Describing Web Topics Meticulously through Word Graph Analysis

Bai Sun, Lei Shi, Liang Kong, Yan Zhang
Department of Machine Intelligence
Peking University, 100871 Beijing, China
{sunbai, shilei, kl, zhy}@cis.pku.edu.cn

Abstract

Topic description is as important as topic detection. In this paper, we propose a novel method to describe Web topics with topic words. Under the assumption that representative words exist in important sentences and have high probability of occurrence with other representative words, two graphs are built, one of which represents the relationship for sentences, the other for words. Considering a topic cluster contains a set of different Web pages, sentence clusters are also introduced. Experimental results on a real data set show that our method achieves excellent performance in both high precision and efficiency, especially when real Web data contain mass of noises.

1. Introduction

Detecting and describing Web hot topics among mass of Web data is an effective way to provide Web users with more accurate and meaningful information. However, this task is very difficult. In this paper, we focus on how to describe Web topics more accurately and effectively.

As a topic cluster contains a set of Web pages, multiple documents summarization can give every topic a description. However, as mentioned in [4][18], these kinds of methods are not perfect enough and the results are often so poor that they can not be applied in topic description. Web users may feel confused with the results given by these methods, especially when real Web data contain mass of noises.

Some methods, such as method based on term frequency, will also fail in topic description because of the heterogeneity of Web data[22]. As mentioned in[12], Web pages main content extraction is a difficult task. In spite of many new methods[17][3] are employed in this area, the results are not satisfactory and there are still lots of noises. This disadvantage can lead to terrible influences on methods such as keywords extraction based on term frequency. As these methods are easily distorted by noises from real Web data, their results are usually poor.

There are many Web pages in a topic cluster. These Web pages contain similar content and discuss the same topic. We propose a method to find keywords, which are called topic words in this paper. It is very significant to describe a topic by topic words . They can provide ample and meticulous information for every topic. Web users can get enough information from topic words without reading the whole topic, so they are effective and valuable.

In this paper, topic words are extracted for every topic based on graph mining, specifically, based on two graphs, sentence graph and word graph. Sentence graph is built according to the relationship between each sentence pair. Word graph is built according to sentence graph and sentence clusters. These two graphs reflect both the external and latent relationship among every word, and this is the reason that our method can achieve excellent performance in both high precision and efficiency.

Our method is especially powerful for Web topic description. The sentence selecting method based on graph together with sentence clustering can make better use of the information of multiple Web pages and reduce the influence of noises in texts. On the contrary, methods with low immunity to noise will be damaged severely.

The rest of the paper is organized as follows. In Section 2, we review related work. The detailed process of our method for topic description is given in Section 3. Section 4 shows the performance of our method. Finally we draw the conclusion in Section 5.

2. Related Work

Topic detection of text has become a mature research area in information retrieval. Topic detection and tracking (TDT) is proposed by James Allan in 1998 [2]. Afterwards much progress has been obtained in topic detection of news broadcast. In [21], the researchers adapted several IR and machine learning techniques for effective event detection and tracking. In some research, the application of probabilistic
models to the topic detection and tracking tasks is developed [8]. Those algorithms also have an excellent effect on topic detection from Web documents, etc.

Topic description has some relations with document summarization. Document summarization has been widely explored in natural language processing and information retrieval. Extraction-based methods and abstraction-based methods [5],[14] are the main methods in this area. Recently, the graph-ranking based methods, TextRank [6] as an example, have been proposed for document summarization. These methods use graph to rank sentences and suggest the top ones. PageRank [11],[16] and HITS[10] are both famous ranking algorithms. These algorithms use link analysis for ranking based on graph. Graph is used and acts well in many other aspects besides ranking. In[15], a novel method based on graph is used to analyze query logs. In[19], a method based on graph mining is used to discover topics about scientific papers, however it is crippled because of low efficiency. In[18], the researchers use a new way based on graph for multiple documents summarization. The method of using graphs is employed in our paper, for both sentence ranking and word ranking.

3. Finding Topic Words Based on Graph Analysis

3.1. Overview

The framework of Web topic analysis is shown in Figure 1. In this paper we focus on topic description which is demonstrated in the second part of Figure 1.

