# Investigating Learning Approaches for Blog Post Opinion Retrieval

Shima Gerani, Mark Carman, Fabio Crestani

Faculty of Informatics, University of Lugano, Lugano, Switzerland
{shima.gerani,markjcarman,fabio.crestani}@lu.unisi.ch

**Abstract.** Blog post opinion retrieval is the problem of identifying posts which express an opinion about a particular topic. Usually the problem is solved using a 3 step process in which relevant posts are first retrieved, then opinion scores are generated for each document, and finally the opinion and relevance scores are combined to produce a single ranking. In this paper, we study the effectiveness of classification and rank learning techniques for solving the blog post opinion retrieval problem. We have chosen not to rely on external lexicons of opinionated terms, but investigate to what extent the list of opinionated terms can be mined from the same corpus of relevance/opionion assessments that are used to train the retrieval system. We compare popular feature selection methods such as the *weighted log likelihood ratio* and *mutual information* for use both in selecting terms for training an opinionated document classifier and also as term weights for generating simpler (not learning based) aggregate opinion scores for documents. We thereby analyze what performance gains result from learning in the opinion detection phase. Furthermore we compare different learning and not learning based methods for combining relevance and opinion information in order to generate a ranked list of opinionated posts, thereby investigating the effect of learning on the ranking phase.

**Key words:** Opinion Retrieval, Blog Post, Learning Methods

## 1 Introduction

Unlike traditional Web pages, which tend to contain primarily factual information, blog posts often contain opinionated content expressing the views of blog authors on a variety of topics such as political issues, product launches, fashion trends, health services, and so on. This wealth of opinionated content can be very useful for those whose job it is to gauge users' thoughts and perceptions about different concepts, including marketing departments, political pundits, anthropology researchers, etc.

Mining this opinionated content so as to separate the opinionated and relevant content from the "background noise" that is present in the blogosphere (the totality of all blogs) constitutes an interesting and challenging research problem. We view this problem from an Information Retrieval (IR) perspective and investigate the specific task of *retrieving blog posts that express an opinion about*

*a particular topic*, where the topic is described using a short keyword query. We refer to this task as *blog post opinion retrieval*. The objective of the task is to return a list of posts that have been ranked by the likelihood that the user is expressing an opinion about the topic. Various techniques are used to estimate this probability. We discuss our approach along with related approaches below.

The opinion retrieval problem is usually tackled in 3 stages. In the first phase, a standard information retrieval system is used to rank posts by relevance, and the highest ranking posts are selected as a candidate set of documents. In the second phase, opinion scores are calculated for each candidate document, and in the third phase, the opinion and relevance scores are combined so as to produce a single ranking. A variety of different techniques have been employed in previous studies for realizing each of these phases and it is not clear from these studies to what extent learning (either in the opinion scoring phase or the ranking phase) or the feature selection method or the use of external datasets of opinionated terms/sentences/documents is responsible for the performance improvements demonstrated. Our intention in this paper is to remedy this situation by focusing our investigation on the use of learning in the different phases of opinion retrieval and comparing learning techniques directly with baseline (non learning) techniques. We limit our study to use only data available in opinion retrieval relevance assessments in order to focus our investigation on what can be learnt automatically from the assessment data.

The paper is structured as follows. We first discuss some recent work on blog post opinion retrieval. We then define formally the problem and describe different learning approaches for solving it. Finally, we describe the empirical comparison we performed between different techniques for learning in opinion retrieval.

## 2 Related Work

The opinion retrieval task was introduced as a main task in TREC 2006 and continued in TREC 2007 and 2008. Some 20 groups participated in 2007, with most following a 3 step algorithm. In the fist step, traditional IR is used to find topic relevant documents. In the second step, an opinion score or opinion rank is given to the documents which are the results of the IR step. In the last (third) step a ranking method is used to rank documents according to their opinionatedness about the query. We can classify the approaches according to different aspects including the opinion word list used, the term weighting methods, the level of classification, the classification technique and the training data collection used. We now discuss some of the better performing and more innovative approaches seen at TREC 2007 as far as they are related to this paper. We refer the interested reader to [1].

