IS_SUM: A Multi-Document Summarizer based on Document Index Graphic and Lexical Chains

Quan Zhou, Le Sun
Institute of Software
Chinese Academic of Sciences
{zhouquan03, sunle}@iscas.cn

Jian-Yun Nie
Département d'informatique et de recherche opérationnelle
Université de Montréal
{nie}@iro.umontreal.ca

Abstract

IS_SUM is a summarizer developed at Institute of Software (IS) of Chinese Academy of Sciences for DUC 2005. We adopt a new way for clustering and summarizing documents by integrating Document Index Graphic (DIG) [7] with Lexical Chains [5]. Our results show the benefit of integrating DIG with Lexical Chains.

1 Introduction

IS_SUM is a summarizer we developed during our participation to the DUC 2005 Task. Our approach is based on Lexical Chains [5], which has been successfully used for summarization. In our experiments, we further integrate Document Index Graphics (DIG) with Lexical Chains in order to deal with long documents efficiently. The result reported in this paper is produced by the initial version of IS_SUM. We made some modifications to increase the Precision/Recall ratio of the summarization output afterwards.

For the summarization task of this year, we made the following assumption according to the instructions and guidelines of DUC [1] [2], that is, all the documents in test data are more or less related to its topic without exception. The test documents selected by the organizer are selected by volunteers who produced the topic by choosing several documents that are highly relative to the topic from original 50 TREC documents. We believe such documents are rather related to the given topic. So we do not remove any documents that may be unrelated.

According to this assumption, we finally designed our system into 4 modules in sequence: preprocessing, clustering, summarization, and compression. We use sentence extraction strategy, but include lexical analysis to improve the extraction results. After preprocessing, we calculate text similarity to cluster documents. Grouping documents related to some common topics into clusters can help construct lexical chains of the documents. Finally a compression module based on MMR algorithm [20] removes redundancy and reduces the summary to the required length.

The rest of the paper is organized as: Section 2 reviews previous work with focus on DIG and lexical chains. Section 3 presents our approach that is based on lexical chain and DIG. The system architecture and algorithm are show in detail in this section. The evaluation results of our system are presented in section 4. Section 5 summarizes the paper and shows the future work.

2 Previous work

Summarization is a difficult task. “This requires semantic analysis, discourse processing, and inferential interpretation. Those actions are proved to be highly complex by the researchers and their systems in the last decade” [3].

According to the level of semantic analysis, summarization methods can be roughly classified into the following 3 categories:

1) Based on extraction. These methods analyze the sentences similarity and extract the most important sentences to form the

summary. MEAD [17] is an example of this category.

2) Based on simple semantic analysis such as Lexical Chain. We first construct a tree structure of the origin document, and then score the every chain to select the strongest chains as output.

3) Based on deep semantic analysis. For example, Marcu [18] proposed an approach based on the construction of a rhetorical tree that uses explicit discourse markers and heuristic rules to decide which is the best rhetorical tree for a given document.

Those 3 categories cover most of the existing summarizer approaches. In DUC 2005, topics are created to reflect explicitly the specific interests of a potential user in a task context and capture some general user/task preferences in a simple user profile [11]. The 'general' profile task requires a deeper analysis of the documents, while the 'specific' profile task could be met by extraction. In order to meet the requirements of both 'general' and 'specific' tasks, we choose lexical chains as the framework for our system for its flexibility.

Lexical Chain is first proposed by Morris and Hirst [4]. It has been used in a variety of IR and NLP applications including summarization in which it is used as an intermediate text representation. The first summarizer, which uses Lexical Chain, was implemented by Barzilay and Elhadad [5]. Lexical chain is used to weigh the contribution of a sentence to the main topic of a document. Sentences with high occurences of chain words are extracted and presented as a summary of that document. Brunn et al. [6] suggests "calculating the chain scores with the pair-wise sum of the chain word relationship strengths in the chains. Then, sentences are ranked based on the number of 'strong' chain words they contain." [10]

In general, there are several steps to build a summary using lexical chain: 1) forming the chains to represent the origin document, 2) scoring the chains and 3) select the 'strongest' chains to form a summary.

Barzilay [5] pointed out that one of the limitations of lexical chain method is its inefficiency to deal with long documents. In fact, the basic lexical chain method only takes into account the weight of single words rather the weight of both words and phrases. It is known that in order to better capture the structure of documents, the model should be able to represent the sentence structure in addition of words in the document.