![Figure 1. Topic Detection and Description](image)

As we know, producing a description for every topic is preferred for it can provide Web users with more specific and accurate information about a particular topic. In this paper, a set of words which are called topic words is used as the description for a topic. A topic cluster generally contains a set of web pages, which focus on the same object or event, but unfortunately, the content of these pages are always mixed up with mass of noises and there is no effective method to remove these noises. In this paper we use graphs to describe the relationships between sentences and words.

By analyzing the relationships, topic words for each topic are finally extracted.

As shown in Figure 2, important sentences can be selected by step 1. We analyse the relationship between each sentence pair and build a graph for all the sentences. All the sentences belonging to a particular topic can be ranked according to this graph. Step 2 makes an improvement for step 1. Different from dealing with just one single document, the object for topic description is a set of relevant documents. In step 2, clustering analysis is adopted to partition all the sentences from different documents into some clusters, in which one consists of several similar sentences. These clusters can enrich the link structure of word graph. After these two steps, a word graph can be built and hence all the words are ranked according to this word graph. At last, the top-ten words are selected as topic words for every topic.

![Figure 2. Topic Description Algorithm](image)

Our algorithm is based on the assumptions as follows:

1) Representative words have high probability to exist in important sentences.
2) Representative words have high probability of occurrence with other representative words.
3) Sentences and words can be ranked according to their link structure.

3.2. Important Sentences Selecting

The first step of topic description based on topic words is selecting important sentences. As we focus on describing a topic with representative topic words, assuming that these words exist in important sentences, it is necessary to choose the important sentences from the whole sentences collection of a topic’s multiple documents. Meanwhile, important sentences selecting can also reduce the influence of texts with low quality, because noises often exist in unimportance sentences. After important sentences are selected, topic words will be extracted from them.
Given a sentence collection \( S = \{s_i| 0 \leq s_i \leq n\} \), \( S \) contains all the sentences for a topic. As we are dealing with Chinese Web pages, some symbols, such as “!” , “?” , “.” are recognized as end of a sentence. The purpose of our method is extracting topic words, so Chinese segmentation is obligatory. ICTCLAS [7] is a mature Chinese segmentation tool, which is employed in our method.

In this paper, a sentence graph \( GS \) is used to decide important sentences. \( GS \) is a directed graph, in which each sentence is a vertex.

**Definition 1** The affinity weight \( IF(s_i, s_j) \) from \( s_i \) to \( s_j \) is defined as follows:

\[
IF(s_i, s_j) = \frac{MaxCo(s_i, s_j)}{Length(s_j)} \tag{1}
\]

where \( IF(s_i, s_j) \) means the weight of the edge from \( s_i \) to \( s_j \) in graph \( GS \). \( MaxCo(s_i, s_j) \) means the length of the common substring between \( s_i \) and \( s_j \). \( Length(s_j) \) means the length of sentence \( s_j \). As \( GS \) is a directed graph, the edge from \( s_j \) to \( s_i \) can be computed according to \( IF(s_j, s_i) \) with the denominator \( Length(s_i) \).

An adjacency (affinity) matrix \( MS' \) is used to describe the graph with each entry corresponding to the weight of an edge in the graph. \( MS' = (MS'_{i,j})_{n \times n} \) is defined as follows:

\[
MS'_{i,j} = \begin{cases} 
IF(s_i, s_j) & \text{if } i \neq j, \\
0 & \text{if } i = j.
\end{cases} \tag{2}
\]

\( MS \) is a normalized matrix of \( MS' \) in which the sum of every row equals to 1 by:

\[
MS_{i,j} = \begin{cases} 
\frac{MS'_{i,j}}{\sum_{j=1}^{n} MS'_{i,j}} & \text{if } \sum_{j=1}^{n} MS'_{i,j} \neq 0, \\
0 & \text{else}.
\end{cases} \tag{3}
\]

Based on the graph \( GS \) and matrix \( MS \), a score for every sentence \( s_i \) can be deduced from all the other sentences linked with \( s_i \) and this process can be formulated in a recursive form [11][16] as follows:

\[
Score(s_i) = d \times \sum_{j \neq i} Score(s_j) \times MS_{j,i} + \frac{1 - d}{n} \tag{4}
\]

where \( n \) is the total number of sentences, \( d \) is a damping factor which is usually set to 0.85.

