Generic Summarization and Keyphrase Extraction Using Mutual Reinforcement Principle and Sentence Clustering

Hongyuan Zha
Department of Computer
Science & Engineering
Pennsylvania State University
University Park, PA 16802
zha@cse.psu.edu

ABSTRACT

A novel method for *simultaneous* keyphrase extraction and generic text summarization is proposed by modeling text documents as weighted undirected and weighted bipartite graphs. Spectral graph clustering algorithms are used for partitioning sentences of the documents into topical groups with sentence link priors being exploited to enhance clustering quality. Within each topical group, saliency scores for keyphrases and sentences are generated based on a mutual reinforcement principle. The keyphrases and sentences are then ranked according to their saliency scores and selected for inclusion in the top keyphrase list and summaries of the document. The idea of building a hierarchy of summaries for documents capturing different levels of granularity is also briefly discussed. Our method is illustrated using several examples from news articles, news broadcast transcripts and web documents.

Categories and Subject Descriptors

H.3.3 [**Information Systems**]: Information Search and Retrieval—*Clustering*; H.4.m [**Information Systems**]: Miscellaneous—*Text Summarization*; G.1.3 [**Numerical Analysis**]: Numerical Linear Algebra—*Singular value decomposition*; G.2.2 [**Discrete Mathematics**]: Graph Theory—*Graph algorithms*

General Terms

Algorithms, Experimentation, Theory

Keywords

text summarization, keyphrase extraction, mutual reinforcement principle, bipartite graph, graph partitioning, singular value decomposition

SIGIR'02, August 11-15, 2002, Tampere, Finland.

1. INTRODUCTION

Text summarization is an increasingly pressing practical problem due to the explosion of the amount of textual information available. For example, web search engines have exploited the use of text summarization from the very beginning: starting with the extraction of certain number of bytes from the beginning of each document to the more sophisticated query-focused summaries typified by Google's snippets (see also the recent work in [1]). Query-focused summaries provide the users with the useful information for initial relevance judgement so that they can quickly zero in on documents deserving further inspection. In contrast, a generic summary in general distills the most important overall information from a document (or a set of documents), it can be especially useful when the documents are relatively long and contain a variety of topics. With many search engines starting to index documents in postscript and pdf formats, we will see increased availability of long and multi-part documents and the pressing needs for efficiently generating effective generic summaries for those documents. In addition, there is also a great amount of news articles and broadcast transcripts generated daily from various news agencies needing effective summarization.

Automatic text summarization is an extremely active research field making connections with many other research areas such as information retrieval, natural language processing and machine learning [5]. Informally, the goal of text summarization is *to take a textual document, extract content from it and present the most important content to the user in a condensed form and in a manner sensitive to the user's or application's needs* [13]. In this paper we concentrate on the shallow approach of text summarization using sentence and keyphrase extractions. Abstract generation utilizing materials not present in the documents to be summarized is a more challenging problem and will not be addressed here [1, 13, 14]. Two basic approaches to sentence extraction can be distinguished on whether they are supervised or unsupervised in nature. Supervised approaches need human-generated summary extracts for feature extraction and parameter estimation as is typified by the methods in [10, 3] where sentence classifiers are trained using human-generated sentence-summary pairs as training examples. Possible drawbacks of the supervised approaches are domain-dependency and the problems caused by the potential inconsistency of human-generated summaries. In this paper we adopt the unsupervised approach, we *explicitly* model both keyphrases and the

sentences that contain them using weighted undirected and weighted bipartite graphs and generate sentence extracts on the fly without extensive training. Our method can also be viewed as representing a more sophisticated and effective approach to exploiting the term co-occurrence relationship in textual documents.

Many text summarization methods are surveyed in [5, 13, 14]. We mention here several recent approaches that are closest in spirit to our approach. In [18], documents are modeled using undirected graphs with the vertices representing paragraphs, and edge weights representing similarity between two paragraphs. Salient paragraphs are those connected to many other paragraphs with similarity above certain thresholds. In [6], Latent semantic indexing was used as the bases for sentence selection of a given textual document, exploiting the components of multiple singular vectors. However, the singular vectors other than the one corresponding to the largest singular value can have both positive and negative components, making ranking sentences by singular vector component values less meaningful. In [4], QR decomposition with column pivoting applied to term-sentence matrices was used for sentence selection, providing another example of unsupervised summarization methods. To emphasize diversity of topic coverage in a generic summary, in [15] it was proposed to use a variation of the K-means method to cluster the sentences of a document into different topical groups, and then apply a sentence weighting model within each topical group for sentence selection.

