An Effective Statistical Approach to Blog Post Opinion Retrieval

Ben He & Craig Macdonald
Department of Computing Science
University of Glasgow
Scotland, UK
{ben,craigm}@dcs.gla.ac.uk

Jiyin He
Informatics Institute
University of Amsterdam
The Netherlands
jiyinhe@science.uva.nl

Iadh Ounis
Department of Computing Science
University of Glasgow
Scotland, UK
ounis@dcs.gla.ac.uk

ABSTRACT

Finding opinionated blog posts is still an open problem in information retrieval, as exemplified by the recent TREC blog tracks. Most of the current solutions involve the use of external resources and manual efforts in identifying subjective features. In this paper, we propose a novel and effective dictionary-based statistical approach, which automatically derives evidence for subjectivity from the blog collection itself, without requiring any manual effort. Our experiments show that the proposed approach is capable of achieving remarkable and statistically significant improvements over robust baselines, including the best TREC baseline run. In addition, with relatively little computational costs, our proposed approach provides an effective performance in retrieving opinionated blog posts, which is as good as a computationally expensive approach using Natural Language Processing techniques.

Categories and Subject Descriptors: H.3.3 [Information Storage and Retrieval]: Information Search and Retrieval

General Terms: Experimentation, Performance

Keywords: Opinion, Subjectivity, Sentiment, Blog, Statistics, Retrieval

1. INTRODUCTION

The rise on the Internet of blogging, the creation of journal-like web page logs, has created a highly dynamic subset of the World Wide Web, which evolves and responds to real-world events. Indeed, blogs (or weblogs) have recently emerged as a new grassroots publishing medium. The so-called blogosphere (the collection of blogs on the Internet) opens up several new interesting research areas.

A key feature that distinguishes blog content from other Web content is their subjective nature. Bloggers tend to express opinions and comments towards some given targets, such as persons, organisations or products. A study of a query log from a commercial blog search engine found that many blog queries seem to be related to uncovering public opinions about a given target [15]. For example, a user who is planning to buy a given laptop brand might wish to gauge the opinions of other users in the blogosphere about how they rate its features.

There have been several studies on how to find opinions in the Natural Language Processing (NLP) community. For example, Pang et al. proposed to find opinions from movie reviews using machine learning and NLP techniques [21]. However, their approach is based on the assumption that the analysed documents are already known to be relevant. Building a retrieval system to uncover documents that are both opinionated and relevant remains a difficult challenge in information retrieval. Since 2006, the Text REtrieval Conference (TREC) has been running a Blog track and a corresponding opinion finding task for addressing this challenge, namely finding opinionated and relevant blog posts [13, 18]. The opinion finding task is an articulation of a user search task, where a user is trying to uncover what the public opinions are on the blogosphere, towards a given named-entity target [20].

Under the TREC opinion finding task, an important issue in evaluating a blog post opinion finding system is to look at how the system performs over a baseline for which no opinion feature is applied. The baseline retrieves as many relevant documents as possible, regardless of their opinionated nature. Various approaches have been proposed for the TREC Blog track opinion finding task [13, 18]. However, the experimental results in this task have demonstrated considerable difficulty in improving strong retrieval baselines [20]. Indeed, only a handful of groups achieved an improvement over their baseline, using techniques such as NLP (for example integrating OpinionFinder [8]), or SVM classifiers [30]. In general, most of the proposed approaches utilise different external sources of evidence, mostly heuristically, such as a long list of pre-compiled subjective terms, or rare terms, for detecting opinionated documents. However, evidence that can be learnt from the collection itself, by applying appropriate statistical methods, is not adequately utilised. As a consequence, these proposed approaches either involve considerable manual efforts in collecting evidence for opinions, or lead to little improvement over a baseline that does not include any opinion finding feature [18].

In this paper, we propose a statistical and light-weight automatic dictionary-based approach. We show that de-

CIKM'08, October 26–30, 2008, Napa Valley, California, USA.

spite its apparent simplicity, it provides statistically significant improvements over robust baselines, including the best TREC baseline run, without any manual effort. In addition, we show that our proposed approach provides comparable opinion retrieval performances with a sophisticated approach adapting the NLP-based OpinionFinder toolkit [25], while being much less computationally expensive.

The remainder of this paper is organised in three parts. First, we survey previous work on retrieving opinionated blog posts. Next, we present our proposed method using an automatically built dictionary for opinion retrieval. Finally, we provide a thorough evaluation of the proposed method and its variants compared to strong baselines.

