A Method of Hot Topic Detection in Blogs Using *N*-gram Model

Xiaodong Wang
College of Computer and Information Technology, Henan Normal University, Xinxiang, China
Email: wangxiaodong.wang@yahho.com.cn

Juan Wang
College of Computer and Information Technology, Henan Normal University, Xinxiang, China
Email: juaner.510@163.com

Abstract— **Over the last few years, blogs (web logs) have gained massive popularity and have become one of the most important web social media, through which people can get and release information. Hot topic detection in blogs is most commonly used in analyzing network public opinion. A method of hot topic detection using *n*-gram model and hotness of topic evaluation is proposed in this paper. Our approach consists of three steps. First of all, keywords during a given time period are obtained by means of calculating word's weight, and hot keywords are collected by combining keywords. Secondly, based on hot keywords, hot keyword groups are extracted using *n*-gram model. In the third step, hot keyword groups are extracted and hot topics are detected. The hotness of hot topic is evaluated by the value of keywords' weight, which is got in the second step. Evaluations on Chinese corpus show that when the size of *n* for *n*-gram is five, the proposed method is most effective.**

Index Terms—***n*-gram model; blog; hot keyword; hot keyword group; hot topic**

I. Introduction

With the development of WEB 2.0, blogs have gained massive popularity and have become one of the most influential web social media in our times. Anyone with an internet connection can conveniently publish topics. According to a research [1], there are over 175,000 new blogs are created per day by people all over the world, on a great variety of subjects. The huge growth of blogs provides a wealth of information waiting to be extracted. Blogs are becoming an extremely relevant resource for different kinds of studies focused on many useful applications. Accordingly, blogs offer a rich opportunity for detecting hot topics that may not be covered in traditional newswire text. Unlike news reports, blog article expresses a wide range of topics, opinions, vocabulary and writing style. The change in editorial requirements allows blog authors to comment freely on local, national and international issues, while still expressing their personal sentiment. These forms of self published media might also allow topic detection systems to identify developing topics before official news reports can be written.

Most previous approaches in hot topic detection are based on cluster technologies. Li and Wu [2] used an improved algorithm based on K-means clustering method and support vector machine to group the forums into various clusters. Hao and Hu [3] proposed single-pass clustering method to detect topic oriented to BBS (Bulletin Board Systems). Dai [4] and Wang [5] proposed hierarchical clustering method. Zheng and Fang [6] used clustering method and aging theory to detect hot topic on BBS.

Another trend of research is to use Natural Language Processing technology. As two representative methods, Chinese Word Segmentation Technology (CWST) and Named Entity Recognition [7] are utilized. Zhu and Wu [8] used CWST to dig out hot topics based on combination of multiple keywords. Chen [9] presented a noise-filtered model to extract the outburst topics from web forums using terms and participations of users. K. Chen and L. Luesukprasert [10] extracted hot terms by mapping their distribution over time, and identified key sentences through hot terms, then used multidimensional sentence vectors to group key sentences into clusters that represented hot topics. M. Platakis, D. Kotsakos and D. Gunopulos proposed a hot topic detection method during a time interval which was based on bursty discovery. Yadong Zhou, Xiaohong Guan and et al. [11] utilized statistics and correlation of popular words in network traffic content to extract popular topics on the Internet.

Two or three keywords can generally express a topic [12]. Merging multiple keywords [13] can reflect the hot news events in a certain time. Zhou et al. [14] constructed a keyword network based on word frequency and co-occurrences to detect hot topic on professional blogs. Unfortunately, the frequencies of terms describing the same hot topic information are different. If the threshold value is not high enough, some impact terms will be filtered. As a result, the detected keywords will not effectively represent the hot topic information.

How to evaluate the hotness of topic is one of the important problems in hot topic detection. Jianjiang Li, Xuechun Zhang, and et al. [15] evaluated the blog hotness based on text opinion analysis. They took the number of reviews, comments and publication time of the

doi:10.4304/jsw.8.1.184-191

blog topic, and the comment opinion into account. Lan You, Yongping Du and et al. [16] evaluated the hotness of the topic through its popularity, quality and message distribution using Back-Propagation Neural Network based on classification algorithm. Tingting He, Guozhong Qu and et al. [17] presented a semi-automatic hot event detection approach. They evaluated the activity of events firstly, then filtered and sorted the events according to the activity of events, and finally got hot event.

