Automatic Keyphrase Extraction via Topic Decomposition

Zhiyuan Liu, Wenyi Huang, Yabin Zheng and **Maosong Sun**
Department of Computer Science and Technology
State Key Lab on Intelligent Technology and Systems
National Lab for Information Science and Technology
Tsinghua University, Beijing 100084, China
{lzy.thu, harrywy, yabin.zheng}@gmail.com,
sms@tsinghua.edu.cn

Abstract

Existing graph-based ranking methods for keyphrase extraction compute a *single* importance score for each word via a *single* random walk. Motivated by the fact that both documents and words can be represented by a mixture of semantic topics, we propose to decompose traditional random walk into multiple random walks specific to various topics. We thus build a Topical PageRank (TPR) on word graph to measure word importance with respect to different topics. After that, given the topic distribution of the document, we further calculate the ranking scores of words and extract the top ranked ones as keyphrases. Experimental results show that TPR outperforms state-of-the-art keyphrase extraction methods on two datasets under various evaluation metrics.

1 Introduction

Keyphrases are defined as a set of terms in a document that give a brief summary of its content for readers. Automatic keyphrase extraction is widely used in information retrieval and digital library (Turney, 2000; Nguyen and Kan, 2007). Keyphrase extraction is also an essential step in various tasks of natural language processing such as document categorization, clustering and summarization (Manning and Schutze, 2000).

There are two principled approaches to extracting keyphrases: supervised and unsupervised. The supervised approach (Turney, 1999) regards keyphrase extraction as a classification task, in which a model is trained to determine whether a candidate phrase is a keyphrase. Supervised methods require a document set with human-assigned keyphrases as training set. In Web era, articles increase exponentially and change dynamically, which demands keyphrase extraction to be efficient and adaptable. However, since human labeling is time consuming, it is impractical to label training set from time to time. We thus focus on the unsupervised approach in this study.

In the unsupervised approach, graph-based ranking methods are state-of-the-art (Mihalcea and Tarau, 2004). These methods first build a word graph according to word co-occurrences within the document, and then use random walk techniques (e.g., PageRank) to measure word importance. After that, top ranked words are selected as keyphrases.

Existing graph-based methods maintain a *single* importance score for each word. However, a document (e.g., news article or research article) is usually composed of multiple semantic topics. Taking this paper for example, it refers to two major topics, "keyphrase extraction" and "random walk". As words are used to express various meanings corresponding to different semantic topics, a word will play different importance roles in different topics of the document. For example, the words "phrase" and "extraction" will be ranked to be more important in topic "keyphrase extraction", while the words "graph" and "PageRank" will be more important in topic "random walk". Since they do not take topics into account, graph-based methods may suffer from the following two problems:

1. Good keyphrases should be relevant to the major topics of the given document. In graph-based methods, the words that are strongly connected with other words tend to be ranked high,

Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing, pages 366–376,
MIT, Massachusetts, USA, 9-11 October 2010. ©2010 Association for Computational Linguistics

which do not necessarily guarantee they are relevant to major topics of the document.

2. An appropriate set of keyphrases should also have a good coverage of the document's major topics. In graph-based methods, the extracted keyphrases may fall into a single topic of the document and fail to cover other substantial topics of the document.

To address the problem, it is intuitive to consider the topics of words and document in random walk for keyphrase extraction. In this paper, we propose to decompose traditional PageRank into multiple PageRanks specific to various topics and obtain the importance scores of words under different topics. After that, with the help of the document topics, we can further extract keyphrases that are relevant to the document and at the same time have a good coverage of the document's major topics. We call the topic-decomposed PageRank as Topical PageRank (TPR).

In experiments we find that TPR can extract keyphrases with high relevance and good coverage, which outperforms other baseline methods under various evaluation metrics on two datasets. We also investigate the performance of TPR with different parameter values and demonstrate its robustness. Moreover, TPR is unsupervised and language-independent, which is applicable in Web era with enormous information.

TPR for keyphrase extraction is a two-stage process:

1. Build a topic interpreter to acquire the topics of words and documents.

2. Perform TPR to extract keyphrases for documents.

We will introduce the two stages in Section 2 and Section 3.

2 Building Topic Interpreters

To run TPR on a word graph, we have to acquire topic distributions of words. There are roughly two approaches that can provide topics of words: (1) Use manually annotated knowledge bases, e.g., WordNet (Miller et al., 1990); (2) Use unsupervised machine learning techniques to obtain word topics from a large-scale document collection. Since the vocabulary in WordNet cannot cover many words in modern news and research articles, we employ the second approach to build topic interpreters for TPR.

In machine learning, various methods have been proposed to infer latent topics of words and documents. These methods, known as latent topic models, derive latent topics from a large-scale document collection according to word occurrence information. Latent Dirichlet Allocation (LDA) (Blei et al., 2003) is a representative of topic models. Compared to Latent Semantic Analysis (LSA) (Landauer et al., 1998) and probabilistic LSA (pLSA) (Hofmann, 1999), LDA has more feasibility for inference and can reduce the risk of over-fitting.