2. RELATED WORK

We introduce the TREC paradigm for experimenting with opinion retrieval in Section 2.1, and previous approaches to opinion retrieval in Section 2.2.

2.1 The TREC Paradigm for Experimentation of Opinion Retrieval

The TREC Blog opinion finding task has been running since 2006. This task uses the Blog06 collection, representing a large sample crawled from the blogosphere over an eleven week period from December 6, 2005 until February 21, 2006 [12]. The collection is 148GB in size, with three main components consisting of 38.6GB of XML feeds (i.e. the blog), 88.8GB of permalink documents (i.e. a single blog post and all its associated comments) and 28.8GB of HTML homepages (i.e. the main entry to the blog). The permalink documents are used as a retrieval unit for the opinion finding task [13, 18]. There are over 3.2 million permalink documents in the Blog06 collection. In this paper, we follow the TREC setting and experiment on the permalink documents.

Each participating system is evaluated using a set of topics and their associated relevance assessment. For example, a Blog opinion finding topic is included in Figure 1.

```
<top>
<num> Number: 863

<title> netflix

<desc> Description:
Identify documents that show customer opinions
of Netflix.

<narr> Narrative:
A relevant document will indicate subscriber
satisfaction with Netflix.  Opinions about
the Netflix DVD allocation system, promptness
or delay in mailings are relevant.
Indications of having been or
intent to become a Netflix subscriber that do
not state an opinion are not relevant.
</top>
```

Figure 1: Blog 2007 opinion finding task, topic 930.

The relevance assessment procedure for the documents retrieved for the topics had two levels. The first level assesses whether a given blog post, i.e. a permalink, contains information about the target and is therefore relevant. The second level assesses the opinionated nature of the blog post, if it was deemed relevant in the first assessment level [13, 18]. A system's performance in retrieving opinionated blog post is evaluated by how the system performs over a baseline, which retrieves as many relevant documents as possible, independent of whether they contain an opinion or not. For example, in the TREC opinion finding task, submission of baselines was encouraged in TREC 2006, and has been mandatory since TREC 2007[1].

Our experiments in this paper follow this paradigm. We examine if our proposed method brings an improvement when running on top of strong and robust baselines. Moreover, we experiment at the second relevance assessment level, which takes into account only blog posts that are both relevant and opinionated.

2.2 Previous Work on Opinion Retrieval

In this section, we briefly survey previous studies on detecting opinion/sentiment for blog post retrieval. In the literature, the blog opinion retrieval system is usually built on top of a baseline, which retrieves as many relevant documents as possible for a given topic, independent of whether they contain an opinion or not. One or several sources of evidence for opinion/sentiment is (are) used for assigning an opinion score to each retrieved document. The opinion score is then combined with the initial relevance score given by the baseline to produce a final document ranking.

The approach proposed by Yang et al. is a typical example of the above described architecture of the current opinion retrieval approaches [27, 28]. Different sources of evidence were used in their approach, including pre-compiled lists of terms and indicators created by extracting opinionated terms from training data and manual editing. The subjectivity of blog posts is then determined by scoring the density of the potentially subjective indicators found in the posts. Documents are re-ranked by combining the opinion score of each post with the relevance score given by the baseline.

Java et al. applied a meta-learning approach using Support Vector Machine (SVM) classifiers based on a manual opinion term dictionary [9]. Proximity between the opinionated terms in the dictionary and the query terms is considered in the classification. Approaches based on similar ideas were also proposed in the context of the TREC blog track [23, 31]. Zhang et al. performed sentiment analysis on a per-sentence basis [29, 30]. For each query topic, they collected a set of subjective and objective sentences that are used to train a sentence classifier. Each query topic has its specific sentence classifier. They used the Wikipedia as an external source of objective sentences, and RateitAll.com and other Web sources as a source for subjective sentences. They then used SVMs with a default kernel to build the sentence classifier. This sentence classifier then gives a score that is combined with the relevance score produced by their baseline, which retrieves topic-relevant blog posts regardless of the notion of opinion. Mishne proposed a dictionary-based approach based on the use of the General Inquirer, specifically the Osgoods semantic dimensions and emotional categories[2] to produce an opinion score that is linearly combined with the baseline relevance score [16].