These methods above could detect hot topic and evaluate hotness of the topic effectively. However, blog has some particular structure features itself. These methods may not be completely suitable for hot topic detection in blogs. In this paper, employing CWST and taking both the several words in the sentence which locate left and right of the keywords into account, we propose a method to detect hot topic in blogs based on *n*-gram model. Firstly, the hot keywords are extracted based on the co-occurrence information in *n*-gram items which provide more information about hot topic. Then, hot keywords groups are collected by calculating the similarity of *n*-gram items containing keywords. Finally, the hot topics are detected by computing the similarity between keywords and *n*-gram items to decide whether the keyword group represents a hot topic. The hotness of the topic is evaluated by the value of keywords’ weight.

This paper is organized as follows: Section 2 introduces *n*-gram model and gives some concepts. Section 3 presents our approach for hot topic detection and hotness of the topic evaluation in blogs. Section 4 presents the experimental results on Chinese corpus. Finally, we summarize the future work.

II. N-Gram Model And Related Concepts

In the fields of computational linguistics and probability, an *n*-gram is a contiguous sequence of *n* words from a given sequence of text or speech. *N*-gram models are widely used in statistical natural language processing, and approximate matching. Statistical *N*-gram models capturing patterns of local co-occurrence of contiguous words in sentences have been used in various hybrid implementations of Natural Language Processing and Machine Translation systems [18-22]. In this paper, one contiguous sequence of *n* words is referred to as an *n*-gram item, and one *n*-gram item has *n* entries. We use *n*-gram model to present sentence that contains keyword. Words that relate with the keywords can be found in the *n*-gram items, so the detailed information of topic that keywords depict can be detected. The keywords depicting one topic can be detected through calculating the similarity of *n*-gram items that contain keywords.

A blog article consists of three parts: the title, the content and the reply. For a sentence in blog article, first, we define a stopping-word list to remove the words that are irrelevant to the theme of the blog article. Then we do word segmentation and Part-of-Speech tagging. Finally, this sentence is represented by *n*-gram model. An example of five-gram model is given in Fig.1. Every five-gram item has five entries, all of which in an item have some co-occurrence information. Every sentence has several five-gram items.

In process of combining five-gram items, if the number of the entries in the last five-gram item is smaller than five, then we use stop words to fill. The *n*-grams can effectively capture the relationship between words, which have co-occurrence information of keywords.

III. Hot Topic Detection Method In Blogs

Two or three keywords can generally express a topic, but they can't provide the detailed topic information, such as time, place and related people. A topic in blogs is a cluster that is composed of a number of blog articles sharing theme. Each blog article has one or more keywords to depict the theme. For a blog article, in the sentence containing keywords, words that are not far away from the keywords generally provide the topic information. So for the sake of effectiveness, we use *n*-gram model to present sentences, set a distance window of *n* entries, and take the *n*-gram items that containing keywords into account.

The hot topic detection method in blogs is composed of three parts, they are hot keyword extraction method, hot keyword group extraction method and hot topic detection algorithm. In Fig.2, the general process of the method is given. The first step is to get blog articles, and then do preprocessing and word segmentation. Next, extract keyword by calculating the word’s weight, represent the sentence containing keywords with *n*-gram model, and discover the frequency of every *k*-grams where $1 \leqslant k \leqslant n$ by scanning the training dataset and estimating each precision and recall. In the third step, we build the hot keyword extraction algorithm, hot keyword group extraction algorithm and hot topic detection algorithm using n-grams.

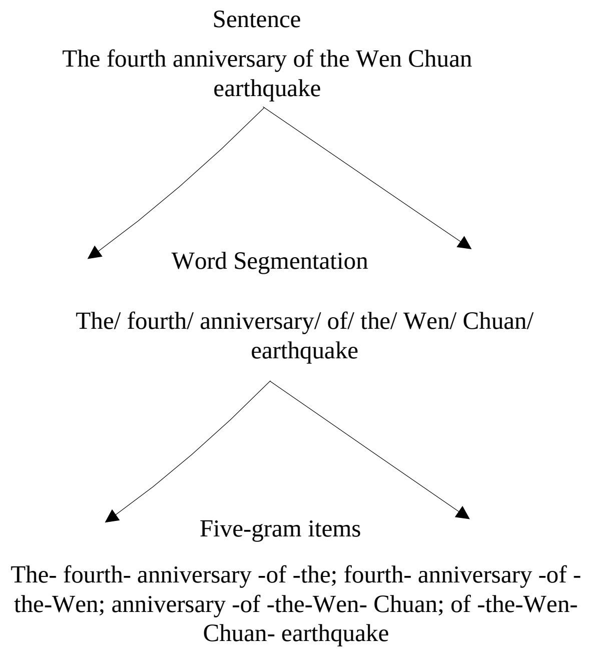

Sentence
The fourth anniversary of the Wen Chuan earthquake
Word Segmentation
The/ fourth/ anniversary/ of/ the/ Wen/ Chuan/ earthquake
Five-gram items
The- fourth- anniversary -of -the; fourth- anniversary -of -the-Wen; anniversary -of -the-Wen- Chuan; of -the-Wen- Chuan- earthquake

