Exploiting Description Knowledge for Keyphrase Extraction

Fang Wang[1](✉), Zhongyuan Wang[2], Senzhang Wang[1], and Zhoujun Li[1]

[1]State Key Laboratory of Software Development Environment
Beihang University, Beijing, China
{fangwang,szwang,lizj}@buaa.edu.cn
[2]School of Information, Renmin University of China, Beijing, China
wzhy@outlook.com

Abstract. Keyphrase extraction is essential for many IR and NLP tasks. Existing methods usually use the phrases of the document separately without distinguishing the potential semantic correlations among them, or other statistical features from knowledge bases such as WordNet and Wikipedia. However, the mutual semantic information between phrases is also important, and exploiting their correlations may potentially help us more effectively extract the keyphrases. Generally, phrases in the title are more likely to be keyphrases reflecting the document topics, and phrases in the body are usually used to describe the document topics. We regard the relation between the title phrase and body phrase as a description relation. To this end, this paper proposes a novel keyphrase extraction approach by exploiting massive description relations. To make use of the semantic information provided by the description relations, we organize the phrases of a document as a description graph, and employ various graph-based ranking algorithms to rank the candidates. Experimental results on the real dataset demonstrate the effectiveness of the proposed approach in keyphrase extraction.

Keywords: Keyphrase extraction, Description relation, Wikipedia.

1 Introduction

A keyphrase is defined as a meaningful and significant expression consisting of a single word (e.g. *information*) or compound words (e.g. *information retrieval*) [1]. The keyphrases of a document can be considered as a very short summary of the document, precisely expressing the primary topics and themes. Automatic keyphrase extraction has been widely used in many applications including document indexing, summarization [2], text classification [3], and information retrieval [4].

Existing approaches can be generally categorized as keyphrase extraction and keyphrase assignment. Keyphrase extraction focuses on choosing the representative phrases from the body of the source document, while keyphrase assignment aims to select the keyphases from a controlled vocabulary. Some previous works [5, 6] have proved that keyphrase assignment tends to yield more precise and readable results. Besides, a predefined controlled vocabulary will favor the extraction of semantic relations between candidates. For example, given the predefined vocabulary, we can pre-extract relations of interest between candidates in the vocabulary. Therefore, this paper focuses on generic keyphrase assignment.

Keyphrase generation has attracted much attention for a long time. Various methods [7, 8, 5, 9–12] have been proposed. Despite their differences, most methods utilize some heuristic features to retrieve a set of words that accurately epitomize the text. TF-IDF [7] is the simplest method, which measures a phrase's frequency in a document compared to its rarity in general use. KEA [8] is one of the most famous approaches. It uses two statistical features: *tf-idf* and *first occurrence*, and trains a Naive Bayes classifier [3] to extract the keyphrases. In Wikify! system [13], a new feature called *Keyphraseness* is proposed for automatic keyword extraction, by leveraging the hyperlinks within Wikipedia. In the state-of-the-art method Maui [9], there are 14 features, including distance between phrases, frequency, phrase length, similarity between phrases, etc. However, most existing work uses statistical features extracted from the local text or other knowledge bases such as Wikipedia. The semantic relations between phrases are ignored. In this paper, we argue that mining the semantic correlations among the phrases may help us better identify keyphrases.

This paper studies how to improve the keyphrase extraction by exploiting the *description* relations among the phrases. Here we define a description relation as the co-occurrence of two phrases in the same article with one appeared in the title and the other appeared in the body. Our intuition is as follows. Generally, the article title can represent the main topics, such that the phrase in the title, short for *title phrase*, tends to be a keyphrase, and it is more likely that the phrase in the body text (keyphrase or not), short for *body phrase*, can be considered as a description of the title phrases. The description relation may provide additional knowledge and more clues for keyphrase extraction. For example, given the knowledge that *search engine* and *language model* used to describe *information retrieval*, when the three phrases co-occur in a new text, we think *information retrieval* tends to be more important than the other phrases and it is more likely to be a keyphrase.

