The BlogVox Opinion Retrieval System*

Akshay Java, Pranam Kolari, Tim Finin, Anupam Joshi and Justin Martineau
University of Maryland, Baltimore County
{aks1, kolari1, finin, joshi, jm1}@cs.umbc.edu

James Mayfield
Johns Hopkins University Applied Physics Laboratory
james.mayfield@jhuapl.edu

Abstract

The BlogVox system retrieves opinionated blog posts specified by ad hoc queries. BlogVox was developed for the 2006 TREC blog track by the University of Maryland, Baltimore County and the Johns Hopkins University Applied Physics Laboratory using a novel system to recognize legitimate posts and discriminate against spam blogs. It also processes posts to eliminate extraneous non-content, including blog-rolls, link-rolls, advertisements and sidebars. After retrieving posts relevant to a topic query, the system processes them to produce a set of independent features estimating the likelihood that a post expresses an opinion about the topic. These are combined using an SVM-based system and integrated with the relevancy score to rank the results. We evaluate BlogVox's performance against human assessors. We also evaluate the individual splog filtering and non-content removal components of BlogVox.

Introduction

Blog posts contain noisy, ungrammatical and poorly structured text. This makes open-domain, opinion retrieval for blogs challenging. In addition any text analytics system that deals with blogs must address two key issues: (i) detecting and eliminating spam blogs and spam comments and (ii) eliminating noise from link-rolls and blog-rolls.

The BlogVox system was developed by the University of Maryland, Baltimore County and the Johns Hopkins University Applied Physics Laboratory to perform the opinion retrieval task defined by the 2006 TREC Blog Track. In this task, a user enters a query for a topic of interest (e.g., March of the Penguins) and expects to see a list of blog posts that express an opinion (positive or negative) about the topic. The results are ranked by the likelihood that they are expressing an opinion about the given topic. The approach used in BlogVox has several interesting features. Two techniques are used to eliminate spurious text that might mislead the judgment of both relevance and opinionatedness. First, we identify posts from spam blogs using a machine-learning based approach and eliminate them from the collection. Second, posts are "cleaned" before being indexed to eliminate extraneous text associated with navigation links, blog-rolls, link-rolls, advertisements and sidebars. After retrieving posts relevant to a topic query, the system applies a set of scoring modules to each producing a vector of features estimating the likelihood that a post expresses an opinion about the topic. These are combined using an SVM-based system and integrated with the overall relevancy score to rank the results.

Opinion extraction and sentiment detection have been previously studied for mining sentiments and reviews in domains such as consumer products (Dave, Lawrence, & Pennock 2003) or movies (Pang, Lee, & Vaithyanathan 2002; Gilad Mishne 2006). More recently, blogs have become a new medium through which users express sentiments. Opinion extraction has thus become important for understanding consumer biases and is being used as a new tool for market intelligence (Glance *et al.* 2005; Nigam & Hurst 2004; Liu, Hu, & Cheng 2005).

Blog posts contain noisy, ungrammatical and poorly structured text. This makes open-domain, opinion retrieval for blogs challenging. In addition any text analytics system that deals with blogs must address two larger issues: (i) detecting and eliminating posts from spam blogs (commonly known as splogs) and spam comments and (ii) eliminating irrelevant text and links that are not part of the post's content.

Recently, Spam blogs, or splogs have received significant attention, and techniques are being developed to detect them. Kolari, et al. (Kolari, Finin, & Joshi 2006) have recently discussed the use of machine learning techniques to identify blog pages (as opposed to other online resources) and to categorize them as authentic blogs or spam blogs (splogs). (Kolari, Java, & Finin 2006) extends this study by analyzing a special collection of blog posts released for the Third Annual Workshop on the Weblogging Ecosystem held at the 2006 World Wide Web Conference. Their findings on spam blogs confirms the seriousness of the problem, the most recent data shows about 64% of "pings" collected from the most popular ping-server for English blogs are from splogs.

Blog posts are complex documents and consist of a core containing the post's real content surrounded by an array of extraneous and irrelevant text, images and links. This "noise" includes links to recent posts, navigational links, advertisements and other Web 2.0 features such as tag rolls, blog rolls, Technorati tags, Flickr links and often accounts

*Partial support was provided by an IBM Fellowship and by NSF awards ITR-IIS-0326460 and TR-IDM-0219649.

for 75% or more of the post's size. The presence of this extra material can make it difficult for text mining tools to narrow down and focus on the actual content of a blog post. Moreover, these features may also reduce search index quality. Discounting for such noise is especially important when indexing blog content. Blog posts are complex documents and consist of a core containing the post's real content surrounded by an array of extraneous and irrelevant text, images and links. This "noise" includes links to recent posts, navigational links, advertisements and other Web 2.0 features such as tag rolls, blog rolls, Technorati tags, Flickr links and often accounts for 75% or more of the post's size. The presence of this extra material can make it difficult for text mining tools to narrow down and focus on the actual content of a blog post. Moreover, these features may also reduce the quality of the search index. Discounting for such noise is especially important when indexing blog content.

The paper is organized as follows: first we give a brief overview of the 2006 TREC Blog Track, and its associated dataset in section . Then we describe our system, BlogVox, in Section . The next three sections explain how BlogVox works. In Section we explain how we deal with splogs. Following that we describe in Section a simple, yet effective, heuristic for cleaning the blog post to remove any extraneous links and other features. Then we explain how BlogVox scores posts for opinion ranking in Section . We present the TREC results along with an evaluation of the post cleaning and splog filtering techniques in Section . Finally, we discuss our results in Section and conclusions in Section .