Figure.1 An example of five-gram model

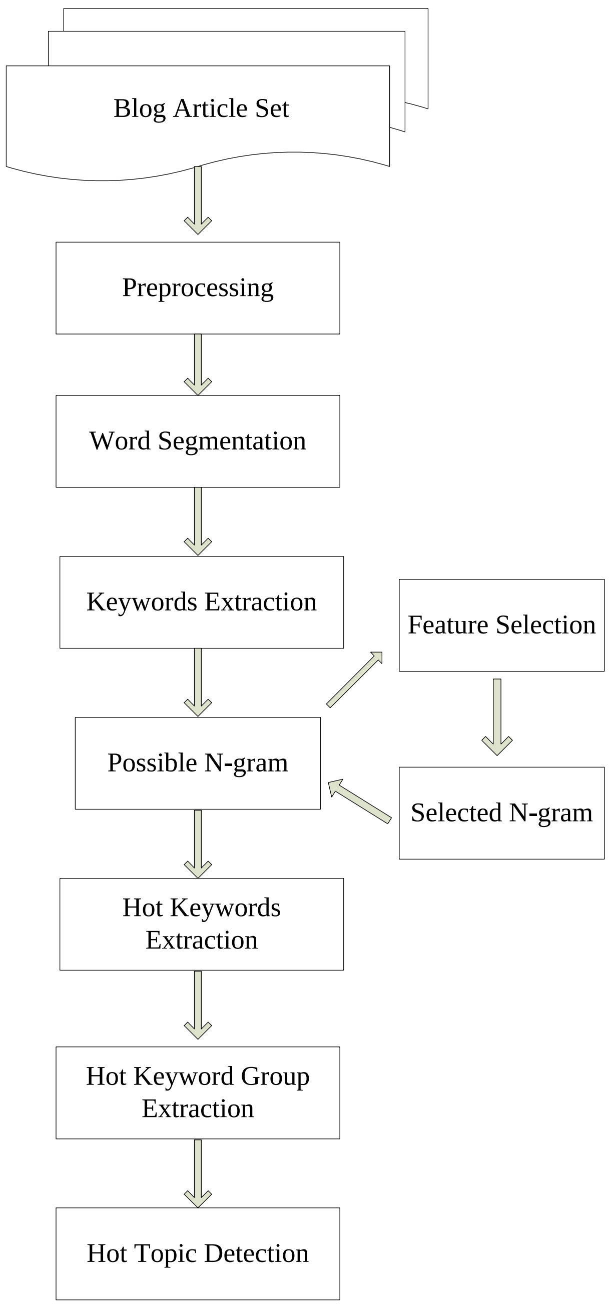


Figure.2 The framework of our approach

A. Hot Keyword Extraction Method

Because words in the title, text, and reply have different influence on theme of a blog article, we set a weight for each keyword, which reflects the importance and hotness of the keyword. The weight can be calculated as

$$weight = \alpha * tf\_tit + \beta * tf\_txt + \gamma * tf\_rly \quad (1)$$

where $\alpha + \beta + \gamma = 1$, and tf_tit is the word frequency in the title, tf_txt is the word frequency in the text, and tf_rly is the word frequency in the reply. In this paper, we set α =0.5, β =0.3, γ =0.2.

We consider the word whose weight exceeds the predefined $threshold_{hot}$ as a keyword, and put it into a keyword list. We create a keyword list for each blog article. The data structure of the node in the keyword list is defined as follows:

```
Typedef   struct{
        String   keyword ;
        Float    weight ;
        String   pos[] ;
}
```

every node in keyword list has three element, *keyword* , *weight* that represents the weight of keyword, and *pos*[] whose member are *n*-gram items containing keyword.

We use *n*-gram model to represent the sentence that contains keywords. In propose of simplification, we do not consider all the *n*-gram items of the sentence, only the *n*-gram items that contain keywords are considered. The keyword, keyword's weight and *n*-gram items containing keyword are stored in the keyword list.

