UTDallas at TREC 2008 Blog Track

Bin Li, Feifan Liu, Yang Liu
Department of Computer Science
The University of Texas at Dallas

Abstract

This paper describes our participation in the 2008 TREC Blog track. Our system consists of 3 components: data preprocessing, topic retrieval, and opinion finding. In the topic retrieval task, we applied Lemur IR toolkit and used various techniques for query expansion. In the opinion finding and polarization task, we employed a feature-based classification approach. Then re-ranking was performed using a linear combination of the opinionated score and the topic relevance score. Our system achieved reasonable performance in this evaluation.

1 System Overview

We participated in several tasks of the 2008 TREC Blog Track. Figure 1 shows the flow diagram of our system. First, data preprocessing is implemented to remove HTML tags and useless context, and extract content from the blog web pages. Second, we apply the Lemur Information Retrieval toolkit to retrieve 1000 relevant documents for each topic. Query terms are selected from the title and descriptions, and weighted according to their TFIDF values. In addition, more weight is given to topical terms and quoted expressions for each topic. Third, for opinion finding, we employ a classification framework that exploits rich linguistic features, including lexical features, polarized features, and sentimental analysis based on mutual information with predefined sentiment terms. The same method is also used for polarity task, but with different class tags in the classifiers. Re-ranking is performed using a simple linear combination of the opinion score and the relevance scores provided by the opinion analysis and the topic retrieval modules respectively.

2 Data Preprocessing

The Blog06 test collection (Ounis et al., 2006) contains more than 3 million permalinks (a total size of about 148G). In order to obtain plain text data for document retrieval and opinion finding, we preprocessed the data using two filters: html-tag filter and non-English Blog filter. This also reduced the size of the data collection and made the subsequent modules more efficient.