The basic idea of our summarization method is to first cluster sentences of a document (or a set of documents) into topical groups and then, within each topical group, to select the keyphrases and sentences by their saliency scores. Our major contributions are 1) proposing the use of sentence link priors resulted from the linear ordering of the sentences in a document to enhance sentence clustering quality; and 2) developing the mutual reinforcement principle for simultaneous keyphrase and sentence saliency score computation. We also alluded to the possibility of building a summary hierarchy based on a hierarchical clustering of the sentences of a document. The rest of the paper is organized as follows: in section 2, we develop the mutual reinforcement principle and its connection to computing the largest singular value triplet of the weight matrix of the term-sentence bipartite graph. In section 3, we introduce the sentence link prior and show how we can incorporate it into the sum-of-squares sentence clustering objective function using spectral clustering techniques. We also discuss link strength selection using generalized cross-validation. In section 4, we describe some experimental results using documents from the newswires, broadcasting transcripts and the World Wide Web. We conclude the paper in section 5 with pointers to future research.

2. THE MUTUAL REINFORCEMENT PRINCIPLE

For each document, we generate two sets of objects: one the set of terms $T = \{t_1, \ldots, t_n\}$ and the other the set of sentences $S = \{s_1, \ldots, s_m\}$ in the document. [1] We build a weighted *bipartite* graph from T and S in the following way: if the term t_i appears in sentence s_j, we then create an edge between t_i and s_j. We can also specify *nonnegative* weights on the edges of the weighted bipartite graph with w_{ij} indicating the weight on the edge (t_i, s_j). For example, we can simply choose w_{ij} to be the number of times t_i appears in s_j. More sophisticated weighting schemes will be discussed later. We denote the weighted bipartite graph by $G(T, S, W)$ where $W = [w_{ij}]$ is the m-by-n weight matrix containing all the pairwise edge weights. For each term t_i and each sentence s_j we wish to compute their saliency scores $u(t_i)$ and $v(s_j)$, respectively. To this end, we state the following *mutual reinforcement principle*:[2]

> A term should have a high saliency score if it appears in many sentences with high saliency scores while a sentence should have a high saliency score if it contains many terms with high saliency scores.

In essence the principle dictates that the saliency score of a term is determined by the saliency scores of the sentences it appears in, and the saliency score of a sentence is determined by the saliency scores of the terms it contains. Mathematically, the above statement is rendered as

$$u(t_i) \propto \sum_{v(s_j) \sim u(t_i)} w_{ij} v(s_j),$$
$$v(s_j) \propto \sum_{u(t_i) \sim u(t_i)} w_{ij} u(t_i),$$

where the summations are over the neighbors of the vertices in question, and $a \sim b$ indicates there is an edge between vertices a and b, i.e., when computing a term score, the summation is over all sentences that contain the term and when computing a sentence score, the summation is over all terms that appear in the sentence. The symbol $\propto$ stands for "proportional to". Now we collect the saliency scores for terms and sentences into two vectors u and v, respectively, the above equation can then be written in the following matrix format

$$u = \frac{1}{\sigma} W v, \quad v = \frac{1}{\sigma} W^T u,$$

where W is the weight matrix of the bipartite graph of the document in question, W^T stands for the matrix transpose of W, and $1/\sigma$ is the proportionality constant. It is easy to see that u and v are the left and right singular vectors of W corresponding to the singular value σ. If we choose σ to be the largest singular value of W, then its is guaranteed that both u and v have *nonnegative* components. The corresponding component values of u and v give the term and sentence saliency scores, respectively. We can rank terms and sentences in decreasing order of their saliency scores, and select the top t terms (with the highest saliency scores) to add to the top term list and the top s sentences (with the highest saliency scores) to add to the summary. Here t and s are some user-defined values, and s can be estimated from the compression rate of the desired summary.

REMARK. For numerical computation of the largest singular value triplet $\{u, \sigma, v\}$, we can use a variation of the power method adapted to the case of singular value triplets: choose an initial value for v to be the vector of all ones. Alternate between the following two steps until convergence,

1. Compute and normalize

$$u = Wv, \quad u = u/\|u\|,$$

[1] The choice is flexible, for example, terms can be words or phrases and sentences can be replaced by paragraphs or other text units.

[2] Similar ideas have also been used to find the hub and authority web pages in a link graph [9].

2. Compute and normalize

$$v = W^T u, \quad v = v/\|v\|,$$

where the vector norm $\|\cdot\|$ can be chosen to the Euclidean norm, and σ can be computed as $\sigma = u^T W v$ upon convergence. For a detailed analysis of the singular value decomposition for related types of matrices, the reader is referred to [19].