In order to get rid of the limitations we mentioned above, we use Document Index Graphics (DIG), which is proposed by Hammouda [15]. "The DIG model is based on graph theory and utilizes graph properties to match any-length phrase from a document to any number of previously seen documents in a time nearly proportional to the number of words of the document."[15]

DIG is a directed graph. Its node represents a unique word in the document and its edge can represent two words in any phrases occurring in the document. Hammouda implemented a document clustering algorithm based on DIG by calculating document similarity using both the weight of phrases and words. It's a good way to solve the problem of long documents. Meanwhile, DIG takes account the phrase weight of a document, so we think it may useful to get a better lexical chain.

3. An approach based on lexical chain and DIG

3.1. DIG representation

We make some modifications to DIG in order to integrate it with lexical chain. Usually all words of the document are used to construct the graph nodes in DIG while in lexical chains only nouns are taken into consideration. In our system we choose both nouns and verbs to represent the document because both of them represent important meanings of a sentence. We call a sequence of nouns and verbs "phrase" in our paper. A phrase in this paper means a set of nouns and verbs which can represent the structure of a sentence. We add the smaller DIG into the chain set, and propose a new algorithm for adding entries and pruning chains.

We pick up document 323 from last year's test data as an example. Figure 1 shows its lexical chain with DIG outputted by our system. In Figure 2, the Document's DIG is presented and only the nouns and verbs whose frequency is higher than 2

is take into account. The construction of DIG and lexical chains is as follows:

Step1: Selected a set of candidate words. Both noun and verb words are take into account.

Step 2: For each candidate, if noun, find the most relative chain; if verb, find the 'phrase' in DIG, and add the verb to the chain where the noun of the 'phrase' appears.

Step 3: If a chain is found, insert the word into the chain and update the chain.

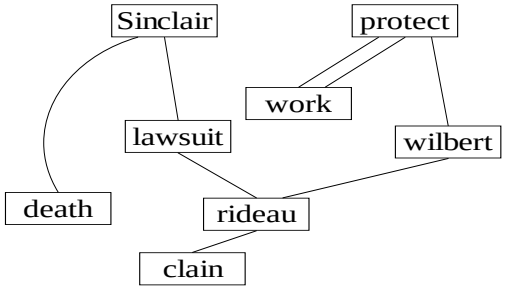

Figure 1: Lexical Chain with DIG

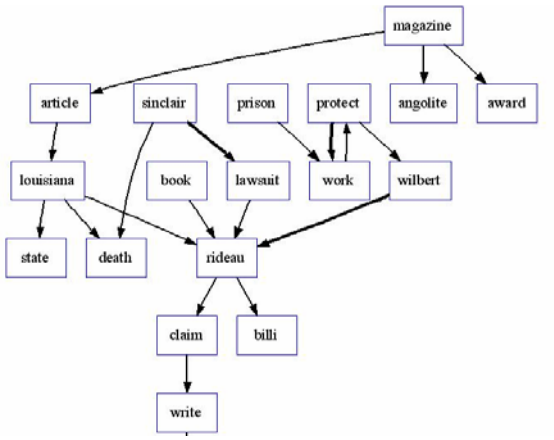

Figure 2: DIG of Document 323/LA010790-0054

3.2 System Architecture

As we described, IS_SUM is separated into 4 modules: Preprocessing, Clustering, Summarization, and Compression (Figure 3). The arrows show data flows in the system. Obviously, we could observe the intermediate output and adjust each module separately.

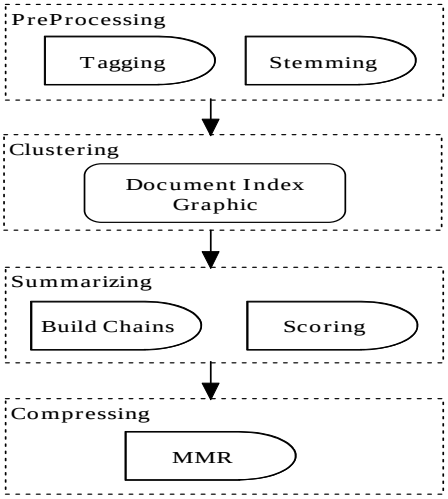

Figure 3: System Architecture

3.3 Preprocessing

The original documents used in DUC are in XML format. The preprocessing module extracts the topics, profiles and documents' bodies by using the technique of DOM. To fulfill other module's requirement, we also do some extra operations within this module. For example, the DOM API is used to divide the body of the documents into sentences and store each sentence with its position, length, content and document number. Both the clustering and summarizing modules need Part of Speech (POS) tags. We choose the Java version of the Stanford English tagger [16] for POS-tagging. We also perform word stemming at the end of this module and calculate the word frequency before store each word in a big table array. Figure 4 shows some details of operations in preprocessing module.