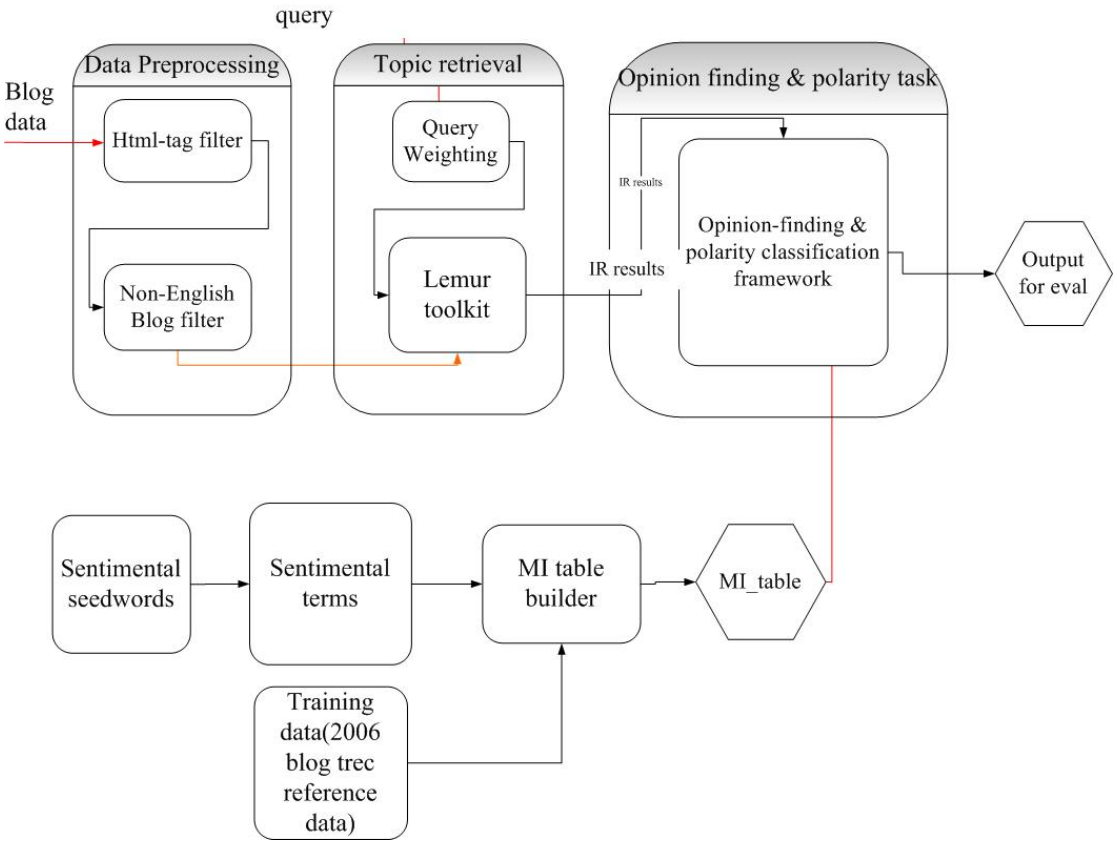

Figure 1. System overview for TREC Blog Track 2008.

First we used an html-tag filer to remove the useless characters and tags in the data. The blog web pages in the corpus have been collected from internet in HTML format. Some HTML tags are used to describe the structure of text-based information in a document, denoting certain text as links, headings, paragraphs, lists, and so on. Examples of such structure markers are "<TABLE>", "<TD>", and "<TR>". Other tags are used to supplement the textual information with interactive forms, embedded images, and other objects. For example, "<script>, <link>" are used for denoting functionalities. These extra characters and tags are generally irrelevant to our topic retrieval and opinion finding tasks and thus need to be removed.

During the html-tag filtering procedure, we treated the content between different markers differently. For some useful tags that provide important information about the topic of the documents, such as "<title></title>", we did not remove the content between the tags. Another example is the tag "<a href> ", which defines a link connection between different documents, and sometimes we cannot remove the link part. For example, in the sentence "The Buck's freshman phenom, Yi, shared his thoughts on being selected with the *Milwaukee Journal Sentinel* ", "*Milwaukee Journal Sentinel*" is a link to another related post and is a meaningful unit in this context. So in this case we only removed the tag "<a href> " and kept the content between the tags. In comparison, the content wrapped by <script> and </script> is pure programming scripts which are irrelevant to the topic, therefore we eliminated both the tags and the content. Table 1 lists all the content-irrelevant tags that we considered in this preprocessing step. After html tag filtering, the size of the blog corpus was reduced to 28G.

<style> <script> <fieldset> <form> <!--comments--> //comments <address> <acronym> <abbr> <server> <select> <option> <strike> <button>

Table 1: Content irrelevant tags.

The second part of data processing is to remove non-English blogs. We noticed that some pages in the data collection are not in English, which may cause problems in subsequent processing. In order to exclude those languages for the topic retrieval and opinion finding modules, we constructed a non-English blog filter based on a heuristic rule. In a blog, if the proportion of English characters along with regular symbols such as ".", "#", "?" is less than a predefined threshold (we used 0.5), we considered it as a non-English blog. This step further reduced the size of the data collection to 24G.

3 Topic Retrieval

Topic queries used in the Blog Track have three fields: title (T), description (D), and narrative structure (N). Each year, 50 topics were used for the blog track evaluation. Up till now, there are 150 topics in total. The following shows an example of a topic:

<num> Number: 851
<title> "March of the Penguins"
<desc> Description:
Provide opinion of the film documentary "March of the Penguins".
<narr> Narrative:
Relevant documents should include opinions concerning the film documentary "March of the Penguins". Articles or comments about penguins outside the context of this film documentary are not relevant.

