Domain-specific Keyphrase Extraction

Yi-fang Brook Wu, Quanzhi Li, Razvan Stefan Bot, Xin Chen
Information Systems Department
New Jersey Institute of Technology
Newark, NJ 07102
{wu, QL23, rsb2, xc7}@njit.edu

ABSTRACT

Document keyphrases provide semantic metadata characterizing documents and producing an overview of the content of a document. They can be used in many text-mining and knowledge management related applications. This paper describes a Keyphrase Identification Program (KIP), which extracts document keyphrases by using prior positive samples of human identified domain keyphrases to assign weights to the candidate keyphrases. The logic of our algorithm is: the more keywords a candidate keyphrase contains and the more significant these keywords are, the more likely this candidate phrase is a keyphrase. To obtain prior positive inputs, KIP first populates its glossary database using manually identified keyphrases and keywords. It then checks the composition of all noun phrases of a document, looks up the database and calculates scores for all these noun phrases. The ones having higher scores will be extracted as keyphrases.

Categories and Subject Descriptors

I.2.7 [**Computing Methodologies**]: Natural Language Processing – Text analysis, Language passing and understanding. H.3.3 [**Information Systems**]: Information Search and Retrieval; H.2.8 [**Information Systems**]: Database Applications – Data mining;

General Terms

Algorithms, Design, Performance

Keywords

Keyphrase Extraction, Document Keyphrase, Text Mining, Document Metadata

1. INTRODUCTION

Document keyphrases are the most important topical phrases for a given document. They provide a concise summary of a document's content, offering semantic metadata summarizing a document. Previous studies have shown that document keyphrases can be used in a variety of applications, such as retrieval engine [4, 3], and document classification and clustering. most documents do not have author-assigned keyphrases and manually assigning keyphrases to documents is costly and time-consuming, so it is necessary to develop an algorithm to automatically generate keyphrases for documents.

Several automatic keyphrase extraction techniques have been proposed in previous studies. Extractor [5] uses nine features to score a candidate phrase. One example of the features is the location of the first occurrence of a phrase in the document. Keyphrases are extracted from candidate phrases based on the examination of the nine features. Kea [2] uses a machine learning algorithm which is based on naïve Bayes' decision rule. It treats keyphrase extraction as a classification task. Two attributes are used to discriminate between keyphrase and non-keyphrase, the TF.IDF score of a phrase and the distance into the document of the phrase's first appearance.

Both Kea and Extractor use a similar way to identify candidate keyphrase: the input text is split up according to phrase boundaries (numbers, punctuation marks, etc.); non-alphanumeric characters and all numbers are deleted; then a phrase is defined as a sequence of one, two, or three words that appear consecutively in the text. The above approach to identifying candidate keyphrase is different from ours. Kea and Extractor both use machine learning approaches. They all need training corpora to train their programs. For each document in the corpus, there must be a target set of keyphrases provided by authors or generated by experts. Our keyphrase extraction system, called KIP, use a different method to identify candidate kephrases, and it uses a domain-specific glossary database to identify keyphrases, instead of a surprised training method.

2. KIP ALGORITHM

KIP algorithm is based on the logic that a noun phrase containing domain-specific keywords and/or keyphrases is likely to be a keyphrase in the domain. The more keywords/keyphrases it contains and the more significant the keywords/keyphrases are, the more likely that this noun phrase is a keyphrase. The pre-identified domain-specific keywords and keyphrases are stored in the glossary database, which is used to calculate scores of noun phrases. Here a pre-defined domain-specific keyword means a single term word, and a pre-defined domain-specific keyphrase means a phrase containing one or more words. KIP operations can be summarized as follows. KIP first extracts a list of keyphrase candidates, which are noun phrases from input documents. Then it examines the composition of a keyphrase candidate and assigns a score to it. The score of a noun phrase is determined mainly based on three factors: its frequency of occurrence in the document, its composition (what words and sub-phrases it contains), and how specific these words and sub-phrases are in the domain of the document. Finally, the noun phrases with higher scores are selected as keyphrases of the document. KIP has the following main components: the part-of-speech tagger, the noun phrase extractor, and the keyphrase extraction tool.

Part-of-speech Tagger. Our part-of-speech tagger is a revised version of the widely used Brill tagger [1].

Noun Phrase Extractor. the noun phrases extractor extracts noun phrases by selecting the sequence of POS tags that are of interests. The current sequence pattern is defined as {[A]} {N}, where A refers to Adjective, N refers to Noun, { } means repetition, and [ ] means optional.

Extracting Keyphrases. In order to calculate the scores for noun

CIKM'05, October 31-November 5, 2005, Bremen, Germany.
ACM 1-59593-140-6/05/0010.

phrases, we use a glossary database containing domain-specific manual keyphrases and keywords, which provide initial weights for the keywords and sub-phrases of a candidate keyphrase.