W. Zhang et al. [2] followed the 3 steps method described above. In the opinion scoring step, they decompose the documents which are retrieved in the IR step, into sentences. Then the classifier labels each sentence as opinionated or not and gives an opinion score to the sentence. The retrieved document is

then labeled as opinionated, if it contains at least one opinionated sentence. They used resources other than the BLOG06 data for training the classifier, including subjective documents collected from review web sites like epinion.com. They also submitted queries to a Web search engine containing opinion indicator phrases such as "reviews", "don't like", etc. and considered the top retrieved documents as opinionated documents. They used the same approach for collecting not opinionated documents from dictionary sites like Wikipedia, and by submitting queries to a search engine for documents that did not contain the terms "review", "comments", etc. They used uni-grams and bi-grams in the collected documents as the features for the SVM classifier, and used the Chi-Square method to reduce the number of features. In step 3, to ensure that opinions are related to the query, they used a linear combination of the relevance and opinion scores weighting both components equally.

Yang et al. [3] considered multiple sources of evidence in their opinion retrieval step by combining scores from multiple opinion detection modules. The first module identified opinions based on common opinion terms. A lexicon is built by identifying terms that occur frequently in opinionated blog posts and infrequently in non opinionated blog posts. The resulting terms were manually filtered and assigned an opinion weight. A second module used the standard set of Wilson subjective terms as a collection independent source of evidence. A third module looked at the low frequency terms as opinion evidence, since in blogs opinion is often expressed in a creative and non standard form. Pronouns such as "I" and "you" appear very frequently in opinionated posts, so a fourth module recognized certain n-grams which begin or end with these terms. The last module recognized acronyms, specifically a subset of Netlingo's chat acronyms and text message shorthands. The scores from each module are combined using a weighted summation, where the weights are estimated from training data. In step 3, the linear combination of relevance and opinion score is used to score and rank documents.

Q. Zhang et al. [4] used the CME (Classification by Minimizing Error) method to assign an opinion score to each sentence of a blog post. They then defined some features based on subjectivity and relevance of all sentences in the post. An SVM classifier is used to assign an opinion score to each document based on the values of the defined features. A subjectivity data set of movie-review data containing 5000 subjective and 5000 objective sentences was used for training the CME classifier, but for the SVM and classifying documents they just used the BLOG06 collection. The blog posts were ranked by the final score that was calculated as the relevance score times the opinion score.

In the above mentioned work, various learning techniques have been employed in combination with external sources of training data and opinionated wordlists, making it difficult to draw conclusions as to the degree to which learning is responsible for the performance gains shown. In this study we limit ourselves to use only relevance assessments for training thereby investigating the performance gain that are due to learning alone. In order to avoid any external lexicons for opinionated words we use an approach similar to Amati et al. [5] who proposed

automatically generating an opinion word list from the TREC 2006 relevance data. Terms are first selected according to a method that is similar to Weighted Log-Likelihood Ratio [6, 7] and then refined according to their document frequency. These terms are then submitted as a query to a search engine (using a parameter free retrieval model) to get the opinion score for the relevant documents. Ranking is done in two steps. In the first step, documents are ranked according to opinion score divided by the content rank, and in the second step, ranking is obtained by the content score divided by the ranking of the previous step.

## 3 Problem Definition

The *blog post opinion retrieval* problem is the problem of developing an effective retrieval function that ranks blog posts according to the likelihood that they are *expressing an opinion about a particular topic*. To define this problem more formally, we introduce some notation. Assume that we have a set of labeled training documents, denoted $D$, and a set of training queries, denoted $q_1, ..., q_n$. For each query $q_i$ we have a set of assessed documents $A_i \subset D$, a subset of which were considered relevant to the query $R_i \subset A_i$, and a subset of the relevant documents have also been marked as opinionated documents $O_i \subset R_i$. We now define the blog post opinion retrieval problem as the problem of learning a retrieval function $f : Q \times D \rightarrow \mathbb{R}$ such that the ranking of documents for each query $q_i \in Q$ is optimal (on average) with respect to a particular performance measure over rankings, $\mathcal{M} : \mathbb{B}^l \rightarrow \mathbb{R}$, (where $l \leq |D|$ is the maximum length of a ranking)[1].