We propose a novel framework for keyphrase extraction by exploiting description relations among candidate keyphrases. We first mine massive description relations from the article corpus. The description relations are then used to help extract keyphrases for a new given article. In this paper, we use Wikipedia as a domain-independent vocabulary for generic keyphrase assignment, and map document phrases to Wikipedia article titles as candidate phrases. Then, we organize the candidates as a description graph according to the extracted description relations. Based on the description graph, various ranking algorithms are employed to scoring the candidates. Extensive experiments are conducted to show the effectiveness of the proposed methods. We have performed keyphrase extraction experiments on an open dataset with human-annotated keyphrases. Experimental results demonstrate the description relationships can significantly improve the performance of keyphrase extraction.

The rest of the paper is organized as follows. Section 2 introduces the framework of the proposed method and then describes each module in detail. In section 3, we evaluate our approach and report the results. We discuss related work in section 4, and conclude this paper in section 5.

2 The Proposed Approach

We propose a unified framework to extract description relations from Wikipedia, and exploit these relations to help us for generic keyphrase extraction. The framework consists of two steps: description relation construction and keyphrase extraction by exploiting the description relations, as shown in Figure 1.

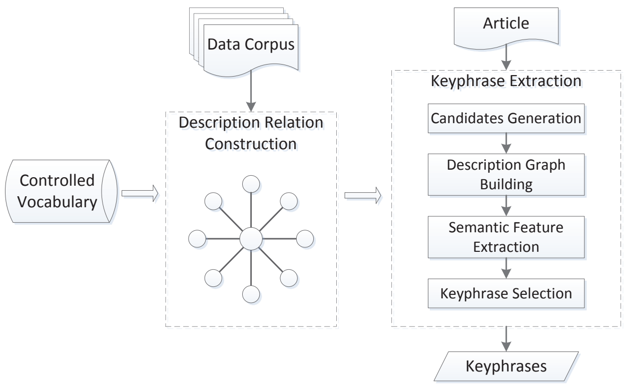


Fig. 1. The flowchart of the proposed approach

Given the predefined controlled vocabulary, the proposed approach first extracts a large number of description relations from the data corpus. The description relation can be seen as a special case of co-occurrence relation. In our description relation case, one phrase is from the title while the other one is from the body text of the article. More extraction details are given in section 2.1. Given an article A for keyphrase extraction, our approach first generates the candidates for A. To represent the description semantic among the candidates, a description graph G_d is built by leveraging the extracted massive description relations. Details are given in section 2.2. Then, based on G_d, various ranking methods are employed to rank the candidates and the top ranked phrases can be selected as the keyphrases. Also, these ranking scores can be seen as new semantic features. For keyphrase extraction, both the statistical features in the article and the new semantic features mined from description relations are incorporated into the keyphrase evaluation process.

2.1 Description Relation Construction

We first give a formal definition of the description relation and then describe our extraction strategy.

Definition 1. ***Description Relation.*** *Given the article corpus* $D = \{A_1, A_2, ..., A_N\}$*, assuming the article* A_i *can be represented as such a tuple* $A_i = (Title_i, Body_i)$*, where* $Title_i$ *represents a set of phrases in the title of* A_i *and* $Body_i$ *represents a set of phrases in the body of* A_i*, there exists a description relation between phrases* p_i *and* p_j*, if the following conditions are satisfied:* $\exists A_k : p_i \in Title_k \wedge p_j \in Body_k$.

This paper uses Wikipedia as a domain-independent vocabulary and the Wikipedia articles are used as the data corpus, since Wikipedia contains plenty of notable entities. Here an entity is a term in Wikipedia vocabulary and we regard it a candidate keyphrase. Each article in Wikipedia provides information for a specific entity and also mentions other entities related to it.

For title phrase extraction, the Wikipedia article title itself can be seen as a title phrase, because the article title is an entity (or *term*) in Wikipedia. For body phrase extraction, previous works use hyperlinks in the article to recognize entities [14–16]. But entities extracted in this way are not sufficient, because there exists many entities without hyperlinks. Besides, some entities may have hyperlinks in previous text, but no links anymore when they are mentioned again. This would lead to insufficient statistic information. This paper uses Wikipedia Miner [17] [1] to identify body phrases. It can detect and disambiguate Wikipedia entities when they are mentioned in documents, which is the state-of-the-art open domain extraction method. As we know, linking free text from web pages to existing knowledge bases (e.g., Wikipedia) is far from trivial. Improving the entity extraction will probably further improve the quality of extracted description relations. We leave this for future work.

