PNR^2: Ranking Sentences with Positive and Negative Reinforcement for Query-Oriented Update Summarization

Li Wenjie[1], Wei Furu[1,2], Lu Qin[1], He Yanxiang[2]

[1]Department of Computing
The Hong Kong Polytechnic University, HK
{csfwei, cswjli, csluqin}
@comp.polyu.edu.hk

[2]Department of Computer Science
and Technology, Wuhan University, China
{frwei, yxhe@whu.edu.cn}

Abstract

Query-oriented update summarization is an emerging summarization task very recently. It brings new challenges to the sentence ranking algorithms that require not only to locate the important and query-relevant information, but also to capture the new information when document collections evolve. In this paper, we propose a novel graph based sentence ranking algorithm, namely PNR^2, for update summarization. Inspired by the intuition that "a sentence receives a *positive* influence from the sentences that correlate to it in the same collection, whereas a sentence receives a *negative* influence from the sentences that correlates to it in the different (perhaps previously read) collection", PNR^2 models both the positive and the negative mutual reinforcement in the ranking process. Automatic evaluation on the DUC 2007 data set pilot task demonstrates the effectiveness of the algorithm.

1 Introduction

The explosion of the WWW has brought with it a vast board of information. It has become virtually impossible for anyone to read and understand large numbers of individual documents that are abundantly available. Automatic document summarization provides an effective means to manage such an exponentially increased collection of information and to support information seeking and condensing goals.

The main evaluation forum that provides benchmarks for researchers working on document summarization to exchange their ideas and experiences is the Document Understanding Conferences (DUC). The goals of the DUC evaluations are to enable researchers to participate in large-scale experiments upon the standard benchmark and to increase the availability of appropriate evaluation techniques. Over the past years, the DUC evaluations have evolved gradually from single-document summarization to multi-document summarization and from generic summarization to query-oriented summarization. Query-oriented multi-document summarization initiated in 2005 aims to produce a short and concise summary for a collection of topic relevant documents according to a given query that describes a user's particular interests.

Previous summarization tasks are all targeted on a single document or a static collection of documents on a given topic. However, the document collections can change (actually grow) dynamically when the topic evolves over time. New documents are continuously added into the topic during the whole lifecycle of the topic and normally they bring the new information into the topic. To cater for the need of summarizing a dynamic collection of documents, the DUC evaluations piloted update summarization in 2007. The task of update summarization differs from previous summarization tasks in that the latter aims to dig out the salient information in a topic while the former cares the information not only salient but also novel.

Up to the present, the predominant approaches in document summarization regardless of the nature and the goals of the tasks have still been built upon the sentence extraction framework.

Proceedings of the 22nd International Conference on Computational Linguistics (Coling 2008), pages 489–496
Manchester, August 2008

Under this framework, sentence ranking is the issue of most concern. In general, two kinds of sentences need to be evaluated in update summarization, i.e. the sentences in an early (old) document collection A (denoted by S^A) and the sentences in a late (new) document collection B (denoted by S^B). Given the changes from S^A to S^B, an update summarization approach may be concerned about four ranking issues: (1) rank S^A independently; (2) re-rank S^A after S^B comes; (3) rank S^B independently; and (4) rank S^B given that S^A is provided. Among them, (4) is of most concern. It should be noting that both (2) and (4) need to consider the influence from the sentences in the same and different collections.

In this study, we made an attempt to capture the intuition that

> "A sentence receives a *positive* influence from the sentences that correlate to it in the same collection, whereas a sentence receives a *negative* influence from the sentences that correlates to it in the different collection."

We represent the sentences in A or B as a text graph constructed using the same approach as was used in Erkan and Radev (2004a, 2004b). Different from the existing PageRank-like algorithms adopted in document summarization, we propose a novel sentence ranking algorithm, called PNR^2 (Ranking with Positive and Negative Reinforcement). While PageRank models the positive mutual reinforcement among the sentences in the graph, PNR^2 is capable of modeling both positive and negative reinforcement in the ranking process.