REMARK. The above general weighted bipartite graph model can be extended further by adding vertex weights to the terms and/or sentences. Both types of weights can incorporate certain kind of prior information, for example, the weight of a sentence vertex can be increased if it contains certain bonus words; we can also modify the weight of a sentence vertex based on its position in the document. In general, let D_T and D_S be two diagonal matrices the diagonal elements of which represent the weights of the term vertices and sentence vertices, respectively. Then instead of finding the largest singular value triplet of the edge weight matrix W, we compute the largest singular value triplet $\{u, \sigma, v\}$ of the scaled matrix $D_T W D_S$. A specific sentence vertex weighting scheme will be discussed later.

3. CLUSTERING SENTENCES INTO TOPICAL GROUPS

The saliency score computation discussed in Section 2 can be more effective if it is applied within each topical group of a document (or a set of documents). To this end we discuss effective algorithms for sentence clustering with the purpose to reveal the latent topical structure of textual documents. The idea of using sentences clustering has also been recently used in [18, 15]. For sentence clustering we first build an *undirected weighted* graph with vertices representing the sentences of a document and two sentences s_i and s_j are linked by an edge if there are terms shared by the two sentences, we also specify an edge weight w_{ij} for the edge (s_i, s_j), in general w_{ij} indicates the similarity between the two sentences s_i and s_j, and there are many different ways for their specification. We denote the resulting graph as $G(S, W_S)$, where S is the set of sentences and $W_S = [w_{ij}]$ gives the edge weights. It is quite easy to verify that W_S has the same *sparsity pattern* as $W^T W$, where W is the weight matrix for the bipartite graph introduced in Section 2.

3.1 Incorporating sentence link priors

Documents consist of sentences arranged in a *linear* order, and near-by sentences in terms of this intrinsic linear order tend to be about the same topic. The fact that topical groups within a document are usually made of sections of consecutive sentences is a strong prior that needs to be fully exploited during the sentence clustering process. The prior which we call sentence link prior will be especially helpful when dealing with broadcasting transcripts where there are no syntactic cues such as paragraph heading/ending to indicate topic boundaries. To ease discussion, we call s_i and s_j are *near-by* to each other if s_i is followed by s_j in the linear order of the document (or s_j is followed by s_i, since the graph we consider is undirected anyway).

The weighted undirected graph framework we are using for sentence clustering naturally lends itself to incorporating the sentence link prior. In general, pairs of vertices (sentences) with large similarity weight tend to be clustered into the same group. A simple approach to taking advantage of this goes as follows: we strengthen the similarity weight between near-by sentences, i.e., we modify the weights for all the near-by sentence pairs,

$$\hat{w}_{ij} = \begin{cases} w_{ij} + \alpha & \text{if } s_i \text{ and } s_j \text{ are near-by} \\ w_{ij} & \text{otherwise} \end{cases}$$

We call α the sentence link strength, and the modified weight matrix is denoted by $W_S(\alpha)$. There are $n-1$ modifications in total for a document with n sentences. There is also the possibility of strengthen the similarity of pairs of sentences that are two (or more) edges away from each other. Notice that incorporating sentence link prior is different from the requirement in text segmentation: we do allow several sections of consecutive sentences to form a single topical group.

The parameter α can be considered as a regularization parameter, and we use the idea of generalized cross-validation (GCV) for choosing a good α [12]. For a fixed α, we apply the spectral clustering technique discussed in the next section to $W(\alpha)$ to obtain a set of sentence clusters $\Pi^*(\alpha)$. For any sentence clustering Π we define $\gamma(\Pi)$ to be the number of consecutive sentence segments it generates which is then used as a measure of model complexity for the clustering Π. The idea is to simultaneously minimize the clustering cost function and the model complexity. To this end, we compute a function of α defined as

$$\mathrm{GCV}(\alpha) = \left(n - k - J(W, \Pi^*(\alpha))\right) / \gamma(\Pi^*(\alpha)), \qquad (1)$$

where k is the desirable number of sentence clusters, W is the weight matrix for the term-sentence bipartite graph and $\Pi^*(\alpha)$ is the set of clusters obtained by applying spectral clustering to the modified weight matrix $W(\alpha)$. We then select the α that maximizes the above function as the *estimated* optimal α value.

3.2 Building a summary hierarchy

Summaries can actually capture the essential information of textual documents at different levels of granularity. This issue is closely related to the compression rate and coverage/diversity of summaries [13]. However, the point of view of *hierarchical* sentence clustering seems to provide a rather natural and fruitful way of tackling the issue. We start with a sentence cluster hierarchy with the lower-level clusters (nodes) representing finer structures of the document. Summarization is then done at each node of the hierarchy using all the sentences that belong to this node. Two basic variations exist when carry out summarization at a non-root node of the hierarchy.

