Exploring Simultaneous Keyword and Key Sentence Extraction: Improve Graph-based Ranking Using Wikipedia

[1]Xun Wang [2]Lei Wang [1]Jiwei Li [1]Sujian Li
[1]Key Laboratory of Computational Linguistics, Peking University, Beijing, China
[2]HP Labs China, Beijing, China
{xunwang,jiweili,lisujian} @pku.edu.cn, lei.wang13@hp.com

ABSTRACT
Summarization and Keyword Selection are two important tasks in NLP community. Although both aim to summarize the source articles, they are usually treated separately by using sentences or words. In this paper, we propose a two-level graph based ranking algorithm to generate summarization and extract keywords at the same time. Previous works have reached a consensus that important sentence is composed by important keywords. In this paper, we further study the mutual impact between them through context analysis. We use Wikipedia to build a two-level concept-based graph, instead of traditional term-based graph, to express their homogenous relationship and heterogeneous relationship. We run PageRank and HITS rank on the graph to adjust both homogenous and heterogeneous relationships. A more reasonable relatedness value will be got for key sentence selection and keyword selection. We evaluate our algorithm on TAC 2011 data set. Traditional term-based approach achieves a score of 0.255 in ROUGE-1 and a score of 0.037 and ROUGE-2 and our approach can improve them to 0.323 and 0.048 separately.

Categories and Subject Descriptors
H.3.1 [Information Storage and Retrieval]: Content Analysis and Indexing – abstracting methods.

General Terms
Algorithms, Experimentation

Keywords
Summarization, Graph, Keyword, Markov Chain

1. INTRODUCTION
With the explosion of online information, the need to good automatic summarization techniques amplifies significantly. Key sentence extraction and keyword extraction are two important tasks targeted to represent the core content of one article by using the sentential and lexical expression separately. Many researches [1] [2] did great work in this area. Automatic summarization is widely used in many scenarios, i.e., portable devices (e.g., mobile phones and PDAs) need to provide a shortened and readable summary of news, emails, even text messages. Besides, search engines usually use a snippet which is the piece of content located under each hyperlink in the search result list to show the main information of the corresponding web page.

CIKM'12, October 29–November 2, 2012, Maui, HI, USA.

Most state-of-the-art text summarization systems use extraction approaches in combination with certain ranking algorithms, which are widely used for information retrieval such as HITS [3] and PageRank [4]. TextRank [5] and LexPageRank [6] are good implementation applying the graph-based ranking strategy. They construct a word connectivity matrix or a sentence connectivity matrix, and one layer graph is built on it.

Considering the relationship between sentences and words, Zha [1] proposes a method for keyword extraction and summary generation by exploiting the sentence-word relationships which indicate the impact of sentences on words. Wang et al. [2] improve this method by employing three kinds of relationships: sentence-sentence, word-word, and sentence-word. The interaction between sentences and words is taken into consideration for summarization and keywords selection.

The graph constructing is crucial for ranking algorithms. Previous graph constructing methods mainly based on the statistical features such as term frequencies and co-occurrences without considering the semantic understanding. In general, the relation of two sentences is weighed by calculating their term overlap and cosine value. The semantic relationship could not be accurately captured in the graphs. In addition, the relation of two words is also got by calculating the rate they appear concurrently.

To build a concept-based graph which represents the semantic relations between sentences or words, we involve Wikipedia concept in our system. The "import" word in a sentence will be linked to a specific Wikipedia article, namely Wikipedia concept. A word/phrase can be usually linked to multiple Wikipedia concepts, so we will analysis the context where it occurs and identify which concept it belongs to. Take the word of *apple* for example, there are two concepts associated with it: a fruit name and a company name. Given there is another concept *iphone* around *apple*, we think it is likely to be a company name. The relatedness among concepts can be measured according to their shared link-ins message [8]. As a sentence can be expressed by a concept vector, the relatedness of sentences will is got.