In LDA, each word w of a document d is regarded to be generated by first sampling a topic z from d's topic distribution $\theta^{(d)}$, and then sampling a word from the distribution over words $\phi^{(z)}$ that characterizes topic z. In LDA, $\theta^{(d)}$ and $\phi^{(z)}$ are drawn from conjugate Dirichlet priors α and β, separately. Therefore, θ and ϕ are integrated out and the probability of word w given document d and priors is represented as follows:

$$pr(w|d,\alpha,\beta) = \sum_{z=1}^{K} pr(w|z,\beta)pr(z|d,\alpha), \quad (1)$$

where K is the number of topics.

Using LDA, we can obtain the topic distribution of each word w, namely $pr(z|w)$ for topic $z \in K$. The word topic distributions will be used in TPR. Moreover, using the obtained word topic distributions, we can infer the topic distribution of a new document (Blei et al., 2003), namely $pr(z|d)$ for each topic $z \in K$, which will be used for ranking keyphrases.

3 Topical PageRank for Keyphrase Extraction

After building a topic interpreter to acquire the topics of words and documents, we can perform keyphrase extraction for documents via TPR. Given a document d, the process of keyphrase extraction using TPR consists of the following four steps which is also illustrated in Fig. 1:

1. Construct a word graph for d according to word co-occurrences within d.

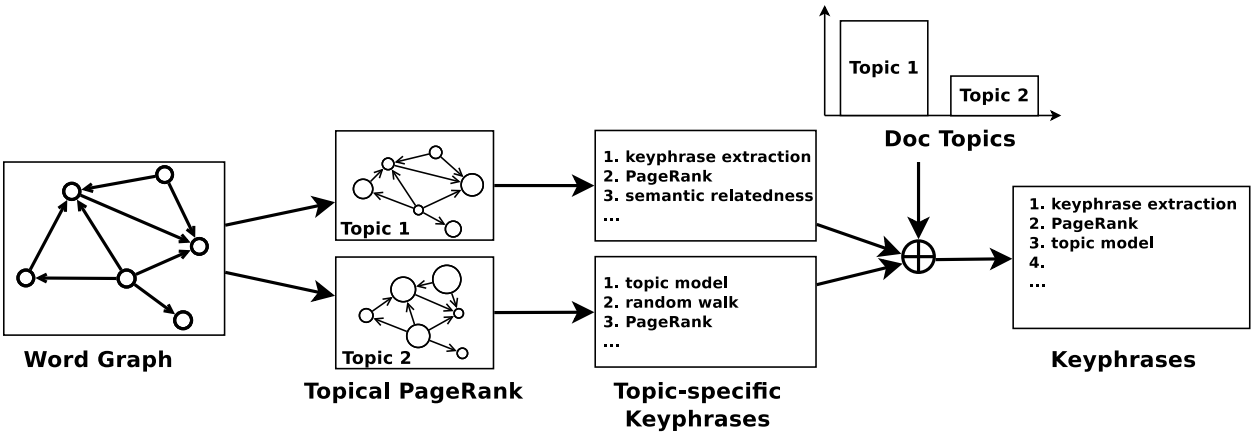


Figure 1: Topical PageRank for Keyphrase Extraction.

2. Perform TPR to calculate the importance scores for each word with respect to different topics.

3. Using the topic-specific importance scores of words, rank candidate keyphrases respect to each topic separately.

4. Given the topics of document d, integrate the topic-specific rankings of candidate keyphrases into a final ranking, and the top ranked ones are selected as keyphrases.

3.1 Constructing Word Graph

We construct a word graph according to word co-occurrences within the given document, which expresses the cohesion relationship between words in the context of document. The document is regarded as a word sequence, and the link weights between words is simply set to the co-occurrence count within a sliding window with maximum W words in the word sequence.

It was reported in (Mihalcea and Tarau, 2004) the graph direction does not influence the performance of keyphrase extraction very much. In this paper we simply construct word graphs with directions. The link directions are determined as follows. When sliding a W-width window, at each position, we add links from the first word pointing to other words within the window. Since keyphrases are usually noun phrases, we only add adjectives and nouns in word graph.

3.2 Topical PageRank

Before introducing TPR, we first give some formal notations. We denote $G = (V, E)$ as the graph of a document, with vertex set $V = \{w_1, w_2, \cdots, w_N\}$ and link set $(w_i, w_j) \in E$ if there is a link from w_i to w_j. In a word graph, each vertex represents a word, and each link indicates the relatedness between words. We denote the weight of link (w_i, w_j) as $e(w_i, w_j)$, and the out-degree of vertex w_i as $O(w_i) = \sum_{j:w_i \to w_j} e(w_i, w_j)$.