The glossary database has two lists (tables): (a) a manual keyphrase list and (b) a manual keyword list. A manual keyphrase could contain one or more words; and a manual keyword means a single word parsed from list (a). The weights are automatically assigned to keywords and keyphrases. The rationale behind this is that it reflects how domain-specific a keyword or keyphrase is in the domain. The more specific a keyword is, the higher weight it has.

A noun phrase's score is defined by multiplying a factor *F* by a factor *S*. *F* is the frequency of this phrase in the document, and *S* is the sum of weights of all the individual words and all the possible combinations of adjacent words within a keyphrase candidate. The score of a noun phrase = $F \times S$.

The sum of weights S is defined as: $S = \sum_{i=1}^{N} w_i + \sum_{j=1}^{M} p_j$

,where w_i is the weight of a word within this noun phrase, and p_j is the weight of a sub-phrase within this noun phrase. The motivation for including the weights of all possible sub-phrases into the phrase score, in addition to the weights of individual words, is to find out if a sub-phrase is a manual keyphrase in the glossary database. If it is, this phrase is expected to be more important. KIP will lookup the keyphrase table and keyword table to obtain the weights for words and sub-phrases. All candidate keyphrases for a document are then ranked in descending order by their scores. The keyphrases of a document can be extracted from the ranked list.

3. EXPERIMENT

we evaluated our system's effectiveness by using the standard information retrieval measures, precision and recall. the document keyphrases assigned by the original author(s) are usually used as the standard keyphrase set. The system-generated keyphrases are compared to the keyphrases assigned by the original author(s). Recall means the proportion of the keyphrases assigned by a document's author(s) that appear in the set of keyphrases generated by the keyphrase extraction system. Precision means the proportion of the extracted keyphrases that match the keyphrases assigned by a document's author(s). We used the Information Systems (IS) domain to perform the experiments.

We also compared KIP to Kea [2] and Extractor [5]. Five hundred papers from four journals and conference proceedings were chosen as the test documents. All these 500 papers had author-assigned keywords. KIP and Kea were compared when the number of extracted keyphrases was 5, 10, 15 and 20, respectively. Due to the limitation of the used version of Extractor, Extractor and KIP were compared only when the number of extracted keyphrases was 5 and 8, respectively. Table 1 and Table 2 show the results.

ACKNOWLEDGEMENT

This project is, in part, supported under NSF grants DUE-#0434581 and DUE-#0434998.

REFERENCE

[1] Brill, E. Transformation-based Error-driven Learning and Natural Language Processing: A Case study in Part-of-speech Tagging. *Computational Linguistics* 21(4), 1995.

[2] Frank, E., Paynter, G., Witten, I., Gutwin, C., and Nevill-Manning, C. Domain-specific keyphrase extraction. *Proceeding of the sixteenth international joint conference on artificial intelligence*, San Mateo, CA, 1999, 668-673.

[3] Jones, S., and Staveley, M. Phrasier: A system for interactive document retrieval using keyphrases. *Proceedings of SIGIR'99:* ACM Press, Berkeley, CA, 1999, 160-167.

[4] Li, Q., Wu, Y .B., Bot, R. S., and Chen, X. Incorporating Document Keyphrases in Search Results. *Proceedings of the Tenth Americas Conference on Information Systems*, New York, New York. 2004

[5] Turney, P. D. Learning algorithm for keyphrase extraction. *Information Retrieval*, 2(4), 2000, 303-336.

Table 1. Precision and Recall for KIP and Kea

Number of extracted keyphrases	**Average Precision ± SD**		Significant test on precision difference	**Average Recall ± SD**		Significant test on recall difference
	KIP	**Kea**	(p-value < 0.05 ?)	**KIP**	**Kea**	(p-value < 0.05 ?)
5	0.27±0.19	0.20±0.18	Yes	0.31±0.22	0.20±0.17	Yes
10	0.19±0.11	0.15±0.12	Yes	0.44±0.24	0.32±0.26	Yes
15	0.15±0.07	0.13±0.10	Yes	0.50±0.23	0.40±0.27	Yes
20	0.12±0.05	0.11±0.08	No	0.54±0.23	0.44±0.28	Yes

Table 2. Precision and Recall for KIP and Extractor

Number of extracted keyphrases	**Average Precision ± SD**		Significant test on precision difference	**Average Recall ± SD**		Significant test on recall difference
	KIP	**Extractor**	(p-value < 0.05 ?)	**KIP**	**Extractor**	(p-value < 0.05?)
5	0.27±0.19	0.24±0.15	No	0.31±0.22	0.26±0.16	Yes
8	0.22±0.13	0.20±0.12	No	0.39±0.24	0.35±0.22	Yes