The heterogeneous relatedness between sentences and words has been proved effective in improving the quality of both summarization and keywords extraction [1][2]. In their work, the heterogeneous relationship are pre-calculated in a large corpus according to the co-occurrences of words in sentences. In this paper, we focus on building a more reasonable semantic graph by using Wikipedia concepts. The heterogeneous sentence-word relationships can be dynamically adjusted according to the involved sentences and words. For a concise article, it should be reasonable that the relatedness between important sentences and important words are closer than others. Based on this assumption, we design a two-level graph ranking algorithm to prove our idea. PageRank, [1] and [2] can all be seen as specific forms of the model. The difference is that [2] is a generalized form of [1] and

Input: On Oct. 31, 1999, a plane carrying 217 mostly Egyptian passengers crashed into the Atlantic Ocean off Massachusetts.

⇩

Results:
<DetectedTopic id="698" title="Atlantic Ocean" weight="0.909"/>
<DetectedTopic id="1645518" title="Massachusetts" weight="0.807"/>
<DetectedTopic id="34553" title="1999" weight="0.678"/>

Fig. 1 Concepts extracted by wikify

graph ranking algorithms, and our method is a generalized form of [2]. Markov models have been widely used for text processing. LexPageRank[5] and the work in [2] can be seen as the application of Markov models. Compared with them, our work is more general.

The paper is organized as follows: Section 2 introduces the construction of homogeneous semantic graph, and Section 3 presents the adjustment of heterogeneous relationship. Section 4 illustrates the experimental results and analysis. Finally Section 5 concludes this paper.

2. TWO-LEVEL GRAPH CONSTRUCTION

There are three kinds of relations in a two-level graph: word-word, sentence-sentence and word-sentence. The first two relations are homogenous and the last one is heterogeneous. Firstly, we build the word-word (concept-concept) graph. Secondly, we construct the sentence-sentence graph and calculate the relatedness values between sentences and words (concepts). Thirdly, we will further adjust the relatedness value of two graphs by considering sentence-word mutual impact. Eventually, a more accurate semantic graph will be obtained to present the article.

2.1 Homogenous Relationship Construction

To build a semantic graph, we use Wikipedia as ontology knowledge. Containing more than 300 million manually labeled entries and more than 90 million links, it's the largest encyclopedia. In Wikipedia, each entry has a page to explain its meanings and pages are linked together via hyperlinks. A lot of work use Wikipedia to calculate the semantic relatedness between these entries. Mihalcea et al. develop a tool (called wikify) which can disambiguate the word and indicate which page it belongs to. The relevance between two Wikipedia pages is calculated by using the sharing link-in message. For two word *k* and *l*:

$$\mathrm{Rel}(\mathrm{k},\mathrm{l}) \propto \frac{\bigcap(\text{linkins of l}, \text{linkins of k})}{\bigcup(\text{linkins of l}, \text{linkins of k})}$$

The relevance score reflects the relationship between words. In addition, the scores can also be leveraged for disambiguating words. Each word will be assigned a score reflecting its relatedness with the sentence. We use these extracted keywords with corresponding scores to represent the original sentences and build the sentence-sentence semantic graph.

Fig 1 shows an example. Three terms, including Atlantic Ocean, Massachusett and 1999, are extracted by using the *wikify* package. The relatedness value of 'Atlantic Ocean' is 0.909 which is got by calculating the percentage of shared link-ins between itself and other words the sentence contains. After it, a sentence S_i can be expressed by a vector:

$$S_i = \{C_{ik}: \mathrm{Score}_{ik}\}$$

The key-value pair $C_{ik}: Score_{ik}$ contains Wikipedia entry (article ID in Wikipedia) and its relatedness value to the sentence S_i. For example, the sentence shown in Fig. 2 can be expressed in the following form:

$$S = \left\{ \begin{matrix} \text{Atlantic Ocean: } 0.909 \\ \text{Massachusetts: } 0.807 \\ \text{1999: } 0.678 \end{matrix} \right\}.$$