One commonly employed performance measure over rankings is the Average Precision (AP) [8]. In the case of opinion retrieval, AP can be written as:

$$AP_i = \frac{1}{|O_i|} \sum_{d \in O_i} P_i(rank(q_i, d)) \tag{1}$$

where $rank(q_i, d)$ is the rank of document $d$ according to the function $f$ for the query $q_i$ and $P_i$ is the precision at that rank: $P_i(r) = \#\{d \in O_i | rank(q_i, d) \leq r\}/r$. Over a set of queries, the Mean Average Precision (MAP) is then calculated as $MAP = \frac{1}{n} \sum_{i=1}^{n} AP_i$. We will use MAP as the performance measure in the experiments section of this paper.

Note that we have deliberately defined the opinion retrieval problem to include only a collection of documents, a set of queries, corresponding relevance and opinion judgements, and an evaluation function $\langle D, [q_i, A_i, R_i, O_i]^n, \mathcal{M} \rangle$. Our aim is to understand how well a learning system can perform at opinion retrieval without needing to recourse to external sources of information. We argue that these other sources of information, such as opinionated term lists, product review corpora and so on, will likely be suboptimal for (and increase the complexity of) the opinion retrieval task due to differences in the application domain.

[1] $\mathbb{B}$ is the set of boolean values(true, false) and $\mathbb{R}$ is the set of real values

## 4 Generating a Blog Post Opinion Score

Obviously if a blog post does not mention a particular topic then the author cannot be expressing an opinion about it. Thus the first step in most opinion retrieval systems is to rank posts by their relevance to the query. Once we have a set of relevant documents, the problem becomes that of estimating a score for the "opinionatedness" of each document. We investigate non-learning (baseline) and learning approaches for doing this. In the first (non-learning) approach we calculate an opinion score for each term in the document and then combine the score over all terms in the document. In the second we train a classification system to distinguish between opinionated and non-opinionated posts using the opinionatedness of each term for feature selection. We then use the confidence of the classifier as an opinion score for the document.

### 4.1 Discovering Opinionated Terms

Before defining different opinion scores for terms we introduce some more notation. Let $O = \cup_i O_i$ be the set of all opinionated documents in our training set and $R = \cup_i R_i$ be the set of all relevant documents, (note that $O \subset R$). The relative frequency of a particular term $t$ in the set $O$ is denoted $p(t|O)$ and calculated as:

$$p(t|O) = \frac{\sum_{d \in O} c(t,d)}{\sum_{d \in O} |d|} \tag{2}$$

where $c(t,d)$ denotes the number of occurrences term $t$ in document $d$ and $|d|$ denotes the total number of words in the document. The relative frequency of terms in the relevant set $p(t|R)$ is defined analogously.

One simple score for comparing the likely "opinionatedness" of terms is the ratio of relative frequencies in the opinionated and relevant sets, which we call the Likelihood Ratio:

$$opinion_{LR}(t) = \frac{p(t|O)}{p(t|R)} \tag{3}$$

A second formula is the technique proposed by Amati et al. [5], which is a slight variation on the standard Weighted Log-Likelihood Ratio [6, 7] feature selection technique, (where the relevant set $R$ replaces the "not-opinionated" set $\bar{O} = R/O$).

$$opinion_{WLLR}(t) = p(t|O) \log \frac{p(t|O)}{p(t|R)} \tag{4}$$

Note that the summation over all terms of the opinion score gives the well-known Kullback-Leibler divergence [9] between the opinionated document set and the relevant document set. This measure quantifies the dissimilarity between the two sets of documents. Terms which cause high divergence are therefore good indicators of opinionatedness.