Before the retrieval process, we built index using the Lemur Information Retrieval Toolkit[1] based on the preprocessed corpus. For each of the topic queries, the retrieval engine returns 1000 relevant documents, along with their relevance scores. We expanded queries using two approaches. For the title and description field retrieval, we applied a TFIDF-based approach to expand query terms to a proper size. For the title field only retrieval, a Google-set based query expansion method was used. We used the built-in pseudo feedback model in the Lemur toolkit. The following subsections explain in detail the query processing for the two conditions.

3.1 Query processing for title and description

A. Term weighting

In our system, we expanded the query for each topic using the terms that appear in <title> <desc> fields. Each term was assigned a TFIDF-based weight indicating how significant it is to that query. TF is the number of occurrences of a term *i* in a query q_j,

$$tf_{i,j} = \frac{n_{i,j}}{\sum_k n_{k,j}}$$

[1] http://www.lemurproject.org/

where $n_{i,j}$ is the number of occurrences of the considered term in query q_j, and the denominator is the number of occurrences of all the terms in query q_j.
IDF is the inverse document frequency, obtained by treating each query as a "document":

$$IDF_i = \log \frac{|Q|}{|\{q_j : t_i \in q_j\}|}$$

where |Q| is the total number of queries, and $|\{q_j : t_i \in q_j\}|$ is the number of queries containing term t_i (that is, $n_{ij} > 0$). The IDF is a measurement of the general importance of the term in the entire query collection. In addition, we also created a global IDF table that is computed using a large blog corpus (about 1/5 blog posts of the blog data collection). Then we chose the lower score from the two IDF tables as the IDF weight of a term. This can be thought of as a smoothing approach that helps to obtain better estimation and reduces noises caused by stop words and some frequently used terms in the blog query collection such as "find", "opinion", "comments".

B. Part-of-Speech filtering
We notice that using query terms composed of only nouns achieved the highest MAP in both 2006 and 2007 blog track evaluation, so we removed all of the other non-noun words from the query terms.

C. Quote weighting
For each topic, we consider the expression inside the quotation marks as the most informative unit, i.e., "March of the Penguin". Therefore, we used the quotes as is, without any part-of-speech filtering for those words. In addition, we assigned the weight for the entire term as twice of the sum of the weights of all the individual terms expanded for this query. This way Lemur will be more sensitive to such kind of quotes. For example, for query 851 as shown in the earlier example, the expanded terms with the weights are "*0.5 documentary, 0.5 film, 1.0 march, 1.5 penguins*", then "March of the Penguins" will have a weight of 2 * (0.5 + 0.5 + 1.0 + 1.5).

3.2 Query processing for title only

Google Sets is an online search engine which predicts related items based on co-occurrence statistics by using the web as a big pool of data. For the Title field only retrieval, we took advantage of this online resource to expand the topic terms to a larger set by picking the top 15 terms for each topic. Then we fed the new query to Lemur toolkit for the retrieval.

4 Opinion Finding

In our system we created a learning based classification framework to assign each topic-relevant blog a posterior probability, which indicates how likely it is opinionated. Then the re-ranking was conducted based on the combination of this posterior probability and the relevance score generated in the topic retrieval stage.

4.1 Classification framework

Since the goal of the opinion finding task is to rank those blogs higher that are more relevant and more opinionated, it is reasonable to detect an opinionated blog based on the part of it that is relevant to a specific topic rather than using the entire blog text. Motivated by this, we selected some topic specific sentences for each relevant blog for opinion analysis. This way, if one blog belongs to more than one topic, it might possibly be assigned different opinionated scores for different topics. We first split a blog into sentences based on sentence boundary detection[2], and used Lemur to retrieve top 5 relevant sentences corresponding to the topic. Then for each retrieved sentence, we also extracted its preceding and following sentences. Thus for each blog, we used a maximum 15 sentences to perform classification.

Before feature extraction, we first conducted some text normalization using regular expressions and rule-based approaches. Currently only months, weeks, numbers are normalized. For example, we replaced "September" with "MONTH", "2008" with "NUM". In addition, for the sentimental seed words (described later), we also replace them in the text with a polarized tag. For example, "good" becomes "O-POS" (positive). The features we explored are listed as follows, including words, part-of-speech, polarized features, and sentimental score statistics.