[1] http://ir.dcs.gla.ac.uk/wiki/TREC-BLOG
[2] http://www.wjh.harvard.edu/~inquirer/

Many other approaches were also proposed. For instance, Amati et al. proposed a semi-automatic method for learning an opinion dictionary from the Blog06 collection [2, 3]. Yang et al. used logistic regression to classify the opinion or non-opinion statements at the sentence level. The model is trained on external corpora and was applied for cross-domain learning [26]. Godbole et al. developed an algorithm to construct a sentiment lexicon by expanding small dimension sets of seed sentiment words. By marking up all sentiment words and the associate entities in the corpus, they assigned a subjectivity score to the text [6]. Ernsting et al. expand the queries using the collection enrichment technique, based on the idea that an external resource could bring useful additional query terms [5]. They also showed that applying Kullback-Leibler (KL) divergence-based language modelling with Jelinek-Mercer smoothing for opinion term weighting markedly hurts the retrieval performance. In this paper, we will explain the reason for the detrimental effect of applying KL-based language modelling for opinion term weighting. Our explanation will also be confirmed by experiments.

In essence, from the first two years of the TREC Blog track opinion finding task, it has proved to be difficult to improve over a reasonably strong topic-relevance baseline (i.e. a system where all opinion finding features are turned off). Indeed, only a few participating groups were able to do so [20]. In the next section, we propose a purely statistical approach for opinion retrieval. Unlike the aforementioned approaches, the proposed technique does not require manual efforts, as the opinion dictionary is automatically derived from the collection itself.

3. THE STATISTICAL DICTIONARY-BASED APPROACH TO OPINION RETRIEVAL

In this section, we propose a statistical approach to retrieving opinionated blog posts. Our proposed approach has four steps. First, it automatically generates a dictionary from the collection without requiring manual effort. Second, it assigns a weight to each term in the dictionary, which represents how opinionated the term is. Third, it assigns an opinion score to each document in the collection using the top weighted terms from the dictionary as a query. Finally, it appropriately combines the opinion score with the initial relevance score produced by the retrieval baseline.

3.1 Dictionary Generation

Our dictionary is automatically derived from the document collection used. To derive the dictionary, we apply the skewed query model to filter out too frequent or too rare terms in the collection [4]. We remove those terms because if a term appears too many or too few times in the collection, then it probably contains too little (e.g. "and") or too specific (e.g. "aanandha") information so that it can not be generalised to different queries in indicating opinion. Using the skewed model, we firstly rank all terms in the collection by their within-collection frequencies in descending order. The terms, whose rankings are in the range $(s \cdot \#terms, u \cdot \#terms)$, are selected in the dictionary. $\#terms$ is the number of unique terms in the collection. s and u are parameters of the skewed model. In this paper, we apply $s = 0.00007$ and $u = 0.001$ as suggested in [4].

A snippet of the automatically generated dictionary derived from the Blog06 collection is shown in Table 1. From this table, we can see that many terms in the dictionary are not necessarily opinionated, since the dictionary generation process is independent of the notion of opinion. However, as we show later in our experiments, they can be good indicators of opinion when they are put into the context of the topic.

girl	director	simply	consider
fall	researcher	build	radio
class	large	respect	version
subscriber	education	flame	October
result	optional	leader	yeah

Table 1: A snippet of the dictionary derived from the Blog06 collection.

3.2 Term Weighting

This section presents how we assign weights to terms in the opinion dictionary. Our approach is inspired by the Divergence From Randomness (DFR) query expansion mechanism, which measures the divergence of a term's distribution in a pseudo-relevance set from its distribution in the whole collection [1]. Our approach assumes a training step. For a set of training queries, we assume that D(Rel) is the document set containing all relevant documents, and D(opRel) is the document set containing all opinionated relevant documents. D(opRel) is a subset of D(Rel). For each term t in the opinion term dictionary, we measure $w_{opn}(t)$, the divergence of the term's distribution in D(opRel) from that in D(Rel). This divergence value measures how a term stands out from the opinionated documents, compared with all relevant, yet not necessarily opinionated, documents. The higher the divergence is, the more opinionated the term is.

