Experiments in Trec 2007 Blog Opinion Task at CAS-ICT

Xiangwen Liao[1,2], Donglin Cao[1,2], Yu Wang[1,2], Wei Liu[1,2], Songbo Tan[1], Hongbo Xu[1] and Xueqi Cheng[1]

[1]Institute of Computing Technology, Chinese Academy of Sciences, Beijing, China

[2]Graduate school of *the Chinese Academy of Sciences*, Beijing, China

liaoxiangwen@software.ict.ac.cn

Abstract

This paper describes our participation in TREC 2007 Blog Track Tasks: Opinion retrieval and Polarity classification. As for Opinion retrieval task, a two-step approach is used to retrieve opinion relevant blog unit (that is blog post and its comments) given a query after filtering Spam blog and extracting blog unit. With Polarity Classification, Drag-push [1] based classifier is employed to get polarity of blog unit. Finally, we introduce all the runs submitted.

1. Introduction

Given a query, the aim of Blog Opinion Retrieval task [2] is to retrieve opinion relevant blog unit using blog data collection as an important test bed. Compared with traditional retrieval task, opinion retrieval is more challenged since it is very difficult to combine opinion (i.e. Positive, negative and natural) with some topic and then rank the blog unit according to the strength of opinion of the query topic. To complete such task, we present a framework that filters spam data, extracts blog unit, retrieves relavant blog unit and finally re-ranks related blog according the strength of blog unit. A machine learning approach is employed to filter different kind of Spam blogs. The resulted permalink is used as input for our string uniqueness based algorithm to extract blog unit. And then topic related result list is retrieved by BM25 [3] and Language Model given a query. In the end, Drag-push [1] based algorithm is adopted to re-rank the result list by opinion strength.

As for polarity classification, there are two strategies used to get polarity according to different representation of each relevant blog unit after filtering Spam Blog and extracting blog unit. First, the blog unit is represented by traditional multidimensional vector of various words. Second, sentiment lexicon constructed by Inquirer [4] is used to compute blog unit vector. Finally, Drag-push based classifier is employed to classify polarity of each blog unit.

2. Filtering Spam Blog

Spam blogs or Splogs are faked blogs, containing gibberish or plagiarized content from other blogs and news sources with the sole purpose of hosting profitable content based ads, promoting affiliate sites unjustifiably [5]. So Splogs can be roughly classified into faked blogs and link spam blogs. The former is more prevalent in the blogosphere and the latter is quite common in other web pages.

For the link spam, we counted the links density which is very simple but effective. A blog is classified as link spam if the number of links in it exceeds the predefined thresholds. The thresholds are trained from a heuristic approach and tuned to get a better accuracy.

For the faked blogs detection, we adopt a machine learning approach, and we treat the task to be a two-category classification problem. Our model is based on SVM [6], which is widely used and

performs well in text classification. We introduce several powerful local features, as follows:

1) We choose the traditional feature such as “bag-of-words”, and “bag-of-word-N-Grams”based on blogs’ textual content, where the occurrence of words in text or content of some specific tags on a page like “title” is used as a feature.
2) We use two new features which are called bag-of-anchors and bag-of-urls. In bag-of-anchors, features are extracted from the anchor text on a page. Similarly, in bag-of-urls, URLs are tokenized on punctuation (\,/,.,?,=,etc.) and each token is used as a feature. For the bag-of-urls, we adopt additional segmentation of the URLs to combat the case that words are glued together as one token with the attempt to avoid being detected (e.g.,businesscardforfree) [7] .
3) Normalized term frequency (TF) and binary forms are employed to represent the features, and we used Mutual Information together with term frequency as our feature selection mechanism [3], which is simple but performs well in practice for SVM.

Finally, we train the libsvm[8] classifier by manually labeling some examples, and detect splog using above features.

4. Extracting Blog Unit (Post and its Comments)

Blog extraction is a kind of difficult job because each blog site has a lot of different blog templates which can be chosen by bloggers. Each template has different styles which greatly enhance the difficulty of extraction.