- The terms used in summarization remain the same for all the nodes;
- The terms used in summarization at a particular node do not include the top terms (i.e., those terms having high saliency scores) from the summarization of its parent nodes.

The idea for the second approach is that we only retain terms that can distinguish finer structures of the document. Another possibility is to use the first approach in the above for all the *leave* nodes of the hierarchy, and at the next higher level carry out a summary of summaries until the root-node is reached. Currently, we have only limited experiences with the ideas proposed above, and we will not elaborate on the issues further.

3.3 Sum-of-squares cost function for sentence clustering and spectral relaxation

We start with the weight matrix W for the bipartite graph $G(T, S, W)$. Here each sentence is represented by a column of $W = [w_1, \ldots, w_n]$ which we will call the sentence vector. A partition Π of the sentence vectors can be written in the following form

$$WE = [W_1, \ldots, W_k], \quad W_i = [w_1^{(i)}, \ldots, w_{n_i}^{(i)}], \qquad (2)$$

where E is a permutation matrix, and W_i is m-by-n_i, i.e., the ith sentence cluster contains the sentence vectors in W_i. For a given partition Π, the associated sum-of-squares cost function is defined as [7]

$$J(W, \Pi) = \sum_{i=1}^{k} \sum_{s=1}^{n_i} \|w_s^{(i)} - m_i\|^2, \quad m_i = \sum_{s=1}^{n_i} w_s^{(i)}/n_i, \quad (3)$$

i.e., m_i is the centroid of the sentence vectors in cluster i, and n_i is the number of sentences in cluster i.

The traditional K-means algorithm is iterative in nature and in each iteration, the following is performed [7]:

1) For each sentence vector w, find the center m_i that is closest to w, and associate w with this center.

2) Compute a new set of centers by taking, for each center, the center of mass of sentence vectors associated with this center.

It can be proved that the above algorithm is equivalent to finding a local minimum of $J(W, \Pi)$ with respect to Π using coordinate descent. Despite the popularity of K-means clustering algorithm, one of its major drawbacks is that the coordinate descent search method is prone to local minima giving rise to some clusters with very few data points. Moreover, it is also not easy to incorporate the sentence link prior into the above sum-of-squares cost function in order to improve sentence clustering quality. Much research has been done on computing refined initial centroids and adding explicit constraints to the sum-of-squares cost function for K-means clustering so that the search can converge to a better local minimum [2]. It was shown, however, that an equivalent formulation of the sum-of-squares minimization can be derived as a matrix trace maximization problem with special constraints; relaxing the constraints leads to a trace maximization problem that possesses optimal *global* solutions [20]. This formulation also makes K-means method easily adaptable to utilizing the sentence link priors discused in the above subsection.

Here we give a brief presentation of the spectral relaxation approach for K-means clustering. Let e be a vector of appropriate dimension with all elements equal to one, it is easy to see that the centroids can be written as

$$m_i = W_i e/n_i.$$

Now let

$$X = \mathrm{diag}(e/\sqrt{n_1}, \ldots, e/\sqrt{n_k}).$$

It was shown in [20] that the sum-of-squares cost function can be written as

$$J(W, \Pi) = \mathrm{trace}(W^T W) - \mathrm{trace}(X^T W^T W X),$$

and its minimization is equivalent to

$$\max\{\ \mathrm{trace}(X^T W^T W X)\ \mid\ X = \mathrm{diag}(e/\sqrt{n_1}, \ldots, e/\sqrt{n_k})\}.$$

Ignoring the special structure of X and let it be an arbitrary orthonormal matrix, we obtain a *relaxed* matrix trace maximization problem

$$\max_{X^T X = I_k} \mathrm{trace}(X^T W^T W X). \qquad (4)$$

An extension of the Rayleigh-Ritz characterization of eigenvalues of symmetric matrices shows that the above maximum is achieved by the first k largest eigenvectors of the Gram matrix $W^T W$ [8, Section 4.3.18]. As a by-product, we also have the following inequality

$$\min_{\Pi} J(W, \Pi) \geq \mathrm{trace}(W^T W) - \max_{X^T X = I_k} \mathrm{trace}(X^T W^T W X)$$

$$= \sum_{i=k+1}^{\min\{m,n\}} \sigma_i^2(W),$$

where $\sigma_i(W)$ is the i largest singular value of W. This gives a *lower bound* for the minimum of the sum-of-squares cost function. The spectral relaxation formulation of the K-means algorithm also makes it easy to consider more general kernel functions: instead of using $w_i^T w_j$ we can compute the Gram matrix using any Mercer kernel $K(x, x')$ to obtain $[K(w_i, w_j)]_{i,j=1}^n$. In particular, we can replace $W^T W$ by $W_S(\alpha)$ after incorporating the link strength α. The assignment of the cluster labels to each sentence is done by using QR decomposition with pivoting (see below).