In information retrieval, a commonly used measure for term weighting is the Kullback-Leibler (KL) divergence from a term's distribution in a document set to its distribution in the whole collection. For instance, Ernsting et al. applied the KL divergence-based language modelling with Jelinek-Mercer smoothing for weighting opinionated terms [5]. However, their experimental results showed that this method has detrimental effect on the retrieval performance. Regarding this problem, we argue that the KL divergence measure considers only the divergence from one distribution to the other, while ignoring how frequent a term occurs in the opinionated documents. As a consequence, the weights of the terms in the opinion dictionary might be biased towards the terms with high KL divergence values, but containing low information in the opinionated document set D(opRel). For example, if a term appears only 3 times in the collection, and twice in the opinionated documents, this term is likely to have a high KL divergence. However, we don't consider this term to show a strong evidence of opinion because it appears only in at most two opinionated documents in the entire collection. Therefore, we rather apply the Bo1 term weighting model based on the Bose-Einstein statistics given by the geometric distribution, which measures how informative a term is in the set D(opRel) against D(Rel) [1]. Using the Bo1 model, the weight of a term t in the opinionated document set D(opRel) is given by:

$$w_{opn}(t) = tf_x \cdot \log_2 \frac{1+\lambda}{\lambda} + \log_2(1+\lambda) \qquad (1)$$

where λ is the mean of the assumed Poisson distribution of the term t in the relevant documents. It is given by tf_{rel}/N_{rel}. tf_{rel} is the frequency of the term t in the relevant documents, and N_{rel} is the number of relevant documents. tf_x is the frequency of the term t in the opinionated documents.

3.3 Generating the Opinion Score

We take the X top weighted terms from the opinion dictionary, and submit them to the retrieval system as a query Q_{opn}. By doing this, the retrieval system assigns a relevance score to each document in the collection using a document weighting model, e.g. the BM25 model [22], or the PL2 Divergence From Randomness (DFR) model [1]. Such a relevance score reflects the extent to which the top weighted opinionated terms are informative in the document, capturing the overall opinionated nature of the document.

We denote the relevance score, assigned for query Q_{opn} for document d, as the opinion score $Score(d, Q_{opn})$. In the next step, this opinion score is combined with the relevance score $Score(d, Q)$, given by the initial document ranking, to produce the final document ranking. Note that the proposed opinion scoring method is light-weight because it is performed during indexing, independently of the retrieval stage.

3.4 Score Combination

We apply two different methods for combining the initial relevance score with the opinion score, namely an intuitive linear combination, and a combination method that maps document opinion scores to probabilities. The initial relevance score is given by a retrieval baseline, which is independent of any expressed opinion in the document. The first method applies a linear combination:

Linear combination:

$$Score_{com}(d, Q) = (1-a)Score(d, Q_{opn}) + a \cdot Score(d, Q) \quad (2)$$

where each score $Score(d, Q_{opn})$ (resp. $Score(d, Q)$) is scaled by dividing the score by the maximum $Score(d, Q_{opn})$ (resp. $Score(d, Q)$). a is the free parameter of the linear combination.

Our second combination method maps each opinion score to the maximum likelihood of the probability $P(opn|d, Q_{opn})$ of being opinionated as follows:

$$P(opn|d, Q_{opn}) = \frac{Score(d, Q_{opn})}{\sum_{d \in Coll} Score(d, Q_{opn})} \quad (3)$$

where $Coll$ is the entire document collection. Since a high $P(opn|d, Q_{opn})$ is supposed to indicate a high degree of opinion expressed in the document, we would like to have a combined score that is an increasing function of $P(opn|d, Q_{opn})$. Therefore, such a probability $P(opn|d, Q_{opn})$ is combined with the initial relevance score using a logarithmic function as follows:

Log. combination:

$$Score_{com}(d, Q) = \frac{-k}{log_2 P(opn|d, Qt)} + Score(d, Q) \quad (4)$$

where k is a free parameter. Both score combination methods use the stored opinion scores of all documents, computed during indexing. Therefore, there is only a negligible additional overhead during retrieval.

4. EXPERIMENTAL ENVIRONMENT AND SETTINGS

We use the Terrier Information Retrieval platform for both indexing and retrieval [19]. In the rest of this section, we describe our experimental environment and settings for evaluating our proposed dictionary-based approach to the blog opinion retrieval task.

4.1 The Blog06 Test Collection and Topics

We base our experiments on the Blog06 collection created for the TREC Blog track [12], which is currently the only available Blog test collection with relevance assessments. Following the official TREC setting [13, 18], we index only the permalinks, which are the blog posts and their associated comments. The permalinks are used as the retrieval units in the TREC Blog track opinion finding task. Each term is stemmed using Porter's English stemmer, and standard English stopwords are removed.