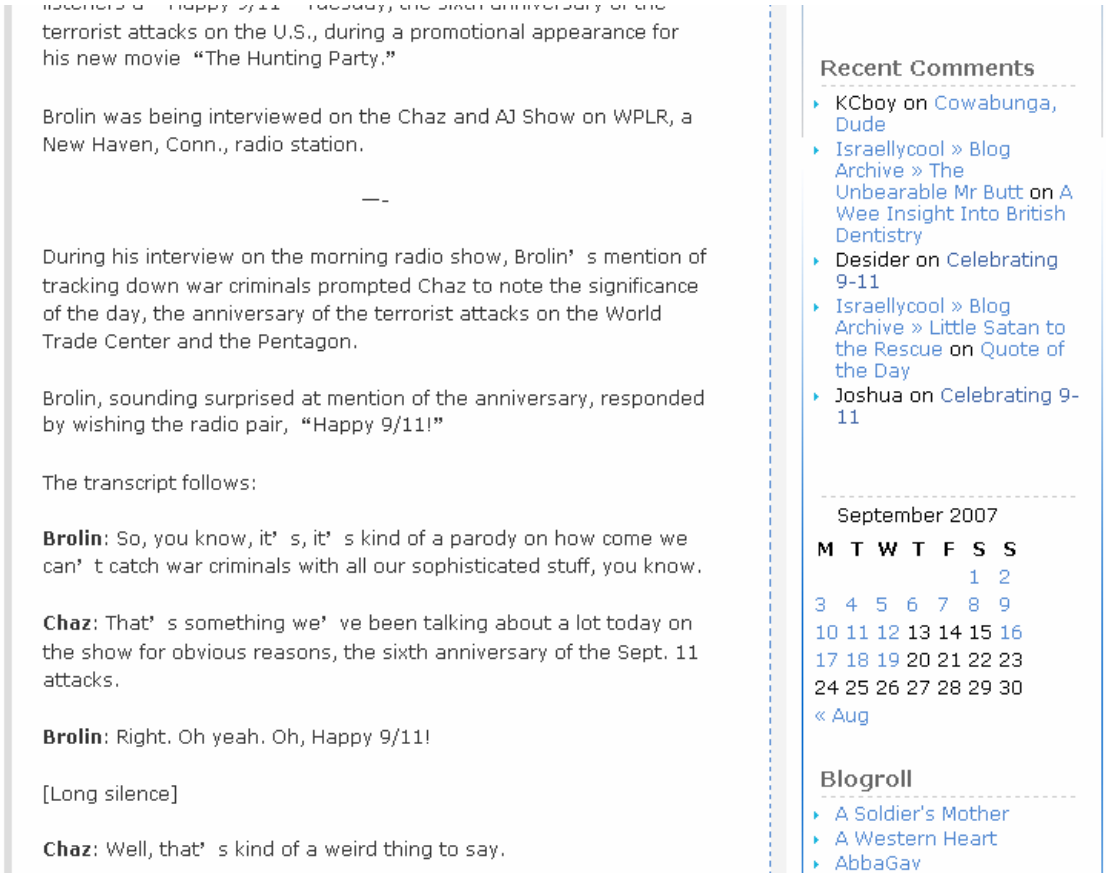

Figure 1 A blog page

While many algorithms for content extraction [9][10], template detection[11][12] and page segmentation[13] already exist, they don't fit well in Blogsphere. According to blog opinion retrieval task, two requirements for the content extraction algorithm are raised. First, this algorithm should run fast enough to meet the updating speed of the whole blogsphere. Second, the extraction result should be precise. Not only the page template but also some dynamic parts of pages should be excluded from the extraction result. Figure 1 shows a blog page. On the right part of this page, there are "Recent Comments", "Calendar", and "Blogroll", all of them should be excluded. Considering these two requirements, most existing algorithms can not satisfy both requires at the same time. A new content extraction algorithm is created to fit with both requires.