We assume this vector partly capture the semantic of the original sentence. What needs to be clarified is that not all the sentences contain Wikipedia entries and those sentences without any Wikipedia entries will be ignored. When building the sentence-sentence graph, the relevance of two sentences:S_iS_j is defined as the following:

if $S_i = \{C_{ik}: \mathrm{Score}_{ik}\}, S_j = \{C_{jl}: \mathrm{Score}_{jl}\}$,

then $\mathrm{Rel(ij)} = \sum_{kl} \mathrm{Score}_{ik}\mathrm{Score}_{jl}\mathrm{Rel}(C_{ik}C_{jl})$;

Score_{ik} evaluates the relevance between sentence i and word k. The relevance value between sentences i and j, namely $\mathrm{Rel(ij)}$, is represented by the sum of pair-wise similarity values for the words they contain.

2.2 Heterogeneous Relationship Adjustment

In the vector shown in Section 2.1, "Atlantic Ocean" has the highest score, but it's not the most representative term in the vector. "Egyptian" and "crashed" hold more information than "Atlantic Ocean" here. The reason is that the relatedness score is calculated in a large corpus without considering the local context. In Wikipedia, the page of "Atlantic Ocean" has much more link-ins than others because many pages citing it would provide a link to it. Hence "Atlantic Ocean" shares the most link-ins with nearby words. It's reasonable but not that suitable. Ignoring the context would usually cause such problems. In previous works, relatedness is also calculated in a corpus ignoring the context. By adjusting the scores according to local information, we try to solve this problem to some extent. We assume that relations between important sentences and important concepts are closer than that between others. It is reasonable for a concise article. So we adjust the sentence-word relatedness scores by their own weights:

$$Rel(S_iC_{ik}) = \beta Score_{ik} + (1-\beta) Score_i * Score_k$$

Score_i is the weight of sentence i and Score_k is the score assigned to word k. Relatedness scores between sentences and words with high scores would increase. In previous works, there's no such adjustment. Here they are influenced by their own weights and changes in the iterative computation until convergence. The parameter of β can be used to tune the weights of initial scores and the adjustment. If $\beta = 1$, there would be no adjustment, and it would be exactly what [2] has done. The adjustment would influence the scores of sentences and words linked together in the next iteration. After the calculation of relevance, we get a two-level graph for the source article. When running the ranking algorithm, the relatedness scores will be adjusted and we will get a more reasonable graph to reflect the semantic relationship.

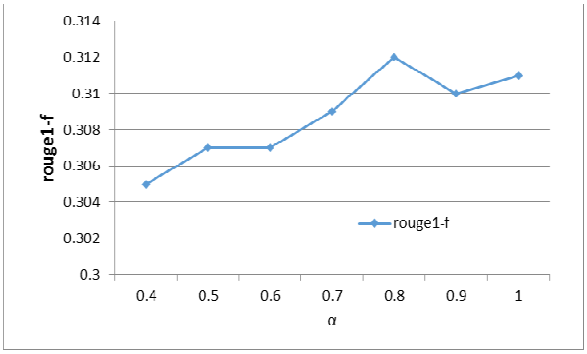

Fig. 2 vs rouge-1 Fscore

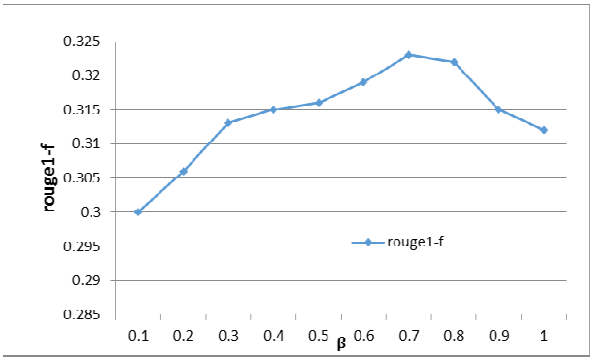

Fig. 3 vs rouge-1 Fscore

3. TWO-LEVEL GRAPH RANKING

We use two dependent Markov processes interacting with each other to perform the ranking on graph.

A typical Markov Chain can be represented by the initial status vector and the transition matrix. The stationary distribution can be calculated via iterations. We have two chains, each for one level, and they interact with each other.