We use the 100 topics from the TREC 2006 & 2007 opinion finding tasks, numbered from 851 to 950. We use the 50 topics from the opinion finding task in 2006 for training, and the 50 topics from TREC 2007 for testing. Each topic contains three topic fields, namely title, description and narrative. We only use the title topic field that contains very few keywords related to the topic. The title-only queries are usually short[3], which is a realistic snapshot of real user queries in practise and the official TREC setting [13, 18].

4.2 Retrieval Baselines

Following the aforementioned TREC opinion finding task paradigm, our baseline retrieves as many relevant documents as possible independently of whether they are opinionated or not.

Firstly, we apply the InLB document weighting model, which is generated from the Divergence from Randomness (DFR) modular framework [19]. The InLB model applies the Inverse Document Frequency and Laplace succession for document weighting [1], and BM25's normalisation function to normalise the term frequency [22]. We use InLB because it provides effective retrieval performance on the Blog06 collection, and because it is a hybrid model combining BM25 and the DFR document weighting paradigm. In InLB, for a given document d and query Q, the relevance score is given by:

$$Score(d, Q) = \sum_{t \in Q} w(d, t) = \sum_{t \in Q} \frac{qtw \cdot tfn}{tfn + 1} \log_2 \frac{N + 1}{df + 0.5} \quad (5)$$

where the query term weight qtw is given by qtf/qtf_{max}; qtf is the query term frequency. qtf_{max} is the maximum query term frequency among the query terms. N is the number of documents in the collection. df is the number of documents containing the query term t. The normalised term frequency tfn is given by BM25's normalisation function [22] as follows:

$$tfn = \frac{tf}{(1 - b) + b \cdot \frac{l}{avg\_l}} \quad (6)$$

where tf is the within-document term frequency, l is the document length and avg_l is the average document length

[3] 1.74 words on average in the 100 topics used.

dwelling	distinguished	Bush	taken
implementation	London	load	bonkers
Colorado	tantalizing	keenly	feisty
defiantly	agitation	torturous	joyfully
trump	gray	Iraq	inventor

Table 2: A snippet of the external opinion dictionary.

in the whole collection. b is a free parameter. In this paper, we set b to 0.2337 based on optimisation on the 50 training topics.

On top of the InLB model, our second baseline applies the pBiL2 randomness model [11], which utilises the query term proximity evidence for retrieval, to favour documents where the query terms appear in close proximity. The model we apply is based on the binomial randomness model. It computes the score of a pair of query terms in a document as follows:

$$\begin{aligned} score(d, Q_2) &= \sum_{p \in Q_2} \frac{1}{pfn+1} \cdot \Big(\\ &- \log_2 (avg\_w - 1)! + \log_2 pfn! \\ &+ \log_2 (avg\_w - 1 - pfn)! \\ &- pfn \log_2 (p_p) \\ &- (avg\_w - 1 - pfn) \log_2 (p'_p) \Big) \end{aligned} \quad (7)$$

where Q_2 is the set of all query term pairs in query Q. $avg\_w = \frac{T - N(ws-1)}{N}$ is the average number of windows of size ws tokens in each document in the collection, N is the number of documents in the collection, and T is the total number of tokens in the collection. $p_p = \frac{1}{avg\_w - 1}$, $p'_p = 1 - p_p$, and pfn is the normalised frequency of the tuple p, as obtained using Normalisation 2[4] [11]. In this paper, Normalisation 2's free parameter c_p is set to 90 based on experiments on the 50 training topics.

4.3 External Opinion Dictionary and Term Weighting

To compare with the dictionary derived from the collection itself, we also manually generate a dictionary compiled from various external linguistic resources such as OpinionFinder [25] and those used in the approaches mentioned in Section 2. The dictionary contains approximately 12,000 English words, mostly adjectives, adverbs and nouns, which are supposed to be subjective. A snippet of this dictionary is shown in Table 2. In this paper, we denote the manually edited dictionary by the *external dictionary*, and we denote the automatically derived one by the *internal dictionary*.

As suggested in Section 3.2, the KL divergence measure does not consider how informative a term is in the opinionated documents. Therefore, the term weights, assigned by the KL divergence measure on one topic set, cannot be generalised to other topics because KL ignores how informative the term is in the opinionated documents. To confirm this argument, in addition to Bo1, we also apply the KL term weighting model based on the KL divergence measure. Using the KL model, the weight of a term t in the opinionated document set D(opRel) is given by [1]:

$$w_{opn}(t) = p(t|D(opRel)) \cdot \log_2 \frac{p(t|D(opRel))}{p(t|D(Rel))} \quad (8)$$

where $p(t|D(opRel)) = tf_x / c(D(opRel))$ is the probability of observing term t in the opinionated document set. tf_x is the frequency of the term t in the opinionated document set, and c(D(opRel)) is the number of tokens in the opinionated document set. $p(t|D(Rel)) = tf_{rel} / c(D(Rel))$ is the probability of observing term t in the relevant document set D(Rel). tf_{rel} is the frequency of t in D(Rel), and c(D(Rel)) is the number of tokens in D(Rel). In the next section, we compare the opinion term weighting using Bo1 with that using KL.