The remainder of this paper is organized as follows. Section 2 introduces the background of the work presented in this paper, including existing graph-based summarization models, descriptions of update summarization and time-based ranking solutions with web graph and text graph. Section 3 then proposes PNR^2, a sentence ranking algorithm based on positive and negative reinforcement and presents a query-oriented update summarization model. Next, Section 4 reports experiments and evaluation results. Finally, Section 5 concludes the paper.

2 Background and Related Work

2.1 Previous Work in Graph-based Document Summarization

Graph-based ranking algorithms such as Google's PageRank (Brin and Page, 1998) and Kleinberg's HITS (Kleinberg, 1999) have been successfully used in the analysis of the link structure of the WWW. Now they are springing up in the community of document summarization. The major concerns in graph-based summarization researches include how to model the documents using text graph and how to transform existing web page ranking algorithms to their variations that could accommodate various summarization requirements.

Erkan and Radev (2004a and 2004b) represented the documents as a weighted undirected graph by taking sentences as vertices and cosine similarity between sentences as the edge weight function. An algorithm called LexRank, adapted from PageRank, was applied to calculate sentence significance, which was then used as the criterion to rank and select summary sentences. Meanwhile, Mihalcea and Tarau (2004) presented their PageRank variation, called TextRank, in the same year. Besides, they reported experimental comparison of three different graph-based sentence ranking algorithms obtained from Positional Power Function, HITS and PageRank (Mihalcea and Tarau, 2005). Both HITS and PageRank performed excellently.

Likewise, the use of PageRank family was also very popular in event-based summarization approaches (Leskovec et al., 2004; Vanderwende et al., 2004; Yoshioka and Haraguchi, 2004; Li et al., 2006). In contrast to conventional sentence-based approaches, newly emerged event-based approaches took event terms, such as verbs and action nouns and their associated named entities as graph nodes, and connected nodes according to their co-occurrence information or semantic dependency relations. They were able to provide finer text representation and thus could be in favor of sentence compression which was targeted to include more informative contents in a fixed-length summary. Nevertheless, these advantages lied on appropriately defining and selecting event terms.

All above-mentioned representative work was concerned with generic summarization. Later on, graph-based ranking algorithms were introduced in query-oriented summarization too when this new challenge became a hot research topic recently. For example, a topic-sensitive version of PageRank was proposed in (OtterBacher et al., 2005). The same algorithm was followed by Wan et al. (2006) and Lin et al. (2007) who further investigated on its application in query-oriented update summarization.

2.2 The DUC 2007 Update Summarization Task Description

The DUC 2007 update summarization pilot task is to create short (100 words) multi-document summaries under the assumption that the reader has already read some number of previous documents. Each of 10 topics contains 25 documents. For each topic, the documents are sorted in chronological order and then partitioned into three collections, "A", "B" and "C". The participants are then required to generate (1) a summary for "A"; (2) an update summary for "B" assuming documents in "A" have already been read; and (3) an update summary for "C" assuming documents in "A" and "B" have already been read. Growing out of the DUC 2007, the Text Analysis Conference (TAC) 2008 planed to keep only the DUC 2007 task (1) and (2).

Each topic collection in the DUC 2007 (will also in the TAC 2008) is accompanied with a query that describes a user's interests and focuses. System-generated summaries should include as many responses relevant to the given query as possible. Here is a query example from the DUC 2007 document collection "D0703A".

<*topic*>
<*num*> D0703A </*num*>
<*title*> Steps toward introduction of the Euro. </*title*>
<*narr*> Describe steps taken and worldwide reaction prior to introduction of the Euro on January 1, 1999. Include predictions and expectations reported in the press. </*narr*>
</*topic*> [D0703A]

Update summarization is definitely a time-related task. An appropriate ranking algorithm must be the one capable of coping with the change or the time issues.

2.3 Time-based Ranking Solutions with Web Graph and Text Graph

Graph based models in document summarization are inspired by the idea behind web graph models which have been successfully used by current search engines. As a matter of fact, adding time dimension into the web graph has been extensively studied in recent literature.