The sentence clustering algorithm based on the modified weight matrix $W_S(\alpha)$ is now summarized as follows:

- Compute the k eigenvectors $V_k = [v_1, \ldots, v_k]$ of $W_S(\alpha)$ corresponding to the largest k eigenvalues.

- Compute the pivoted QR decomposition of V_k^T as

  $$(V_k)^T P = QR = Q[R_{11}, R_{12}],$$

  where Q is a k-by-k orthogonal matrix, R_{11} is a k-by-k upper triangular matrix, and P is a permutation matrix.

- Compute

  $$\hat{R} = R_{11}^{-1}[R_{11}, R_{12}]P^T = [I_k, R_{11}^{-1}R_{12}]P^T.$$

  Then the cluster membership of each sentence is determined by the row index of the largest element *in absolute value* of the corresponding column of $\hat{R}$. This gives rise to the clustering $\Pi^*(\alpha)$ used in (1).

4. EXPERIMENTAL RESULTS

Evaluation for generic text summarization is a very challenging task: although it can be done against a set of documents with manual summarizations, human-generated sentence extracts do, however, tend to differ significantly especially for longer documents [6, 13]. Another approach is to evaluate extrinsically the summarization algorithms based on, for example, their performance on document retrieval or text categorization. Here we will provide some preliminary quantitative and qualitative assessment of our summarization algorithm. We collected a set of ten documents consisting of news articles, news broadcast transcripts and web pages.

- News articles

Document	optimal α	accuracy	estimated α	accuracy	sentence #	cluster #
sf	7.94	82.17%	7.94	82.17%	157	7
pge	63	69.23%	12.59	57.95 %	195	6
flg	19.95	68.07%	125.89	66.87 %	166	7
heart	7.94	87.23%	7.97	87.23%	47	3
enron	39.8	75.86%	14.59	74.71%	87	6
cnn	31.62	71.79%	79.43	64.69%	507	16
cnn1	2.2	73.96%	50.11	69.81%	265	10
cnn2	25.1	64.89%	125.89	60.11 %	544	12
cnn3	1.99	72.52%	12.59	65.32%	223	10
dna	3.16	74.68%	7.94	70.89%	79	8

Table 1: Clustering accuracy with optimal and estimated α

```
sf:  story.news.yahoo.com/news?tmpl=story&u=
     /nyt/20020121/ts_nyt/conduct_of_war_is_
     redefined_by_success_of_special_forces

pge: www0.mercurycenter.com/local/center
     /dereg1201.htm

flg:  www0.mercurycenter.com/local/center
     /fal122301.htm

enron: story.news.yahoo.com/news?tmpl=
       story&cid=68&u=/nyt/20020120/ts_nyt/
       multiple_safeguards_failed_to_detect
       _problems_at_enron
```

- CNN broadcast transcripts

```
cnn : moneyline
cnn1: newsroom
cnn2: Wolf Blitzer repots
cnn3: science and technology week
```

- Web pages

```
dna:
 www.pbs.org/wgbh/nova/neanderthals/mtdna.html
heart:
 www.pbs.org/wgbh/nova/heart/treating.html
```

We manually divide each document into topical groups: for web pages and news articles we rely on the section structure of the documents, and for news broadcast transcripts we rely on the contents of the documents. We notice that the clustering is usually not unique, some clusters can be merged into a bigger cluster and some clusters can be split into several smaller ones to capture the finer structure of the documents. The number of sentences and the number of clusters for each document are listed in the Table 1. (Other items of the table will be discussed later.)

We first illustrate *quantitatively* the difference in sentence clustering resulting from using the sentence link prior. In processing the documents, we deleted words appearing in a stop word list and applied Porter's stemming [16]. We used the following sentence similarity measure to construct the weight matrix W_S: each sentence is represented by a sentence vector using the weight matrix W of the term-sentence bipartite graph introduced in section 2, i.e., columns of W correspond to the sentences of the document in question. The weight matrix $W_S = (w_{ij})$ for the sentence graph is computed with w_{ij} equal to the dot-product between the sentence vectors of sentences s_i and s_j. The sentence vectors are weighted with `tf.idf` weighting and normalized to have Euclidean length one.