5. EXPERIMENTS: OPINION TERM WEIGHTING

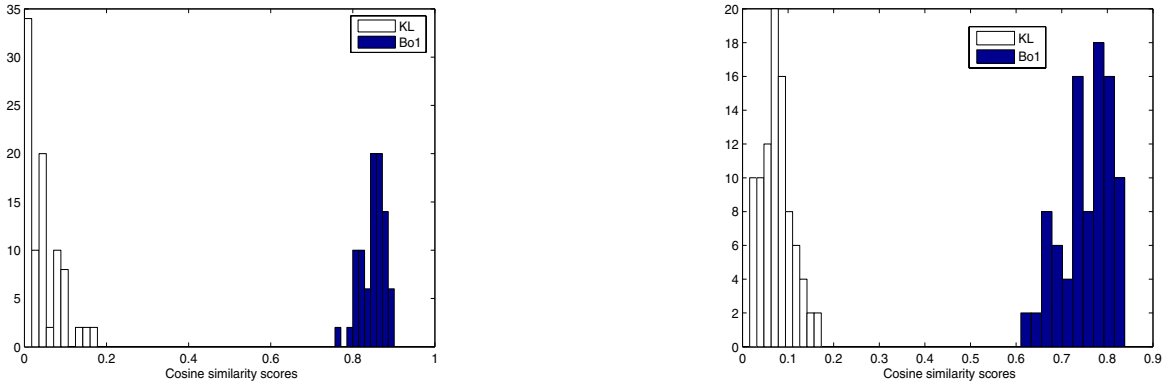
An underlying hypothesis of our proposed approach is that the most opinionated terms, derived from the relevant and opinionated documents for one query set, are also good indicators of opinion for other queries. In this section, we conduct experiments to examine this hypothesis with the use of two different term weighting models, namely KL (see Equation (8)) and Bo1 (see Equation (1)).

We randomly sample from the 50 training topics for 10 times, with each sample having 25 topics. During the sampling process, we ensure that each two samples have a reasonably small overlap (i.e. 65% maximum). For each sample of 25 topics, we rank the terms in the dictionary by their term weights using the corresponding relevance assessments information. Using the relevance assessments in each sample, the weight of each term is measured by the divergence of the term's distribution in the opinionated documents from its distribution in all relevant documents.

We compute the cosine similarity between the weights of the top 100 weighted terms from each two samples from the training topics. Figure 2 plots the distribution of the resulting cosine similarity scores using Bo1 and KL for external and internal opinion dictionaries, respectively. From this figure, we can see that the use of the Bo1 model for term weighting leads to high similarities (with a mean of 0.8487 for the external dictionary, and 0.7531 for the internal dictionary) between the term weights derived from different random samples of the training topics. On the contrary, the use of the KL model leads to a situation where different random samples agree little with each other in terms of the top weighted terms. When using the KL model, the cosine similarity between the top weighted terms from different samples is very low (with a mean of 0.04184 for the external dictionary and 0.07329 for the internal dictionary) as shown in Figure 2. This confirms our argument in Section 3.2 that the term weighting by the KL divergence measure cannot be generalised to different topics because the KL divergence measure ignores how informative a term is in the opinionated document set. This also explains why the KL divergence-based language modelling for opinion term weighting did not work in a previous study [5]. We have also conducted experiments with applying Jelinek-Mercer smoothing for the KL divergence and obtained similar findings. The related results are not included in this paper for brevity.