Basically, the evolution in the web graph stems from (1) adding new edges between two existing nodes; (2) adding new nodes in the existing graph (consequently adding new edges between the existing nodes and the new nodes or among the new nodes); and (3) deleting existing edges or nodes. Berberich et al. (2004 and 2005) developed two link analysis methods, i.e. T-Rank Light and T-Rank, by taking into account two temporal aspects, i.e. freshness (i.e. timestamp of most recent update) and activity (i.e. update rates) of the pages and the links. They modeled the web as an evolving graph in which each nodes and edges (i.e. web pages and hyperlinks) were annotated with time information. The time information in the graph indicated different kinds of events in the lifespan of the nodes and edges, such as creation, deletion and modifications. Then they derived a subgraph of the evolving graph with respect to the user's temporal interest. Finally, the time information of the nodes and the edges were used to modify the random walk model as was used in PageRank. Specifically, they used it to modify the random jump probabilities (in both T-Rank Light and T-Rank) and the transition probabilities (in T-Rank only).

Meanwhile, Yu et al. (2004 and 2005) introduced a time-weighted PageRank, called TimedPageRank, for ranking in a network of scientific publications. In their approach, citations were weighted based on their ages. Then a post-processing step decayed the authority of a publication based on the publication's age. Later, Yang et al. (2007) proposed TemporalRank, based on which they computed the page importance from two perspectives: the importance from the current web graph snapshot and the accumulated historical importance from previous web graph snapshot. They used a kinetic model to interpret TemporalRank and showed it could be regarded as a solution to an ordinary differential equation.

In conclusion, Yu et al. tried to cope with the problem that PageRank favors over old pages whose in-degrees are greater than those of new pages. They worked on a static single snapshot of the web graph, and their algorithm could work well on all pages in the web graph. Yang et al., on the other hand, worked on a series of web graphs at different snapshots. Their algorithm was able to provide more robust ranking of the web pages, but could not alleviate the problem carried by time dimension at each web graph snapshot. This is because they directly applied the original PageRank to rank the pages. In other words, the old pages still obtained higher scores while the newly coming pages still got lower scores. Berberich et al. focused their efforts on the evolution of nodes and edges in the web graph. However, their algorithms did not work

when the temporal interest of the user (or query) was not available.

As for graph based update summarization, Wan (2007) presented the TimedTextRank algorithm by following the same idea presented in the work of Yu et al. Given three collections of chronologically ordered documents, Lin et al. (2007) proposed to construct the TimeStamped graph (TSG) graph by incrementally adding the sentences to the graph. They modified the construction of the text graph, but the ranking algorithm was the same as the one proposed by OtterBacher et al.

Nevertheless, the text graph is different from the web graph. The evolution in the text graph is limited to the type (2) in the web graph. The nodes and edges can not be deleted or modified once they are inserted. In other words, we are only interested in the changes caused when new sentences are introduced into the existing text graph. As a result, the ideas from Berberich et al. cannot be adopted directly in the text graph. Similarly, the problem in web graph as stated in the work of Yu et al. (i.e. "new pages, which may be of high quality, have few or no in-links and are left behind.") does not exist in the text graph at all. More precisely, the new coming sentences are equally treated as the existing sentences, and the degree (in or out) of the new sentences are also equally accumulated as the old sentences. Directly applying the ideas from the work of Yu et al. does not always make sense in the text graph. Recall that the main task for sentence ranking in update summarization is to rank S^B given S^A. So the idea from Yang et al. is also not applicable.

In fact, the key points include not only maximizing the importance in the current new document collection but also minimizing the redundancy to the old document collection when ranking the sentences for update summarization. Time dimension does contribute here, but it is not the only way to consider the changes. Unlike the web graph, the easily-captured content information in a text graph can provide additional means to analyze the influence of the changes.

To conclude the previous discussions, adding temporal information to the text graph is different from it in the web graph. Capturing operations (such as addition, deletion, modification of web pages and hyperlinks) is most concerned in the web graph; however, prohibiting redundant information from the old documents is the most critical issue in the text graph.