After calculating the keyword list for each blog article, we get the hot keywords by merging keyword lists. The algorithm to merge keyword list is described as follows.

```
Algorithm1: Keyword Combination Algorithm
Input: the keyword list of blog article a, the keyword list of blog article b.
Output: merged keyword list.
1)    For ( i =1; i ≤ La.Length ; i ++)
2)        For ( j =1; j ≤ Lb.Length ; j ++)
3)    If(La[ i ].keyword==Lb[ j ].keyword)
4)     Merge(La[ i ],Lb[ j ]);
5)    End For
6)        End For
7)     If( Lb.Length >0)
8) Put all the nodes in Lb into La, delete Lb.
```

In Algorithm 1, *La.Length* is the length of the keyword list of article *a*, and *Lb.Length* is the length of the keyword list of article *b*. The steps to calculate *Merge (La[i], Lb[j])* can be defined as follows:

Step1 Alter the keyword's weight in *La[i].weight*, set *La[i].weight=La[i].weight + Lb[j].weight*;

Step2 Put all the *n*-gram items in *Lb*[*j*].*pos*[] into *La*[*i*].*pos*[] ;

Step3 Remove the repeated *n*-gram items in *La*[*i*].*pos*[] ;

Step4 Delete *Lb[j]* in *Lb.*

Suppose the number of the collected blog articles is *n*, there are *n* keyword lists. We use Algorithm 1 to merge all the keyword lists. Then get the merged keyword list of all the *n* blog articles. If there is a new blog article, it will be combined with the merged keyword list.

For convenience, the value of the keyword's weight is set to between 0 and 1. The normalization value of weight is described as

$$weight(j) = \frac{weight(j)}{\max_{i \in l}(weight(i))} \quad (2)$$

where $weight(j)$ presents the weight of keyword in the jth node in the merged keyword list, and l is the length of the merged keyword list.

We use polling method to look over the weight of each keyword in merged keyword list, if there is a keyword whose weight value is greater than the predefined $threshold_{hot}$, the keyword will be considered as a hot keyword. All the hot keywords are stored in a hot keyword list, which has the same data structure with keyword list.

For each node in hot keyword list, we use its weight value to measure the hotness of its keyword, that is to say, the hotness of a hot keyword is presented with the value of its weight. We rank all the keywords according to their weights' value. The hottest keyword is arranged in the first node of hot keyword list.

B . Hot Keyword Group Extraction Method

One hot keyword can't depict a hot topic comprehensively, so a hot topic always has several hot keywords, all the keywords are contained in the hot keyword list. In this paper, we combine hot keywords that describe the same hot topic to form a hot keyword group. The algorithm to combine hot keywords describing a hot topic is represented as follows.

```
Algorithm 2: Hot Keyword Combination Algorithm
Input: Hot keyword list $Lh$.
Output: Combined hot keyword list.
1)  For( i =1; i < Lh.Length ; i ++)
2)    For(j= i +1;j ≤ Lh.Length ; j++)
3)      sim ( $Lh[i].pos[], Lh[j].pos[]$ );
4)      If ( sim ( $Lh[i].pos[], Lh[j].pos[]$ )> $threshold_{sim}$ )
5)        Merge( $Lh[i], Lh[j]$ );
6)    End For
7)  End For
```

where $sim(Lh[i].pos[], Lh[j].pos[])$ represents the similarity of $Lh[i].pos[]$ and $Lh[j].pos[]$, the higher the value of similarity is, the more information $Lh[i].pos[]$ and $Lh[j].pos[]$ share, and the greater probability of $Lh[i].keyword$ and $Lh[j].keyword$ depicting a hot topic is. Let the number of *n*-gram items in $Lh[i].pos[]$ be *m*, the number of *n*-gram items in $Lh[j].pos[]$ be *k*. The steps of method to compute $sim(Lh[i].pos[], Lh[j].pos[])$ are described as follows:

```
1)  Let th =0;
2)  For (i =1; i ≤m; i ++)
3)    For( j =1; j ≤k; j ++)
4)      $sim(pos[i], pos[j])$ ;
5)      $th = th + sim(pos[i], pos[j])$ ;
6)    End For
7)  End For
```

where *th* represents the value of $sim(pos[i], pos[j])$, and its value is set to zero in the beginning. We use the following normalization formula. (3) to reset the value of *th* at last.

$$th = \sqrt{th}/th \quad (3)$$

$sim(pos[i], pos[j])$ represents the similarity between the *ith* *n*-gram item in $Lh[i].pos[]$ and the *jth* *n*-gram item in $Lh[j].pos[]$. The steps to calculate $sim(pos[i], pos[j])$ are defined as follows.

Step1 Compare the *n* continuous entries in $pos[j]$ with those in $pos[i]$, if they match, $sim_n(pos[i], pos[j])$ =1, else $sim_n(pos[i], pos[j])$ =0.