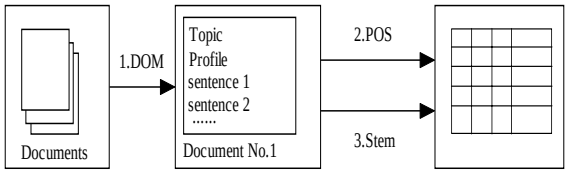

Figure 4: Preprocessing Module

3.4 Document Clustering

The original documents often contain many (more or less important) topics. It is difficult to create a multi-document summary from those for individual documents. Therefore, many

multi-document summarizers first cluster the documents in order to determine the main subjects among these documents. Then a summary is created so as to focus on the common topics of these documents. Many of the documents clustering methods today are based on VSM (Vector Space Model). Documents are represented as a feature vector of the terms (words) that appear in the entire document set. Each feature vector contains term weights of the terms appearing in that document. Similarity between documents can be calculated using one of the similarity measures, such as cosine or Jaccard measure.

The similarity measure we use here is one adapted from Hammouda [7]. Hammond's similarity measure takes account all the words occur in the document. However, we only use the noun and verb occur in the document to build our lexical chains. So it is inefficient to use Hammond's similarity measure. We use a simplified version for phrase weighting according to both the term frequency and the phrase frequency (See Equation 1 below).

Equation 1 consisted of two parts: the phrase weight $Sim_p(D_1, D_2)$ and term weight $Sim_t(D_1, D_2)$.

$$Sim(D_1, D_2) = \alpha Sim_p(D_1, D_2) + (1-\alpha) Sim_t(D_1, D_2) \quad (1)$$

$$Sim_t(D_1, D_2) = \cos(D_1, D_2) = \frac{D_1 \cdot D_2}{\|D_1\| \|D_2\|} \quad (2)$$

$$Sim_p(D_1, D_2) = \sqrt{\sum_1^m [L(p) * (f_{1i} + f_{2i})]^2} \quad (3)$$

We set α at 0.5 empirically in our experiments. $Sim_t(D_1, D_2)$ is calculate by cosine measure where the vectors *D1* and *D2* are represented as term weights using *tf*idf* weighting scheme. Where $L(p)$ is the length of the phrase; m is the number of phrases; f_{1i} is phrase frequency in the D_i.

The formula calculates text similarity between each pair of documents in order to cluster them separately. Finally we separate these documents into small clusters in this module, and each cluster has its own DIG.

3.5 Summarizing

The initial version for our summarizer is based on the traditional chain scoring algorithm proposed by Hirst and St-Onge [8]. Firstly, all words that are a noun or a verb in WordNet are selected. Then "the relatedness of two noun words is determined in terms of the distance between their occurrences in document and the shape of the path that connect them in the WordNet thesaurus"[10]. Three different relationships between candidate noun words are defined: extra-strong, strong and medium-strong. Extra-strong relations are lexical repetitions of a noun word and strong relations are synonyms or near synonyms. Strong relations can also indicate a shared hypernym/hyponym or meronym/holonym, that is, one noun word is a parent-node or child-node of the other in the WordNet topology. Medium-strong relations follow the rule set laid out by St. Onge. [10] These rules govern the shape of the paths that are allowable in the WordNet structure. Then a greedy algorithm is used to disambiguate the multi-sense noun: a multi-sense word's sense is determined only by the senses of other noun words that occur before it in the document. In addition, when processing the verb word, we try to find the phrase that contain this verb, and then add the verb to the chain where the noun of the phrase appears.

It's assumed that the summary should be written at a level of granularity that is consistent with the granularity requested in the user profile. In our system, the two types of profiles (general and specific) have an effect on entry selection during the chains building. The general profile does not need all the address and name entries, so we decrease them by a threshold. Meanwhile the key phrases extracted in the clustering step are good entry from the viewpoint of lexical cohesion. So a chain containing more phrases is given a higher score. The strongest chains of each cluster are selected create the summary once all the chains have been built. However, usually the results are more than 250 words, which are required by DUC. So the results are submitted to the compression module.

3.6 Compression

In order to minimize redundancy, we use an

algorithm called Maximum Marginal Relevance (MMR) [20], which is proposed by Carbonell and Goldstein. It aims at having high relevance of the summary to the query or the document topic, while keeping redundancy in the summary low. A number of features of sentence, such as content words, chronological order, query/topic similarity, anti-redundancy and pronoun penalty, are used in the algorithm. Sentences chosen for inclusion in summary are those that are maximally similar to the document or query, while maximally dissimilar to the sentences already included in the summary.