3 Positive and Negative Reinforcement Ranking for Update Summarization

Existing document summarization approaches basically follow the same processes: (1) first calculate the significance of the sentences with reference to the given query with/without using some sorts of sentence relations; (2) then rank the sentences according to certain criteria and measures; (3) finally extract the top-ranked but non-redundant sentences from the original documents to create a summary. Under this extractive framework, undoubtedly the two critical processes involved are sentence ranking and sentence selection. In the following sections, we will first introduce the sentence ranking algorithm based on ranking with positive and negative reinforcement, and then we present the sentence selection strategy.

3.1 Ranking with Positive and Negative Reinforcement (PNR^2)

Previous graph-based sentence ranking algorithms is capable to model the fact that a sentence is important if it correlates to (many) other important sentences. We call this positive mutual reinforcement. In this paper, we study two kinds of reinforcement, namely positive and negative reinforcement, among two document collections, as illustrated in Figure 1.

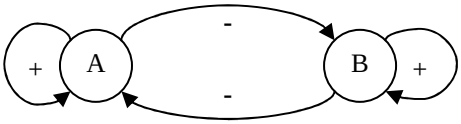

Figure 1 Positive and Negative Reinforcement

In Figure 1, "A" and "B" denote two document collections about the same topics ("A" is the old document collection, "B" is the new document collection), S^A and S^B denote the sentences in "A" and "B". We assume:

1. S^A performs positive reinforcement on its own internally;
2. S^A performs negative reinforcement on S^B externally;
3. S^B performs negative reinforcement on S^A externally;
4. S^B performs positive reinforcement on its own internally.

Positive reinforcement captures the intuition that a sentence is more important if it associates to the other important sentences in the same collection. Negative reinforcement, on the other hand, reflects the fact that a sentence is less

important if it associates to the important sentences in the other collection, since such a sentence might repeat the same or very similar information which is supposed to be included in the summary generated for the other collection.

Let R_A and R_B denote the ranking of the sentences in A and B, the reinforcement can be formally described as

$$\begin{cases} R_A^{(k+1)} = \alpha_1 \cdot M_{AA} \cdot R_A^{(k)} + \beta_1 \cdot M_{AB} \cdot R_B^{(k)} + \gamma_1 \cdot \vec{p}_A \\ R_B^{(k+1)} = \beta_2 \cdot M_{BA} \cdot R_A^{(k)} + \alpha_2 \cdot M_{BB} \cdot R_B^{(k)} + \gamma_2 \cdot \vec{p}_B \end{cases} \quad (1)$$

where the four matrices M_{AA}, M_{BB}, M_{AB} and M_{BA} are the affinity matrices of the sentences in S^A, in S^B, from S^A to S^B and from S^B to S^A. $W = \begin{bmatrix} \alpha_1 & \beta_1 \\ \beta_2 & \alpha_2 \end{bmatrix}$ is a weight matrix to balance the reinforcement among different sentences. Notice that $\beta_1, \beta_2 < 0$ such that they perform negative reinforcement. $\vec{p}_A$ and $\vec{p}_B$ are two bias vectors, with $0 < \gamma_1, \gamma_2 < 1$ as the damping factors. $\vec{p}_A = \left[1/n\right]_{n\times 1}$, where n is the order of M_{AA}. $\vec{p}_B$ is defined in the same way. We will further define the affinity matrices in section 3.2 later. With the above reinforcement ranking equation, it is also true that

1. A sentence in S^B correlates to many new sentences in S^B is supposed to receive a high ranking from R_B, and
2. A sentence in S^B correlates to many old sentences in S^A is supposed to receive a low ranking from R_B.

Let $R = [R_A \; R_B]^T$ and $\vec{p} = [\gamma_1 \cdot \vec{p}_A \; \gamma_2 \cdot \vec{p}_B]^T$, then the above iterative equation (1) corresponds to the linear system,

$$(I - M) \cdot R = \vec{p} \quad (2)$$

where, $M = \begin{bmatrix} \alpha_1 M_{AA} & \beta_1 M_{AB} \\ \beta_2 M_{BA} & \alpha_2 M_{BB} \end{bmatrix}$.