Topical PageRank is based on PageRank (Page et al., 1998). PageRank is a well known ranking algorithm that uses link information to assign global importance scores to web pages. The basic idea of PageRank is that a vertex is important if there are other important vertices pointing to it. This can be regarded as voting or recommendation among vertices. In PageRank, the score $R(w_i)$ of word w_i is defined as

$$R(w_i) = \lambda \sum_{j:w_j \to w_i} \frac{e(w_j, w_i)}{O(w_j)} R(w_j) + (1 - \lambda) \frac{1}{|V|}, \quad (2)$$

where λ is a *damping factor* range from 0 to 1, and $|V|$ is the number of vertices. The damping factor indicates that each vertex has a probability of $(1 - \lambda)$ to perform random jump to another vertex within this graph. PageRank scores are obtained by running Eq. (2) iteratively until convergence. The second term in Eq. (2) can be regarded as a smoothing factor to make the graph fulfill the property of being aperiodic and irreducible, so as to guarantee that PageRank converges to a unique stationary dis-

tribution. In PageRank, the second term is set to be the same value $\frac{1}{|V|}$ for all vertices within the graph, which indicates there are equal probabilities of random jump to all vertices.

In fact, the second term of PageRank in Eq. (2) can be set to be non-uniformed. Suppose we assign larger probabilities to some vertices, the final PageRank scores will prefer these vertices. We call this *Biased PageRank*.

The idea of Topical PageRank (TPR) is to run Biased PageRank for each topic separately. Each topic-specific PageRank prefers those words with high relevance to the corresponding topic. And the preferences are represented using random jump probabilities of words.

Formally, in the PageRank of a specific topic z, we will assign a topic-specific preference value $p_z(w)$ to each word w as its random jump probability with $\sum_{w \in V} p_z(w) = 1$. The words that are more relevant to topic z will be assigned larger probabilities when performing the PageRank. For topic z, the topic-specific PageRank scores are defined as follows:

$$R_z(w_i) = \lambda \sum_{j:w_j \to w_i} \frac{e(w_j, w_i)}{O(w_j)} R_z(w_j) + (1-\lambda) p_z(w_i). \quad (3)$$

In Fig. 1, we show an example with two topics. In this figure, we use the size of circles to indicate how relevant the word is to the topic. In the PageRanks of the two topics, high preference values will be assigned to different words with respect to the topic. Finally, the words will get different PageRank values in the two PageRanks.

The setting of preference values $p_z(w)$ will have a great influence to TPR. In this paper we use three measures to set preference values for TPR:

- $p_z(w) = pr(w|z)$, is the probability that word w occurs given topic z. This indicates how much that topic z focuses on word w.
- $p_z(w) = pr(z|w)$, is the probability of topic z given word w. This indicates how much that word w focuses on topic z.
- $p_z(w) = pr(w|z) \times pr(z|w)$, is the product of hub and authority values. This measure is inspired by the work in (Cohn and Chang, 2000).

Both PageRank and TPR are all iterative algorithms. We terminate the algorithms when the number of iterations reaches 100 or the difference of each vertex between two neighbor iterations is less than 0.001.

3.3 Extract Keyphrases Using Ranking Scores

After obtaining word ranking scores using TPR, we begin to rank candidate keyphrases. As reported in (Hulth, 2003), most manually assigned keyphrases turn out to be noun phrases. We thus select noun phrases from a document as candidate keyphrases for ranking.

The candidate keyphrases of a document is obtained as follows. The document is first tokenized. After that, we annotate the document with part-of-speech (POS) tags [1]. Third, we extract noun phrases with pattern `(adjective)*(noun)+`, which represents zero or more adjectives followed by one or more nouns. We regard these noun phrases as candidate keyphrases.

After identifying candidate keyphrases, we rank them using the ranking scores obtained by TPR. In PageRank for keyphrase extraction, the ranking score of a candidate keyphrase p is computed by summing up the ranking scores of all words within the phrase: $R(p) = \sum_{w_i \in p} R(w_i)$ (Mihalcea and Tarau, 2004; Wan and Xiao, 2008a; Wan and Xiao, 2008b). Then candidate keyphrases are ranked in descending order of ranking scores. The top M candidates are selected as keyphrases.

In TPR for keyphrase extraction, we first compute the ranking scores of candidate keyphrases separately for each topic. That is for each topic z we compute

$$R_z(p) = \sum_{w_i \in p} R_z(w_i). \quad (4)$$

By considering the topic distribution of document, We further integrate topic-specific rankings of candidate keyphrases into a final ranking and extract top-ranked ones as the keyphrases of the document. Denote the topic distribution of the document d as $pr(z|d)$ for each topic z. For each candidate keyphrase p, we compute its final ranking score as

[1] In experiments we use Stanford POS Tagger from `http://nlp.stanford.edu/software/tagger.shtml` with English tagging model `left3words-distsim-wsj`.

follows:

$$R(p) = \sum_{z=1}^{K} R_z(p) \times pr(z|d). \quad (5)$$

After ranking candidate phrases in descending order of their integrated ranking scores, we select the top M as the keyphrases of document d.