The TREC Blog Track

The 2006 TREC Blog track, organized by NIST, asked participants to implement and evaluate a system to do "opinion retrieval" from blog posts. Specifically, the task was defined as follows: build a system that will take a query string describing a topic, e.g., "March of the Penguins", and return a ranked list of blog posts that express an opinion, positive or negative, about the topic.

For training and evaluation, NIST provided a dataset of over three million blogs drawn from about 80 thousand blogs. The TREC dataset consisted of a set of XML formatted files, each containing blog posts crawled on a given date. The entire collection consisted of over 3.2M posts from 100K feeds (Macdonald & Ounis 2006). These posts were parsed and stored separately for convenient indexing, using the HTML parser tool [1]. Non-English blogs were ignored in addition to any page that failed to parse due to encoding issues.

In order to make the challenge realistic NIST explicitly included 17,969 feeds from splogs, contributing to 15.8% of the documents. There were 83,307 distinct homepage URLs present in the collection, of which 81,014 could be processed. The collection contained a total of 3,214,727 permalinks from all these blogs.

TREC 2006 Blog Track participants built and trained their systems to work on this dataset. Entries were judged upon an automatic evaluation done by downloading and running, without further modification to their systems, a set of fifty test queries.

BlogVox Design

Compared to domain-specific opinion extraction, identifying opinionated documents about a randomly chosen topic from a pool of documents that are potentially unrelated to the topic is a much more difficult task. Our goal for this project was to create a system that could dynamically learn topic sensitive sentiment words to better find blog posts expressing an opinion about a specified topic. After cleaning the TREC 2006 Blog Track dataset in the pre-indexing stage, blog posts are indexed using Lucene, an open-source search engine. Given a TREC query BlogVox retrieves a set of relevant posts from the Lucene index and sends the posts to the scorers. Using a SVM BlogVox ranks each document based upon the score vector generated for the document by the set of scorers show in Figure 2. Section explains how the individual scorers, some of which employ learning algorithms, function.

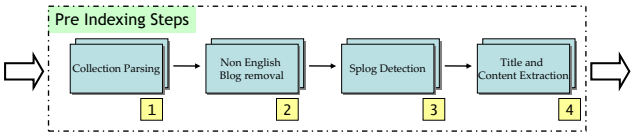

Figure 1: BlogVox text Preparation steps: 1. Parse the TREC corpus 2. Remove non English posts 3. Eliminate splogs from the collection (Section) 4. Remove spurious material from the DOM tree. (Section)

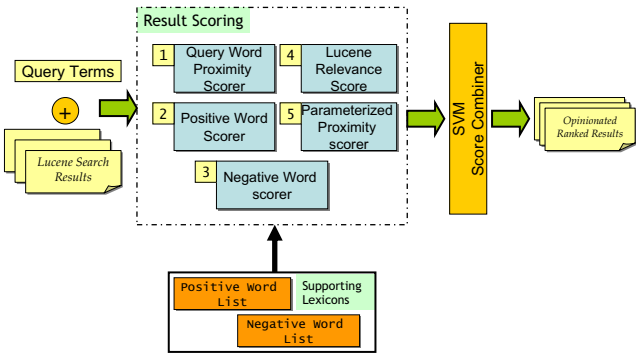

Figure 2: After relevant posts are retrieved, they are scored by various heuristics and an overall measure of oinionatedness computed by a SVM.

We tuned Lucene's scoring formula to perform document length normalization and term specific boosting [2]. Lucene

[1] http://htmlparser.sourceforge.net/

[2] http://lucene.apache.org/java/docs/scoring.html

internally constructs an inverted index of the documents by representing each document as a vector of terms. Given a query term, Lucene uses standard Term Frequency (TF) and Inverse Document Frequency (IDF) normalization to compute similarity. We used the default parameters while searching the index. However, in order to handle phrasal queries such as "United States of America" we reformulate the original query to boost the value of exact matches or proximity-based matches for the phrase.

Identifying and Removing Spam

Two kinds of spam are common in the blogosphere (i) spam blogs or splogs, and (ii) spam comments. We first discuss spam blogs, approaches on detecting them, and how they were employed for BlogVox.

Problem of Spam Blogs

Splogs are blogs created for the sole purpose of hosting ads, promoting affiliate sites (including themselves) and getting new pages indexed. Content in splogs is often auto-generated and/or plagiarized, such software sells for less than 100 dollars and now inundates the blogosphere both at ping servers (around 75% (Kolari 2005)) that monitor blog updates, and at blog search engines (around 20%, (Kolari *et al.* 2006b)) that index them. Spam comments pose an equally serious problem, where authentic blog posts feature auto-generated comments that target ranking algorithms of popular search engines. A popular spam comment filter [3] estimates the amount of spam detected to be around 93%.

Figure 3 shows a splog post indexed by a popular blog search engine. As depicted, it features content plagiarized from other blogs (ii), displays ads in high paying contexts (i), and hosts hyperlinks (iii) that create link farms. Scores of such pages now pollute the blogosphere, with new ones springing up every moment. Splogs continue to be a problem for web search engines, however they present a new set of challenges for blog analytics.

Detecting Splogs

Splogs are well understood to be a specific instance of the more general spam web-pages (Gyöngyi & Garcia-Molina 2005). Though offline graph based mechanisms like TrustRank (Gyöngyi, Garcia-Molina, & Pedersen 2004) are sufficiently effective for the Web, the blogosphere demands new techniques. The quality of blog analytics engines is judged not just by content coverage, but also by their ability to quickly index and analyze recent (non-spam) posts. This requires that fast online splog detection/filtering (Kolari, Finin, & Joshi 2006)(Salvetti & Nicolov 2006) be used prior to indexing new content.