The initial \( Score(s_i) \) for every \( i \) is set to 1. By an iterative algorithm according to the equation above, the new score for every sentence is computed iteratively until the computation is converged. After the computation, every \( Score(s_i) \) represents the importance of \( s_i \), which will be used to establish a word graph.

3.3. Topic Words Extraction by Word Graph

Word graph \( GW \) is undirected, with each distinct word belonging to a topic as its vertex. The key point is the method to add edge between two vertexes. In this paper edges are added to \( GW \) according to sentence scores \( Score(s_i) \) and sentence clusters. Sentence clusters will be introduced in the next part as an improvement.

**Definition 2** \( Support(i, j) \) is the supportiveness between \( word_i \) and \( word_j \), which is defined as:

\[
support(i, j) = \sum_{i \in P, j \in P} Score(s_p) \tag{5}
\]

where \( i, j \) represent \( word_i \), \( word_j \) and \( P \) represents sentence \( s_p \) respectively. The meaning of \( Support(i, j) \) is as follows:

1. Assuming that \( word_i \) is a representative word, if \( word_j \) has many co-occurrences with \( word_i \), \( word_j \) has a high probability to be a representative word, too.
2. Assuming that sentence \( s_i \) is an important sentence, if \( word_j \) has many occurrences in \( s_i \), \( word_j \) has a high probability to be a representative word.

Based on the two assumptions above, \( Support(i, j) \) is employed to represent the weight of the edge between \( word_i \) and \( word_j \) in graph \( GW \). An adjacency (affinity) matrix \( MW \) is used to describe a graph with each entry corresponding to the weight of an edge in the graph \( GW \). \( MW = (MW_{i,j})_{m \times m} \) is defined as follows:

\[
MW_{i,j} = \begin{cases} 
Support(i, j) & \text{if } i \neq j, \\
0 & \text{if } i = j.
\end{cases} \tag{6}
\]

Each initial \( MW_{i,j} \) is set to 0, which means that there is no edge between \( word_i \) and \( word_j \). If there is a co-occurrence of \( word_i \) and \( word_j \) in sentence \( s_p \), \( Score(s_p) \) is added to \( Support(i, j) \) to describe the edge between \( word_i \) and \( word_j \).

Matrix \( MW \) can be used to rank all the words of a topic with a similar method as employed in sentence ranking.

3.4. Sentence Clustering

Unlike the single document summarization which only needs to use the relationship between sentences in only one document, topic description must consider the relationship between sentences across documents. Considering two different sentences from different documents, \( s_p \) and \( s_q \), \( s_p \) has a structure as “ A is a B ”, and \( s_q \) has a similar structure as “ A is a C ”. A, B and C are three different words. Sentence score \( Score(s_i) \) can only provide the affiliation between word A and B , word A and C, which makes the affiliation between word B and C missed. There should be
an edge between word B and C in GW, either. So sentence clustering is employed as an improvement for using sentence score only.

Famous clustering methods include hierarchical clustering algorithms, nearest neighbor clustering, artificial neural networks, fuzzy clustering and so on, all of which are used widely[1]. Using these methods, all sentences are divided into different clusters. Sentences in cluster \( c_i \) are correlative. As what has been done in sentence ranking, each cluster \( c_i \) has a score \( CScore(c_i) \) to indicate its importance. Time complexity is not considered for this task can be finished very soon.

**Definition 3** \( CScore(c_i) \) is the importance of cluster \( c_i \) defined as:

\[
CScore(c_i) = \sum_{s_p \in c_i} Score(s_p)
\]  

(7)

in which \( CScore(c_i) \) is from \( Score(s_p) \). With \( CScore(c_i) \), the \( Support(i, j) \) in equation 5 can be updated as:

\[
Support(i, j) = (1 - \lambda) \sum_{i,j \in p} Score(s_p) + \lambda \sum_{i,j \in q} CScore(c_q)
\]  

(8)

where \( \lambda \) is a constant which describes the proportion when the two parts are combined. Experiments will show the influence of \( \lambda \) to the final result.

In this paper, K-means clustering algorithm [13] is employed. We compare several clustering methods, and find the final result is not sensitive to the clustering results when the number of the clusters is large to a certain degree, such as what we choose as three times than the average number of sentences per document.