MMR can generate the summary with any desired length. We adopt sentence properties which has used at the first step of the summarization, such as, positions, words sequence and term frequency in MMR.

4 Evaluation

The official result we submitted to DUC is evaluated by ROUGE 1.5 [12]. Table 1 is the Linguistic Quality score, which is judged on five aspects: grammaticality, non-redundancy, referential clarity, focus, structure and coherence. They are identified by L1, L2...L5 in the table.

System	L1	L2	L3	L4	L5
IS-SUM	3.96	4.76	3.24	3.49	2.67

Table1: Average Linguistic Quality

Our system produced good performance on non-redundancy, but not on structure and coherence. This is because our method does not take syntax into consideration during sentence generation.

Table 2 is responsiveness score of our system. The score is an integer between 1 and 5, with 1 being the least responsive and 5 the most responsive. The bold column is the result of our system.

There are 2 different profiles (General & Specific) in DUC 2005. All the 32 systems show a better performance in the latter summary (specific) and the average scores for the entire 25 topic requires general summary are all below 3. Our system produced almost the same performance in specific topic and general topic. It is due to the fact that our algorithm takes both terms and phrases into consideration.

Number	AVG	**IOS**	Number	AVG	**IOS**
D301	3.125	**3**	D398	2.375	**1**
D307	2.0625	**3**	D400	2.25	**3**
D311	2.78125	**2**	D401	2.78125	**3**
D313	3.6875	**4**	D404	3.46875	**4**
D321	2.21875	**2**	D407	2	**2**
D324	3	**2**	D408	2.3125	**1**
D331	2.4375	**3**	D413	2.65625	**2**
D332	1.65625	**2**	D422	2.0625	**3**
D343	2.34375	**3**	D426	1.96875	**2**
D345	2.25	**2**	D428	2.96875	**2**
D346	2.65625	**3**	D431	2.375	**2**
D347	2.46875	**2**	D434	2.3125	**2**
D350	2.8125	**3**	D435	2.09375	**2**
D354	2.46875	**2**	D436	2.3125	**3**
D357	1.90625	**2**	D438	3.46875	**4**
D360	1.6875	**2**	D442	2.40625	**3**
D366	2.46875	**2**	D446	2.5625	**2**
D370	2	**2**	D632	2.15625	**2**
D374	2.65625	**2**	D633	3.1875	**1**
D376	2.125	**2**	D654	2.40625	**3**
D383	1.875	**1**	D671	1.8125	**1**
D385	2.15625	**2**	D683	2	**2**
D389	2.28125	**1**	D694	2.25	**1**
D391	1.59375	**1**	D695	2.4375	**2**
D393	2.1875	**1**	D699	1.96875	**2**

Table2 Responsiveness

We compare the Models of Document D438 with the summary output by IS_SUM in order to see whether the similar summary would have similar DIG. Figure 6a is a part of DIG of the whole 4 models of D438 (that is the summary results given by human), and the right one, Figure 6b is output of our system. The node of ‘year’ and ‘month’ in right figure shows it has many words that describe time or date. This may be because it contains some sentence about specific event. The size of the in the right graphic show their weight in summary. For example, nodes like ‘UK’ and ‘Tourist’ are important in the summary. Obviously we found such entries are also the focus of the left one. So we may work out a formula for using DIG to evaluate summary later. Although DIG didn’t