We employ statistical models to detecting splogs as described by (Kolari *et al.* 2006b), based on supervised machine learning techniques, using content local to a page, enabling fast splog detection. These models are based solely on blog home-pages, and are based on a training set of 700 blogs and 700 splogs. Statistical models based on local blog

[3] http://akismet.com

Figure 3: A typical splog, plagiarizes content (ii), promotes other spam pages (iii), and (i) hosts high paying contextual advertisements

Feature	*Precision*	*Recall*	*F1*
words	.887	.864	.875
urls	.804	.827	.815
anchors	.854	.807	.830

Table 1: *SVM Models with 19000 word features and 10000 each of URL and anchor text features (ranked using Mutual Information) can be quite effective for splog detection.*

features perform well on spam blog detection. See Table 1. The bag-of-words based features slightly outperforms bag-of-outgoingurls (URL's tokenized on '/') and bag-of-outgoinganchors. Additional results using link based features are slightly lower that local features, but effective nonetheless. Interested readers are referred to (Kolari *et al.* 2006b) for further details. Therefore, BlogVox used only local features to detect splogs.

Comment spam

Comment spam occurs when a user posts spam inside a blog comment. Comment spam is typically managed by individual bloggers, through moderating comments and/or using comment spam detection tools (e.g. Akismet) on blogging platforms. Comment spam and splogs share a common purpose. They enable indexing new web pages, and promoting their page rank, with each such page selling online merchandise or hosting context specific advertisements. Detecting and eliminating comment spam (Mishne, Carmel, & Lempel 2005) depends largely on the quality of identifying comments on a blog post, part of which is addressed in the next section.

Identifying Post Content

Most extraneous features in blog post are links. We describe two techniques to automatically classify the links into content-links and extra-links. Content links are part of either the title or the text of the post. Extra links are not directly related to the post, but provide additional information such as: navigational links, recent entries, advertisements, and blog rolls. Differentiating the blog content from its chaff is further complicated by blog hosting services using different templates and formats. Additionally, users host their own blogs and sometimes customize existing templates to suit their needs.

Web page cleaning techniques work by detecting common structural elements from the HTML Document Object Model (DOM) (Yi & Liu 2003; Yi, Liu, & Li 2003). By mining for both frequently repeated presentational components and content in web pages, a site style tree is constructed. This tree structure can be used for data cleaning and improved feature weighting. Finding repeated structural components requires sampling many web pages from a domain. Although blogs from the same domain can share similar structural components, they can differ due to blogger customization. Our proposed technique does not require sampling and works independently on each blog permalink.

Instead of mining, we used a simple general heuristic. Intuitively extraneous links tend to be tightly grouped containing relatively small amounts of text. Note that a typical blog post has a complex DOM tree with many parts, only one of which is the content of interest in most applications.

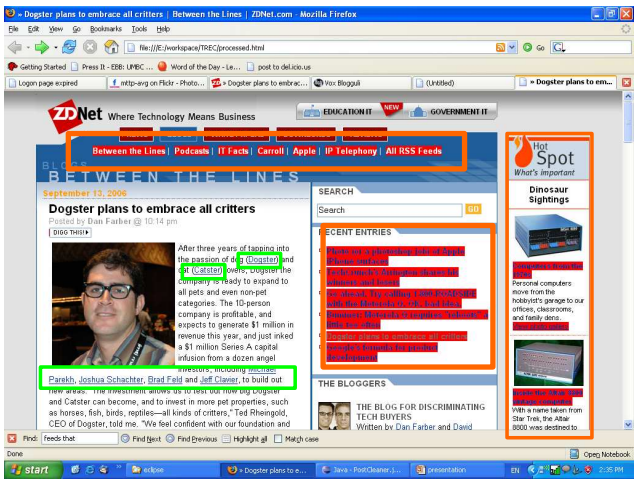

Figure 4: A typical blog post containing navigational links, recent posts, advertisements, and post content with additional links in it. Highlighted links are eliminated by the approximation heuristic.

After creating the DOM tree we traverse it attempting to eliminate any extraneous links and their corresponding anchor text, based upon the preceding and following tags. A link **a** is eliminated if another link **b** within a θ_{dist} tag distance exists such that:

Algorithm 1 Blog post cleaning heuristic

```
Nodes[] tags = tags in the order of the depth first traversal of the DOM tree
for all i such that 0 ≤ i ≤ |tags| do
  dist = nearestLinkTag(tags, i);
  if dist ≤ θ_dist then
    eliminate tags[i]
  end if
end for
```

Procedure 2 int nearestLinkTag(Nodes[] tags, int pos)

```
minDist = |tags|
textNodes = 0
textLength = 0
title = false;
for all j such that pos − θ_dist ≤ j ≤ pos + θ_dist do
  node = tags[j]
  if j = 0||j = pos||j > (|tags| − 1) then
    continue
  end if
  if node instanceOf TextNode then
    textNodes++;
    textLength += node.getTextLength();
  end if
  dist = |pos − j|
  if node instanceOf LinkNode && dist < minDist then
    minDist = dist
  end if
  if node instanceOf TitleNode then
    title = true
  end if
end for
ratio = textLength / textCount
if ratio > α_avgText||title == true then
  return tags.size()
end if
return minDist
```

- No title tags (H1, H2...) exist in a θ_{dist} tag window of **a**.
- Average length of the text bearing nodes between **a** and **b** is less than some threshold.
- **b** is the nearest link node to **a**.

The average text ratio between the links, $\alpha_{avgText}$ was heuristically set to 120 characters and a window size, θ_{dist} of 10 tags was chosen. The Algorithm 1 provides a detailed description of this heuristic.

