ISCAS at DUC 2006

Quan Zhou, Le Sun, Yuanhua Lv
Institute of Software
Chinese Academic of Sciences
{zhouquan03, sunle, yuanhua04}@iscas.cn

Abstract

This paper describes the architecture of the summarization system IS_SUM from Institute of Software, Chinese Academy of Sciences for DUC2006. The improvements on lexical chain algorithm are given in detail in order to enhance its efficiency and adapt it to query based summarization. We conclude our paper with the different evaluation results and the very primary analysis.

1 Introduction

The initial system IS_SUM v0.1 is designed during our participation to DUC2005. We analyzed the benefit and drawback of our algorithm based on our DUC2005 evaluation results and developed IS_SUM v0.2 for DUC 2006. We still use lexical chain algorithm as our main extraction algorithm. Some modifications are made to lexical chain algorithm in order to ameliorate the efficiency and adapt it to query based summarization.

The rest of the paper is organized as: Section 2 introduces lexical chain summarization algorithm; Section 3 describes system architecture; the evaluation result is given in section 4; section 5 shows the future work.

2 Lexical Chain Algorithm

There are about three famous implementations of lexical chains. Morris [3] first gave the logical description of his implementation which uses Roget dictionary in 1991. WordNet is introduced to lexical chains by Hirst [4]. He adopted a strategy that may choose the word's sense depending on those words occurred ahead of this word. Obviously this strategy may bring misunderstanding. Barzilay [5] designed an optimizing strategy which would insure considering all the sense of the candidate words. She applied the strategy to generate coherent summaries.

It's a pity that Barzilay only described a single document summarizer. When adapting it to multi-document and query based summarization, we found several limits:

1) With the size of input documents became bigger, the time/space cost of algorithm is increasing dramatically.
2) When summarizing for a query, the algorithm doesn't take any weight of query into account.

Those 2 points are the main drawbacks of the current lexical chain algorithm. We've solved them in our new implementation.

2.1 Candidate words

The preprocessing procedure is mainly for choosing the candidate words. We considered three types of the words in documents: the noun compounds, the noun entries and the name entities. We followed Barzilay [5] to introduce the noun compounds in summarization. We developed a simple script to recognize only noun-noun compounds in a single document. We used Stanford-tagger[6] for POS tagging to annotate noun entries and used GATE[7] to annotate name entities. The candidate words for lexical chains consist of noun compounds and noun entries. Name entities are used for sentence selection described in next Section.

2.2 Chain Building

In the view of text summarization, we only need those strongest lexical chains that can represent the main ideas of documents. These chains are available for summary. So we want to generate the strongest chains in priority when building the chains. Our implementation of chain building is as follows:

1) Sort the candidate words in the descend sequence of word frequency.
2) Pick up the first word in the sequence, get its sense set from WordNet; create empty chains for each sense; insert with the word in a specific sense.
3) For each chains created above, search the rest of the sequence to see if there's words can be insert into the

chain. Store the chains after searching.

4) Remove the first word from the sequence and process the second one using step 2) and 3) until half of the candidate words have been processed.
5) Merge the generated chains and remove some of them to insure one word occurs one time in the chain set; Score the last chains.

We processed the words of high frequency in priority. These words strengthened the score of the chain they exist. Strongest chains often contained these words. So we can get the strongest chains directly. Meanwhile, every chain word has a specific sense which eliminated the word misunderstanding.

2.2 Summary generating

Our summarizer is based on sentence selection. Firstly we picked some documents according to the relevance between the document and topic. The relevance is given by using a search engine. We scored candidate sentences with the following formula, and picked the sentences with high score:

$$Score = \alpha P(chain) + \beta P(entity) + \gamma P(query) + \delta P(rank)$$

Where $P(chain)$ *is the sum of the scores of the chains whose words contained by the sentence;* $P(entity)$ *is the number of name entity existing in both topic and the sentence;* $P(query)$ *is the sum of the co-occurrence times of key words in topic and sentences;* $P(rank)$ *is the rank of the document that contains the sentence.*

Each score has been normalized first. We selected the sentence with high score until reached the words number limit.

In our experiment, we found $P(chain)$ and $P(query)$ affect the system's performance notably. Finally 4 parameters were set to 0.4, 0.1, 0.4, and 0.1.

We compare the manual summary with our generated summary. The manual ones often contains shorter sentence so as to include more content in the limited words. So we decided to import a sentence compression algorithm. A sentence is first parsed to a syntactic tree. Then the tree is passed to a compressor based on a machine learning technique[9].

3 System Architecture

IS_SUM v0.2 is separated into 3 modules: Preprocessing, Modeling and Summarizing (See Figure 1). The arrows in Figure 1 show data flows in the system.

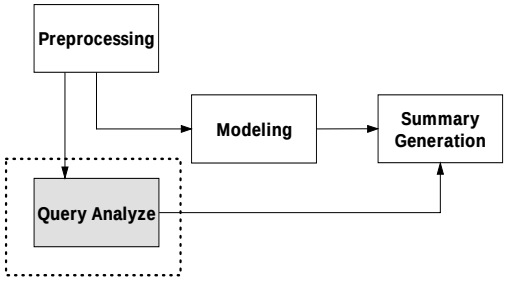

Figure 1 System Architecture

3.1 Preprocessing

We construct a simple search engine at first. It ranks all documents according to their relevance of the topic. The first 10 documents are considered as the most relevant documents with the topic. They are chosen to generate the summary.