4 Experiments

4.1 Datasets

To evaluate the performance of TPR for keyphrase extraction, we carry out experiments on two datasets.

One dataset was built by Wan and Xiao [2] which was used in (Wan and Xiao, 2008b). This dataset contains 308 news articles in DUC2001 (Over et al., 2001) with $2,488$ manually annotated keyphrases. There are at most 10 keyphrases for each document. In experiments we refer to this dataset as NEWS.

The other dataset was built by Hulth [3] which was used in (Hulth, 2003). This dataset contains $2,000$ abstracts of research articles and $19,254$ manually annotated keyphrases. In experiments we refer to this dataset as RESEARCH.

Since neither NEWS nor RESEARCH itself is large enough to learn efficient topics, we use the Wikipedia snapshot at March 2008 [4] to build topic interpreters with LDA. After removing non-article pages and the articles shorter than 100 words, we collected $2,122,618$ articles. After tokenization, stop word removal and word stemming, we build the vocabulary by selecting $20,000$ words according to their document frequency. We learn LDA models by taking each Wikipedia article as a document. In experiments we learned several models with different numbers of topics, from 50 to $1,500$ respectively. For the words absent in topic models, we simply set the topic distribution of the word as uniform distribution.

4.2 Evaluation Metrics

For evaluation, the words in both standard and extracted keyphrases are reduced to base forms using Porter Stemmer [5] for comparison. In experiments we select three evaluation metrics.

The first metric is precision/recall/F-measure represented as follows,

$$p = \frac{c_{correct}}{c_{extract}}, \quad r = \frac{c_{correct}}{c_{standard}}, \quad f = \frac{2pr}{p+r}, \quad (6)$$

where $c_{correct}$ is the total number of correct keyphrases extracted by a method, $c_{extract}$ the total number of automatic extracted keyphrases, and $c_{standard}$ the total number of human-labeled standard keyphrases.

We note that the ranking order of extracted keyphrases also indicates the method performance. An extraction method will be better than another one if it can rank correct keyphrases higher. However, precision/recall/F-measure does not take the order of extracted keyphrases into account. To address the problem, we select the following two additional metrics.

One metric is *binary preference measure* (Bpref) (Buckley and Voorhees, 2004). Bpref is desirable to evaluate the performance considering the order in which the extracted keyphrases are ranked. For a document, if there are R correct keyphrases within M extracted keyphrases by a method, in which r is a correct keyphrase and n is an incorrect keyphrase, Bpref is defined as follows,

$$\text{Bpref} = \frac{1}{R} \sum_{r \in R} 1 - \frac{|n \text{ ranked higher than } r|}{M}. \quad (7)$$

The other metric is *mean reciprocal rank* (MRR) (Voorhees, 2000) which is used to evaluate how the first correct keyphrase for each document is ranked. For a document d, $rank_d$ is denoted as the rank of the first correct keyphrase with all extracted keyphrases, MRR is defined as follows,

$$\text{MRR} = \frac{1}{|D|} \sum_{d \in D} \frac{1}{rank_d}, \quad (8)$$

where D is the document set for keyphrase extraction.

Note that although the evaluation scores of most keyphrase extractors are still lower compared to

[2] http://wanxiaojun1979.googlepages.com.

[3] It was obtained from the author.

[4] http://en.wikipedia.org/wiki/Wikipedia_database.

[5] http://tartarus.org/~martin/PorterStemmer.

other NLP-tasks, it does not indicate the performance is poor because even different annotators may assign different keyphrases to the same document.

4.3 Influences of Parameters to TPR

There are four parameters in TPR that may influence the performance of keyphrase extraction including: (1) window size W for constructing word graph, (2) the number of topics K learned by LDA, (3) different settings of preference values $p_z(w)$, and (4) damping factor λ of TPR.

In this section, we look into the influences of these parameters to TPR for keyphrase extraction. Except the parameter under investigation, we set parameters to the following values: $W = 10$, $K = 1,000$, $\lambda = 0.3$ and $p_z(w) = pr(z|w)$, which are the settings when TPR achieves the best (or near best) performance on both NEWS and RESEARCH. In the following tables, we use "Pre.", "Rec." and "F." as the abbreviations of precision, recall and F-measure.

4.3.1 Window Size W

In experiments on NEWS, we find that the performance of TPR is stable when W ranges from 5 to 20 as shown in Table 1. This observation is consistent with the findings reported in (Wan and Xiao, 2008b).