Next we present a machine learning approach to the link classification problem. From a large collection of blog posts, a random sample of 125 posts was selected. A human evaluator judged a subset of links (approximately 400) from these posts. The links were manually tagged either content-links or extra-links. Each link was associated with a set of features. Table 2 summarizes the main features used. Using

ID	Features
1	Previous Node
2	Next Node
3	Parent Node
4	Previous N Tags
5	Next N Tags
6	Sibling Nodes
7	Child Nodes
8	Depth in DOM Tree
9	Char offset from page start
10	links outside the blog?
11	Anchor text words
12	Previous N words
13	Next N words

Table 2: Features used for training an SVM for classifying links as content links and extra links.

Method	*Precision*	*Recall*	*F1*
baseline heuristic	0.83	0.87	0.849
svm cleaner (tag features)	0.79	0.78	0.784
svm cleaner (all features)	0.86	0.94	0.898

Table 3: *Data cleaning with DOM features on a training set of 400 HTML Links.*

this feature set an SVM model was trained [4] to recognize links to eliminate. The first set of features (1-7) was based on the tag information. The next set of features (8-9) was based on position information and the final set of features (10-13) consisted of word-based features. Using features (1-7) yields a precision of 79.4% and recall of 78.39%, using all our features (1-13) yields a precision of 86.25% and recall of 94.31% under 10-fold cross validation.

We compared the original baseline heuristic against human evaluators. The average accuracy for the baseline heuristic is about 83% with a recall of 87%.

Scoring

To improve the quality of opinion extraction results, it is important to identify the title and content of the blog post because the scoring functions and the Lucene indexing engine can not differentiate between text present in the links and sidebars from text present in content of the blog post. Thus, a post which has a link to a recent post titled 'Why I love my iPod' would be retrieved as an opinionated post even if the post content is about some other topic. This observation lead to the development of our first scorers.

As shown in figure 2, a number of heuristics are employed to score the results based on the likelihood that it contains an opinion about the query terms. These scorers work by using both document level and individual sentence level features. Some of the scoring heuristics were supported by a hand-crafted list of 915 generic postive and 2712 negative sentiment words.

[4] `http://svmlight.joachims.org/`

The following is a brief description of each scoring function:

Query Word Proximity Scorer finds the average number of sentiment terms occurring in the *vicinity* of the query terms using a window size of 15 words before and after the query terms. If the query is a phrasal query, the presence of sentiment terms around the query was weighted twice.

Parametrized Proximity Scorer was similar to the Query Word Proximity Scorer. However, we used a much smaller dictionary which was divided into two subsets: highly polar sentiment words, and the relatively less polar words. We used parameters to specify the window of text to search for sentiment words (five and fifteen), and to boost sentiment terms around phrase queries (one and three). This resulted in a total of eight scorers.

Positive and Negative Scorers counted the number of sentiment words (positive, negative) in the entire post.

Lucene Relevance Score was used to find how closely the post matches the query terms.

We also experimented with other scoring functions, such as adjective word count scorer. This scorer used an NLP tool to extract the adjectives around the query terms. However, this tool did not perform well mainly due to the noisy and ungrammatical bsentences present in blogs.

Once the results were scored by these scoring modules, we used a meta-learning approach to combine the scores using SVMs. Our SVMs were trained using a set of 670 samples of which 238 were positive (showed a sentiment) and the rest were negative. Using the polynomial kernel with degree gave the best results with precision of 80% and recall of 30%. The model was trained to predict the probability of a document expressing opinion. This value was then combined with the Lucene relevance score to produce final runs.

Evaluation

The opinion extraction system provides a testbed application for which we evaluate different data cleaning methods. There are three criteria for evaluation: i) improvements in opinion extraction task with and without data cleaning ii) performance evaluation for splog detection iii) performance of the post content identification.

Splog Detection Evaluation

Our automated splog detection technique identified 13,542 blogs as splogs. This accounts for about 16% of the identified homepages. The total number of splog permalinks is 543,086 or around 16% of the collection, which is very close to the 15.8% explicitly included by NIST. While the actual list of splogs are not available for comparison, the current estimate seem to be close. To prevent the possibility of splogs skewing our results permalinks associated with splogs were not indexed.

Given a search query, we would like to estimate the impact splogs have on search result precision. Figure 5 shows the distribution of splogs across the 50 TREC queries. The quantity of splogs present varies across the queries since splogs are query dependent. For example, the topmost spammed query terms were 'cholesterol' and 'hybrid cars'.

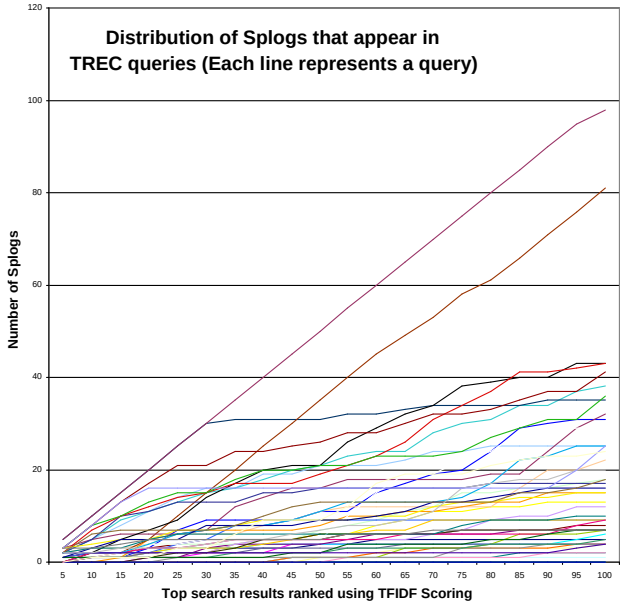

Figure 5: The number of splogs in the top x results for 50 TREC queries. Top splog queries include “cholesterol” and “hybrid cars”