Each document is processed with following operations: finding sentences boundary; lemmatizing words; calculating word frequency; doing POS tagging and Name Entity annotating; recognizing noun compounds and forming the candidate set at last.

3.2 Modeling

Build the chain set for each document with the algorithm described above. Then merge all the 10 chain set. Two chains can be merged only when there's at least one word existed in both chains and the word must have the same sense in both chains. Finally we got a chain set of input documents for summarization.

3.3 Summarizing

Score each candidate sentence first. Then pass the long sentences (more than 30 words) into a syntactic parser developed by Eugene Charniak[10], then compressed them into short ones with the compressor[9]. After compression, select the sentence with high score until the number of total words reaches 250.

4 Evaluation

We give two evaluation results of DUC evaluation and pyramid evaluation as follows:

4.1 Linguistic Quality

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.

Table1 Average Linguistic Quality

System	L1	L2	L3	L4	L5
IS-SUM	3.96	4.36	3.24	3.49	2.67

After improvement, we achieved higher score in Focus. We owe it to our sentence compression algorithm. The shorter sentences are more acceptable by summarization.

4.2 Responsiveness

Table 2 is the responsiveness score. We got rank 14 in 34 systems. We think the current query analysis really help to increase the responsiveness of summary. But it's still may lost some key content of summary. Key words and name entities are not enough for query based summary. We should continue to extract new features of topic.

Table 2 DUC 2006 Responsiveness

SYS	Responsiveness	SYS	Responsiveness
27	2.84	12	2.22
23	2.76	35	2.2
31	2.6	4	2.18
2	2.46	10	2.16
24	2.44	9	2.12
5	2.42	22	2.12
28	2.42	7	2.08
14	2.42	32	2.08
6	2.36	29	2.08
13	2.36	21	2.08
33	2.28	25	2.06
20	2.28	15	2.06
34	2.24	1	2
3	2.22	19	1.98

4.3 ROUGE Evaluation

Table 3 shows the ROUGE score of our summarizer.

Table 3 DUC 2006 ROUGE

System	ROUGE-2	ROUGE-SU4
Highest	0.09558	0.15529
IS_SUM	0.07923	0.13590
Baseline	0.04947	0.09788

We ranked 14 in 34 systems The performance of our system in this year is better than last year.

4.3 Pyramid Evaluation

We also participated in the Pyramid evaluation. We compared our system with the highest score of each document. The result is shown in Table 4

Table 3 IS_SUM v0.2 Pyramid

Doc No	Highest	IS_SUM
D0601	0. 264	0.2299
D0603	0.210	0.0857
D0605	0.175	0.0619
D0608	0.385	0.1538
D0614	0.476	0.3016
D0615	0.220	0.16
D0616	0.362	0.1552
D0617	0.290	0.2456
D0620	0.434	0.0482
D0624	0.406	0.3438
D0627	0.372	0.093
D0628	0.246	0.2
D0629	0.317	0.122
D0630	0.309	0.1912
D0631	0.587	0.4239
D0640	0.433	0.2667
D0643	0.478	0.318
D0645	0.3	0.3
D0647	0.264	0.1111
D0650	0.264	0.066

We ranked 8 in 24 systems. We observed the average score of our system is low. Pyramid evaluation is more precise in summary content. Sentence compressor helps to conclude more content in the summary. But we find same content included in different summary sentences. We may try to find some methods to eliminate the content overlapping existed in our summary in the following version of IS_SUM.

5 Conclusion and Future work

We made some modifications to lexical chain algorithm in order to ameliorate the efficiency and adapt it to query-based summarization. We extracted the sentences and formed the summary based on this improved lexical chain algorithm. From the primary analysis to the evaluation result we can see that the lexical chain method is useful on extract important sentences but not good at eliminate

same content unit. Lexical chain could be considered as a transition from extraction to abstraction. In order to produce abstraction summary, we are going to include some deep semantic analysis methods in the future.

Reference

[1] DUC 2005/2006 http://duc.nist.gov

[2] Quan Zhou，Le Sun，Jian-Yun Nie，IS_SUM :A Multi-Document Summarizer based on Document Index Graphic and Lexical Chains，2005

[3] 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.

[4] Hirst G. and St-Onge，D.，Lexical chains as representation of context for the detection and correction of malapropisms，，Cambridge，MA: The MIT Press，1998

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

[6] The Stanford Natural Language Processing Group http://nlp.stanford.edu/software/tagger.shtml

[7] GATE http://gate.ac.uk

[8] Pyramid Evaluation http://www1.cs.columbia.edu/~becky/DUC2006/2006-pyramid-guidelines.html

[9] Jacob Balazer, Sentence Compression Using a Machine Learning Technique, http:// www.eecs.umich.edu/~balazer/sc/

[10] Eugene Charniak, Edge-based best-first chart parsing, Proceedings of the Sixth Workshop on Very Large Corpora ,1998