Keyphrase Extraction Based on Topic Relevance and Term Association

Decong Li\textsuperscript{a}, Sujian Li\textsuperscript{a,*} Wenjie Li\textsuperscript{b}, Congyun Gu\textsuperscript{a}, Yun Li\textsuperscript{c}

\textsuperscript{a}Key Laboratory of Computational Linguistics (Peking University), Ministry of Education Beijing 100871, China
\textsuperscript{b}Department of Computing, The Hong Kong Polytechnic University, Kowloon, Hong Kong
\textsuperscript{c}Institute of Linguistics, Chinese Academy of Social Sciences, China

Abstract

Keyphrases are concise representation of documents and usually are extracted directly from the original text. This paper proposes a novel approach to extract keyphrases. This method proposes two metrics, named topic relevance and term association respectively, for determining whether a term is a keyphrase. Using Wikipedia knowledge and betweenness computation, we compute these two metrics and combine them to extract important phrases from the text. Experimental results show the effectiveness of the proposed approach for keyphrases extraction.

Keywords: Keyphrase Extraction; Topic Relevance; Term association; Betweenness

1 Introduction

With the rapid development of Internet and explosive increase of information, keyphrases as a concise representation of documents can save human a lot of time from reading the whole text. Automatic keyphrase extraction has drawn a lot of attention for a long time. And keyphrases are also important to many other natural language processing tasks such as information retrieval, text categorization, text clustering and so on [7].

Keyphrases, also named as keywords or key terms, are a set of words or phrases that are closely related to the main topic of a document and can express the main content of a document. Generally, a document is expressed from several aspects and keyphrases are expected to describe the main topic, which is termed here as topic relevance. On the other hand, a keyphrase exhibits higher association with other terms in describing a document, and we call it term association. In previous studies, keyphrases are usually extracted from the document with consideration of one of these two factors - topic relevance or term association. In this paper, with the help of
Wikipedia knowledge, we construct a semantic graph, based on which topic relevance and term association are combined to extract keyphrases.

The remainder of this paper is organized as follows. Section 2 briefly introduces the background of semantic graph construction, betweenness based clustering and sorting, and the related work on keyphrases extraction evaluated with topic relevance and term association. Section 3 describes our method, mainly focused on how to consider both topic relevance and term association. Section 4 reports the experimental results. And section 5 concludes our work.

2 Background and Related Work

Currently keyphrases are acquired using various extraction techniques based on features of the phrases and those phrases ranked with highest scores are extracted from the document as keyphrases. These features reflect one of the following two properties directly or indirectly: one is whether the phrase reflects the content of the main topics of the document, and we call it topic relevance. The other considers the relationship between the phrase and other phrases in the document, and we call it term association.

The commonly used method is to extract keyphrases using $\text{TF.IDF}$ [17] and then various variant forms appear with more features added. Keyphrase extraction is converted to a classification task and supervised learning techniques are widely used to tune feature weights for extracting keyphrases. The famous KEA selects several features and trains the classifier using Naïve Bayes model. However, each keyphrase candidate is usually judged independently without considering their relations with other candidates. In addition, supervised learning methods need a large amount of training corpus, which not only requires a lot of human labor but also bears the subjectivity from the person who constructs the corpus.

Graph based methods overcome the problems of term association and training corpus. For these methods, a document is modeled as a graph where vertices represent terms and edges represent the relations between terms. Graph-based algorithms such as PageRank and HITS are used to rank these terms, and high ranked terms are selected as keyphrases [10].

Another kind of approach to extract keyphrases is clustering techniques. Generally, a document can be described from several aspects and keyphrases are expected to express the main topic. Clustering algorithms are implemented to find all the topics and the high evaluated clusters tend to correspond to the main topics of the document. Then all the terms in the main topics are selected as keyphrases [4].

In our work, both term association and topic relevance will be considered for extracting keyphrases. The main process will be introduced respectively as follows.

2.1 Construction of Semantic Graph

Generally a document is modeled as a semantic graph with phrases as vertexes. The edges and their weights are defined differently. Here, we adopt semantic relatedness based on Wikipedia to define an edge’s weight. Wikipedia has become the largest on-line encyclopedia of the world, and kinds of methods computing Wikipedia-based semantic relatedness have appeared. In this paper we use a method similar to that of D. Turdakov and P. Velikhov [14] to calculate the semantic relatedness.
2.2 Clustering

Clustering algorithms group the vertexes of the graph into several communities. Since for our task it is difficult to estimate the number of communities in advance, obviously hierarchical clustering algorithms are more fit. Among various hierarchical clustering algorithms, Girvan-Newman (GN) algorithm [11] is good at finding community structure of the network. Its key concept is edge betweenness, which reflects an edge’s bottleneckness through calculating the sum of the probabilities that the shortest paths (also called geodesics) between all pairs of vertexes go through this edge. The ordinary GN algorithm only treats unweighted graph, therefore lose the weight information which is important for clustering. Newman generalized this algorithm to weighted networks and invented weighted Girvan-Newman (WGN) algorithm [12], which is employed in our work for clustering.