To measure the quality of sentence clustering, we use a variation of the confusion matrix which is frequently used for measuring classifier accuracy. We assume the manually generated section number gives the *true* cluster label of each sentence. We then compute the accuracy of our clustering algorithm against the section labels. In particular, for a k cluster case, we compute a k-by-k confusion matrix $C = [c_{ij}]$ with c_{ij} the number of sentences in cluster i that belong to section j. It is actually quite subtle to compute the accuracy using the confusion matrix because we do not know which cluster matches which section. An optimal way is to solve the following maximization problem

$$\max\{\ \text{trace}(CP)\ \mid\ P \text{ is a permutation matrix}\},$$

and divide the maximum by the total number of sentences to obtain the clustering accuracy. This is equivalent to finding a perfect matching of a complete weighted bipartite graph which can be solved using Kuhn-Munkres algorithm [11]. In our computation, we used a greedy algorithm to compute a sub-optimal solution.

As an illustration, for a sequence of α values, we applied the spectral clustering algorithm to the weight matrix $W_S(\alpha)$ of the document `dna`, and in Figure 1 we plot the clustering accuracy against α. In the same figure, we also contrast the results of sentence clustering with and without link priors, the one without link priors tend to be more fragmented. Based on the results for the ten documents, a general conclusion seems to be that the clustering algorithm matches the section structure of the document poorly when there is no near-by sentence constraints (i.e., $\alpha = 0$). With too large an α value, sentence similarities are overwhelmed by link strength, the results are also poor. Our generalized cross-validation method seems to be quite effective at selecting a good α that produces clustering quality close to that given by the optimal α. In Table 1, we list the optimal α and the corresponding sentence clustering accuracy for each document. We also list the estimated α using GCV(α) discussed in section 3.1 and the corresponding sentence clustering accuracy. In general, clustering accuracy is a relatively flat function of α, and the estimated α even though may differ considerably from the optimal α still produces clustering accuracy that matchs well the best clustering accuracy, as

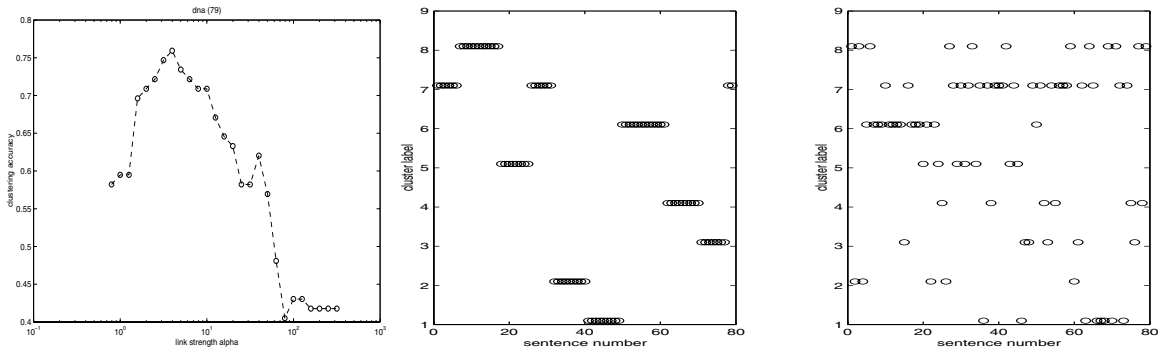

Figure 1: (Left) sentence clustering accuracy versus sentence link strength α. Sentence cluster distribution with link prior (middle) and without link prior (right)

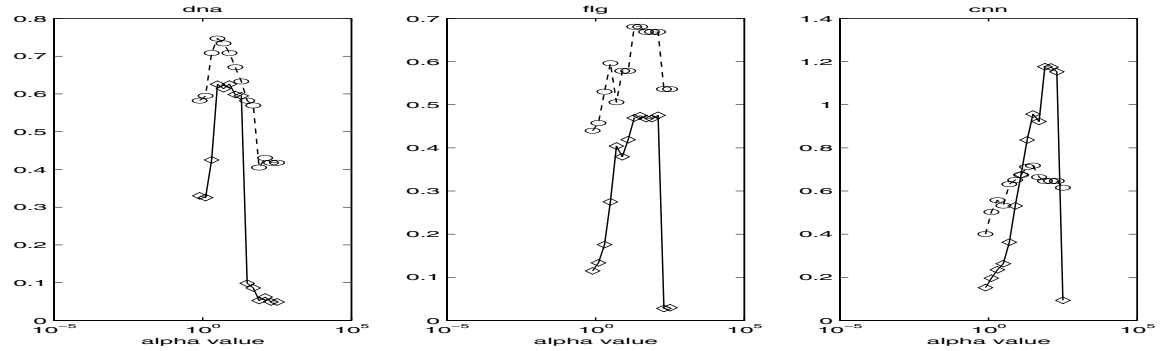

Figure 2: Clustering accuracy vs. α (circles); GCV(α) vs. α (diamonds)

can be seen from Figure 2.