Initial Distribution: represents the initial scores of sentences, represents the initial scores of concepts. They are non-negative and normalized vectors.

Transition Matrix:

, here is the relevance between sentence and sentence .

, here is the relevance between word and word .

——————————— ……. (1)

, here is the relevance between word and word .

, here is the relevance between word and sentence .

All matrixes are normalized that the sum of each row equals to 1 as . Obviously we get before they are normalized.

A major difference between our method and previous methods is that when running the algorithm, we adjust the relevance between sentences and concepts according to their own weights.

For the k-th iteration, k>1, firstly we update the relevance values between sentences and concepts:

……. (2)

And then the relevance matrix is normalized again by row and by column and we get the new . After the normalization, the relevance scores for sentences and concepts are updated as:

…… (3)

…… (4)

(3) and (4) are repeated until stationary distributions , are achieved. They are the final scores of sentences and words.When , the relevance values between sentences and words do not change from one iteration to another. That is the situation in [2]. Process (2) (3) and (4) are repeated until converge. is assigned to sentences as final scores and assigned to concepts. Top-ranked sentences are selected as summary and top-ranked words as keyword after proper post-processing. We also get new scores for sentence-word relatedness by taking context into consideration.

4. EXPERIMENT

4.1 Experiment Set-up

We perform a series of experiments to evaluate the performance of our new graph and ranking algorithm. We built an automatic summarization system in which traditional term-based graph and concept-based graph are constructed respectively. Some popular ranking algorithms such as PageRank, HITS and a hybrid Rank (combining the results of Hits Rank and PageRank) are selected. Our proposed compound Markov chain model is applied on both graphs. Experiments were done on TAC 2011 automatic summarization task data which contains more than 200 news reports. 10% are used for tuning parameters and 90% are used for testing. ROUGE was used as the evaluation tool. ROUGE has been adopted by TAC for years and has also been widely used by summarization researchers because the ROUGE scores are highly homogenous with the scores assigned by human. For the keyword selection, we use f-score at top 5 and top 10 words.

4.2 Parameters Tuning

For choosing the best values for , we implement classic PageRank algorithm and Wan(2007)'s [2] unified model. From Formula (3) and (4) we can see that represents the weight of the homogeneous relationship, that is the impact of sentences to sentences and that of words to words, while (1- represents the weight of heterogeneous relationship. We randomly choose 22 documents for parameters tuning and the left are used for testing.

A series of experiments are conducted to find the suitable value for . Firstly we set =1, which means the local context are ignored completely. That is .

Then we need to find the most suitable for our system. Obviously when , it's the same as the traditional PageRank method: the heterogeneous relationships between sentences and concepts are ignored and only the homogenous relations are taken into consideration. When , only the heterogeneous relationships are used which does not makes sense obviously. So we experiment on . Results are shown in Fig2.

From the results we can see when , it performs best. In following experiments would be set as 0.8. Since 0.8>0.5, conclusion that the impact of homogeneous relations between sentences is more important than the heterogeneous relationships can be drawn for summarization task. Considering the impact of

concepts on sentences, we can achieve better results than traditional ones. $\boldsymbol{\beta}$ in Formula (2) represents the weight of the old scores and (1- $\boldsymbol{\beta}$) represents the impact of local context when updating these scores. We also try to find a proper value for $\boldsymbol{\beta}$ that works well with our corpus and model. Firstly we set $\boldsymbol{\alpha = 0.8}$. When $\boldsymbol{\beta = 1}$, it would be exactly the same as above. And when $\boldsymbol{\beta = 0}$, the relatedness between sentence and concepts are completely decide by the local context. Fig 3 shows the various results as $\boldsymbol{\beta}$ changes. From Fig 3, we can see the best value for $\boldsymbol{\beta}$ is 0.7 in our testing data set.