In this paper, we use English Wikipedia[2] as the data corpus, and extract the description relation using the method described as above. For gathering statistical information, the frequency of each extracted phrases are recorded. Phrases and their description relations are extracted in forms of triples: $<v_i, v_j, n_{ij}>$, where v_i is the title phrase, v_j is the body phrase, and n_{ij} is the frequency, e.g., $<information\ retrieval, search\ engine, 7>$.

Noises may exist, since not every phrase in the article directly relates with the title. Intuitively, the phrases with more co-occurrence frequency are more likely to be related with the title phrase. Thus we keep the phrases whose $frequency > 2$ and filter others to reduce noises. Finally, we got 3.8 million phrases and 68 million corresponding description relations.

2.2 Keyphrase Extraction

The process of our keyphrase extraction can be divided into four steps: candidate generation, description graph building, semantic feature extraction and keyphrase selection. In this subsection, we describes each steps in detail.

[1] http://wikipedia-miner.sourceforge.net
[2] In this paper, we used enwiki dump progress on 20130503. http://en.wikipedia.org/wiki/Wikipedia:Database_download

Candidates Generation Candidate generation traditionally uses phrases extracted from the document itself, but can be made more consistent by choosing entities with respect to Wikipedia [9]. Similar to the body phrase extraction, we use Wikipedia Miner to map the document d to Wikipedia entities. The detected entities are considered the candidate keyphrases.

Description Graph Building This paper deeply mines the mutual semantic relatedness between candidates by leveraging fruitful description relations. Specifically, candidate keyphrases $V = \{v_1, v_2, ..., v_n\}$ in the given article A are constructed as a description graph G_d based on the description relations among them. Each node in G_d is a candidate phrase. If two nodes have a description relationship, then they will be connected with an edge directed from the body phrase to title phrase.

The description graph G_d can reflect the global description relation information between the candidates. We use an affinity matrix M to describe G_d, with each entry corresponding to the weight of an edge in the graph. $M = (M_{i,j})_{|V|\times|V|}$ is defined as follows:

$$M_{i,j} = \begin{cases} w(v_i, v_j) & : \ if\ v_i\ connects\ to\ v_j\ and\ i \neq j \\ 0 & : \ otherwise \end{cases} \tag{1}$$

where $w(v_i, v_j)$ is the edge weight. This paper uses the count of the description frequency between v_i and v_j as the weight. Then M is normalized to $\widetilde{M}$ as follows to make the sum of each row equal to 1:

$$\widetilde{M}_{i,j} = \begin{cases} \frac{M_{i,j}}{\sum_j^{|V|} M_{i,j}} & : \ if\ \sum_j^{|V|} M_{i,j} \neq 0 \\ 0 & : \ otherwise \end{cases} \tag{2}$$

Semantic Feature Extraction Based on the built description graph, we mine the mutual semantic information between the candidates by making use of their description relations. Graph-based ranking methods [18, 10, 19, 20] have been widely used in keyphrase extraction. Also, there are other graph-based ranking methods [21–23], but more information is usually needed in these algorithms, such as the personalized vector in Personalized PageRank [21]. We believe leverage more information may further improve the ranking results, we leave this for future work. In this paper, we employ the following graph-based ranking methods for candidate keyphrase selection.

– *Phrase Degree* is the degree of v_i in G_d, quantifies the relatedness of a candidate phrase to other candidates. The value is normalized by the max degree in this graph. Notice that an edge indicates that one phrase used to describe the other one. Hence the degree value reflects the frequency that the candidate described (or was described by) other candidates.

$$Degree(v_i) = Deg(v_i) = |Nei(v_i)| \tag{3}$$

– *Phrase Closeness* measures how close a phrase is to other phrases. The higher the value, the more likely is the candidate to be a keyphrase.

$$Closeness(v_i) = \frac{1}{D_{avg}(v_i)} = \frac{n-1}{\sum_{j\neq i}^{n} g(v_i, v_j)} \tag{4}$$

where $g(v_i, v_j)$ is the number of hops in the shortest path between v_i and v_j.
– *Phrase Betweenness* counts the number of shortest paths that pass one node. Nodes with high betweenness are important in connecting other phrases and tend to be more informative.