Up to now, the PNR^2 is still query-independent. That means only the content of the sentences is considered. However, for the tasks of query-oriented summarization, the reinforcement should obviously bias to the user's query. In this work, we integrate query information into PNR^2 by defining the vector $\vec{p}$ as $\vec{p}_i = rel(s_i \mid q)$, where $rel(s_i \mid q)$ denotes the relevance of the sentence s_i to the query q.

To guarantee the solution of the linear system Equation (2), we make the following two transformations on M. First M is normalized by columns. If all the elements in a column are zero, we replace zero elements with $1/n$ (n is the total number of the elements in that column). Second, M is multiplied by a decay factor θ ($0 < \theta < 1$), such that each element in M is scaled down but the meaning of M will not be changed.

Finally, Equation (2) is rewritten as,

$$(I - \theta \cdot M) \cdot R = \vec{p} \quad (3)$$

The matrix $(I - \theta \cdot M)$ is a strictly diagonally dominant matrix now, and the solution of the linear system Equation (3) exists.

3.2 Sentence Ranking based on PNR^2

We use the above mentioned PNR^2 framework to rank the sentences in both S^A and S^B simultaneously. Section 3.2 defines the affinity matrices and presents the ranking algorithm.

The affinity (i.e. similarity) between two sentences is measured by the cosine similarity of the corresponding two word vectors, i.e.

$$M[i, j] = sim(s_i, s_j) \quad (4)$$

where $sim(s_i, s_j) = \frac{\vec{s}_i \bullet \vec{s}_j}{\|\vec{s}_i\| \cdot \|\vec{s}_j\|}$. However, when calculating the affinity matrices M_{AA} and M_{BB}, the similarity of a sentence to itself is defined as 0, i.e.

$$M[i, j] = \begin{cases} sim(s_i, s_j) & i \neq j \\ 0 & i = j \end{cases} \quad (5)$$

Furthermore, the relevance of a sentence to the query q is defined as

$$rel(s_i, q) = \frac{\vec{s}_i \bullet \vec{q}}{\|\vec{s}_i\| * \|\vec{q}\|} \quad (6)$$

Algorithm 1. *RankSentence*(S^A, S^B, q)

Input: The old sentence set S^A, the new sentence set S^B, and the query q.
Output: The ranking vectors R of S^A and S^B.
1: Construct the affinity matrices, and set the weight matrix W;
2: Construct the matrix $A = (I - \theta \cdot M)$.
3: Choose (randomly) the initial non-negative vectors $R^{(0)} = [1 \cdots 1]^T$;
4: $k \leftarrow 0$, $\nabla \leftarrow 0$;
5: **Repeat**
6: $R_i^{(k+1)} = \frac{1}{a_{ij}} \left(\vec{p}_i - \sum_{j<i} a_{ij} R_j^{(k+1)} - \sum_{j>i} a_{ij} R_j^{(k)} \right)$;
7: $\nabla \leftarrow \max\left(\left\| R^{(k+1)} - R^{(k)} \right\| \right)$;
8: $R^{(k+1)}$ is normalized such that the maximal element in $R^{(k+1)}$ is 1.

9: $k \leftarrow k + 1$;
10: **Until** $\nabla < \zeta$ 1;
11: $R \leftarrow R^{(k)}$;
12: **Return**.

Now, we are ready to adopt the Gauss-Seidel method to solve the linear system Equation (3), and an iterative algorithm is developed to rank the sentences in S^A and S^B.

After sentence ranking, the sentences in S^B with higher ranking will be considered to be included in the final summary.

3.3 Sentence Selection by Removing Redundancy

When multiple documents are summarized, the problem of information redundancy is more severe than it is in single document summarization. Redundancy removal is a must. Since our focus is designing effective sentence ranking approach, we apply the following simple sentence selection algorithm.