This can be explained as follows. The clusters are used as an improvement of using sentences only, so the number of the clusters determines to what degree the \( Support(i, j) \) is improved. If this number is large, which means the clusters may be small and it is strict to improve \( Support(i, j) \) only in a small range. If this number is small, which means the clusters may be large, it is relatively not very strict and \( Support(i, j) \) can be improved in a large range. The second case may bring in error because the clusters are too large to be accurate, some wrong supports may be added in \( Support(i, j) \). Finally, this number of clusters is set N times as the average number of sentences per document. In this paper, N is 3.

This new \( Support(i, j) \), different from the old one, combines the sentence scores with the cluster scores by \( \lambda \). The significance of this new \( Support(i, j) \) is, it provides hidden and more abundant connections between words by automatical clustering instead of using sentences which provides only external connections. The new \( Support(i, j) \) can strengthen the links between important sentences, and weaken the links between noises and other sentences.

In the undirected graph \( GW \) with every word as its vertex, updated \( support(i, j) \) is finally used to describe every edge in this graph. With the updated \( support(i, j) \), matrix \( MW \), of which each row is normalized to 1 as \( MW_{i,j} = \frac{WM_{i,j}}{\sum_{j=1}^{n} WM_{i,j}} \), is used to rank all the words using a recursive algorithm and every word \( w_i \) is given a score \( WScore(w_i) \) as:

\[
WScore(w_i) = d \times \sum_{j \neq i} WScore(w_j) \times MW_{j,i} + \frac{1 - d}{n}
\]  

(9)

This algorithm is the same as Equation 4. Finally, the top ten representative words are selected as topic words.

Because of the disadvantage of Chinese segmentation, some Chinese words are cut into individual characters. To avoid this drawback, a greedy filtration algorithm is employed. However, this filtration is not the focus of this paper.

### 4 Experiments

#### 4.1 Data Set and Baselines

There is no mature data set for our method of topic description on Chinese Web pages, so our experiments are conducted on a data set crawled from the real Web. This collection contains about 6GB data from News Web Sites, Forums and BBS( for BBS and Forums we only crawled the “Top ten Hot Topics for Today” these Web sites provided and for News Sites we crawled all the news of that day), with an amount of 211,385 Web pages for a whole month. As Web pages’ main content extraction is necessary, the method in [12] is employed in our experiment.

To evaluate our topic description method, we take two previous methods as baselines, TF and KeyGraph. TF uses term frequency as the ranking factor. KeyGraph[20] is a famous method used in many area for keyword extraction, which combines both term frequency and term importance.

#### 4.2 Evaluation

In our study, we use \( P(\text{precision}) \), \( R(\text{recall}) \) and \( F(\text{measure}) \) in Equation 10 as evaluation metrics. Ten students worked as volunteers and gave a ranked list of twenty words for each topic. For each topic, we use their rankings to select the top fifteen words and consider these fifteen words in collection \( T \) are true and can describe the topic correctly. Top thirty words selected by our method, TF and KeyGraph are compared with these fifteen words.

\[
\begin{align*}
Precision & = \frac{t_p}{t_p + f_p}, \\
Recall & = \frac{t_p}{t_p + f_n}, \\
F-Measure & = \frac{2 \times Precision \times Recall}{Precision + Recall}
\end{align*}
\]  

(10)

where \( t_p \) is the ratio of retrieved value to relevant value, \( f_p \) is the ratio of retrieved value to irrelevant value and \( f_n \) is the ratio of un-retrieved value to relevant value.
Considering that the sequence for the words is important, Spearman rank correlation coefficient is used. Spearman rank correlation coefficient[9] is defined as:

\[
\rho = 1 - \frac{6 \times \sum d_i^2}{n^3 - n},
\]

(11)

where \(n\) means the number of words, \(d_i\) means the distance between automatic result and true result.

### 4.3 Topic Description Results

We randomly select three topics and our experimental results are based on these three topics. Topic 1 is about LiuXiang Out of the Race. Topic 2 is about SanLu’s Quality Issues in Milk Powder. Topic 3 is about ChenShuibian in Jail. The three topics’ detailed information is in Table 1.