$$Betweenness(v_i) = \sum_{v_s \neq v_i \neq v_t \in V}^{n} \frac{\sigma_{st}(v_i)}{\sigma_{st}} \tag{5}$$

where σ_{st} is number of shortest paths between v_s and v_t, and $\sigma_{st}(v_i)$ is the number of which pass v_i.
– *Phrase PageRank* can reflect the phrase importance in G_d. The more importance, the more likely is the candidate to be a keyphrase. It can be deduced from those of all other phrases linked with it and it can be formulated in the following recursive form:

$$PRscore(v_i) = \mu \cdot \sum_{j\in V, j\neq i} PRscore(v_j) \cdot \widetilde{M}_{j,i} + \frac{1-\mu}{|V|} \tag{6}$$

The matrix form is:

$$\boldsymbol{\lambda} = \mu \cdot \widetilde{M}^T \cdot \boldsymbol{\lambda} + \frac{1-\mu}{|V|} \cdot \boldsymbol{e} \tag{7}$$

where $\boldsymbol{\lambda} = [PRscore(v_i)]_{|V|\times 1}$ is the vector of phrase saliency scores, $\boldsymbol{e}$ is a vector with all elements equal to 1 and μ is the damping factor.
– *Phrase PageRankPrior* is a variant of the PageRank score, as shown in Eq. 8. We define a vector of prior probabilities $PR = \{pr(v_1), pr(v_2), ..., pr(v_n)\}$ such that the probabilities sum to 1, where $pr(v_i)$ denotes the relative importance (or "prior bias") we attach to node v_i. In this paper, we regard the normalized *tf-idf* values of the candidates in article d as prior probabilities for each candidate, in considering that candidates with higher *tf-idf* are more important in general.

$$PRRscore(v_i) = \mu \cdot \sum_{j\in V, j\neq i} PRRscore(v_j) \cdot \widetilde{M}_{j,i} + (1-\mu) \cdot pr(v_i) \tag{8}$$

Keyphrase Selection After the scores of all candidate phrases in the document have been computed, candidate phrases are selected and evaluated for the article A. We obtain five graph-based ranking scores by making use of the description relations. As an unsupervised method, all the candidate phrases in A are ranked in decreasing order of the phrase scores and the top k phrases are selected as the keyphrases of A. k ranges from 1 to 15 in this study. Also, these graph-based scores can be used as new semantic features in supervised methods, such as Naive Bayes [8], Decision Tree [9] and SVM [24].

3 Experiments

In this section, we first calculate the ranking scores described in section 2.2. Next, we use them as unsupervised ranking methods to assess the their effectiveness separately. Finally, we incorporate these scores into a state-of-the-art supervised method to evaluate their performances in keyphrase extraction.

3.1 Feature Effectiveness

In this subsection, we evaluate the effectiveness of various ranking methods in ranking the candidate keyphrases. The dataset and evaluation metrics are first introduced.

Dataset The open dataset Wiki20[3] is used to evaluate the ranking methods. It consists of 20 computer science related documents. Each document was manually annotated independently by 15 teams of graduate students. For each document, we merge the label results from all the 15 teams together as the gold standard. Thus each document has about 34 keyphrases on average in this experiment.

Metrics To evaluate the proposed rankings, we resort to three popular metrics used in keyphrase extraction - *precision*, *recall*, and *F-Score*. Let the classification results be divided into four subsets, namely TP (correct results), FP (unexpected results), FN (missing results) and TN (correct absence of results). Then it comes the definitions of precision (P) and recall (R):

$$precision = \frac{|TP|}{|TP \cup FP|}, \quad recall = \frac{|TP|}{|TP \cup FN|}$$

F-score (F) is a measure of a test's accuracy. It considers both the precision and the recall to compute the score. The general formula for positive real β is:

$$F_\beta = (1+\beta^2) \times \frac{precision \times recall}{\beta^2 precision + recall} \tag{9}$$

We match the keyphrases in the gold standard with those the rankings provide, and calculate the averaged precision, recall and F-score ($\beta = 1$) over all the documents. In the evaluation, we check the performance of each ranking method over the top 5, 10 and 15 ranked keyphrases. We rank the ranking methods by F-score over the top 15 candidates.

Evaluation In this study, we use the open source toolkit JUNG[4] to implement various ranking algorithms. In computing PageRank score (Eq. 6) and PageRankPrior score (Eq. 8), we set $\mu = 0.15$. All these features are normalized by dividing them with the highest value in the candidates separately. To assess the

Table 1. Performance of the proposed rankings vs. TF-IDF, ranked by F-score.