A. Lexical features

In addition to the bag-of-words feature, we also considered the following n-gram lexical features. Note that w_i and p_i is the word form and its part-of-speech tag.

- Combination of a word and its part-of-speech tag: $w_i p_i$
- Bigram words: $w_{i-1} w_i$
- Bigram of word and the neighboring part-of-speech tag: $w_{i-1} p_i$ and $w_i p_{i-1}$
- Trigram features: $w_{i-1} w_i w_{i+1}$ and $p_{i-1} p_i p_{i+1}$
- Trigram syntactic patterns: $w_{i-1} p_i w_{i+1}$ and $p_{i-1} w_i p_{i+1}$

Note that when we extracted the above features, we obtained some polarized features due to the normalization of those sentimental seed words mentioned above. For instance, if the i^{th} word is a positive sentimental seed word, we obtain polarized bigrams and trigram such as "w_{i-1}*_O-POS*", "*O-POS_*p_{i+1}", "p_{i-1}*_O-POS_*p_{i+1}" and "w_{i-1}*_O-POS_*w_{i+1}".

B. Sentimental features based on Mutual Information (MI) score

A *Sentimental score* is used to evaluate sentiment polarity of a textual context. The following describes the steps we used to compute this score.

.

(i) Generating sentiment terms

First of all, we manually selected a group of *sentimental seed words* as shown in Table 2. Then, based on a large review corpus[3], we computed the statistics of co-occurrences between the seed words and other adjectives appearing around them (we applied a window size 3 to limit the co-occurring span of the adjective words). Adjectives that co-occur with any seed word over 10 times are considered as *sentiment terms*. With a further human judgment on the polarity of the generated new terms, we

[2] The sentence boundary detection is done by using "mxterminator", a toolkit developed by Adwait Ratnaparkhi, University of Pennsylvania.

[3] This corpus comprises of 2000 movie reviews from (Pang et al., 2002), custom reviews from (Hu and Liu, KDD-2004) and 256 hotel reviews.

compiled 50 positive sentimental terms and 50 negative ones. Some examples of the new selected sentiment terms are shown in Table 3.

Positive	good, excellent, wonderful
Negative	bad, poor, terrible

Table 2. Sentiment seed words.

Positive	good, excellent, wonderful, relaxing, glorious, delicious, priceless, decorated, helpful, superb, …
Negative	bad, poor, terrible, worse, absent, stupid, problematic, boring, threatening, …

Table 3. Newly generated sentiment terms.

(ii) Calculating MI for adjectives

Once we have the sentiment terms, we will use them to compute the MI scores between each of them and an adjective. Here MI is defined as a score to evaluate the polarity strength of an adjective for both positive and negative categories. The following formula shows the MI score for an adjective word for the positive polarity:

$$MI^{+}(w_i) = \frac{\sum_{n \in St_+} Co(w_i, n, win)}{N}$$

where w_i refers to the target adjective; S_{t+} is the collection of positive sentimental terms; N is the size of S_{t+}; Co is the number of times that w_i co-occurs with a sentimental term n within a contextual window size of win (5 in our experiments). The co-occurrence statistics were obtained using the blog track 2006 reference collection. Similarly we calculate the MI score of the adjective word for its negative polarity using all the negative sentiment terms.

(iii) Calculating sentence-level sentiment scores

After the above steps, each adjective has a positive and a negative MI score. To calculate the sentiment score for each sentence, we simply added all of its adjectives' MI scores for positive and negative respectively.

Based on the sentimental scores for each sentence, we calculated the following statistics as features for a blog for opinion classification.

- Mean of the sentence sentiment scores for positive and negative, respectively.
- Mean of the difference between positive and negative scores among sentences.
- Mean of the ratio of positive and negative score among sentences.

