Select an appropriate chain

We will select an appropriate chain for each candidate word. Morris and Hirst introduce five types of thesaural relations of a candidate word in a chain [15]. In this paper, we only use the first one: two words have a category in their index entries. It is the most frequent relation, can be computed rapidly and include a large set of closely related words. Roget's structure plays an important role in this step. We can easily get two relations in terms of the Roget's thesaurus: Repetition of the same word and inclusion in the same head. The particular information of Roger's thesaurus is illustrated in IV.

C. Insert candidate word into the chain.

In this section, we use the semantic similarity to measure the relation between the candidate word and the lexical chain. This is an important step, most open to interpretation. First of all, we choose a candidate word as the head of a chain, and record its line number in the document, and then we scan the candidate words of the document. In this process, a threshold value is given to compare the semantic similarity value between the head of the chain and other candidate words. If the semantic similarity value be equal or greater than the threshold, then we insert the candidate word into the chain. For each candidate word we use this method to build lexical chain. However, what worth mention is that if a word has already been used to start a chain, then when we meet it the next time we just insert it into the chain rather than use it to start a chain again .This reduces the complexity by a large factor! We use a score value to record the length of the chain. If one candidate has different scores in different chains, we would choose the highest score value chain as the best chain. Thus, each candidate word can only belong to one chain. In our paper, the threshold value is set to 10, that means the following two relations would likely share in the same chain:

R1: reiteration of the same string. For example, computer, computer.

R2: the couple phrases belong to the same head, paragraph or part-of-speech. This method can be used to build lexical chains. The parameters head number, part-of-speech and reference name as parameter are used to identify a specific sense of a word or phrase. For example, bank and slope are likely in the same chain because both of them are in head 209 of Roget's thesaurus.

VI. EVALUATION

In this paper, we propose a method with a combination of KEA and lexical chain. We use the following expression to compute the "TFIDF":

$$TFIDF = \frac{\alpha * freq(P,D) + \beta * score}{size(D)} * \left(-\log_2 \frac{df(P)}{N} \right), \quad (6)$$

where α, β are regulatory factors, *score* is the length of the corresponding lexical chain. Thus the feature calculating not only uses basic statistics but also considers the semantics in the document. And then Naïve Bayes technical will be used to build keyphrases extraction model.

Evaluating keyphrases has shown to be subjective and difficult in many previous works. According to Medelyan and Witten [11], the evaluation of keyphrase extraction algorithms requires multiple judgments and cannot rely on a single set of keyphrases provided by a paper's author. To evaluate our performance, we tested our system by using a collection of dataset that provided by Nguyen Thuy Dung, professor of (NUS) [18]. All of the documents in the dataset include more than 80 subjects and they are all selected from ACM conference with length of 4 pages to 12. To make sure the annotated keyphrase more correct, the organizer not only consider the author's annotated phrases but also invited students and staffs in School of Computing, National University of Singapore, to participate in the tasks. In our paper, we randomly choose 40 documents of the dataset as our train set and 60 papers as our test set.

There are two criteria to measure the effectiveness: precision and recall. Precision is the fraction of retrieved instances that are relevant, while recall is the fraction of relevant instances that are retrieved. Both precision and recall are therefore based on an understanding and measure of relevance . They are defined as:

$$P = \frac{tp}{tp + fp}, \; R = \frac{tp}{tp + fn}, \quad (7)$$

1. tp is the correct result.
2. fp is the unexpected result.
3. fn is the missing result.

Table 1 shows the performance of our candidate extraction method and KEA approaches using the current standard evaluation method over top 5th, 10th, 15th candidates. In this table, we set $\alpha = 1.0, \beta = 1.0$.

We change the weight of the lexical chain score to test our data. Here set the number of extracted keyphrases equals 10. The result is shown in table 2. From table1 and table 2 we can know that when $\alpha = 1.0, \beta = 1.0$ and the number of keyphrases extracted equals 10, our results get higher score than traditional KEA both in precision and recall: 5.7% higher in precision and 6.6% in recall. Maybe the results are not ideal, and the better combination of statistical method with semantic method is left for the future work.

TABLE I. AVERAGE PRICISION AND RECALL($\alpha=1.0, \beta=1.0$)

		Number of keyphrases extracted		
		5	10	15
Average pricision	KEA	35.3%	29.3%	24.9%
	Improved KEA	34.0%	35.0%	24.9%
Average recall	KEA	22.0%	35.0%	44.4%
	Improved KEA	21.4%	41.6%	44.4%

TABLE II. AVERAGE PRICISION AND RECALL(NUM=10)

		Regulatory factors		
		$\alpha=0.5, \beta=1.0$	$\alpha=1.0, \beta=1.0$	$\alpha=1.0, \beta=0.5$
Average pricision	KEA	29.3%	29.3%	29.3%
	Improved KEA	31.0%	35.0%	31.0%
Average recall	KEA	35.0%	35.0%	35.0%
	Improved KEA	37.5%	41.6%	37.5%

VII. CONCLUSION AND FUTURE WORK

Keyphrases usually can reflect the theme information of a document. Good keyphrases are beneficial to improve the readers' reading speed and can make deeper understanding of the documents. This paper proposed an approach that based on the lexical chain to improve the KEA keyphrases extraction. This algorithm is actually more efficient compared with the KEA. It not only calculates the basic statistical information but also rise to semantic level, which can make big contribution to mining the deep theme information to reach a relatively better effect.