Method	Rank	Top 5 candidates			Top 10 candidates			Top 15 candidates		
		P (%)	R (%)	F (%)	P (%)	R (%)	F (%)	P (%)	R (%)	F (%)
PageRankPrior	1	37.67	5.76	10.00	35.50	11.20	**17.03**	31.56	15.07	**20.10**
Betweenness	2	37.00	5.60	9.69	33.00	9.93	15.17	29.06	13.89	**18.65**
TF-IDF	3	46.00	6.93	**11.99**	36.50	11.10	16.90	28.67	13.02	17.77
PageRank	4	32.00	4.72	8.19	32.00	9.63	14.72	29.00	12.84	17.67
Degree	5	34.00	5.07	8.79	28.00	8.21	12.62	25.33	11.31	15.52
Closeness	6	12.00	1.77	3.07	21.50	6.35	9.74	24.00	10.71	14.69

effectiveness of the proposed rankings, we compare them with *TF-IDF*. Table 1 shows the ranking results.

As can be seen from Table 1, the proposed PageRankPrior performs best over the top 10 and 15 candidates, and Betweenness also outperforms the strong feature TF-IDF over the top 15 candidates, which shows that the description relation does benefit keyphrase extraction. Moreover, we can see that the PageRank method can perform as well as the baseline TF-IDF over the top 15 candidates, but it is inferior to PageRankPrior, which demonstrates that using the *TF-IDF* as the prior probability is helpful to improve the extraction. Although TF-IDF performs well in the top 5 ranking, it is inferior to our approaches in the top 10 and top 15 rankings. It implies the proposed semantic features are good features that can be combined with TF-IDF, to improve keyphrase extraction.

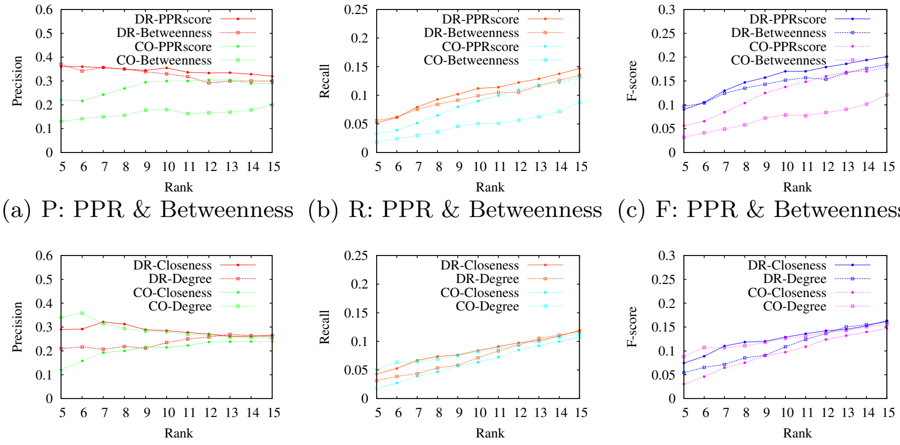
To assess the effectiveness of the description relation in keyphrase extraction, we compare the description relation with co-occurrence relation. Instead of building description graph (Section 2.2), we construct a phrase co-occurrence graph G_c for the given document. Similarly, the node in G_c is the candidate phrase. But in G_c, two nodes will be connected with an undirected edge if they have a co-occurrence relation[5] in the given document. Also, we use JUNG to calculate the ranking scores based on G_c, and compare the description based features with the co-occurrence based features on Wiki20. The *precision*, *recall* and *F-score* are used as the metrics.

Figure 2 shows the performance curves, where "DR" denotes the description relation, "CO" for co-occurrence relation and "PPRscore" for PageRankPrior score. We omit PageRank since it has the similar trend with PageRankPrior.

We draw the following observations from Figure 2:

(1) Almost all the description based features beat the the co-occurrence based rankings, which indicates that the description relation performs better than co-occurrence relation in keyphrase extraction. It implies that the description relation could provide additional knowledge and more clues, such as the mutual description information between two phrases.
(2) The performances of the two relations varies widely in different ranking methods. In PageRankPrior and Betweenness, the description relation per-

[3] http://maui-indexer.googlecode.com/files/wiki20.tar.gz
[4] http://jung.sourceforge.net/
[5] Here we use sentence-wise co-occurrence, i.e., two phrases co-occurred in a sentence.