We investigate also another formula for estimating the opinionatedness of a term, based on the concept of Mutual Information (MI) [9]. Mutual Information, also sometimes referred to as Information Gain, is often used for feature selection in machine learning [10].

$$opinion_{MI}(t) = \sum_{\mathcal{T} \in \{t \in d, t \notin d\}} \sum_{\mathcal{O} \in \{d \in O, d \notin O\}} p(\mathcal{T}, \mathcal{O}) \log \frac{p(\mathcal{T}, \mathcal{O})}{p(\mathcal{T}), p(\mathcal{O})} \tag{5}$$

Here we calculate the joint and marginal probabilities are calculated using the document frequency in the sets $O$, $\bar{O}$ and $R$ as follows:

$$\begin{gathered} p(t \in d, d \in O) = df(t, O)/|R| \\ p(t \in d, d \notin O) = df(t, \bar{O})/|R| \\ p(t \in d) = df(t, R)/|R| \\ p(d \in O) = |O|/|R| \end{gathered}$$

where $df(t, O) = \#\{d \in O | t \in d\}$ is the number of documents in $O$ containing the term $t$.

A related feature selection measure, also based on document frequencies, is the $\chi^2$ score:

$$opinion_{\chi^2}(t) = |D| \frac{[p(t \in d, d \in O)p(t \notin d, d \notin O) - p(t \in d, d \notin O)p(t \notin d, d \in O)]^2}{p(t \in d)p(t \notin d)p(d \in O)p(d \notin O)} \tag{6}$$

### 4.2 Document Opinion Scores

In order to calculate an opinion score for an entire document, we can simply calculate the average opinion score over all the words in the document:

$$opinion_{avg}(d) = \sum_{t \in d} opinion(t)p(t|d) \tag{7}$$

where $p(t|d) = c(t, d)/|d|$ is the relative frequency of term $t$ in document $d$.

Alternatively, as stated previously, we can train a classification system and in particular a Support Vector Machine (SVM) to recognize opinionated documents. Training data in this case will be the set of all relevant documents $R$, with the set of opinionated documents $O$ being positive examples of the class. We can then use the confidence of the classifier (i.e. the distance from the hyperplane) as the opinion score for each document[2]. The per term opinion score is used in this case only for feature selection, with the $m$ highest scoring terms being selected as features for the classifier. We use the relative frequency of each of these terms in the document as the feature weight:

[2] Note that the classifier is *not* being used to perform regression estimation as the training examples have categorical class labels $c \in \{0, 1\}$ as opposed to a range of values, e.g. $c \in [0, 1]$.

$$opinion_{SVM}(d) = f_{SVM}(p(t_1|d), ..., p(t_m|d)) \quad (8)$$

where the function $f_{SVM}()$ is the output (confidence) of the trained SVM for a particular document and $m$ is the size of the feature set (vocabulary).

## 5 Combining Relevance and Opinion Scores

In order to combine the relevance score with the opinion score we again investigate both learning and non-learning approaches.

### 5.1 Simple Combinations

We investigate simple algorithms for combining the two scores. Our first approach is simply to rerank the top N results according to the opinion score:

$$basicScore(q_i, d) = opinion(d) \quad (9)$$

The second approach is to simply weight the opinion score by the relevance score for the document:

$$product(q_i, d) = relevance(q_i, d) * opinion(d) \quad (10)$$

where $relevance(q_i, d)$ is the relevance score that was given to document $d$ for query $q_i$. This score could be a simple content-based retrieval function like BM25 [11] and Language Modeling [12], or it could be combined using a Rank Learning approach [13] with additional information including the content of incoming hyperlinks and tag data from social bookmarking websites.

A third approach is to take the rankings produced by the opinion score and the relevance score and merge them. The simplest way to do that is to use Borda Fuse [14], which is simply to calculate a score based on the sum of the individual ranks.

$$borda(q_i, d) = rank_{opin}(q_i, d) + rank_{rel}(q_i, d) \quad (11)$$

where $rank_{opin}(q_i, d)$ and $rank_{rel}(q_i, d)$ is the rank of document $d$ in the opinion and relevance rankings for query $q_i$ respectively. Whenever there are ties in the ranking (i.e. two or more documents have the same opinion or relevance score), the rank is the mean of the tied rankings (e.g. two documents tied at position 10 would both be given the rank 10.5).