**Table 1. Three Topics**

<table>
<thead>
<tr>
<th>Data set</th>
<th>Web pages</th>
<th>Sentences</th>
<th>Words</th>
</tr>
</thead>
<tbody>
<tr>
<td>Topic 1</td>
<td>516</td>
<td>7512</td>
<td>7675</td>
</tr>
<tr>
<td>Topic 2</td>
<td>491</td>
<td>7632</td>
<td>8381</td>
</tr>
<tr>
<td>Topic 3</td>
<td>494</td>
<td>8282</td>
<td>8796</td>
</tr>
</tbody>
</table>

In Figure 3, results for Topic 1 of our method, TF and KeyGraph are illustrated. For each method, the top ten words are shown as ten is considered to be enough to represent that topic. We can see that TF and KeyGraph select some words which are not recommended by our method, such as word “Hubei Province”, “Companies”, and they are not in collection \(T\). Our method is different from TF, frequent and unrepresentative words are not considered because they seldom exist in important sentences although with high frequency.

**Figure 3. Example: Description for Topic 1**

For the three topics above, the accuracy of each method is shown in Table 2, including \(P\), \(R\), \(F\) and \(\rho\). As mentioned above, single Chinese characters should be removed, although they enrich the link structure. Figure 4 shows the result for Topic 1 both before and after single Chinese characters filtration.

### 4.4 The Influence of \(\lambda\)

\(\lambda\) acts on the combination of \(Score(s_k)\) and \(CScore(s_k)\), which determines the proportion of \(CScore(s_k)\) added into \(Support(i, j)\). Figure 5 shows \(\lambda\)’s influence to \(\rho\).

From Figure 5, \(\rho\) has positive correlation with \(\lambda\) in an approximate range of \([0.1, 0.5]\). With the increase of \(\lambda\) in this range, \(\rho\) is getting better which means \(CScore(s_k)\) plays a more important role than \(Score(s_k)\) in \(Support(i, j)\), while with the increase of \(\lambda\) in this range, \(\rho\) is getting

**Figure 4. P R and F for Topic 1**

**Figure 5. \(\lambda\) and \(\rho\)**

**Table 2. Comparison to Baselines**

<table>
<thead>
<tr>
<th></th>
<th>P</th>
<th>R</th>
<th>F</th>
<th>(\rho)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Topic 1</td>
<td>0.646</td>
<td>0.317</td>
<td>0.376</td>
<td>0.884</td>
</tr>
<tr>
<td></td>
<td>TF</td>
<td>0.561</td>
<td>0.281</td>
<td>0.329</td>
</tr>
<tr>
<td></td>
<td>KG</td>
<td>0.516</td>
<td>0.276</td>
<td>0.325</td>
</tr>
<tr>
<td>Topic 2</td>
<td>0.784</td>
<td>0.325</td>
<td>0.407</td>
<td>0.859</td>
</tr>
<tr>
<td></td>
<td>TF</td>
<td>0.690</td>
<td>0.359</td>
<td>0.395</td>
</tr>
<tr>
<td></td>
<td>KG</td>
<td>0.466</td>
<td>0.267</td>
<td>0.354</td>
</tr>
<tr>
<td>Topic 3</td>
<td>0.507</td>
<td>0.256</td>
<td>0.335</td>
<td>0.859</td>
</tr>
<tr>
<td></td>
<td>TF</td>
<td>0.472</td>
<td>0.244</td>
<td>0.331</td>
</tr>
<tr>
<td></td>
<td>KG</td>
<td>0.410</td>
<td>0.228</td>
<td>0.318</td>
</tr>
</tbody>
</table>
worse which means $Score(s_k)$ play more important part than $CScore(s_k)$ in $Support(i, j)$. Topic 2 is stable because the result of this topic is perfect so that $\rho$ changes little with $\lambda$. We finally set $\lambda$ to 0.25. This value is considered to be able to provide more meaningful hidden links between words, meanwhile less errors.

5 Conclusion

Our method based on graph combines sentence scores and sentence cluster scores, which is more adaptive for Web topic description than term frequency and KeyGraph. Experimental results show that our method performs well with both high precision and efficiency, especially when real Web data contain mass of noises. Methods based on term frequency would suggest topic words which have high occurrences, however, these words have a high probability to be wrong because the real Web data contain mass of noises. This indicates that our method is more appropriate for Web topic description. Our method can avoid some negative influence from term frequency. Sentence clusters are employed to discover hidden links between words and can enrich the link structure. Experiments show that this improvement is effective.

References