C. Classifier setup

We used the annotated Blog-2006 and Blog-2007 data as training data and development data respectively. There are 4 opinion tags in the blog annotation, among which "2, 3, 4" correspond to the opinionated blogs. When using a binary classification (opinionated vs. not), "2, 3, 4" tags correspond to positive instances and "1" is negative class. We can also train a 4-way classifier based on those four

tags, then we assign blogs with “2,3,4” hypotheses as opinionated blogs. A comparison between different classification paradigms will be presented later.

4.2 Re-ranking

Opinion classification is applied to the relevant documents returned by the first blog retrieval module. Each blog has an associated posterior probability of being opinionated. Then we computed the final score using a linear combination of the opinionated measurement (*S_opi*) and the relevance score (*S_rel*) from topic retrieval:

$$final = (1-\lambda) * S\_opi + \lambda * S\_rel$$

where λ is a parameter to adjust the balance between being relevant and opinionated.

5. Polarity Task

To detect the polarity of a blog, we trained a classifier using blog instances with class tags “2, 3, 4”. Once we assign those tags for the relevant blogs returned by the topic retrieval module, we extracted the sentimental positive (“4”) and negative (“2”) ones. Then we applied the same re-ranking processing as in opinion finding, generating the ranked list of positive and negative blogs. In addition, considering some blogs classified as mixed polarity (tag “3”) might also belong to the positive or negative ones, we used those mixed polarity blogs to further expand the negative and positive ranked lists based on the posterior probabilities associated with positive and negative tags. In other words, from the hypothesized mixed opinion blogs, we selected the ones with higher posterior probability for the positive tags and added them to the end of the existing positive ranked list in the order of the posterior probabilities, until we reached 400 blogs (400 is an empirical number we chose). The same approach is also used for the negative ranked list.

For the polarity task we compared three different classification strategies on the development set, and based on the results, in our final submitted runs, we chose the approach described above (i.e., the last setting below).

- One stage with 4-way classification

  A 4-way classifier was trained to distinguish blogs as no-opinion, negative opinion, mixed opinion, and positive opinion, similar to the 4-way classification performed in opinion finding section. Directly based on the hypothesis from this classifier, we can generate a positive and negative ranked list respectively.

- Two-stage with successive binary classification and 3-way classification

  In this two-stage approach, we trained a 3-way classifier using blog instances with “2,3,4” tags and applied it only to those blogs which are recognized as opinionated ones by a binary classifier used for opinion finding. Then blogs classified as “2” and “4” were selected for the final negative and positive lists.

- One stage with 3-way classification

  This 3-way classifier was trained with tags “2,3,4”, and applied directly to the entire 1000 blogs recognized as relevant ones for each topic, generating the positive and negative lists.

6 System Performance

6.1 Performance on previous Blog-2006 and 2007 data

A. Topic retrieval

We conducted various experiments on the development set (Blog 2006 and 2007 data) to finalize our submitted system parameters. Table 4 shows the results for the title only (T) task using and without using Google-set based query expansion. The improved results suggest that the expanded terms produced by Google-set are helpful for query expansion.

Year	T only (Map/P5)	T only, with Google-set query expansion (Map/P5)
2006	0.29/0.63	0.31/0.63
2007	0.33/0.64	0.35/0.66

Table 4. Topic retrieval results using title only.

We also evaluated topic retrieval performance using TD and TDN for query expansion respectively. In Table 5, results show a performance degradation when including words in the <narr> field (adding N) in the query terms. This might be due to some noisy terms introduced when using this field.

Year	TD (Map/P5)	TDN (Map/P5)
2006	0.37/0.74	0.34/0.72
2007	0.43/0.76	0.40/0.74

Table 5. Comparison between using TD and TDN for topic retrieval.