### 5.2 Using a Rank Learner

Our final approach to ranking blog posts by their opinionatedness relies on a learning framework. In this case we train a Learning to Rank system [13] to take both the relevance score and the opinion score for each document into account when producing an output ranking. Training data in this case is the set

of assessed documents $A_i$ for each query $q_i$, where positive examples are those documents judged to be both relevant and opinionated, $O_i$.

The advantage of the learning approach is that we do not need to decide explicitly how best to combine the forms of evidence, but can rely on historical data for fine tuning the retrieval.

## 6 Experiments

In this section we discuss the experiments we conducted in order to determine the usefulness of different learning frameworks for performing blog post opinion retrieval.

### 6.1 Test Collection & Methodology

We use the 2008 TREC blog track opinion retrieval collection for our experiments in this paper. The collection contains 3.2 million documents built during a crawl in December 2005. The dataset also contains a set of 150 topics (queries) and corresponding relevance and opinion judgements.

In order to facilitate direct comparison between systems, five relevance retrieval baselines are provided by the TREC organizers, selected from the best performing retrieval runs. For our experiments we chose to use baseline 4 because it demonstrated the best performance in terms of both relevance retrieval and opinion retrieval with MAP values of 0.4776 and 0.3543 respectively.

Note that the baseline TREC runs are purely relevance retrieval and not opinion retrieval systems. They score relatively well at opinion retrieval simply because the majority of blog posts that are relevant to a topic are also express an opinion about it. When evaluating opinion retrieval systems therefore, one must compare the MAP score of the opinion retrieval system with the MAP (for opinionated posts) of the baseline system.

In order to avoid overfitting the data we perform 10 fold cross-validation on the 150 query dataset. Thus for each fold we had 135 queries in the training set and 15 queries in the test set. The training and test data was kept separate *in all phases* of the experiment. Training data was used in the feature selection phase in order to determine highly opinionated terms, in the classification phase to learn a model for detecting opinionated documents, and in the ranking phase to train the rank learner to best combine opinion and relevance scores for each document.

### 6.2 Discovering Opinionated Terms

In this section we investigate the different techniques discussed in section 4.1 for discovering terms which are good indicators of opinionated content. Table 1 shows a list of the most opinionated terms in the collection according to the different weighting schemes outlined in section 4.1. It would appear from the list that the Likelihood Ratio (LR) is not a useful weighting scheme. The Weighted

| Rank | LR | WLLR | MI | $\chi^2$ |
|---|---|---|---|---|
| 1 | jeffgannon | 8217 | just | just |
| 2 | ruberri | who | think | think |
| 3 | opotho | peopl | know | know |
| 4 | mtlouie | muslim | like | like |
| 5 | mcclatchi | like | don | don |
| 6 | spaeth | think | ve | who |
| 7 | alschul | just | who | ve |
| 8 | tomwright | don | love | sai |
| 9 | hairlessmonkeydk | 2006 | sai | love |
| 10 | snoopteddi | sai | littl | littl |
| 11 | amlp | februari | well | well |
| 12 | mccammon | can | am | am |
| 13 | quicklink | post | actual | peopl |
| 14 | rrlane | comment | didn | actual |
| 15 | deggan | 12 | peopl | didn |
| 16 | wpf | right | ll | ll |
| 17 | fatimah | know | thought | thought |
| 18 | pixnap | pm | let | let |
| 19 | onim | cartoon | wai | see |
| 20 | martouf | decemb | see | wai |
| 21 | perfess | islam | still | still |
| 22 | pemolin | see | feel | feel |
| 23 | bka | rsquo | reason | reason |
| 24 | suec | film | hate | hate |
| 25 | thecitizen | time | god | go |
| 26 | tvtattl | want | go | want |
| 27 | scocca | will | sure | sure |
| 28 | rutten | onli | yes | god |
| 29 | fws | wai | believ | yes |
| 30 | pav | am | want | try |

**Table 1.** The highest scoring opinion bearing terms according to different opinion scores. The scores were calculated over the training data for a single fold, consisting of the opinionated and relevant sets of documents for 135 queries.

Log Likelihood (WLLR) does a reasonable job of discovering terms that are likely to indicate opinionated content, such as "think" and "like", but the document frequency based Mutual Information (MI) appears to do even better. Not surprisingly, the $\chi^2$ measure seems to rank terms almost identically to MI.