2.3 Ranking

Graph-based algorithms can be used to rank the vertexes of a graph. In [10, 13], PageRank is used to extract keywords. Another method computing vertex betweenness could also be used to rank vertexes. Vertex betweenness introduced by L. C. Freeman [3] has been applied into internet analysis, social network analysis, bioinformatics and other network-related fields successfully. Similar to edge betweenness, it computes the sum of the probabilities that the geodesics between all pairs of vertexes go through a vertex to express the vertex’s influence in the network. In this paper, we use Brandes’ algorithm [1] which runs faster than the traditional ways of computing vertex betweenness, and use vertex betweenness as an important factor to choosing keyphrases.

3 Our Method

Our work mainly focuses on how to assure the extracted keyphrases are high in both topic relevance and term association, including the following steps: Firstly, construct a semantic graph for the document; Secondly, compute term association and cluster the terms for determining the topic relevance; Finally, assign those terms that are high in both topic relevance and term association as keyphrases.

To describe our work more clearly, we use a news article “Microsoft’s Sales Tumble on PC Weakness” as an instance document, which tells that Microsoft’s revenues has declined amid falling global demand in 2009.

3.1 Semantic Graph Construction

We firstly model the document as a weighted graph $G = (V, E, W)$, where $V$ represents a set of terms selected from the document, and $E$ and $W$ represent the semantic relatedness among terms in $V$.

We use Wikipedia to select terms as keyphrase candidates. If several adjacent words make up a phrase that appears in Wikipedia, we select the phrase to compose of the graph. For any two terms $v_i$ and $v_j$, their semantic relatedness is defined as $w(v_i, v_j)$, which is computed as in [14],

$$w(v_i, v_j) = \frac{x_i \cdot x_j}{|x_i|^2 + |x_j|^2}$$ (1)
Here terms $v_i$ and $v_j$ are represented respectively by two vectors $x_i$ and $x_j$, which are from the corresponding Wikipedia articles. Each dimension of a vector means a link in the Wikipedia articles, and its value is binary. If a link exists, the value is set 1, otherwise 0.

### 3.2 Topic Relevance and Term Association

Generally, a document is described by many topics $t_1, t_2, \cdots, t_n$, among which only a few topics $t'_1, t'_2, \cdots, t'_m (m < n)$ are closely related to the main content of the document, named main topics. Each topic is usually represented by a set of terms. Then it is important to discriminate whether a term belongs to one of those main topics.

The first step is to cluster all terms into $n$ topics. We make the following hypothesis: if a set of terms in a document relate to each other closely, they belong to a common topic. Based on the semantic graph $G$, we adopt WGN algorithm [12]. It iteratively divides the graph by finding the edge which has the highest edge betweenness. Let $\sigma_{ij}(e)$ be the number of geodesics between vertex $v_i$ and $v_j$ passing $e$, $\sigma_{ij}$ be the number of geodesics between $v_i$ and $v_j$, and $w$ is the edge weight, then the edge betweenness $B_E(e)$ is defined as

$$ B_E(e) = \frac{1}{w} \sum_{v_i \neq v_j \in V} \frac{\sigma_{ij}(e)}{\sigma_{ij}} $$

(2)

This algorithm utilizes modularity $Q$ to choose the best division of graph $G$. Here $Q$ is defined as follows

$$ Q = \frac{1}{2m} \sum_{ij} \left[ W_{ij} - \frac{k_i k_j}{2m} \right] \delta(c_i, c_j) $$

(3)

where the $\sigma$-function $\sigma(u, v)$ is 1 if $u = v$ and 0 otherwise, $W_{ij}$ represents the weight of the edge between vertex $v_i$ and $v_j$, $m = \frac{1}{2} \sum_{ij} W_{ij}$, $k_i = \sum_j W_{ij}$ and $c_i$ is the community to which $v_i$ is assigned.

The next step after clustering is, to determine the main topics. We find that in most cases, the terms in a document’s title concisely represent all the main topics of the document. Then we label the topics which includes terms in a title as main topics.

Table 1 illustrates the topic relevance of each term for the example document, where 1 means the term is in the main topic, and the title terms are printed with bold italic type.

Then we continue to select appropriate keyphrase in the main topic. Two factors are consid-

<table>
<thead>
<tr>
<th>topicID</th>
<th>Term</th>
<th>Topic Relevance</th>
<th>topicID</th>
<th>Term</th>
<th>Topic Relevance</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Customer</td>
<td>0</td>
<td>24</td>
<td>Fuel</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>Improve</td>
<td>0</td>
<td>24</td>
<td>Technology</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>\ldots</td>
<td>0</td>
<td>25</td>
<td>Apple</td>
<td>1</td>
</tr>
<tr>
<td>23</td>
<td>Business operations</td>
<td>1</td>
<td>25</td>
<td>Network</td>
<td>1</td>
</tr>
<tr>
<td>23</td>
<td>\ldots</td>
<td>1</td>
<td>25</td>
<td>Software</td>
<td>1</td>
</tr>
<tr>
<td>23</td>
<td>Sales</td>
<td>1</td>
<td>25</td>
<td>\ldots</td>
<td>1</td>
</tr>
<tr>
<td>24</td>
<td>Fuel</td>
<td>0</td>
<td>25</td>
<td>Microsoft</td>
<td>1</td>
</tr>
<tr>
<td>24</td>
<td>Technology</td>
<td>0</td>
<td>25</td>
<td>Personal computer</td>
<td>1</td>
</tr>
</tbody>
</table>
First, it tends to associate with many other terms in the document, which is measured by vertex betweenness \[3\]. Second, it usually appear frequently in the document. Let \(v\) be a vertex in graph \(G\), \(\sigma_{ij}(v)\) be the number of geodesics between vertex \(v_i\) and \(v_j\) passing \(v\), and \(\sigma_{ij}\) as before, then vertex betweenness \(B_V(v)\) is given by