Step2 Cut the last entry in $pos[j]$, compare the *n-1* continuous entries with the *n* continuous entries in $pos[i]$. If the *(n-1)*-gram in $pos[j]$ is the part of *n*-gram in $pos[i]$, $sim_{n-1}(pos[i], pos[j])$ =*n-1/n*, else $sim_{n-1}(pos[i], pos[j])$ *=0.*

Step3 Repeat Step2 until all the remaining continuous entries in $pos[j]$ are compared. In each step, if they match, $sim_p(pos[i], pos[j]) = p/n$, else $sim_p(pos[i], pos[j]) = 0$, where *p* changes from *n-2* to 1.

Step4 The $sim(pos[i], pos[j])$ is defined as

$$sim(pos[i], pos[j]) = \sum_{p=1}^{n} sim_p(pos[i], pos[j]) \quad (4)$$

In Algorithm 2, the method to calculate $Merge(Lh[i], Lh[j])$ is represented as follows:

(1) Put the hot keywords in $Lh[j]$ into $Lh[i]$;

(2) Let $Lh[i].weight = Lh[i].weight + Lh[j].weight$;

(3) Put the *n*-gram items in $Lh[j].pos[]$ into $Lh[i].pos[]$, and remove the repeated *n*-gram items in $Lh[i].pos[]$;

(4) Delete $Lh[j]$ in Lh.

The hot keyword group in combined hot keyword list is the combination of several hot keywords that depict the same topic. Let $L[i]$ be a node in the combined hot keyword list L, then $L[i].keyword$ contains the keywords that have same information in their *n*-gram items, $L[i].weight$ presents the sum of the value of all keywords' weight in $L[i].keyword$, and $L[i].pos[]$ contains the *n*-gram items containing keyword. The *n*-gram items in $L[i].pos$ contain the context information of keywords in the sentence which includes these keywords, so the *n*-gram items descript the time and place information of hot topic in detail.

Due to the complexity of Chinese sentence, the detected keyword may not reflect the blog article theme, and the keywords in $L[i].keyword$ may not describe the hot topic. If a keyword in $L[i].keyword$ reflect a topic, then the keyword will appear in more than one *n*-gram items in $L[i].pos[]$. So the keyword group represents a hot topic candidate, we need to calculate the similarity between the keyword group and their *n*-gram items to decide whether the keyword group represents a hot topic.

C. Hot Topic Detection Algorithm

Let $\mathrm{K}=\{\mathrm{k}_1,\mathrm{k}_2,...,\mathrm{k}_\mathrm{n}\}$ represent the combined hot keyword list. Suppose the number of keywords in node k_i is *m*, the number of *n*-gram items in $k_i.[pos]$ is *p*, and the set of keywords is represented by $S=\{s_1,s_2,s_3,...,s_m\}$. The method to detect hot topic is described as follows:

Algorithm 3: Hot Topic Detection Algorithm
Input: Combined hot keyword list
Output: Hot topics

Step1 Calculate the similarity between s_i and $k_i.[pos]$ using the following formula.(5)

$$sim\_s_i(s_i,k_i.[pos]) = p_i / p \quad (5)$$

where p_i presents the number of *n*-gram items containing s_i in $k_i.[pos]$.

Step2 Repeat step1 until all the keywords in $\{s_1,s_2,s_3,...,s_m\}$ have been calculated similarity with $k_i.[pos]$.

Step3 The similarity between $k_i.keyword$ and $k_i.[pos]$ is defined as

$$sim(k_i.keyword,k_i.[pos]) = \sum_{i=1}^{m} sim\_s_i(s_i,k_i.[pos]) \quad (6)$$

Step4 If $sim(k_i.keyword,k_i.[pos])$ is greater than the predefined $threshold_{tpc}$, then k_i represents a hot topic.

The detected hot topic is represented by the keyword group in combined keyword list, the hotness of the topic is measured by the value of keywords' weight in keyword list. We rank the detected hot topics according to their hotness. Suppose the number of keywords in a keyword group is *m*, the hotness of the topic that the keyword group represents can be defined as

$$hotness = \sum_{i=1}^{m} weight(keyword_i) \quad (7)$$

where $weight(keyword_i)$ represents the value of the *ith* keyword's weight in keyword group.

IV. Experiment Analysis And Result

At present, there is no public blog corpus, we design a crawler to get blog articles from Sina Blog, Tencent Blog, which are two of most popular blogs in China. All the data were published from May 1, 2012 to May 14, 2012. There are up to 7980 articles. All articles have been already preprocessed, such as word segmentation, Part-of-Speech tagging and unknown words recognition.

The corpus was divided into two data sets:

Training Set: Published from May 1 to May 7, which contains 4480 articles.

Testing Set: Published from May 8 to May 14, which contains 3500 articles.

The experiment contains two steps. In first step, we use the 4480 articles to train. The size of *n* in *n*-gram, the values of $threshold_{hot}$, $threshold_{top}$, $threshold_{sim}$, and $threshold_{tpc}$ should be set in this step. In second step, we use the remaining 3500 articles to test and evaluate the parameters.