### 6.3 Opinionated Document Detection

Here we investigate the use of the term scores for feature selection when we want to use the learning methods for calculating the opinion score for documents. We did some initial investigations on how performance on training data was affected by the number of features and we chose reasonable number of features (2 percent of features which is 6395 terms on average). The selected features

| | Weighted Log Likelihood (WLLR) | Mutual Information (MI) |
|---|---|---|
| Precision | 64.1% | 62.7% |
| Recall | 92.5% | 98.1% |
| F1 | 75.7% | 76.5% |

**Table 2.** Comparing different feature selection methods for training an SVM to recognize opinionated blog posts.

are not optimum and more feature selection analysis would be needed in order to find the best number of features but it was beyond the scope of this paper. We only compare the WLLR and ML term weighting methods, because the LR does not seem a good weighting method and $\chi^2$ ranks term pretty much the same as MI. We rank terms according to these weighting methods and choose the top 2 percent of the terms as features. In order to train a classifier we use the SVM-light tool [15] with the linear kernel and other default parameters. We use the TREC qrels to evaluate the system in terms of *precision*, *recall* and the combined *F1* score, which are defined as follows, where $C$ is the set of documents classified as opinionated:

$$Precision = |C \cap O|/|C|$$
$$Recall = |C \cap O|/|O|$$
$$F1 = 2|C \cap O|/(|C| + |O|)$$

The results for this experiment are shown in Table 2. The precision values are relatively low for both feature selection methods. Mutual Information based feature selection achieved a very high recall rate of 98% and despite slightly lower precision than WLLR, achieved a better $F1$. The low precision rates may indicate that the bag-of-words assumption is too simplistic for this problem and that richer representation of documents may be needed to improve the classifier performance.

### 6.4 Opinionated Document Retrieval

For generating the final ranking of documents, we tested simple combinations of opinion and relevance scores and also a particular rank learner named SVMmap [13] which is designed to optimize rankings for the Mean Average Precision (MAP) performance measure.

The results for the experiment are given in Table 3. We report both the opinion retrieval MAP scores as well as the percentage change with respect to the opinion retrieval score for the baseline (relevance retrieval) system. The different ranking methods are listed vertically, while the techniques for generating a document opinion score are divided horizontally. One can see that only the Borda Fuse and Rank Learning using SVMmap are useful for combining the opinion and relevance scores and that SVMmap performs the best for most of the cases.

The results show that combining the learning methods far outperformed the other approaches and 17% improvement is seen compared to 3.8% improvement

| | Expected Value | | Learning (SVM) | |
|---|---|---|---|---|
| | WLLR | MI | WLLR | MI |
| opinion | 0.282 (-7.3%) | 0.292 (-6.2%) | 0.341 (-1.4%) | 0.351 (-0.3%) |
| opinion*relevance | 0.305 (-4.9%) | 0.357 (0.3%) | 0.342 (-1.2%) | 0.343 (-1.1%) |
| Borda Fuse | 0.381(2.6%) | 0.381 (2.6%) | 0.392(3.8%) | 0.372 (1.8%) |
| Rank Learning | 0.388 (3.4%) | 0.365 (1.1%) | **0.531 (17.65%)** | **0.529 (17.54%)** |

**Table 3.** Mean Average Precision (MAP) for opinion retrieval on the TREC Blog Post dataset using different opinion scores for documents and different options for combining opinion and relevance scores. Percentages show improvements over the best baseline opinion retrieval ranking.

in the best case for the Borda Fuse. This is a statistically significant result according to the two-tailed paired T-test at the 0.01% confidence level (p-value= 1.57E-07).

The comparison between the two feature selection methods: WLLR and MI did not provide a conclusive result. According to the paired T-test, the p-value of 0.46 shows the difference not to be significant.

## 7 Conclusions and Future Work

We have shown that learning can be effective both for generating opinion scores for individual documents and also for learning a ranking function that combines opinion evidence with relevancy into a single ranking function. A distinct advantage of our approach is that by performing multiple levels of learning, we maximize the use of available training data while not relying on external sources of opinion-bearing word lists, that may actually not be well-suited to the blog opinion retrieval task.