Algorithm 2. *GenerateSummary*(*S*, *length*)

Input: sentence collection *S* (ranked in descending order of significance) and *length* (the given summary length limitation)
Output: The generated summary Π
$\Pi \leftarrow \{\}$;
$l \leftarrow$ *length*;
For $i \leftarrow 0$ to |*S*| **do**
  $threshold \leftarrow \max\left(sim(s_i, s) \middle| s \in \Pi\right)$;
  If *threshold* <= 0.9[2] **do**
    $\Pi \leftarrow \Pi \bigcup s_i$;
    $l \leftarrow l - len(s_i)$;
    If (l <= 0) **break**;
  End
End
Return Π.

4 Experimental Studies

4.1 Data Set and Evaluation Metrics

The experiments are set up on the DUC 2007 update pilot task data set. Each collection of documents is accompanied with a query description representing a user's information need. We simply focus on generating a summary for the document collection "B" given that the user has read the document collection "A", which is a typical update summarization task.

Table 1 below shows the basic statistics of the DUC 2007 update data set. Stop-words in both documents and queries are removed[3] and the remaining words are stemmed by Porter Stemmer[4]. According to the task definition, system-generated summaries are strictly limited to 100 English words in length. We incrementally add into a summary the highest ranked sentence of concern if it doesn't significantly repeat the information already included in the summary until the word limitation is reached.

	A	B
Average number of documents	10	10
Average number of sentences	237.6	177.3

Table 1. Basic Statistics of DUC2007 Update Data Set

As for the evaluation metric, it is difficult to come up with a universally accepted method that can measure the quality of machine-generated summaries accurately and effectively. Many literatures have addressed different methods for automatic evaluations other than human judges. Among them, ROUGE[5] (Lin and Hovy, 2003) is supposed to produce the most reliable scores in correspondence with human evaluations. Given the fact that judgments by humans are time-consuming and labor-intensive, and more important, ROUGE has been officially adopted for the DUC evaluations since 2005, like the other researchers, we also choose it as the evaluation criteria.

In the following experiments, the sentences and the queries are all represented as the vectors of words. The relevance of a sentence to the query is calculated by cosine similarity. Notice that the word weights are normally measured by the document-level TF*IDF scheme in conventional vector space models. However, we believe that it is more reasonable to use the sentence-level inverse sentence frequency (ISF) rather than document-level IDF when dealing with sentence-level text processing. This has been verified in our early study.

4.2 Comparison of Positive and Negative Reinforcement Ranking Strategy

The aim of the following experiments is to investigate the different reinforcement ranking strategies. Three algorithms (i.e. PR(B),

[1] ζ is a pre-defined small real number as the convergence threshold.
[2] In fact, this is a tunable parameter in the algorithm. We use the value of 0.9 by our intuition.
[3] A list of 199 words is used to filter stop-words.
[4] http://www.tartarus.org/~martin/PorterStemmer.
[5] ROUGE version 1.5.5 is used.

PR(A+B), PR(A+B/A)) are implemented as reference. These algorithms are all based on the query-sensitive LexRank (OtterBacher et al., 2005). The differences are two-fold: (1) the document collection(s) used to build the text graph are different; and (2) after ranking, the sentence selection strategies are different. In particular, PR(B) only uses the sentences in "B" to build the graph, and the other two consider the sentences in both "A" and in "B". Only the sentences in "B" are considered to be selected in PR(B) and PR(A+B/A), but all the sentences in "A" and "B" have the same chance to be selected in PR(A+B). Only the sentences from B are considered to be selected in the final summaries in PNR^2 as well. In the following experiments, the damping factor is set to 0.85 in the first three algorithms as the same in PageRank. The weight matrix W is set to $\begin{bmatrix} 1 & -0.5 \\ -0.5 & 1 \end{bmatrix}$ in the proposed algorithm (i.e. PNR^2) and $\gamma_1 = \gamma_2 = 0.5$. We have obtained reasonable good results with the decay factor θ between 0.3 and 0.8. So we set it to 0.5 in this paper.