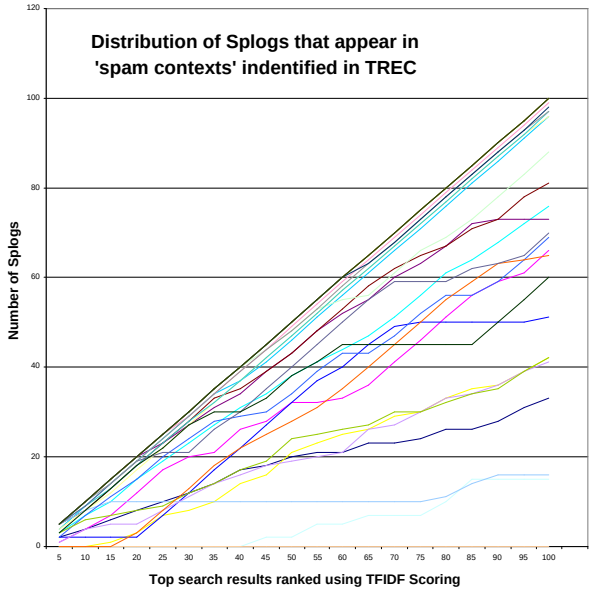

Figure 6: The number of splogs in the top x results of the TREC collection for 28 highly spammed query terms. Top splog queries include ’pregnancy’, ’insurance’, ’discount’

Such queries attract a target market, which advertiser can exploit.

The description of the TREC data (Macdonald & Ounis 2006) provides an analysis of the posts from splogs that were added to the collection. Top informative terms include ‘insurance’, ‘weight’, ‘credit’ and such. Figure 6 shows the distribution of splogs identified by our system across such spam terms. In stark contrast from Figure 5 there is a very high percentage of splogs in the top 100 results.

Post Cleaning Evaluation

In BlogVox data cleaning improved results for opinion extraction. Figure 7 highlights the significance of identifying and removing extraneous content from blog posts. For 50 TREC queries, we fetched the first 500 matches from a Lucene index and used the baseline data cleaning heuristic. Some documents were selected only due to the presence of query terms in sidebars. Sometimes these are links to recent posts containing the query terms, but can often be links to advertisements, reading lists or link rolls, etc. Reducing the impact of sidebar on opinion rank through link elimination or feature weighing can improve search results.

Table 3 shows the performance of the baseline heuristic and the SVM based data cleaner on a hand-tagged set of 400 links. The SVM model outperforms the baseline heuristic. The current data cleaning approach works by making a decision at the individual HTML tag level; we are currently working on automatically identifying the DOM subtrees that correspond to the sidebar elements.

Trec Submissions

The core BlogVox system produces results with two measures. The first is a relevance score ranging from 0.0 to 1.0, which is the value returned by the underlying Lucene query system. The second was a measure of opinionatedness, which was a real number greater than 0.0. We produced the sim numbers for each of the runs from a weighted average of the two numbers after normalizing them using the standard Z-normalization technique.

The baseline run was exectuted on the uncleaned dataset using a selection of what we anticipated to be the seven best scorer features and with an equal weighting for relevance and opinionatedness. This run was also the best performing run amongst our official runs. Runs two through five were made on the semi-cleaned dataset and using a larger set of eleven scorer features. After normalizing the result scores, we used weights of (1,1), (1,2), (1,3) and (2,1).

Figure 8 shows the results from the TREC submissions for opinion retrieval. Figure 9 shows the results for the topic relevance. The Mean Average Precision (MAP) for opinion retrieval of the original TREC submissions was 0.0764 and the R-Prec was around 0.1307. The MAP for topic relevance was about 0.1288 with an R-Prec of 0.1805. After inspection of the code, it appeared that this may have been due to a minor bug in the original code that was used for the official run. Upon correcting this and re-executing the run, we found that the MAP for opinion task was about 0.128 and for retrieval was about 0.1928. A final run was performed by running the queries against an index recreated by clean-