B. Opinion finding

For the system development for opinion finding and polarity recognition, we used the reference relevant documents, instead of the baseline results from our topic retrieval module. Therefore the results shown below are generally better than those in the submitted runs. The experiments reported here are mainly used to finalize the parameters for the opinion and polarity finding task.

Table 6 shows the results for opinion finding on the Blog 2007 data using different classifiers, classification strategies, and text normalization choices. The classifier was trained using the Blog 2006 data. For the 4-way classification, we show results using SVM and the maximum entropy (Maxent) classifier. Since Maxent performs much better than SVM, we chose to use Maxent for other experiments. We can see that using the Maxent classifier, the binary classification framework outperforms the 4-way classification, especially after text normalization.

Classifier	4-way vs. binary	Text normalization	Map/P@5
SVM	4-way	No	0.018/0.32
Maxent	4-way	No	0.384/0.536
Maxent	binary	No	0.4/0.66
Maxent	binary	Yes	0.45/0.68

Table 6. Opinion finding results on Blog-2007 data.

C. Polarity task

Polarity results on the development data are shown in Table 7. As described in Section 5, we tried different classification strategies on this task, among which one-stage method with a 3-way classifier obtained the best results compared to the other settings. Similar to the opinion finding task, the SVM classifier performed poorly for both opinion finding and polarity task. Both text normalization and the expansion using blogs with mixed opinion (tag "3") based on the corresponding posterior probabilities yielded performance gain on the development set.

Classifier	2-pass vs. 1-pass	Classification strategy	Text normalization	Expansion from mixed	Pos (Map/P5)	Neg (Map/P5)
SVM	One	4-way	No	No	.009/.29	.031/.73
Maxent	One	4-way	No	No	.26/.54	.07/.51
Maxent	Two	bin+3-way	No	No	.16/.51	.14/.61
Maxent	Two	bin+3-way	Yes	No	.20/.51	.13/.59
Maxent	One	3-way	Yes	No	.26/.47	.16/.58
Maxent	One	3-way	Yes	Yes	.35/.47	.27/.58

Table 7. Results for the polarity task on Blog-2007 data.

6.2 Submission performance at Blog-2008

A. Topic retrieval

Table 8 shows the topic retrieval results of our submitted runs. "SplBaseT" denotes the run using only the topic information, and "SplBaseTD" used both topic and description information of the query. The query processing approaches for these runs are described in Section 3. Using both title and description (TD) yielded better performance, consistent with what we have observed from the Blog-2007 results.

RunID	Map	P@5	P@10
SplBaseT	0.3077	0.6000	0.5960
SplBaseTD	0.3298	0.6480	0.6380

Table 8. Topic retrieval performance at Blog-2008.

B. Opinion finding

Table 9 shows the best result for opinion finding we obtained based on different baselines. The different performance suggests that the quality of topic retrieval has a great impact on opinion finding.

Baseline	Map
1	0.3251
2	0.2789
3	0.3565
4	0.3844
5	0.3036

Table 9. Opinion finding performance at Blog-2008.

In addition, we employed feature scaling in our classifier, which scales each feature into the range

between 0 and 1. Table 10 shows the MAP results with and without scaling. Overall, there seems to be slight improvement due to feature value scaling.

Baseline	1	2	3	4	5
No scaling	0.3242	0.2754	0.3565	0.3071	0.2629
Scaling	0.3251	0.2789	0.3565	0.3465	0.2874

Table 10. Comparison between feature scaling and non-scaling. Results shown are the MAP results on Blog-2008 data.

Finally we show the effect of the interpolation weight when combining the topic relevance score and the opinion score. Results are shown in Figure 2 using different weights for different baseline systems. We notice that most baselines (3, 4 and 5) prefer the higher weight on the relevance score, indicating a more dominant role of the topic retrieval component. This is not true for all baselines due to different retrieval qualities and representation of relevance scores.