Size	Pre.	Rec.	F.	Bpref	MRR
5	0.280	0.345	0.309	0.213	0.636
10	0.282	0.348	0.312	0.214	0.638
15	0.282	0.347	0.311	0.214	0.646
20	0.284	0.350	0.313	0.215	0.644

Table 1: Influence of window size W when the number of keyphrases $M = 10$ on NEWS.

Similarly, when W ranges from 2 to 10, the performance on RESEARCH does not change much. However, the performance on NEWS will become poor when $W = 20$. This is because the abstracts in RESEARCH (there are 121 words per abstract on average) are much shorter than the news articles in NEWS (there are 704 words per article on average). If the window size W is set too large on RESEARCH, the graph will become full-connected and the weights of links will tend to be equal, which cannot capture the local structure information of abstracts for keyphrase extraction.

4.3.2 The Number of Topics K

We demonstrate the influence of the number of topics K of LDA models in Table 2. Table 2 shows the results when K ranges from 50 to 1,500 and $M = 10$ on NEWS. We observe that the performance does not change much as the number of topics varies until the number is much smaller ($K = 50$). The influence is similar on RESEARCH which indicates that LDA is appropriate for obtaining topics of words and documents for TPR to extract keyphrases.

K	Pre.	Rec.	F.	Bpref	MRR
50	0.268	0.330	0.296	0.204	0.632
100	0.276	0.340	0.304	0.208	0.632
500	0.284	0.350	0.313	0.215	0.648
1000	0.282	0.348	0.312	0.214	0.638
1500	0.282	0.348	0.311	0.214	0.631

Table 2: Influence of the number of topics K when the number of keyphrases $M = 10$ on NEWS.

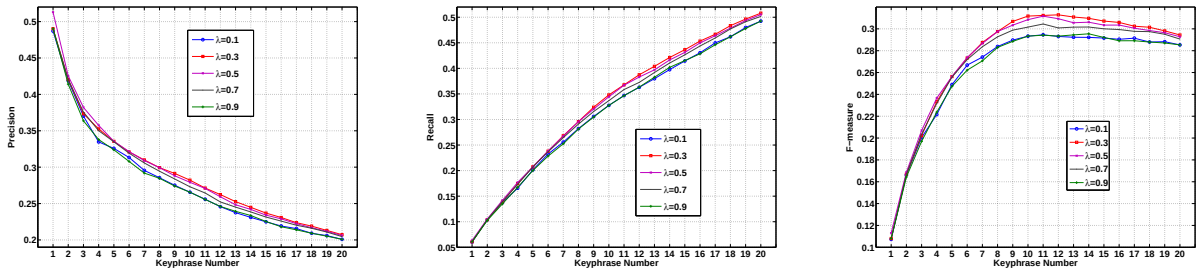
4.3.3 Damping Factor λ

Damping factor λ of TPR reconciles the influences of graph walks (the first term in Eq.(3)) and preference values (the second term in Eq.(3)) to the topic-specific PageRank scores. We demonstrate the influence of λ on NEWS in Fig. 2. This figure shows the precision/recall/F-measure when $\lambda = 0.1, 0.3, 0.5, 0.7, 0.9$ and M ranges from 1 to 20. From this figure we find that, when λ is set from 0.2 to 0.7, the performance is consistently good. The values of Bpref and MRR also keep stable with the variations of λ.

4.3.4 Preference Values

Finally, we explore the influences of different settings of preference values for TPR in Eq.(3). In Table 3 we show the influence when the number of keyphrases $M = 10$ on NEWS. From the table, we observe that $pr(z|w)$ performs the best. The similar observation is also got on RESEARCH.

In keyphrase extraction task, it is required to find the keyphrases that can appropriately represent the topics of the document. It thus does not want to extract those phrases that may appear in multiple topics like common words. The measure $pr(w|z)$ assigns preference values according to how frequently that words appear in the given topic. Therefore, the



(a) Precision (b) Recall (c) F-measure

Figure 2: Precision, recall and F-measure of TPR with $\lambda = 0.1, 0.3, 0.5, 0.7$ and 0.9 when M ranges from 1 to 20 on NEWS.

common words will always be assigned to a relatively large value in each topic-specific PageRank and finally obtain a high rank. $pr(w|z)$ is thus not a good setting of preference values in TPR. In the contrast, $pr(z|w)$ prefers those words that are focused on the given topic. Using $pr(z|w)$ to set preference values for TPR, we will tend to extract topic-focused phrases as keyphrases.

Pref	Pre.	Rec.	F.	Bpref	MRR
$pr(w\|z)$	0.256	0.316	0.283	0.192	0.584
$pr(z\|w)$	0.282	0.348	0.312	0.214	0.638
prod	0.259	0.320	0.286	0.193	0.587

Table 3: Influence of three preference value settings when the number of keyphrases $M = 10$ on NEWS.

4.4 Comparing with Baseline Methods

After we explore the influences of parameters to TPR, we obtain the best results on both NEWS and RESEARCH. We further select three baseline methods, i.e., TFIDF, PageRank and LDA, to compare with TPR.