In future work we plan to investigate the effects of introducing different extensions and refinements to our basic framework. Firstly, we would like to consider term proximity and weight a document's opinion score by the distance between opinionated terms and query terms [2]. In this way, posts expressing opinions about topics unrelated to the query should have their scores reduced. Secondly, we plan to investigate the granularity at which we represent documents. Opinion detection systems often work at the sentence level, training classifiers to recognize opinionated sentences rather than entire documents [2, 4]. Gathering training data in this case may be problematic, however, as TREC assessments mark entire posts as being opinionated and not individual sentences. Moreover, we could extend the document features to include not only unigrams, but also bigrams or trigrams in an attempt to capture more complicated opinion expressions [3, 4]. The disadvantage of these more complicated models is that they require a much larger number of training examples to learn a model.

## References


1. Macdonald, C., Ounis, I., Soboroff, I.: Overview of the trec-2007 blog track. In: Proceedings of the Sixteenth Text REtrieval Conference (TREC 2007). (2007)
2. Zhang, W., Yu, C., Meng, W.: Opinion retrieval from blogs. In: CIKM '07: Proceedings of the sixteenth ACM conference on Conference on information and knowledge management, New York, NY, USA, ACM (2007) 831–840
3. Yang, K., Yu, N., Zhang, H.: Widit in trec 2007 blog track: Combining lexicon-based methods to detect opinionated blogs. In: Proceedings of the Sixteenth Text REtrieval Conference (TREC 2007). (2007)
4. Zhang, Q., Wang, B., Wu, L., Huang, X.: Fdu at trec 2007: Opinion retrieval of blog track. In: Proceedings of the Sixteenth Text REtrieval Conference (TREC 2007). (2007)
5. Amati, G., Ambrosi, E., Bianchi, M., Gaibisso, C., Gambosi, G.: Automatic construction of an opinion-term vocabulary for ad hoc retrieval. In: Proceedings of the 30th European Conference on IR Research (ECIR). (2008) 89–100
6. Nigam, K., McCallum, A.K., Thrun, S., Mitchell, T.: Text classification from labeled and unlabeled documents using em. Machine Learning **39**(2-3) (2000) 103–134
7. Ng, V., Dasgupta, S., Arifin, S.M.N.: Examining the role of linguistic knowledge sources in the automatic identification and classification of reviews. In: Proceedings of the COLING/ACL on Main conference poster sessions, Morristown, NJ, USA, Association for Computational Linguistics (2006) 611–618
8. Carterette, B., Allan, J., Sitaraman, R.: Minimal test collections for retrieval evaluation. In: SIGIR '06: Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval, New York, NY, USA, ACM (2006) 268–275
9. Manning, C.D., Schütze, H.: Foundations of Statistical Natural Language Processing. The MIT Press (June 1999)
10. Sebastiani, F.: Machine learning in automated text categorization. ACM Computing Surveys **34** (2002) 1–47
11. Robertson, S., Zaragoza, H., Taylor, M.: Simple bm25 extension to multiple weighted fields. In: CIKM '04: Proceedings of the thirteenth ACM international conference on Information and knowledge management. (2004) 42–49
12. Zhai, C., Lafferty, J.: A study of smoothing methods for language models applied to information retrieval. ACM Transactions on Information Systems (TOIS) **22**(2) (2004) 179–214
13. Yue, Y., Finley, T., Radlinski, F., Joachims, T.: A support vector method for optimizing average precision. In: SIGIR '07: Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval, New York, NY, USA, ACM (2007) 271–278
14. Montague, M., Aslam, J.A.: Condorcet fusion for improved retrieval. In: CIKM '02: Proceedings of the eleventh international conference on Information and knowledge management, New York, NY, USA, ACM (2002) 538–548
15. Joachims, T.: Making large-scale SVM learning practical. In Schölkopf, B., Burges, C., Smola, A., eds.: Advances in Kernel Methods - Support Vector Learning. MIT Press, Cambridge, MA (1999) 169–184