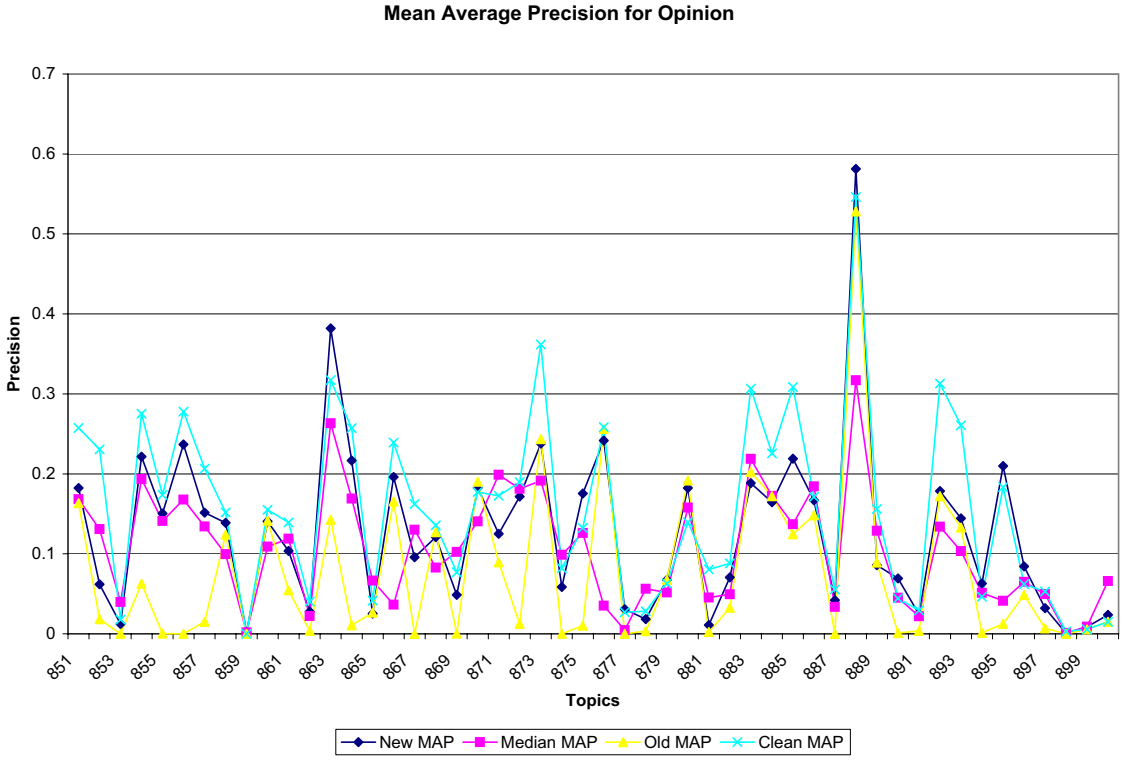

Figure 8: Mean average precision (for opinion) of original TREC submission UABas11 ,updated runs and clean index runs.

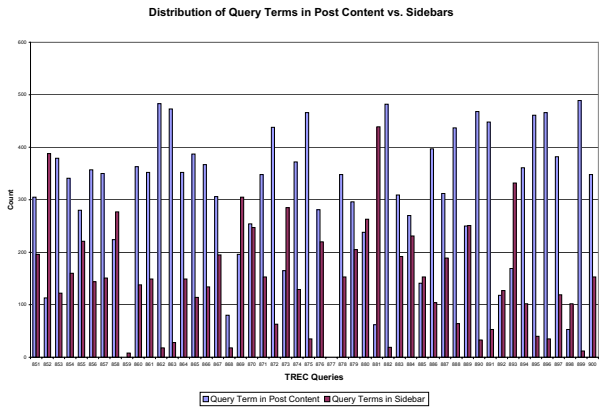

Figure 7: Documents containing query terms in the post title or content vs. exclusively in the sidebars, for 50 TREC queries, using 500 results fetched from the Lucene index.

run	opinion		topic relevance	
	map	r-prec	map	r-prec
UABas11	**0.0764**	**0.1307**	**0.1288**	**0.1805**
UAEx11	0.0586	0.0971	0.0994	0.1367
UAEx12	0.0582	0.0934	0.0985	0.1355
UAEx13	0.0581	0.0923	0.0978	0.1360
UAEx21	0.0590	0.0962	0.0998	0.1366
Corrected	**0.1275**	**0.202**	**0.1928**	**0.2858**
Cleaned	**0.1548**	**0.2388**	**0.2268**	**0.3272**

Table 4: The results for the opinion and topic relevance performance of different runs

ing all the posts using heuristics described in Section . Table 4 summarizes the results obtained. We find that cleaning significantly improved both opinion and retrieval scores of our system. Figure 11 compares the precision recall curves

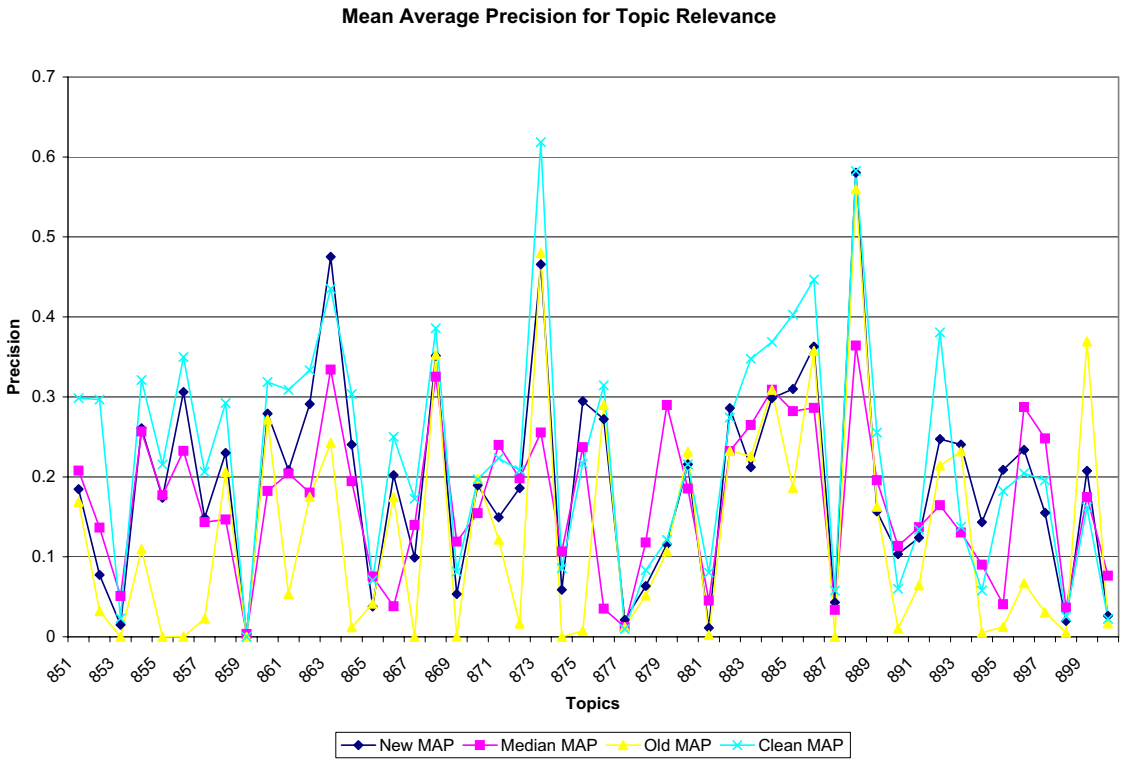

Figure 9: Mean average Precision (for topic relevance) of original TREC submission UABas11, updated runs and clean index runs.

for these these runs.