Notice that the three PageRank-like graph-based ranking algorithms can be viewed as only the positive reinforcement among the sentences is considered, while both positive and negative reinforcement are considered in PNR^2 as mentioned before. Table 2 below shows the results of recall scores of ROUGE-1, ROUGE-2 and ROUGE-SU4 along with their 95% confidential internals within square brackets.

	ROUGE -1	ROUGE -2	ROUGE-SU4
PR(B)	0.3323 [0.3164,0.3501]	0.0814 [0.0670,0.0959]	0.1165 0.1053,0.1286]
PR(A+B)	0.3059 [0.2841,0.3256]	0.0746 [0.0613,0.0893]	0.1064 [0.0938,0.1186]
PR(A+B/A)	0.3376 [0.3186,0.3572]	0.0865 [0.0724,0.1007]	0.1222 [0.1104,0.1304]
PNR^2	**0.3616** [0.3464,0.3756]	**0.0895** [0.0810,0.0987]	**0.1291** [0.1208,0.1384]

Table 2. Experiment Results

We come to the following three conclusions. First, it is not surprising that PR(B) and PR(A+B/A) outperform PR(A+B), because the update task obviously prefers the sentences from the new documents (i.e. "B"). Second, PR(A+B/A) outperforms PR(B) because the sentences in "A" can provide useful information in ranking the sentences in "B", although we do not select the sentences ranked high in "A". Third, PNR^2 achieves the best performance. PNR^2 is above PR(A+B/A) by 7.11% of ROUGE-1, 3.47% of ROUGE-2, and 5.65% of ROUGE-SU4. This result confirms the idea and algorithm proposed in this work.

4.3 Comparison with DUC 2007 Systems

Twenty-four systems have been submitted to the DUC for evaluation in the 2007 update task. Table 3 compares our PNR^2 with them. For reference, we present the following representative ROUGE results of (1) the best and worst participating system performance, and (2) the average ROUGE scores (i.e. AVG). We can then easily locate the positions of the proposed models among them.

	PNR^2	**Mean**	**Best / Worst**
ROUGE-1	**0.3616**	0.3262	0.3768/0.2621
ROUGE2	**0.0895**	0.0745	0.1117/0.0365
ROUGE-SU4	**0.1291**	0.1128	0.1430/0.0745

Table 3. System Comparison

4.4 Discussion

In this work, we use the sentences in the same sentence set for positive reinforcement and sentences in the different set for negative reinforcement. Precisely, the old sentences perform negative reinforcement over the new sentences while the new sentences perform positive reinforcement over each other. This is reasonable although we may have a more comprehensive alternation. Old sentences may express old topics, but they may also express emerging new topics. Similarly, new sentences are supposed to express new topics, but they may also express the continuation of old topics. As a result, it will be more comprehensive to classify the whole sentences (both new sentences and old sentences together) into two categories, i.e. old topics oriented sentences and new topic oriented sentences, and then to apply these two sentence sets in the PNR^2 framework. This will be further studied in our future work.

Moreover, in the update summarization task, the summary length is restricted to about 100 words. In this situation, we find that sentence simplification is even more important in our investigations. We will also work on this issue in our forthcoming studies.

5 Conclusion

In this paper, we propose a novel sentence ranking algorithm, namely PNR^2, for update summarization. As our pilot study, we simply assume to receive two chronologically ordered document collections and evaluate the summaries

generated for the collection given later. With PNR^2, sentences from the new (i.e. late) document collection perform positive reinforcement among each other but they receive negative reinforcement from the sentences in the old (i.e. early) document collection. Positive and negative reinforcement are concerned simultaneously in the ranking process. As a result, PNR^2 favors the sentences biased to the sentences that are important in the new collection and meanwhile novel to the sentences in the old collection. As a matter of fact, this positive and negative ranking scheme is general enough and can be used in many other situations, such as social network analysis etc.

Acknowledgements

The research work presented in this paper was partially supported by the grants from RGC of HKSAR (Project No: PolyU5217/07E), NSF of China (Project No: 60703008) and the Hong Kong Polytechnic University (Project No: A-PA6L).