The TFIDF computes the ranking scores of words based on words' $tfidf$ values in the document, namely $R(w) = tf_w \times \log(idf_w)$. While in PageRank (i.e., TextRank), the ranking scores of words are obtained using Eq.(2). The two baselines do not use topic information of either words or documents. The LDA computes the ranking score for each word using the topical similarity between the word and the document. Given the topics of the document d and a word w, We have used various methods to compute similarity including cosine similarity, predictive likelihood and KL-divergence (Heinrich, 2005), among which cosine similarity performs the best on both datasets. Therefore, we only show the results of the LDA baseline calculated using cosine similarity.

In Tables 4 and 5 we show the comparing results of the four methods on both NEWS and RESEARCH. Since the average number of manual-labeled keyphrases on NEWS is larger than RESEARCH, we set $M = 10$ for NEWS and $M = 5$ for RESEARCH. The parameter settings on both NEWS and RESEARCH have been stated in Section 4.3.

Method	Pre.	Rec.	F.	Bpref	MRR
TFIDF	0.239	0.295	0.264	0.179	0.576
PageRank	0.242	0.299	0.267	0.184	0.564
LDA	0.259	0.320	0.286	0.194	0.518
TPR	**0.282**	**0.348**	**0.312**	**0.214**	**0.638**

Table 4: Comparing results on NEWS when the number of keyphrases $M = 10$.

Method	Pre.	Rec.	F.	Bpref	MRR
TFIDF	0.333	0.173	0.227	0.255	0.565
PageRank	0.330	0.171	0.225	0.263	0.575
LDA	0.332	0.172	0.227	0.254	0.548
TPR	**0.354**	**0.183**	**0.242**	**0.274**	**0.583**

Table 5: Comparing results on RESEARCH when the number of keyphrases $M = 5$.

From the two tables, we have the following observations.

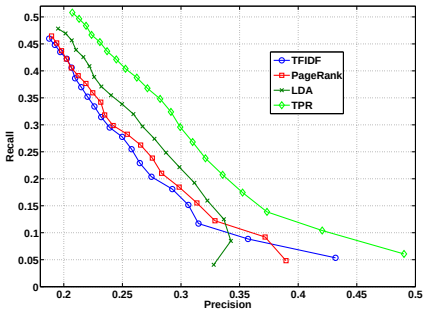


Figure 3: Precision-recall results on NEWS when M ranges from 1 to 20.

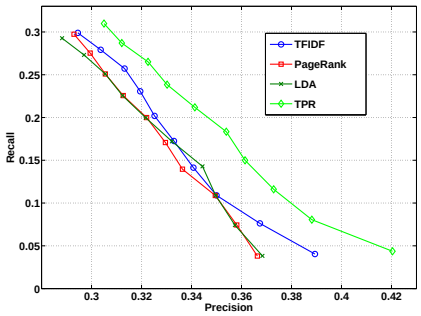


Figure 4: Precision-recall results on RESEARCH when M ranges from 1 to 10.

First, TPR outperform all baselines on both datasets. The improvements are all statistically significant tested with bootstrap re-sampling with 95% confidence. This indicates the robustness and effectiveness of TPR.

Second, LDA performs equal or better than TFIDF and PageRank under precision/recall/F-measure. However, the performance of LDA under MRR is much worse than TFIDF and PageRank, which indicates LDA fails to correctly extract the first keyphrase earlier than other methods. The reason is: (1) LDA does not consider the local structure information of document as PageRank, and (2) LDA also does not consider the frequency information of words within the document. In the contrast, TPR enjoys the advantages of both LDA and TFIDF/PageRank, by using the external topic information like LDA and internal document structure like TFIDF/PageRank.

Moreover, in Figures 3 and 4 we show the precision-recall relations of four methods on NEWS and RESEARCH. Each point on the precision-recall curve is evaluated on different numbers of extracted keyphrases M. The closer the curve to the upper right, the better the overall performance. The results again illustrate the superiority of TPR.

4.5 Extracting Example

At the end, in Table 6 we show an example of extracted keyphrases using TPR from a news article with title "Arafat Says U.S. Threatening to Kill PLO Officials" (The article number in DUC2001 is AP880510-0178). Here we only show the top 10 keyphrases, and the correctly extracted ones are marked with "(+)". We also mark the number of correctly extracted keyphrases after method name like "(+7)" after TPR. We also illustrate the top 3 topics of the document with their topic-specific keyphrases. It is obvious that the top topics, on "Palestine", "Israel" and "terrorism" separately, have a good coverage on the discussion objects of this article, which also demonstrate a good diversity with each other. By integrating these topic-specific keyphrases considering the proportions of these topics, we obtain the best performance of keyphrase extraction using TPR.