We think that the retrieval performance could be improved by using the following approaches: use of query expansion modules, applying relevance feedback and using the description and narrative fields from the TREC queries to formulate the final Lucene query.

Discussion

For TREC runs, we used an index on blog posts that had not been cleaned for all of the runs. For run one we evaluated these uncleaned posts using a complement of seven heuristics. For runs two through five, we retrieved a fixed number of post ids using the index of uncleaned data and then cleaned the resulting posts "on the fly". A larger set of eleven heuristic scoring functions was used for these runs. After cleaning a post, we did a heuristic check to ensure that at least some of the query terms remained. If not, the post was discarded. We believe that this ad hoc approach significantly lowered our precision scores for these runs due to at least three reasons. First, the relevance scores were computed by Lucene on the uncleaned posts and were not accurate for the cleaned versions since the term frequencies for both the collection and for each document were altered. Second, discarding many of the posts after the cleaning reduced the number of available results, already low due to the impending deadline. Finally, the cleaned posts were in many cases likely to be less relevant that their scores would indicate due to the removal of query words.

Manual inspection of some of the results showed that there were a number of matches that were due to the presence of the query terms in extraneous links. In order to verify the effectiveness of cleaning we created a new index using only the cleaned versions of the posts. We find that using this cleaner index improved not only retrieval results but also effective mean average precision for opinion retrieval. As can be observed from Figure 10, in almost all the cases the mean average precision for the runs on cleaned data outperform those on unclean data. The queries for which data

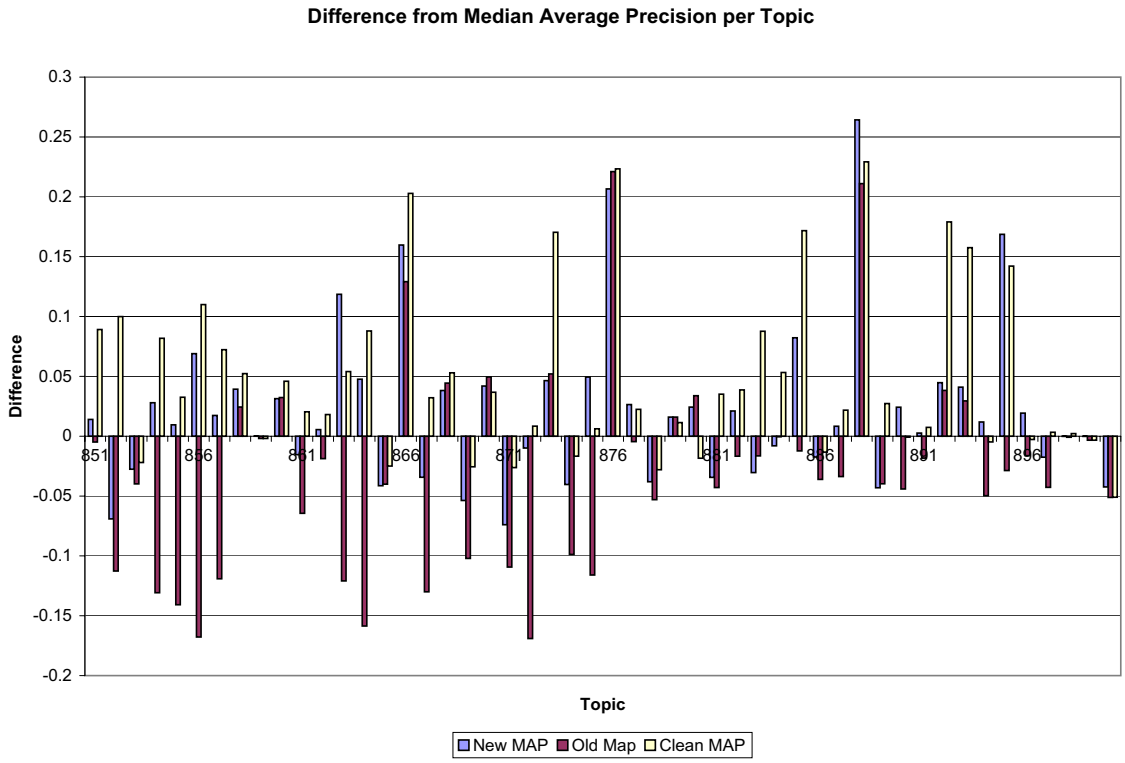

Figure 10: Difference of MAP from Median for original TREC submission UABas11, updated runs and clean index runs.

cleaning made a significant improvement were “larry summers”, “bruce bartlett”, “Fox News Report” and “zyrtec”. Comparing these with Figure 7 indicates that these were also queries that contained a higher number of matches that had the terms exclusively in the sidebar. On the other hand for queries like ‘audi’, ‘oprah’ and ‘colbert report’ the cleaned runs had a lower precision possibly due to the strict thresholds for cleaning.

Conclusion

We developed the BlogVox system as an opinion retrieval system for blog posts as part of the 2006 TREC Blog Track. This task requires processing an ad hoc queries representing topics and retrieving posts that express an opinion about them. Our initial experiments with the blog post collection revealed two problems: the presence of spam blogs and the large amounts of extra, non-content text in each posts.

We identified posts from spam blogs using a machine-learning based approach and eliminated them from the collection. The remaining posts were “cleaned” before being indexed to eliminate extraneous text associated with navigation links, blog-rolls, link-rolls, advertisements and sidebars. After retrieving posts relevant to a topic query, the system applies a set of scoring modules to each producing a vector of features estimating the likelihood that a post expresses an opinion about the topic. These are combined using an SVM-based system and integrated with the overall relevancy score to rank the results.