References

Klaus Berberich, Michalis Vazirgiannis, and Gerhard Weikum. 2004. G.T-Rank: Time-Aware Authority Ranking. In Algorithms and Models for the Web-Graph: Third International Workshop, WAW, pp 131-141.

Klaus Berberich, Michalis Vazirgiannis, and Gerhard Weikum. 2005. Time-Aware Authority Ranking. Journal of Internet Mathematics, 2(3): 301-332.

Klaus Lorenz Berberich. 2004. Time-aware and Trend-based Authority Ranking. Master Thesis, Saarlandes University, Germany.

Sergey Brin and Lawrence Page. 1998. The Anatomy of a Large-scale Hypertextual Web Search Engine. Computer Networks and ISDN Systems, 30(1-7):107-117.

Gunes Erkan and Dragomir R. Radev. 2004a. LexPageRank: Prestige in Multi-Document Text Summarization, in Proceedings of EMNLP, pp365-371.

Gunes Erkan and Dragomir R. Radev. 2004b. LexRank: Graph-based Centrality as Salience in Text Summarization, Journal of Artificial Intelligence Research 22:457-479.

Jon M. Kleinberg. 1999. Authoritative Sources in Hyperlinked Environment, Journal of the ACM, 46(5):604-632.

Jure Leskovec, Marko Grobelnik and Natasa Milic-Frayling. 2004. Learning Sub-structures of Document Semantic Graphs for Document Summarization, in Proceedings of LinkKDD Workshop, pp133-138.

Wenjie Li, Mingli Wu, Qin Lu, Wei Xu and Chunfa Yuan. 2006. Extractive Summarization using Intra- and Inter-Event Relevance, in Proceedings of ACL/COLING, pp369-376.

Chin-Yew Lin and Eduard Hovy. 2003. Automatic Evaluation of Summaries Using N-gram Co-occurrence Statistics, in Proceedings of HLT-NAACL, pp71-78.

Ziheng Lin, Tat-Seng Chua, Min-Yen Kan, Wee Sun Lee, Long Qiu, and Shiren Ye. 2007. NUS at DUC 2007: Using Evolutionary Models for Text. In Proceedings of Document Understanding Conference (DUC) 2007.

Rada Mihalcea and Paul Tarau. 2004. TextRank - Bringing Order into Text, in Proceedings of EMNLP, pp404-411.

Rada Mihalcea. 2004. Graph-based Ranking Algorithms for Sentence Extraction, Applied to Text Summarization, in Proceedings of ACL (Companion Volume).

Jahna OtterBacher, Gunes Erkan, Dragomir R. Radev. 2005. Using Random Walks for Question-focused Sentence Retrieval, in Proceedings of HLT/EMNLP, pp915-922.

Lucy Vanderwende, Michele Banko and Arul Menezes. 2004. Event-Centric Summary Generation, in Working Notes of DUC 2004.

Xiaojun Wan, Jianwu Yang and Jianguo Xiao. 2006. Using Cross-Document Random Walks for Topic-Focused Multi-Document Summarization, in Proceedings of the 2006 IEEE/WIC/ACM International Conference on Web Intelligence, pp1012-1018.

Xiaojun Wan. 2007. TimedTextRank: Adding the Temporal Dimension to Multi-document Summarization. In Proceedings of 30th ACM SIGIR, pp 867-868.

Lei Yang, Lei Qi, Yan-Ping Zhao, Bin Gao, and Tie-Yan Liu. 2007. Link Analysis using Time Series of Web Graphs. In Proceedings of CIKM'07.

Masaharu Yoshioka and Makoto Haraguchi. 2004. Multiple News Articles Summarization based on Event Reference Information, in Working Notes of NTCIR-4.

Philip S. Yu, Xin Li, and Bing Liu. 2004. On the Temporal Dimension of Search. In Proceedings of the 13th International World Wide Web Conference on Alternate Track Papers and Posters, pp 448-449.

Philip S. Yu, Xin Li, and Bing Liu. 2005. Adding the Temporal Dimension to Search – A Case Study in Publication Search. In Proceedings of the 2005 IEEE/WIC/ACM International Conference on Web Intelligence.