As an example, Tables 3 & 4 contain the top 20 weighted terms derived from one of the 10 random samples, from the

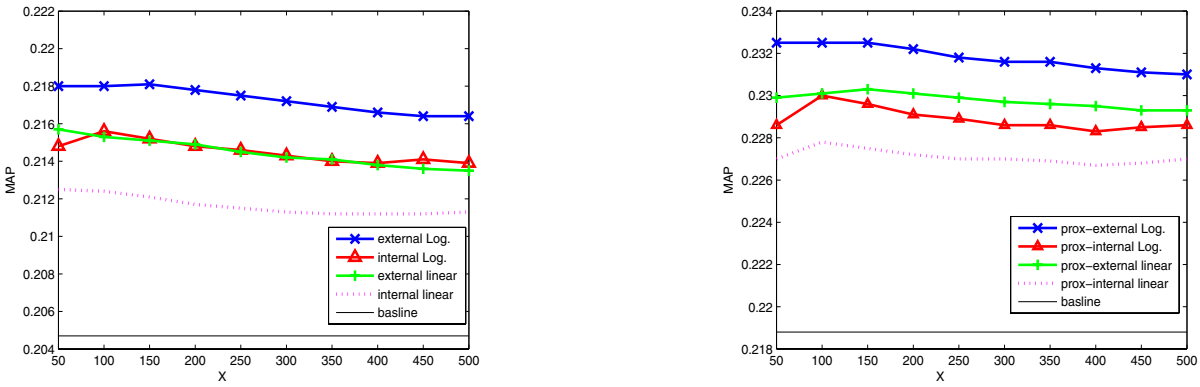
[4]Normalisation 2 is a term frequency normalisation method that assumes a decreasing density of term frequency with document length [1]



(a) External dictionary

(b) Internal dictionary

Figure 2: Cosine similarity distribution between the top 100 weighted terms from different samples of topics using Bo1 and KL with external and internal opinion dictionaries.



(a) Bo1 (no proximity)

(b) Bo1 (with proximity)

Figure 3: Parameter X (i.e. the number of top weighted opinion terms) against the mean best MAP obtained on the validation sets with Log. or linear combination. *Bo1* is used for term weighting.

Bush	movie	film	point
war	president	media	long
talk	nation	maked	give
watch	sure	white	let
Iraq	man	big	doesn't

Table 3: An example of the top 20 weighted terms from the internal dictionary on one of the sampled topic sets.

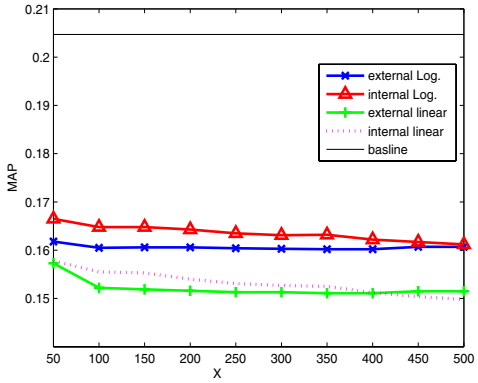
thinking	February	people	see
know	say	Bush	movie
January	want	only	show
report	film	work	American
story	come	being	read

Table 4: An example of the top 20 weighted terms from the external dictionary on one of the sampled topic sets.

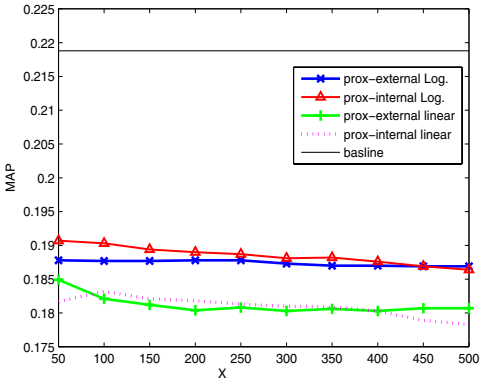
internal and external dictionary, respectively. From these two tables, we find that terms in both internal and external dictionaries, e.g. "Bush", "war", "movie" and "Iraq", are often related to controversial topics for which bloggers tend to express opinions. In the next sections, we show that both dictionaries actually result in comparable opinion retrieval performances.

6. EXPERIMENTS: VALIDATION

In this section, we describe our experiments for training the parameter X (i.e. the number of top-ranked terms in the dictionary used for assigning opinion scores to documents) and the free parameters a and k in Equations (2) & (4). For training X, we reuse the 10 samples of topics created in the previous section. For each sample, the 25 chosen topics are used for assigning term weights to the terms in the dictionary. The other 25 remaining topics in the training set are used for validation. We call this set of 25 remaining topics the *validation* set. For each set of 25 sampled topics, we use the corresponding relevance assessment to compute the opinion score $Score(d, Q_{opn})$. Different values of X are used in our experiments. In this paper, we report only results with X ranging from 50 to 500 with an interval of 50, since larger or smaller X values do not result in better re-