(a) P: PPR & Betweenness (b) R: PPR & Betweenness (c) F: PPR & Betweenness

(d) P: Degree & Closeness (e) R: Degree & Closeness (f) F: Degree & Closeness

Fig. 2. Description based features vs. co-occurrence based features

forms much better than the co-occurrence relation. However, in Degree and Closeness, the performance difference between description relation and co-occurrence are very small. This also suggests that it is critical to employ the right method for making use of the additional knowledge.

3.2 Keyphrase Extraction Performance

In this subsection, we incorporate the proposed semantic features into a state-of-the-art supervised method Maui [9] to evaluate their overall performances in keyphrase extraction. The Wiki20 dataset is used again as the benchmark, but this time we follow the evaluation approach used in Maui [25] to measure the quality of keyphrases assigned to the test documents by our method via comparing them with those assigned by each team of human annotators.

Metric For fair comparison, we use the same metric as the Maui, namely the inter-indexer consistency [26]. It compares the results with those assigned by each team of the standard data in Eq. 10:

$$Consistency(A, B) = \frac{2 \times |A \cap B|}{|A| + |B|} \tag{10}$$

where A and B represent the two keyphrase sets assigned by the model and the standard data respectively. Here, we first average the sum of all consistency scores between the model and one team on the 20 test documents, then average the results over all the 15 teams as the final score. The evaluation is performed with leave-one-out cross-validation (LOOCV). The higher the consistency value, the more accurate the method is in assigning the keyphrases.

Result We reproduce Maui model following the work by Medelyan and Witten [25]. We use the same version of Wikipedia dump[6] as Muai did. In our reproduced version, the results are nearly the same as Maui, indicating that our rebuilt model works. We compare the performance of our model (added our semantic features) with three baselines: TF-IDF, KEA++ [9] and Maui. Table 2 gives the comparison results of the baseline methods.

Table 2. Performance Comparison with Baselines: consistency with the standard data (%), over the top 5 ranked candidates.

Method	Learning Approach	Consistency		
		Min.	Avg.	Max.
TF-IDF*	Unsupervised	5.7	8.3	14.7
KEA++*	Naive Bayes	15.5	22.6	27.3
Maui	Bagging decision trees	22.4	31.6	38.1
Our model	Bagging decision trees	**27.4**	**33.6**	**38.8**

* The result is referred from Medelyan[25].

We can see that adding description relation based features does improve the performance of Maui. We use Student's t-test to see whether the average consistency gained by our model is significant. The null hypothesis is $H_0: \mu_1 = \mu_0$ and the alternative hypothesis is $H_1: \mu_1 \neq \mu_0$. In the formulations, μ_1 indicates our result and μ_0 represents the average consistency gained by Maui, which is 31.6. We have 15 samples, namely the 15 consistency values between our model and the 15 teams of annotators. The t-test result, p-value, is 0.02, lower than the significance level 0.05. We conclude that the title-body relation can significantly improve the performance of keyphrase extraction on the studied dataset.

Furthermore, we evaluate the performances of Maui and our model Maui+ (added our semantic features) on a per document basis (Fig.3). Maui's consistency on each document is referenced from Medelyan [25]. For each document, the consistency between our model and the standard data is computed by averaging the inter-indexer consistency over all 15 teams. The vertical axis represents the consistency difference between machine results and standard data.

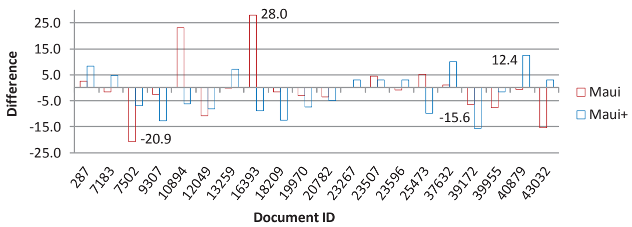


Fig. 3. Difference with standard data per test document

[6] http://sourceforge.net/projects/wikipedia-miner/files/data/en_20090306/

As can be seen from Fig. 3, the fluctuations of our model are much smaller than those of Maui. In statistics, *standard deviation* can show how much variation or dispersion there is from the average. The value in our model is 8.4 and 11.0 in Maui. Therefore, we believe that adding new semantic features to Maui makes it perform more stable on each test document, apart from the improvement in accuracy.