We come up with sixteen sets of experiments to finalize the parameters in the algorithms above. All the cases are shown in TABLE Ⅰ.

For all the cases, we implemented the method described in section 3 to find out the best values for the parameters. We use precision and recall to evaluate the performance of the cases above. The results are shown in TABLEⅡ.

TABLE Ⅰ
THE VALUES OF PARAMETERS

N	THD_{hot}	THD_{top}	THD_{sim}	THD_{tpc}	Cases
3	0.25	0.30	0.30	0.5	1
	0.30	0.35	0.35	0.55	2
	0.35	0.40	0.40	0.60	3
	0.40	0.45	0.45	0.65	4
4	0.25	0.30	0.30	0.5	5
	0.30	0.35	0.35	0.55	6
	0.35	0.40	0.40	0.60	7
	0.40	0.45	0.45	0.65	8
5	0.25	0.30	0.30	0.5	9
	0.30	0.35	0.35	0.55	10
	0.35	0.40	0.40	0.60	11
	0.40	0.45	0.45	0.65	12
6	0.25	0.30	0.30	0.5	13
	0.30	0.35	0.35	0.55	14
	0.35	0.40	0.40	0.60	15
	0.40	0.45	0.45	0.65	16

Remark: THD represents en *threshold*

TABLE Ⅱ
EVALUATION OF PARAMETERS

Approaches	Precision	Recall
Case1	63.14%	68.36%
Case2	65.68%	67.64%
Case3	69.89%	66.99%
Case4	70.26%	63.26%
Case5	62.18%	70.42%
Case6	67.70%	69.68%
Case7	71.81%	68.09%
Case8	72.24%	65.38%
Case9	73.14%	78.36%
Case10	75.68%	77.64%
Case11	79.89%	76.99%
Case12	80.26%	73.26%
Case13	70.20%	73.42%
Case14	72.73%	74.71%
Case15	74.85%	76.98%
Case16	75.34%	76.19%

As shown in TABLE Ⅱ, Case11 gives us the best result. That is N=5, $threshold_{hot}$ =0.35, $threshold_{top}$ =0.40, $threshold_{sim}$ =0.45, $threshold_{tpc}$ =0.60.

In second step, we use the remaining 3500 articles to test. In addition, we implemented other methods as compare experiments. The methods we used for comparison are Multiple Keywords Combination (MKC) method, single-pass clustering (SPC) method.

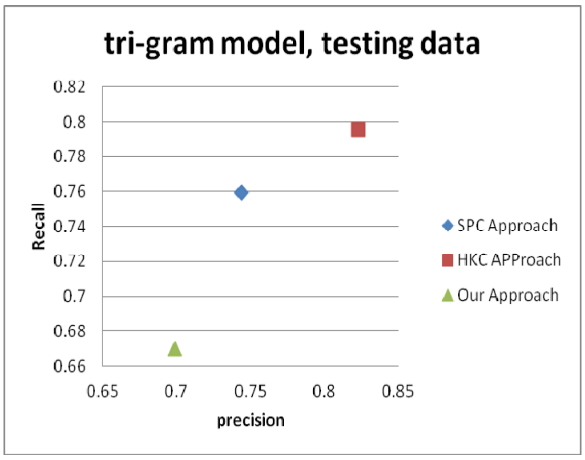
TABLE Ⅲ
EVALUATION OF COMPARED METHODS

Approaches	Num	Precision	Recall
SPC	46	74.39%	75.93%
HKC	52	82.32%	79.54%
Our approach	56	83.32%	79.98%

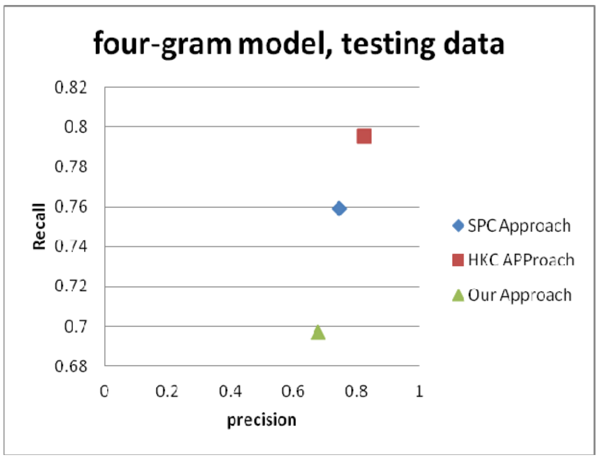
Remark: Num represents the numbers of detected hot topics.