In Table 7 we also show the extracted keyphrases of baselines from the same news article. For TFIDF, it only considered the frequency properties of words, and thus highly ranked the phrases with "PLO" which appeared about 16 times in this article, and failed to extract the keyphrases on topic "Israel". LDA only measured the importance of words using document topics without considering the frequency information of words and thus missed keyphrases with high-frequency words. For example, LDA failed to extract keyphrase "political assassination", in which the word "assassination" occurred 8 times in this article.

5 Related Work

In this paper we proposed TPR for keyphrase extraction. A pioneering achievement in keyphrase extraction was carried out in (Turney, 1999) which regarded keyphrase extraction as a classification task. Generally, the supervised methods need manually annotated training set which is time-consuming and in this paper we focus on unsupervised method.

TPR (+7)
PLO leader Yasser Arafat(+), Abu Jihad, Khalil Wazir(+), slaying Wazir, political assassination(+), Palestinian guerrillas(+), particulary Palestinian circles, Israeli officials(+), Israeli squad(+), terrorist attacks(+)
TPR, Rank 1 Topic on "Palestine"
PLO leader Yasser Arafat(+), United States(+), State Department spokesman Charles Redman, Abu Jihad, U.S. government document, Palestine Liberation Organization leader, political assassination(+), Israeli officials(+), alleged document
TPR, Rank 2 Topic on "Israel"
PLO leader Yasser Arafat(+), United States(+), Palestine Liberation Organization leader, Israeli officials(+), U.S. government document, alleged document, Arab government, slaying Wazir, State Department spokesman Charles Redman, Khalil Wazir(+)
TPR, Rank 3 Topic on "terrorism"
terrorist attacks(+), PLO leader Yasser Arafat(+), Abu Jihad, United States(+), alleged document, U.S. government document, Palestine Liberation Organization leader, State Department spokesman Charles Redman, political assassination(+), full cooperation

Table 6: Extracted keyphrases by TPR.

TFIDF (+5)
PLO leader Yasser Arafat(+), PLO attacks, PLO offices, PLO officials(+), PLO leaders, Abu Jihad, terrorist attacks(+), Khalil Wazir(+), slaying wazir, political assassination(+)
PageRank (+3)
PLO leader Yasser Arafat(+), PLO officials(+), PLO attacks, United States(+), PLO offices, PLO leaders, State Department spokesman Charles Redman, U.S. government document, alleged document, Abu Jihad
LDA (+5)
PLO leader Yasser Arafat(+), Palestine Liberation Organization leader, Khalil Wazir(+), Palestinian guerrillas(+), Abu Jihad, Israeli officials(+), particulary Palestinian circles, Arab government, State Department spokesman Charles Redman, Israeli squad(+)

Table 7: Extracted keyphrases by baselines.

Starting with TextRank (Mihalcea and Tarau, 2004), graph-based ranking methods are becoming the most widely used unsupervised approach for keyphrase extraction. Litvak and Last (2008) applied HITS algorithm on the word graph of a document for keyphrase extraction. Although HITS itself worked the similar performance to PageRank, we plan to explore the integration of topics and HITS in future work. Wan (2008b; 2008a) used a small number of nearest neighbor documents to provide more knowledge for keyphrase extraction. Some methods used clustering techniques on word graphs for keyphrase extraction (Grineva et al., 2009; Liu et al., 2009). The clustering-based method performed well on short abstracts (with F-measure 0.382 on RESEARCH) but poorly on long articles (NEWS with F-measure score 0.216) due to two non-trivial issues: (1) how to determine the number of clusters, and (2) how to weight each cluster and select keyphrases from the clusters. In this paper we focus on improving graph-based methods via topic decomposition, we thus only compare with PageRank as well as TFIDF and LDA and do not compare with clustering-based methods in details.

In recent years, two algorithms were proposed to rank web pages by incorporating topic information of web pages within PageRank (Haveliwala, 2002; Nie et al., 2006). The method in (Haveliwala, 2002), is similar to TPR which also decompose PageRank into various topics. However, the method in (Haveliwala, 2002) only considered to set the preference values using $pr(w|z)$ (In the context of (Haveliwala, 2002), w indicates Web pages). In Section 4.3.4 we have shown that the setting of using $pr(z|w)$ is much better than $pr(w|z)$.

Nie et al. (2006) proposed a more complicated ranking method. In this method, topical PageRanks are performed together. The basic idea of (Nie et al., 2006) is, when surfing following a graph link from vertex w_i to w_j, the ranking score on topic z of w_i will have a higher probability to pass to the same topic of w_j and have a lower probability to pass to a different topic of w_j. When the inter-topic jump probability is 0, this method is identical to (Haveli-

wala, 2002). We implemented the method and found that the random jumps between topics did not help improve the performance for keyphrase extraction, and did not demonstrate the results of this method.

6 Conclusion and Future Work

In this paper we propose a new graph-based framework, Topical PageRank, which incorporates topic information within random walk for keyphrase extraction. Experiments on two datasets show that TPR achieves better performance than other baseline methods. We also investigate the influence of various parameters on TPR, which indicates the effectiveness and robustness of the new method.