Our algorithm relies on such an assumption: given a bunch of pages from a site, main content sections contain more unique strings than other sections. It seems reasonable since main content sections are more likely to provide new information to people.

With this assumption, we propose an effective algorithm, as follows:

1) For every page of the blog, divide it into text segments. Every text in a web page that can be seen by users after rendering is called a text segment. In web pages, text segments are separated by html tags, process instructions, comments, etc.
2) Build a hash map from all pages of the blog, keys of the map are text segments, values of the map are occurrence times of keys.
3) Erase all text segments whose occurrence times are bigger than a given threshold.

3. Opinion Retrieval

The aim of blog opinion retrieval task is to retrieve opinion relevant blog unit using blog data collection as an important test bed given a query. In our system, a two-step approach is used to retrieve opinion relevant blog unit.

First, the Firtex platform[14] is used to index the blog unit collection constructed after filtering spam blog and extracting procession. Since BM25 algorithm always effectively retrieves related document in ad hoc retrieval task, it is also employed to retrieve relevant blog unit as input for re-ranking according to strength of opinion in our system. In this Model, we have:

$$CIW = \frac{TF(k_1+1)}{K+TF} \log \frac{(r+0.5)(N-n-R+r+0.5)}{(R-r+0.5)(n-r+0.5)}$$

Where $K = k_1 * ((1-b) + b\frac{DL}{AVDL})$.

The parameters in this formula are K_1, b, r and R. Since we have no relevance information in advance, we set r = R =0. Experiments in former TREC suggest a value of b around 0.75 is good. So we also set b=0.75. We tune K_1 by experiments using 2006 TREC data and set it 2.75.

Language Model is used in our Firtex system besides the BM25 algorithm.In order to improve the performance of Language Model, we used Dirichlet smoothing method. Our ranking function is as follows:

$$p(w \mid d) = \frac{c(w;d) + \mu\, p(w \mid C)}{\sum_{w} c(w;d) + \mu}$$

Where d denotes a document, w denotes the word, C denotes the document collection. c(w;d) denotes the number of word w in document d. p(w|C) denotes the probability of the word w in the

whole document collection C. μ is the only parameter which can be adjusted in language model. Our experiments in TREC 2006 corpus show that the best result is achieved when the value of μ is 2500.

Comparing BM25 Model with Language Model, our experiments show that Language Model is a little better than BM25 Model. Finally, the related topic result is obtained in Language Model.

After retrieving the related topic result lists, our system re-ranks the blog unit and constructs 6 submitted runs: 2 simple statistics runs, 3 Drag-push classifier based runs, and 1 relevance run. Simple statistics runs are constructed by just computing the sum score of sentiment words (specified by Inquirer) in blog unit, normalizing the sum score, and then re-ranking the result lists. The difference of the two submitted runs is: SmpStm gets sum score after stemming blog unit and lexicon words while BaseSmp not. Drag-push classifier is employed to get 3 Drag-push classifier based runs:

1) Train a Drag- push classifier using TREC 2006 Blog Task result lists;
2) Score each 2007 Blog Unit by the sum of similarity to positive core and negative core;
3) Re-rank each result lists to form submitted run.

While all the 3 Drag-push classifier based runs use the above method to get result lists, there is difference among them: DrapCom just uses relevant blog unit to train and gets result lists; DrapOpi represents training and tests blog unit by lexicon words; DrapStem represents training and tests blog unit by stemmed lexicon words.

Run	Map	R-precision	p@10
Relevent	0.3041	0.3600	0.4460
BaseSmp	0.1433	0.1843	0.2300
DrapCom	0.1564	0.1999	0.2460
DrapOpi	0.1659	0.2176	0.2920
DrapStm	0.1537	0.2047	0.2640
SmpStm	0.1409	0.1788	0.2320

Table 1 Result of submitted opinion search runs

For all the six runs, *Relevent* run gets best MAP, R-precision and p@10. The reason leading to such result is that after retrieving relevant blog units we re-rank the relevant result lists according to strength of opinion. So it is possible that some less relevant but more strength opinion units get higher rank than more relevant and less strength opinion units. But the test evaluation seems to prefer more relevant and less strength opinion units.