As in TABLE Ⅲ, the result shows that, under the same experiment condition, the HKC method improves the effectiveness compared with SPC, and our approach got the best performance.

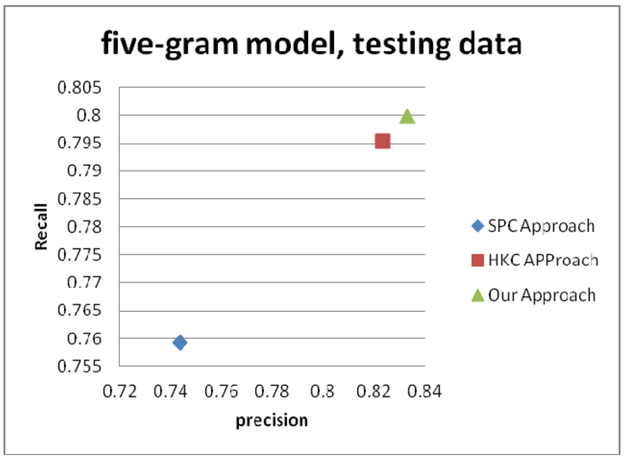
Fig.3 gives the results of comparison of our approach and other two approaches with feature selection on testing data in tri-gram, four-gram, five-gram and six-gram models. In Fig.3, we can see that our approach outperforms SPC and HKC approaches in most case, and we get the best results when we use five-gram model. That is to say, five-grams can well reflect the relationship of keywords, so the keywords depicting a topic can be effectively clustered together.



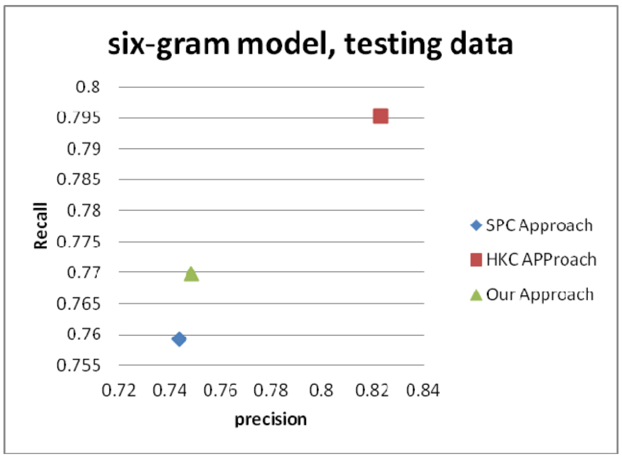
(a) Tri-grams



(b) Four-grams



(c) Five-grams



(d) Six-grams

Figure.3 Select n-grams

The detected top four blog topics using the proposed method based on five-gram model is shown in TABLE Ⅳ. This list is sorted by weight that represents the hotness of topic in descending order.

TABLE Ⅳ
TOP FOUR HOT TOPICS IN BLOGS DURING & MAY TO 14 MAY

Topic	Experimental Result
1	The fourth anniversary of the Wen Chuan earthquake
2	What will you do on Mother's Day
3	Why oppose the war in the south China sea between China and Philippines
4	Refined oil price falls for the first time

V. Conclusion And Future Work

This paper presents a hot topic detection and hotness of the topic evaluation method using *n*-gram model, which can improve hot topic retrieval in blogs. The main contributions of the paper are as follows. First, we analyze the *n*-gram model. Second we propose an effective way to validate the importance of words in blog article according to the features of blog article. Third, we apply *n*-gram model to design the algorithm to detect hot topic. Experiment results on Chinese corpus show that the proposed method is promising. However, there are still some shortages. First, the experiment data is trend to be incomplete in real life application. Second, the method to optimize the parameters is just through repeated experiments. Third, user participation degree, opinion communication degree of blog article should be considered. How to improve the coverage of experiment data, find the optimization algorithm to adjust the threshold and take more features of blog article into account will be conducted in the future.

Acknowledgment

This work was supported by the Scientific and Technology project of Henan Province of China (No. 102300410198) and (NO. 082102210007).