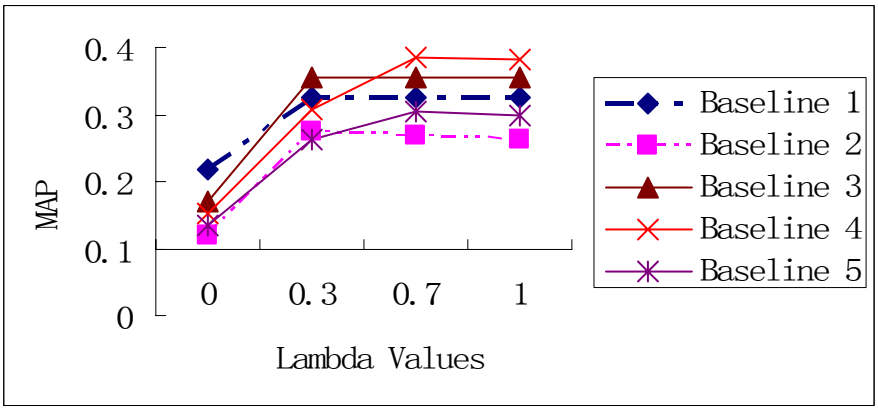

Figure 2. Performance curve using different combination weights for the relevance score and opinion score on different baselines.

C. Polarity task

For the polarity task, we submitted two results for each baseline corresponding to different combination weights for the relevance score (0.3 and 0.7). We noticed that different baselines prefer different weights. The better results between the two runs are shown in Table 12 for each baseline.

	Baseline	Map	P@5
Positive	1	0.1110	0.2286
	2	0.0760	0.1388
	3	0.1120	0.1592
	4	0.1350	0.2286
	5	0.1108	0.2041
Negative	1	0.0815	0.1917
	2	0.0719	0.1583
	3	0.0888	0.1833
	4	0.0962	0.2250
	5	0.0746	0.1792

Table 12. Polarity task performance at Blog-2008.

6.3 Comparisons among performance on different datasets

In Table 13, we show the MAP scores of our best runs on opinion finding and polarity tasks based on different datasets for comparison (Blog06, 07, and 08). We can see that the performance on Blog-2008 is worse compared to Blog06 and Blog 07. This is as expected since our submitted system is trained on Blog-2006 and Blog-2007 reference data. How to improve the generalization of our systems will be addressed in our future work.

Dataset	Opinion (NOpMMs4_0.3)	Positive (NTrMM4_0.7)	Negative (NTrMM4_0.7)
Blog-2006	0.57	0.19	0.18
Blog-2007	0.70	0.31	0.20
Blog-2008	0.35	0.14	0.10

Table 13. Opinion finding and polarity detection results for Blog 06, 07, and 08 based on the best performing systems.

7 Conclusion

In this paper, we described our system in three tasks in the TREC blog track 2008: topic retrieval, opinion finding, and polarity detection. Different query expansion methods have been evaluated for topic retrieval. Our results show that query expansion on Title and Description fields with appropriate weighting can yield better performance. Extensive experiments have also been conducted on the Blog 2007 development data for opinion finding and polarization, proving that the strategies we employed in our system, such as text normalization, polarized features, sentimental features, classification mechanism, are very helpful to improve the system's performance.

We will further explore more features and conduct more detailed experiments to evaluate the contribution of different features for opinion finding and polarity task. In addition, we plan to investigate a more effective and systematic combination framework between relevance score and opinionated score of each blog during the re-ranking process.

Reference

1. Bo Pang, Lilian Lee, and Sharivakumar Vaithyanathan. 2002. Thumbs up? Sentiment classification using machine learning techniques. In Proceedings of EMNLP.
2. Minqing Hu and Bing Liu, 2004. Mining and summarizing customer reviews. In Proceedings of ACM SIGKDD.
3. Iadh Ounis, Maarten de Rijke, Craig Macdonald, Gilad Mishne, and Ian Soboroff. 2006. Overview of the TREC-2006 Blog Track.