For the computation of keyphrase and sentence saliency scores, we considered sentence vertex weights when applying the mutual reinforcement principle. Specifically, for the i-th sentence we apply the weight

$$\frac{\log_2(\text{number of terms in sentence} + 1)}{\log_2(i + \text{total number of sentences})}.$$

The idea is to mitigate the influence of long sentences by scaling each sentence by a factor proportional to the length of the sentence in terms of number of words; at the same time sentences close to the beginning of the document get a small boost based on their positions in the document.

For a detailed illustration we use document `dna` which consists of a story lead plus seven sections with sections headings given below

1. Tracing Ancestry with MtDNA (title) (1–6)
2. Nuclear DNA vs mitochondrial DNA (7–15)
3. Inheriting mtDNA (16–25)
4. Defining mitochondrial ancestors (26–37)
5. Finding mitochondrial ancestors (38–52)
6. Dating mitochondrial ancestors (53–63)
7. Neanderthals and mtDNA (63–70)
8. Final note (71–79)

There are 79 sentences in total, and we number the sentences consecutively starting from 1. The numbers in parentheses in the above indicate sentence sequence range in each section.

In Table 2, for $\alpha = 3.5$, we look at the clusters generated by listing the top five terms and the top one sentence in each topical groups. We also list the sentence numbers in the each topical group in parentheses.

The clustering result matches the section structure quite well except for cluster 8 which actually consists of two sentence sequences. Taking a closer look at the sentences in cluster 8, we found that sentences 1 to 4 discuss issues related the common ancestor "Eve", and section 4 with section heading *Defining mitochondrial ancestors* is about the same topic. So here sentence similarities win over sentence link strength. We have also applied the mutual reinforcement principle to all the sentences in the document `dna`, and the first few keywords and sentences extracted are related to the main topic of document. Similar analysis has also been applied to the other nine documents with quite similar results, the details of which will not be reported here due to the lack of space (see `http://www.cse.psu.edu/ zha/sum.html`).

5. CONCLUSIONS

In this paper we presented a novel method for simultaneous extraction of keyphrases and sentences from textual documents. We explore the sentence link priors embedded in the linear ordering of a document to enhance the quality of clustering sentences of documents into topical groups. We also develop the mutual reinforcement principle to compute

Cluster 1 (71–79) mtdna matern recent recombin alter In fact, recent studies show that paternal mtDNA can on rare occasions enter an egg during fertilization and alter the maternal mtDNA through recombination.	Cluster 2 (47–52) group tree similar famili comput Later, with the help of a computer program, they put together a sort of family tree, grouping those with the most similar DNA together, then grouping the groups, and then grouping the groups of groups.
Cluster 3 (53–61) ancestor mutat mtdna rate live For instance, if they took the mutation rate to be one in every 1,000 years and knew that there was a difference of 10 mutations between the mtDNA of people living today and the mtDNA of an ancestor who lived long ago, then they could infer that the ancestor lived 10,000 years ago.	Cluster 4 (62–70) modern neanderth mtdna european anthropologist This was an unwelcome finding for anthropologists who believe that there was some interbreeding between Neanderthals and early modern humans living in Europe (which might have helped to explain why modern Europeans possess some Neanderthal-like features);
Cluster 5 (36–46) mtdna live mutat inherit long Even though everyone on Earth living today has inherited his or her mtDNA from one person who lived long ago, our mtDNA is not exactly alike.	Cluster 6 (16–24) egg nuclear cell mtdna mothers Whenever an egg cell is fertilized, nuclear chromosomes from a sperm cell enter the egg and combine with the egg's nuclear DNA, producing a mixture of both parents' genetic code.
Cluster 7 (16–24) human mtdna live scientist includ In recent years, scientists have used mtDNA to trace the evolution and migration of human species,including when the common ancestor to modern humans and Neanderthals lived -- though there has been considerable debate over the validity and value of the findings.	Cluster 8 (1–4, 26–35) ancestor live common ev mtdna It also does not mean that the mtDNA originated with this "Eve"; she and her contemporaries also had their own "most recent common ancestor though matrilineal descent," a woman who lived even further into the past who passed on her mtDNA to everyone living during "Eve's" time.