In the future, we would like to use more data corpuses to evaluate our system. So far, there exist no standard data corpuses to test the keyphrase extraction algorithm. A common way is to compare the extracted keyphrases with the authors' keyphrases. But there are several problems by this method. First, authors' keyphrases do not always appear in the document to which they belong. Second, authors rarely provide more than a few keyphrases—far fewer than may be extracted automatically. Fourth, authors' keyphrases are available for a limited number and type of documents [19]. We will do more research on looking for a more scientific and objective way to evaluate the automatic extraction result. Moreover, we will look for a better combination of statistical method with semantic method and consider more features of the text documents or web documents into building the extraction model to improve our results.

REFERENCES

[1] Y.Chen, F.S.Tai,and K.L.Chan. "Machine learning techniques for business blog search and mining,"Expert Systems with Applications, 2008, 35(3) : 581-590

[2] H.F.Zhu, F.Bao. "Continuous keyword search on multiple text streams," Proceeding of the IEEE International Conference on Communications . Glasgow , England, 2007: 1336-1341

[3] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. Witten, "The WEKA Data Mining Software: An Update,"SIGKDD Explorations, vol. 11, 2009.

[4] W.D.Yang, B.L.Shi. "Schema aware keyword search over XML streams," Proc. IEEE International Conference on Computer and Information Technology.Fuku shima, Japan, 2007: 29-34

[5] Y.Xu, J.T.Li, B.Wang, C.M.Sun, S.Zhang. "A study on feature select ion algorithm in text categorization technology," Journal of Computer Research and Development , 2008, 45(4) : 596-602(in Chinese)

[6] H.S.Chang, C.C.Hsu. "Using topic keyword clusters for automatic document clustering," Proc. 3rd International Conference on Information Technology and Applications. Sydney, Australia, 2005: 419- 424.

[7] I.H.Witten, G.W.Paynter, E.Frank, C.Gutwin, C.G.Nevill-Manning. "KEA: Practical automatic keyphrase extraction," Proc. 4th ACM conference on Digital libraries, Berkeley, California,US, 1999: 254 - 256.

[8] B.Krulwich and C.Burkey. "Learning user information interests through the extraction of the semantically significant phrases,"Proc.AAAI 1996 Spring Symposium on Machine Learning in Information Access.AAAI Press,California.1996

[9] A.MuQnoz. "Compound key word generation from document databases using a hierarchical clustering ART model," Intelligent Data Analysis, 1(1): Elsevier,Amsterdam. 1996

[10] A.M. Steier and R.K.Belew. "Exporting phrases: A statistical analysis of topical language,"In: R Casey and B Croft, Eds., Second Symposium on Document Analysis and Information Retrieval, 1993:179~190.

[11] O.Medelyan and I.H.Witten "Thesaurus based automatic keyphrase indexing," Proc. 6th ACM/IEEE-CS joint conference on Digital libraries, ACM Press,2006, pp. 296–297.

[12] T.D.Nguyen and M.-Y.Kan. "Keyphrase extraction in scientific publications," In Proceedings of ICADL,2007.

[13] http://www.nzdl.org/Kea/description.html.

[14] M. Jarmasz and S.Szpakowicz. "Roget's Thesaurus and Semantic Similarity," Proc. Conference on Recent Advances in Natural Language Processing (RANLP 2003), Borovets, Bulgaria, September, 212-219.

[15] J. Morris and G.Hirst "Lexical cohesion computed by thesaural relations as an indicator of the structure of text," Computational Linguistics, 17(1), (1991) 21–45.

[16] http://en.wikipedia.org/wiki/Lexical_chain.

[17] M. Jarmasz and S. Szpakowicz. "Not As Easy As It Seems: Automating the Construction of Lexical Chains Using Roget's Thesaurus," Proc.16th Canadian Conference on Artificial Intelligence (AI 2003), Halifax, Canada, June, 544-549.

[18] http://wing.comp.nus.edu.sg/downloads/keyphraseCorpus/.

[19] S. Jones and G.W.Paynter. "Human evaluation of Kea, an automatic keyphrasing system," First ACM/IEEE-CS Joint Conference on Digital Libraries, Roanoke,Virginia, June24-29,2001,ACMPress,148-156.