REFERENCES

[1] M. Platakis, D. Kotsakos, and D. Gunopulos. Discovering Hot Topics in the Blogosphere. In: Proceeding of the Second Panhellenic Scientific Student Conference on Informatics, Related Technologies and Applications EUREKA, pp. 122-132(2008)
[2] N. Li, and D. D. Wu. Using text mining and sentiment analysis for online forums hotspot detection and forecast [J]. Decision Support System, 48(2), 2010, 354-368
[3] X. Hao, and Y. Hu. Topic detection and tracking oriented to BBS. in: Proceedings of the 2010 International Conference on Computer, Mechatronics, Control and Electronic Engineering(CMCE), 4(2010), 154-157
[4] X. Y. Dai, Q. C. Chen, X. L. Wang, and J. Xu. Online topic detection and tracking of financial news based on hierarchical clustering. in: Proceedings of the Ninth International Conference on Machine learning and Cybernetics(ICMLC), 6(2010),3341-3346
[5] C. H. Wang, M. Zhang, S. P. Ma, and L. Y. Ru. Automatic hot event detection using both media and user attention . Journal of Computational Information Systems, 4(3), 2008, 985-992
[6] Y. Yang, J. Zhang, J. Carbonell, and C. Jin. Topic-conditioned Novelty Detection, in: Proceedings of the 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining'02, 2002, 668-693
[7] D. H. Zheng, and F. Li. Hot Topic Detection on BBS Using Aging Theory. Web Information System and Mining Lecture Notes in Computer Science, 5854(2009), 129-138
[8] S. D. Zhu, X. H. Wu, and J. P. Fan. Analysis of Bulletin Board System Hot Topic Based on Multiple Keywords Combination. in: Management and Service Science (MASS), 2011 International Conference , 8(2011), 1-4
[9] Y. Chen, X. Q. Cheng, and S. Yang. Outburst Topic Detection for Web Forums. Journal of Chinese Information Processing, 24(3), 2010, 29-36
[10] K. Chen, L. and Luesukprasert. Hot topic extraction based on timeline analysis and multidimensional sentence modeling. Knowledge and Data Engineering, IEEE Transactions on, vol.19: pp.1016-1025, Aug.2007
[11] Y. D. Zhou, X. H. Guan and et al. Approach to extracting hot topics based on network traffic content. Frontiers of Electrical and Electronic Engineering, vol.4: pp.20-23,2009
[12] H. X. Li, H. P. Zhang, and et al. Keywords based hot topic detection on Internet[C]. in: Proceedings of the 5th CCIR, 2009,134-143(in Chinese)

[13] K.Zheng, X. M. Shu, and H. Y. Yuan. Hot Spot Information Auto-detection Method of Network Public Opinion, Computer Engineering, 36(3),2010,4-6
[14] J. J Li, X. H. Zhang, and et al. Blog Hotness Evaluation Model Based on Text Opinion Analysis, Proceedings of the 2009 Eighth IEEE International Conference on Dependable, Autonomic and Secure Computing, p.235-240, December 12-14, 2009
[15] E. Z. Zhou, N. Zhong, and Y. F. Li. Hot Topic Detection in Professional Blogs. Lecture Notes in Computer Science, 6890(2011), 141-152
[16] L. You, Y. P. Du, and et al. BBS Based Hot Topic Retrieval Using Back-Propagation Neural Network. Proceedings of 1st International Joint Conference on Natural Language Processing, China, Springer-Verlag, 2005, pp.139-148
[17] T. T. He, G. Z. Qu, and et al. Semi-automatic Hot Event Detection. In proceedings of the 2nd International Conference on Advanced Data Mining and Applications, 2006, LNAI 4093, pp.1008-1016
[18] Knight, K, Hatzivassiloglou, V. Two-Level, Many-Paths Generation. In: Proceedings of the 33rd Annual Meeting of the Association for Computational Linguistics(ACL-95), Cambridge, MA(1995)252–260
[19] Brown, R, Frederking, R. Applying Statistical English Language Modeling to Symbolic Machine Translation. In: Proceedings of the Sixth International Conference on Theoretical and Methodological Issues in Machine Translation, Leuven, Belgium(1995)221–239
[20] Langkilde, I, Knight, K. Generating Word Lattices from Abstract Meaning Representation. Technical report, Information Science Institute, University of Southern California(1998)
[21] Bangalore, S, Rambow, O. Corpus Based Lexical Choice in Natural Language Generation. In: Proceedings of the 38th Annual Meeting of the Association for Computational Linguistics(ACL2000), Hongkong, China(2000)
[22] Habash, N. Dorr, B. Traum, D. Hybrid Natural Language Generation from Lexical Conceptual Structures. Machine Translation17(2003)

Xiaodong Wang received his M.E. degree in Computer Science from Tsinghua University, China in 1993 and received his Ph.D. degree in Information technology in Education from East China Normal University, China in 2003. He is an associate professor in the College of Computer Science, Henan Normal University, China. His current areas of interest include Ontology and Knowledge Engineering.
Email: wangxiaodong.wang@yahho.com.cn

Juan Wang is major in Computer Application Technology. She is an undergraduate student at Henan Normal University, China. Her research interests include Natural Language Processing , Ontology and Knowledge Engineering.
Email:juaner.510@163.com