Table 2: Clustering and sentence extract results for document dna. For each cluster we list five top words and one sentence.

keyphrase and sentence saliency scores within each topical groups. We have also illustrated our method using a variety of documents with different characteristics. Many issues need further investigation: 1) more research needs to be done for the determination of optimal link strength α; 2) other possible ways for sentence clustering. One promising approach is to use a two-stage method: first segment the sentences and then cluster the segments into topical groups; 3) the issues of replacing the use of simple terms by noun phrases, this will impact how the weight matrix W and W_S will be constructed, and in general the construction of sentence similarity using various resources beyond lexical match. We have experimented with using Ramshaw and Marcus noun phrase chuncker for extracting noun phrases [17]; 4) more extensive and systematic experimentation of the method for both single and multiple documents [5]; and 5) extension to translingual summarization [21].

Acknowledgement. The research was supported in part by NSF grant CCR-9901986. The author also wants to thank the referees for many constructive comments and suggestions.

6. REFERENCES

[1] A. L. Berger and V. O. Mittal. OCELOT: a system for summarizing Web pages. Proceedings of the 23rd International Conference on Research in Information Retrieval (SIGIR '00), pp. 144 - 151, 2000.

[2] P. S. Bradley and Usama M. Fayyad. *Refining Initial Points for K-Means Clustering.* Proc. 15th International Conf. on Machine Learning, 91–99, 1998.

[3] W. T. Chuang and J. Yang. Extracting sentence segments for text summarization: a machine learning approach Proceedings of the 23rd International Conference on Research in Information Retrieval (SIGIR '00), pp. 152–159, 2000.

[4] J. Conroy and D. P. O'Leary. Text Summarization via Hidden Markov Models and Pivoted QR Matrix Decomposition. Technical Report, Dept. Comp. Sci., CS-TR-4221, Univ. Maryland, 2001.

[5] Document Understanding Conference. `http://www-nlpir.nist.gov/projects/duc/`.

[6] Y. Gong and X. Liu. Generic text summarization using relevance measure and latent semantic analysis. Proceedings of the 24th International Conference on Research in Information Retrieval (SIGIR '01), pp. 19–25, 2001.

[7] J.A. Hartigan and M.A. Wong. (1979). *A K-means Clustering Algorithm.* Applied Statistics, 28:100–108.

[8] R. Horn and C. Johnson. *Matrix Analysis.* Cambridge University Press, 1985.

[9] J. Kleinberg. Authoritative Sources in a Hyperlinked Environment. Proc. ACM-SIAM SODA, 1998.

[10] J. Kupiec, J. Pedersen and F. Chen. A Trainable Document Summarizer. Proceedings of the 18th International Conference on Research in Information Retrieval (SIGIR '95), pp. 55-60, 1995.

[11] L. Lovasz and M.D. Plummer. (1986) *Matching Theory.* Amsterdam: North Holland.

[12] Z. Luo. Clustering under Spatial Contiguity Constraint: A penalized K-means method. Technical Report, Department of Statistics, Penn State University, 2001.

[13] I. Mani. *Automatic Summarization.* John Benjamins Pub Co., 2001.

[14] I. Mani and M. Maybury. *Advances in automatic text summarization.* MIT Press, 1999.

[15] T. Nomoto and Y. Matsumoto. A new approach to unsupervised text summarization. Proceedings of the 24th International Conference on Research in Information Retrieval (SIGIR '01), pp. 26–34, 2001.

[16] M. Porter. The Porter Stemming Algorithm. `www.tartarus.org/ martin/PorterStemmer`

[17] L. A. Ramshaw and M. P. Marcus. Text Chunking Using Transformation-Based Learning. *Proceedings of the Third ACL Workshop on Very Large Corpora"*, Cambridge MA, USA, 1995.

[18] G. Salton, A. Singhal, M. Mitra and C. Buckley. Automatic text structuring and summarization. 341–355, *Advances in Automatic Text Summarization*, edited by I. Mani and M. Maybury, 1999.

[19] H. Zha and Z. Zhang. On Matrices with Low-rank-plus-shift Structures: Partial SVD and Latent Semantic Indexing,. *SIAM Journal of Matrix Analysis and Applications,* 21:522-536, 1999.

[20] H. Zha, M. Gu, X. He, C. Ding, and H. Simon. Spectral Relaxation for K-means Clustering. Advances in Neural Information Processing Systems, 14, eds. T. Dietterich, S. Becker, Z. Ghahramani, MIT Press, 2002.

[21] H. Zha and X. Ji. Correlating Multilingual Documents via Bipartite Graph Modeling. To appear in Proceedings of the 25th International Conference on Research in Information Retrieval (SIGIR '02), 2002.