4.3 Summarization Evaluation

We compare our algorithm with traditional ranking algorithms on traditional term-based graphs and concept-based graph. Considering the fairness, these methods follow the same pre-processing and post-processing steps as possible. And in our method, $\alpha = 0.8$, $\beta = 0.7$. Results are as follows:

Table 1: Results of Traditional and Our Methods

Algorithms	Graph	ROUGE-1	ROUGE-2
HITS	Traditional	0.285	0.033
	Semantic	0.309	0.05
PageRank	Traditional	0.285	0.033
	Semantic	0.311	0.052
Hybrid	Traditional	0.285	0.033
	Semantic	0.194	0.025
Ours	Traditional	0.255	0.037
	Semantic	**0.323**	**0.048**

Concept graphs show better performance than traditional ones. Our new method with concept graphs shows the best results.

4.4 Keywords Evaluation

We manually selected the keywords for 22 documents form TAC 2011 corpus. And then we perform our algorithm and traditional ones. For simplicity, we use only the LexRank algorithm as a contrast experiment. For each document, 10 keywords are selected, including words and phrases which is composed by two or more words. The compound Markov method uses the same graph and follows the same strategy in the previous experiment and the parameters are the same, too. For both methods, the top-10 words (including phrases) are selected for each article. The result is shown in table 2, from which, we can see that our method performs better than traditional LexRank algorithm.

	Top5	Top10
LexRank	29%	44%
Ours	32%	53%

Table 2: Keyword evaluation

The key words extraction also benefits from the sentences selection and the semantic graphs. A byproduct is the adjustment of relatedness scores between sentences and words. Take the previous sentence as an example, the relatedness scores are shown in table 3. The new scores are relatively more reasonable than old ones. Hence this method can also be used to improve such annotation.

Word	Old Score	New Score
Atlantic Ocean	0.909	0.807 ↓
Massachusetts	0.8	0.778 ↑
1999	0.678	0.720 ↑

Table 3: Relatedness scores before and after adjustment

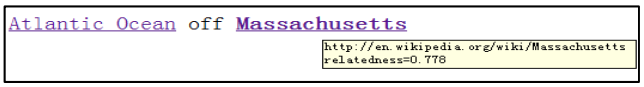

Fig 4: A sample of results

When displayed, hyperlinks are used to provide more information for users' conveniences, as Fig4 shows. Keywords are in bold, as "**Massachusetts**". Note it's just a segment of the real summary, but still can demonstrate our advantages.

5. Conclusions

We propose a two-level graph ranking algorithm for keyword and key sentence selection. Previous Markov models such as PageRank can be described in this unified model. Two tasks both benefit from the improvement of calculation of both homogenous and heterogeneous relations. For future work, we will try other methods to adjust the graph, to gain a more reasonable semantic graph exploiting the context. We can also adjust the homogenous relatedness scores to get a more suitable semantic graph.

6. Acknowledgement

This work is partly supported by National Nature Science Foundation program of China (No: 90920011), National Social Science Foundation program (No: 10CYY023), National Key Technology R&D Program (No: 2011BAH10B04-03), and National High Technology Research and Development Program of China (No. 2012AA011101).

7. REFERENCES

[1] H. Zha. 2002. Generic summarization and keyphrase extraction using mutual reinforcement principle and sentence clustering. SIGIR2002.

[2] X. Wang, J. Yang, J. Xiao,2007, Towards an Iterative Reinforcement Approach for Simultaneous Document Summarization and Keyword Extraction, ACL2007.

[3] J. M. Kleinberg. 1999. Authoritative sources in a hyperlinked environment. Journal of the ACM , 46(5):604–632.

[4] S. Brin, L. Page. 1998. The anatomy of a large scale hypertextual Web search engine. Computer Networks and ISDN Systems, 30(1–7).

[5] R. Mihalcea and P. Tarau. 2004. TextRank: Bringing order into texts. EMNLP2004.

[6] G. ErKan and D. R. Radev. 2004. LexPageRank: Prestige in multi-document text summarization. EMNLP2004.

[7] R. Mihalcea and A. Csomai. 2007. Wikify!: linking documents to encyclopedic knowledge. CIKM'07

[8] M. David, Ian H. Witten, 2008, Learning to Link with Wikipedia, CIKM'08.