Our evaluation of the BlogVox results showed that both splog elimination and post cleaning significantly increased the performance of the system. The overall performance as measured by the *mean average precision* and *R-precision* scores showed that the system worked well on most of the fifty test queries. We believe that the system can be improved by increasing the accuracy of the post-cleaning and refining the opinion scorers.

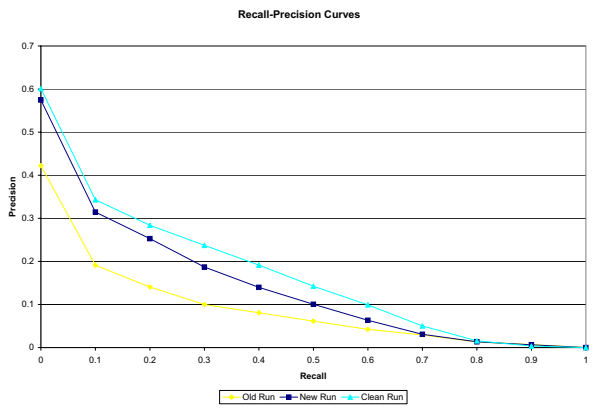

Figure 11: Precision Recall curves for original TREC submission UABas11, updated runs and clean index runs.

References

Dave, K.; Lawrence, S.; and Pennock, D. M. 2003. Mining the peanut gallery: opinion extraction and semantic classification of product reviews. In *WWW*, 519–528.

Gilad Mishne, N. G. 2006. Predicting movie sales from blogger sentiment. In *AAAI 2006 Spring Symposium on Computational Approaches to Analysing Weblogs (AAAI-CAAW 2006)*.

Glance, N. S.; Hurst, M.; Nigam, K.; Siegler, M.; Stockton, R.; and Tomokiyo, T. 2005. Deriving marketing intelligence from online discussion. In *KDD*, 419–428.

Gyöngyi, Z., and Garcia-Molina, H. 2005. Web spam taxonomy. In *First International Workshop on Adversarial Information Retrieval on the Web*.

Gyöngyi, Z.; Garcia-Molina, H.; and Pedersen, J. 2004. Combating web spam with TrustRank. In *Proceedings of the 30th International Conference on Very Large Databases*, 576–587. Morgan Kaufmann.

Hatcher, E., and Gospodnetić, O. 2004. *Lucene in Action*. Manning Publications Co.

Java, A.; Kolari, P.; Finin, T.; Mayfield, J.; Joshi, A.; and Martineau, J. 2007. BlogVox: Separating Blog Wheat from Blog Chaff. In *Proceedings of the Workshop on Analytics for Noisy Unstructured Text Data, 20th International Joint Conference on Artificial Intelligence (IJCAI-2007)*.

Kolari, P.; Java, A.; Finin, T.; Mayfield, J.; Joshi, A.; and Martineau, J. 2006a. Blog Track Open Task: Spam Blog Classification. Technical report. TREC 2006 Blog Track.

Kolari, P.; Java, A.; Finin, T.; Oates, T.; and Joshi, A. 2006b. Detecting Spam Blogs: A Machine Learning Approach. In *Proceedings of the 21st National Conference on Artificial Intelligence (AAAI 2006)*.

Kolari, P.; Finin, T.; and Joshi, A. 2006. SVMs for the Blogosphere: Blog Identification and Splog Detection. In *Proceedings of the AAAI Spring Symposium on Computational Approaches to Analysing Weblogs*. AAAI Press.

Kolari, P.; Java, A.; and Finin, T. 2006. Characterizing the Splogosphere. In *Proceedings of the 3rd Annual Workshop on Weblogging Ecosystem: Aggregation, Analysis and Dynamics, 15th World Wid Web Conference*.

Kolari, P. 2005. Welcome to the splogosphere: 75% of new pings are spings(splogs). [Online; accessed 22-December-2005; `http://ebiquity.umbc.edu/blogger/?p=429`].

Liu, B.; Hu, M.; and Cheng, J. 2005. Opinion observer: analyzing and comparing opinions on the web. In *WWW '05: Proceedings of the 14th international conference on World Wide Web*, 342–351. New York, NY, USA: ACM Press.

Macdonald, C., and Ounis, I. 2006. The trec blogs06 collection: Creating and analyzing a blog test collection. Technical report. Department of Computer Science, University of Glasgow Tech Report TR-2006-224.

Mishne, G.; Carmel, D.; and Lempel, R. 2005. Blocking blog spam with language model disagreement. In *AIRWeb '05 - 1st International Workshop on Adversarial Information Retrieval on the Web, at WWW 2005*.

Nigam, K., and Hurst, M. 2004. Towards a robust metric of opinion. In *Exploring Attitude and Affect in Text: Theories and Applications, AAAI-EAAT 2004*.

Pang, B.; Lee, L.; and Vaithyanathan, S. 2002. Thumbs up? sentiment classification using machine learning techniques. In *Proceedings of EMNLP 2002*.

Salvetti, F., and Nicolov, N. 2006. Weblog classification for fast splog filtering: A url language model segmentation approach. In *Proceedings of the Human Language Technology Conference of the NAACL, Companion Volume: Short Papers*, 137–140. New York City, USA: Association for Computational Linguistics.

Yi, L., and Liu, B. 2003. Web page cleaning for web mining through feature weighting. In *Proceedings of Eighteenth International Joint Conference on Artificial Intelligence, IJCAI-03*.

Yi, L.; Liu, B.; and Li, X. 2003. Eliminating noisy information in web pages for data mining. In *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD-2003*.