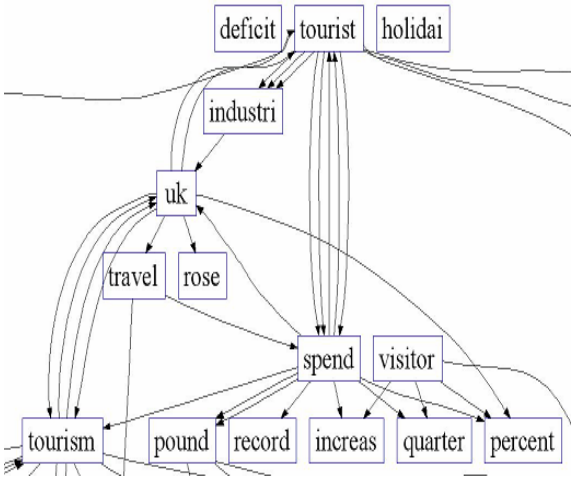

Figure 6a: Four Models' DIG

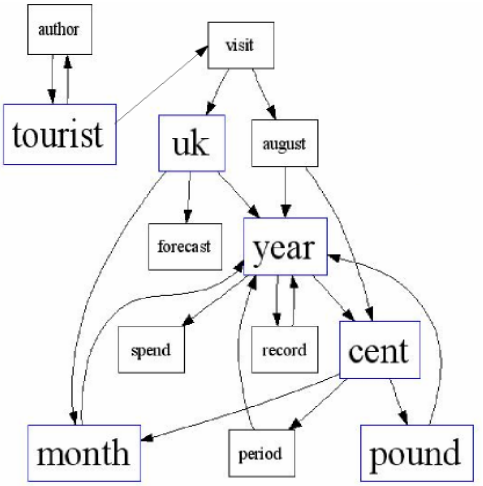

Figure6b: One Peer's DIG

produce precise summary as we expected, it has shown their features by which we can enhance our system in the future.

5 Conclusion and Future work

We adopt a new way for document clustering and summarization based on the DIG and Lexical Chains. A phrase-based module is introduced into the traditional Lexical Chain method. The results show the benefit of integrating DIG with Lexical Chains.

We also observed several problems of our method. This method can at least be improved on the following aspects:

- Ambiguous Anaphora: Several heuristics have been proposed in the literature to address this problem [5]. A preferable solution would be introducing an anaphora resolution algorithm before outputting the final results.
- Performance: Although the main system are written in Java with some aid tools written in Perl, sometimes the required memory is too large and the execution time is too long even for an experimental system.
- Evaluation: We should try to evaluate our system with different methods in order to get the cues to increase the performance. The Pyramid method proposed by DUC this year is a good option.

6 Acknowledgements

This work is supported by Beijing New Star Plan of Technology & Science (NO.H020820790130) and the National Science Fund of China under contract 60203007.

References

[1] NIST, 2005 DUC Topic Development Task for Summarization, Document Understanding Conference, 2005.

[2] NIST, 2005 DUC Summary Writing Task, Document Understanding Conference, 2005.

[3] Dragomir R. Radev, Text summarization, ACM SIGIR tutorial, 2004.

[4] Morris, J. and Hirst, Lexical cohesion computed by thesaural relations as an indicator of the structure of text, Computational Linguistics 17(1): 21.43, 1991.

[5] Barzilay R. and Elhadad M, Using Lexical Chains for Summarization. ACL/EACL-97 summarization workshop Pp 10.18, Madrid, 1997.

[6] Brunn M., Chali Y., and Pancake C, Text summarization using lexical chains, In Workshop on Text Summarization in conjunction with the ACM SIGIR Conference 2001, New Orleans, Louisiana, 2001.

[7] Hammouda, K., Kamel, M.: Efficient phrase-based document indexing for web document clustering. IEEE Transactions on Knowledge and Data Engineering 16 (2004) 1279¨C1296.

[8] St. Onge, Detection and Correcting Malapropisms with Lexical Chains, M.Sc Thesis, University of

Toronto, Canada, 1995.
[9] Madhavi K. Ganapathiraju, Relevance of Cluster size in MMR based Summarizer: A Report 11-742: Self-paced lab in Information Retrieval, November 26, 2002
[10] William Doran, Nicola Stokes, Joe Carthy, and John Dunnion, Comparing Lexical Chain-based Summarization Approaches Using an Extrinsic Evaluation, 2004.
[11] DUC 2005 http://duc.nist.gov
[12] ISI ROUGE www.isi.edu/%7Ecyl/ROUGE/
[13] Miller G, Introduction to WordNet, 1990
[14] Paice C, and Husk G, Towards the automatic recognition of anaphoric features in English text: The impersonal pronoun “it”. Computer Speech and Language 2:109－132, 1991.
[15] Khaled M. Hammouda, Mohamed S. Kamel, Document Similarity Using a Phrase Indexing Graph Model, 2004
[16] The Stanford Natural Language Processing Group http://nlp.stanford.edu/software/tagger.shtml
[17] Dragomir R. Radev, Jahna Otterbacher_, Hong Qi_, and Daniel Tam, MEAD ReDUCs: Michigan at DUC 2003,2003
[18] Marcu, Daniel. 1997. From Discourse Structures to Text Summaries. In The Proceedings of the ACL’97/EACL’97 Workshop on Intelligent Scalable Text Summarization, 82–88
[19] Shuhua Liu, [ppt] Text Summarization -- In Search of Effective Ideas and Techniques，2004
[20] J. Carbonell and J. Goldstein. 1998. The use of MMR, diversity-based reranking for reordering documents and producing summaries. In Proc. ACM SIGIR, pages 335--336.