Run	R-accuracy	p@10	p@20
DrapComSub	0.0689	0.1140	0.1050
DrapOpiSub	0.0801	0.0980	0.0870
DrapStm	0.0818	0.1060	0.1020

Table 2 Result of submitted polarity runs

4. Polarity Task

There are three runs we submitted this year. In our system, we employ Drag-push classifier to get the polarity of each blog unit. The Drag-push classifier takes advantage of training errors to

successively refine the classification model of the Centroid Classifier. We train the Drag-push classifier using TREC 2006 Blog Task result lists, use the resulting classifier to classify each related blog unit and then construct the submitted runs. The difference of the three runs is the representation of blog unit: DrapComSub uses bags of word; DrapOpiSub bases lexicon words; DrapStm adopts stemmed lexicon words.

The result illustrates that using blog unit as a process unit is no enough to classify polarity. It is necessary to extract opinion relevant fragment and use such finer granular fragment to classify blog unit.

5. References

[1]Songbo Tan, Xueqi Cheng, etc. A Novel Refinement Approach for Text Categorization.CIKM2005, October 31–November 5, 2005, Bremen, Germany.

[2]Iadh Ounis, Maarten de Rijke, etc. Overview of the TREC-2006 Blog Track, http://trec.nist.gov/pubs/trec15/papers/BLOG06.OVERVIEW.pdf

[3] Jones K.S., Walker S.,& Robertson S.E.A Probabilistic Model of Information Retrieval: Development and Comparative Experiments Part 2.Information Processing and Management, 2000, 36:779～808

[4] Philip J. Stone, Dexter C. Dunphy, etc. The General Inquirer: A Computer Approach to Content Analysis. The MIT Press, 1966.

[5]Pranam Kolari,Akshay Java and Tim Finin. Characterizing the Splogosphere.*WWW2006, May 22-26, 2006, Edinburgh,UK*

[6] Boser, B. E,Guyon, I. M and Vapnik, V. N. 1992. A training algorithm for optimal margin classifiers. In *COLT '92: Proceedings of the fifth annual workshop on Computational learning theory*, 144–152. New York: ACMPress.

[7]Pranam Kolari,Tim Finin and Anupam Joshi. SVMs for the Blogosphere:Blog Identification and Splog Detection.2006,*American Association forArtificial Intelligence.*

[8] libsvm, http://www.csie.ntu.edu.tw/~cjlin/libsvmtools/#3.

[9] Shian-Hua Lin, Jan-Ming Ho. Discovering Informative Content Blocks from Web Documents. In Proc. of the SIGKDD'02 Conf., pages 588-593, 2002

[10] Lan Yi, Bing Liu, Xiaoli Li. Eliminating Noisy Information in Web Pages for Data Mining. In Proc.of the SIGKDD'03 Conf., pages 296-305, 2003

[11] Ziv Bar-Yossef, Sridhar Rajagopalan. Template Detection via Data Mining and its Applications. In Proc. of the WWW'02 Conf., pages 580-591, 2002

[12] Ling Ma, Nali Goharian, Abdur Chowdhury. Extracting unstructured Data from Template Generated Web Document. In Proc. of the CIKM'03 Conf., pages 516-519, 2003

[13] Deng Cai, Shipeng Yu, Ji-rong Wen and Wei-Ying Ma. Extracting Content Structure for Web Pages based on Visual Representation. In Proc. of the APWeb'03 Conf., number 2642 in LNCS, pages 406-417, 2003

[14] Ruijie Guo,Xueqi Cheng,etc. "Efficient On-line Index Maintenance for Dynamic Text Collections by Using Dynamic Balancing Tree". In *Proceedings of the 16th ACM Conference on Information and Knowledge Management (CIKM 2007)*. Lisbon, Portugal, November 2007.