4 Related Work

Keyphrases provide semantic metadata that summarize and characterize documents. It is very useful in document indexing, summarization information retrieval, etc. Many work has been devoted to this fundamental task.

In the unsupervised paradigm, the key term extraction problem is framed as a ranking problem. *TF-IDF* [7] is the simplest unsupervised method. It is an effective ranking strategy [27], but it tends to give high score for rare phrases that appear frequently in a document. Grineva, et al. [10] modeled the document as a graph of semantic relationships between terms of that document. They argued that terms related to the main topics of the document tended to bunch up into densely interconnected subgraphs or communities. By leveraging graph community detection techniques and Wikipedia, key terms were selected from important sub graphs in their work. Another unsupervised method named CKE [11] was proposed for indexing scientific documents. They first indexed the references cited in the document and then used the keyphrases assigned to those references for generating a set of high-likelihood keyphrases for the document.

In the supervised paradigm, keyphrase extraction is formulated as a classification problem. Generally, the supervised methods take full advantage of keyphrase features. KEA [8] is the most famous supervised one. In KEA, the candidate terms were represented using two features: *TF-IDF* and *first occurrence*. The classifier was trained using Naive Bayes. It only analyzed simple statistical properties of candidate terms. Maui [9] is a state-of-the-art method of this area. Apart from statistical features such as distance between phrases, frequency and phrase length, Maui utilized more semantic information extracted from Wikipedia, including node degree, inverse Wikipedia linkage, keyphraseness[7], similarity between phrases, etc. But most features only reflect general semantic information of candidates through Wikipedia, cannot capture the semantic relatedness between candidates. There was also a workshop for this area [28]. The F-score of the best-performing system was 27.5%. They concluded that there was still room for improvement.

Since the keyphrase candidates are not independent to each other, we argue that mining their mutual semantic relatedness helps identify keyphrases. In this paper, we exploit the description relation between phrases to improve keyphrase extraction, under the assumption that the description relation could provide additional knowledge and more clues. Experimental results demonstrate that description relation is very helpful to keyphrase extraction.

[7] the number of times a candidate term occurs as a key term in the training documents

5 Conclusion

This paper proposes a novel framework to keyphrase extraction by leveraging the description knowledge. We first extract a large-scale description relations from Wikipedia. Than for each document, we construct a description graph by making use of the description relationships between candidates, and employ graph-based ranking methods to mine more semantic features to improve keyphrase extraction. Extensive experiments are conducted to assess effectiveness of the proposed features. The results show the description relation based features are comparable to the strong feature - *TF-IDF*. The proposed features can also be incorporated in supervised methods. Experimental results on an open dataset demonstrate that incorporating the proposed semantic features in the state-of-the-art supervised method can improve the extraction result significantly.

This work suggests some interesting directions for future work. It would be very interesting to leverage more semantic knowledge to improve keyphrase extraction. For example, the taxonomy knowledge may provide extra topic information about the document. We can further explore different methods (e.g. SimRank) to rank the candidate phrases in the description graph.

Acknowledgments. This work was supported by NSFC (Grand Nos. 61170189, 61370126, 61202239), the Research Fund for the Doctoral Program of Higher Education (Grand No. 20111102130003), the Fund of the State Key Laboratory of Software Development Environment (Grand No. SKLSDE-2013ZX-19), and Microsoft Research Asia Fund (Grand No. FY14-RES-OPP-105). This work was partially supported by the National Key Basic Research Program (973 Program) of China under grant No. 2014CB340403 and the National Natural Science Foundation of China under grant No. M13210007.

References

1. X. Wan and J. Xiao, "Exploiting neighborhood knowledge for single document summarization and keyphrase extraction," *ACM Transactions on Information Systems (TOIS)*, vol. 28, no. 2, p. 8, 2010.
2. Z. Liu, P. Li, Y. Zheng, and M. Sun, "Clustering to find exemplar terms for keyphrase extraction," in *EMNLP*, pp. 257–266, ACL, 2009.
3. E. Frank, G. W. Paynter, I. H. Witten, C. Gutwin, and C. G. Nevill-Manning, "Domain-specific keyphrase extraction," *IJCAI*, pp. 668–673, 1999.
4. S. Jones and M. S. Staveley, "Phrasier: a system for interactive document retrieval using keyphrases," in *SIGIR*, pp. 160–167, ACM, 1999.
5. O. Medelyan and I. H. Witten, "Thesaurus based automatic keyphrase indexing," in *JCDL*, pp. 296–297, ACM, 2006.
6. M. Song, I. Y. Song, R. B. Allen, and Z. Obradovic, "Keyphrase extraction-based query expansion in digital libraries," in *JCDL*, pp. 202–209, ACM, 2006.
7. G. Salton and M. J. McGill, "Introduction to modern information retrieval," 1986.
8. I. H. Witten, G. W. Paynter, E. Frank, C. Gutwin, and C. G. Nevill-Manning, "Kea: Practical automatic keyphrase extraction," in *Conference on Digital libraries*, vol. 3, pp. 147–151, ACM, 1999.