\[
B_V(v) = \sum_{v_i \neq v \neq v_j \in V} \frac{\sigma_{ij}(v)}{\sigma_{ij}}
\]

(4)

In our work, we adopt Brandes’ algorithm which can computes vertex betweenness efficiently \[1\].

Finally we define term association \(R_{term}(v_i)\) of term \(v_i\) as the following formula.

\[
R_{term}(v_i) = B_V(v_i) \times TF(v_i)
\]

(5)

### 3.3 Keyphrases Extraction

With consideration of both topic relevance and term association, we select keyphrases. We compute a score \(S(v_i)\) for a term \(v_i\) with topic relevance \(R_{topic}(v_i)\) and term association \(R_{term}(v_i)\):

\[
S(v_i) = R_{topic}(v_i) \times R_{term}(v_i)
\]

(6)

An important observation from the example document is that, in the curve which sorts all the terms in descending order according to the term scores, there is an obvious cutoff point that separates the curve into two sections. The front section of the curve has high values and then declines sharply with low values. The curve for our example document is shown in Figure 1. Here the cutoff is “Sales”.

![Score curve for the example document (non-zero only)](image)

Fig. 1: Score curve for the example document (non-zero only)

### 4 Experiments

In this section, we prove some propositions mentioned above through experiments and evaluate the results of our method. We first try to get the cutoff point mentioned in Section 3.3 precisely.
Then we compare our results with some other approaches. To construct the test corpus, we selected 200 English news articles from some well-known English media. 10 graduate students participated in our work to label the keyphrases manually. In Wikipedia, all terms expressing the same concept are redirected to one Wikipedia article, whose title is called norm term here. For evaluation, we transform all the automatically and manually extracted keyphrases into norm terms. Here we adopt traditional measures to evaluate our methods: precision, recall and F-measure.

4.1 Cutoff between Keyphrases and Non-Keyphrases

As mentioned in Section 3.3, the cutoff point can be obviously found in the score curve to separate keyphrase and non-keyphrases. Through this observation, we utilize the difference between two adjacent scores to get the cutoff. For a document, let \( v_1, v_2, \ldots, v_m \) be a sequence of terms having non-zero scores, with their scores \( s_1, s_2, \ldots, s_m \) in descending order, and \( d_i \) be the difference between two adjacent scores (namely \( d_i = s_i - s_{i+1} \)). We propose a parameter \( \alpha \) which represents a proportion of \( m \), and check the difference \( d_i \) starting at the top \( \alpha \)-position of the sequence of sorted terms. Whenever \( d_i > d_{i+1} \), \( t_{i+1} \) is the cutoff.

Generally, the cutoff point is located between top 10\% and 30\% of the curve. We set \( \alpha \) as 10\%, 15\%, 20\%, 25\% and 30\% respectively. For each \( \alpha \), we get the corresponding cutoffs and evaluate the result. With \( \alpha \) as 20\% we get the highest F-measure 43.4\%.

4.2 Evaluation of Extracted Keyphrases

In this section we compare our approach with several existing approaches, including KEA, a supervised learning approach, and TextRank [10] and Community-based [4], two graph-based unsupervised approaches.

The performance of the four approaches is shown in Table 2, where KEA gives the worst result, TextRank works with similar performance to community-based method. Our method performs a little better than the community-based method which gets the second place in our evaluation.

<table>
<thead>
<tr>
<th>Approach</th>
<th>Precision</th>
<th>Recall</th>
<th>F-measure</th>
</tr>
</thead>
<tbody>
<tr>
<td>KEA</td>
<td>0.284</td>
<td>0.415</td>
<td>0.337</td>
</tr>
<tr>
<td>TextRank</td>
<td>0.328</td>
<td>0.524</td>
<td>0.403</td>
</tr>
<tr>
<td>Community-based</td>
<td>0.317</td>
<td>0.561</td>
<td>0.405</td>
</tr>
<tr>
<td>Ours</td>
<td>0.364</td>
<td>0.537</td>
<td>0.434</td>
</tr>
</tbody>
</table>

5 Conclusion

In this paper we introduce a novel approach to extract keyphrases. This approach calculates topic relevance and term association respectively with the help of betweenness computation, and utilizes these two metrics to extract keyphrases more precisely. Experimental results show that compared with other keyphrases extraction approaches, our approach performs better.
References