We consider the following research directions as future work.

1. In this paper we obtained latent topics using LDA learned from Wikipedia. We design to obtain topics using other machine learning methods and from other knowledge bases, and investigate the influence to performance of keyphrase extraction.

2. In this paper we integrated topic information in PageRank. We plan to consider topic information in other graph-based ranking algorithms such as HITS (Kleinberg, 1999).

3. In this paper we used Wikipedia to train LDA by assuming Wikipedia is an extensive snapshot of human knowledge which can cover most topics talked about in NEWS and RESEARCH. In fact, the learned topics are highly dependent on the learning corpus. We will investigate the influence of corpus selection in training LDA for keyphrase extraction using TPR.

Acknowledgments

This work is supported by the National Natural Science Foundation of China under Grant No. 60873174. The authors would like to thank Anette Hulth and Xiaojun Wan for kindly sharing their datasets. The authors would also thank Xiance Si, Tom Chao Zhou, Peng Li for their insightful suggestions and comments.

References

David M. Blei, Andrew Y. Ng, and Michael I. Jordan. 2003. Latent dirichlet allocation. *Journal of Machine Learning Research*, 3:993–1022, January.

C. Buckley and E.M. Voorhees. 2004. Retrieval evaluation with incomplete information. In *Proceedings of SIGIR*, pages 25–32.

David Cohn and Huan Chang. 2000. Learning to probabilistically identify authoritative documents. In *Proceedings of ICML*, pages 167–174.

M. Grineva, M. Grinev, and D. Lizorkin. 2009. Extracting key terms from noisy and multi-theme documents. In *Proceedings of WWW*, pages 661–670.

Taher H. Haveliwala. 2002. Topic-sensitive pagerank. In *Proceedings of WWW*, pages 517–526.

G. Heinrich. 2005. Parameter estimation for text analysis. *Web: http://www. arbylon. net/publications/text-est.*

Thomas Hofmann. 1999. Probabilistic latent semantic indexing. In *Proceedings of SIGIR*, pages 50–57.

Anette Hulth. 2003. Improved automatic keyword extraction given more linguistic knowledge. In *Proceedings of EMNLP*, pages 216–223.

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

T.K. Landauer, P.W. Foltz, and D. Laham. 1998. An introduction to latent semantic analysis. *Discourse Processes*, 25:259–284.

Marina Litvak and Mark Last. 2008. Graph-based keyword extraction for single-document summarization. In *Proceedings of the workshop Multi-source Multilingual Information Extraction and Summarization*, pages 17–24.

Zhiyuan Liu, Peng Li, Yabin Zheng, and Maosong Sun. 2009. Clustering to find exemplar terms for keyphrase extraction. In *Proceedings of EMNLP*, pages 257–266.

C.D. Manning and H. Schutze. 2000. *Foundations of statistical natural language processing*. MIT Press.

Rada Mihalcea and Paul Tarau. 2004. Textrank: Bringing order into texts. In *Proceedings of EMNLP*, pages 404–411.

George A. Miller, Richard Beckwith, Christiane Fellbaum, Derek Gross, and Katherine Miller. 1990. WordNet: An on-line lexical database. *International Journal of Lexicography*, 3:235–244.

Thuy Nguyen and Min-Yen Kan. 2007. Keyphrase extraction in scientific publications. In *Proceedings of the 10th International Conference on Asian Digital Libraries*, pages 317–326.

Lan Nie, Brian D. Davison, and Xiaoguang Qi. 2006. Topical link analysis for web search. In *Proceedings of SIGIR*, pages 91–98.

P. Over, W. Liggett, H. Gilbert, A. Sakharov, and M. Thatcher. 2001. Introduction to duc-2001: An intrinsic evaluation of generic news text summarization systems. In *Proceedings of DUC2001*.

L. Page, S. Brin, R. Motwani, and T. Winograd. 1998. The pagerank citation ranking: Bringing order to the web. *Technical report, Stanford Digital Library Technologies Project, 1998*.

Peter D. Turney. 1999. Learning to extract keyphrases from text. *National Research Council Canada, Institute for Information Technology, Technical Report ERB-1057*.

Peter D. Turney. 2000. Learning algorithms for keyphrase extraction. *Information Retrieval*, 2(4):303–336.

E.M. Voorhees. 2000. The trec-8 question answering track report. In *Proceedings of TREC*, pages 77–82.

Xiaojun Wan and Jianguo Xiao. 2008a. Collabrank: Towards a collaborative approach to single-document keyphrase extraction. In *Proceedings of COLING*, pages 969–976.

Xiaojun Wan and Jianguo Xiao. 2008b. Single document keyphrase extraction using neighborhood knowledge. In *Proceedings of AAAI*, pages 855–860.