9. O. Medelyan, E. Frank, and I. H. Witten, "Human-competitive tagging using automatic keyphrase extraction," in *EMNLP*, pp. 1318–1327, ACL, 2009.
10. M. Grineva, M. Grinev, and D. Lizorkin, "Extracting key terms from noisy and multitheme documents," in *WWW*, pp. 661–670, ACM, 2009.
11. A. E. Mahdi and A. Joorabchi, "A citation-based approach to automatic topical indexing of scientific literature," *Journal of Information Science*, vol. 36, no. 6, pp. 798–811, 2010.
12. A. Joorabchi and A. E. Mahdi, "Automatic keyphrase annotation of scientific documents using wikipedia and genetic algorithms," *Journal of Information Science*, vol. 39, no. 3, pp. 410–426, 2013.
13. R. Mihalcea and A. Csomai, "Wikify!: linking documents to encyclopedic knowledge," in *CIKM*, pp. 233–242, ACM, 2007.
14. E. Yeh, D. Ramage, C. D. Manning, E. Agirre, and A. Soroa, "Wikiwalk: random walks on wikipedia for semantic relatedness," in *TextGraphs Workshop*, pp. 41–49, ACL, 2009.
15. D. Milne, "Computing semantic relatedness using wikipedia link structure," in *Proceedings of the new zealand computer science research student conference*, Citeseer, 2007.
16. A. Fogarolli, "Word sense disambiguation based on wikipedia link structure," in *ICSC'09*, pp. 77–82, IEEE, 2009.
17. D. Milne and I. H. Witten, "An open-source toolkit for mining wikipedia," *Artificial Intelligence*, vol. 194, pp. 222–239, 2013.
18. C. Huang, Y. Tian, Z. Zhou, C. X. Ling, and T. Huang, "Keyphrase extraction using semantic networks structure analysis," in *ICDM*, pp. 275–284, IEEE, 2006.
19. W. Zhang, W. Feng, and J. Wang, "Integrating semantic relatedness and words' intrinsic features for keyword extraction," in *IJCAI*, pp. 2225–2231, AAAI, 2013.
20. S. Lahiri, S. R. Choudhury, and C. Caragea, "Keyword and keyphrase extraction using centrality measures on collocation networks," *arXiv*, 2014.
21. L. Page, S. Brin, R. Motwani, and T. Winograd, "The pagerank citation ranking: Bringing order to the web.," 1999.
22. G. Jeh and J. Widom, "Simrank: a measure of structural-context similarity," in *KDD*, pp. 538–543, ACM, 2002.
23. S. Wang, S. Xie, X. Zhang, Z. Li, P. S. Yu, and X. Shu, "Future influence ranking of scientific literature," in *SDM*, pp. 749–757, SIAM, 2014.
24. X. Jiang, Y. Hu, and H. Li, "A ranking approach to keyphrase extraction," in *SIGIR*, pp. 756–757, ACM, 2009.
25. O. Medelyan, *Human-competitive automatic topic indexing*. PhD thesis, The University of Waikato, 2009.
26. L. Rolling, "Indexing consistency, quality and efficiency," *Information Processing & Management*, vol. 17, no. 2, pp. 69–76, 1981.
27. K. S. Hasan and V. Ng, "Conundrums in unsupervised keyphrase extraction: making sense of the state-of-the-art," in *ICCL Posters*, pp. 365–373, ACL, 2010.
28. S. N. Kim, O. Medelyan, M.-Y. Kan, and T. Baldwin, "Semeval-2010 task 5: Automatic keyphrase extraction from scientific articles," in *ACL workshop*, pp. 21–26, ACL, 2010.