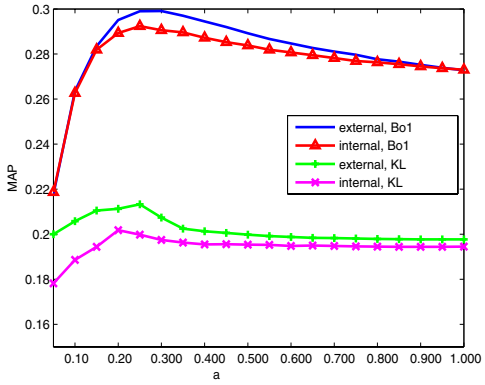


(a) KL (no proximity)

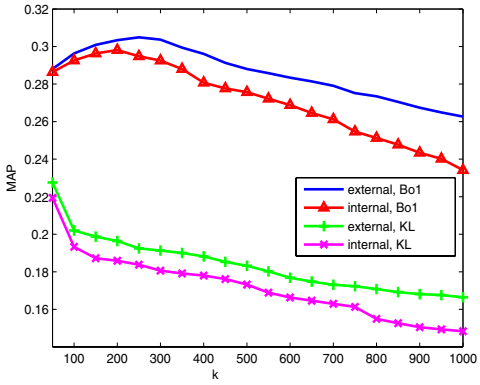


(b) KL (with proximity)

Figure 4: Parameter X (i.e. the number of top weighted opinion terms) against the mean best MAP obtained on the validation sets with probability (prob.) or linear combination. *KL* is used for term weighting.



(a) Linear combination



(b) Log. combination

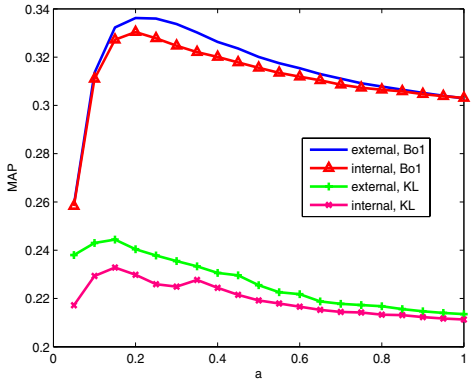
Figure 5: The combination parameter (a or k) against MAP obtained on the test topics using linear or Log. combination.

trieval performance according to our experimental results. For each set of opinion score $Score_i(d, Q_{opn})$, assigned by using the X_i top weighted terms in the dictionary, we tune the parameter of each score combination method (a and k in Equations (2) & (4)) by maximising Mean Average Precision (MAP) on the validation topic set. The resulting maximised MAP, using the opinion scores assigned by the X_i top weighted terms in the dictionary on the jth validation set, is denoted as $MAP_{max}(X_{i,j})$. The optimised X value is then the X_i that gives the highest $\overline{MAP_{max}(X_{i,j})}$, the mean $MAP_{max}(X_{i,j})$ over the 10 validation sets.

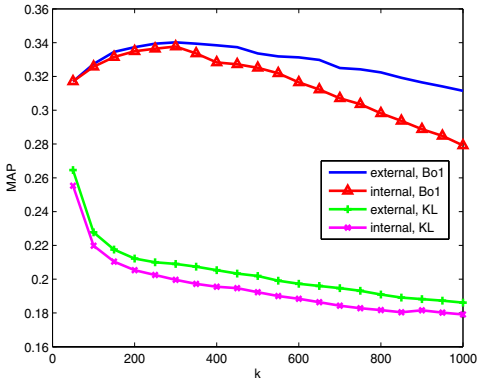
Figures 3 and 4 plot $\overline{MAP_{max}(X_{i,j})}$ against different X values used in the validation process, using Bo1 and KL for term weighting, respectively. From Figure 3, we can see that the use of Bo1 for term weighting results in a consistent improvement over the baseline using InLB, with and without the use of term proximity. On the contrary, the use of KL for term weighting leads to a marked degradation of the retrieval performance (see Figure 4) compared to the baseline. Such a degradation is statistically significant according to the Wilcoxon signed-rank matched-pairs test[5] at 0.01 level. This observation is expected since when KL is used for term weighting, different samples have little agreement on the most opinionated terms according to Figure 2. Moreover, Figure 3 shows that using Bo1 for term weighting, the resulting retrieval performance of our approach is stable over a wide range of X values. In particular, $X = 100$ provides the best retrieval performance across the 10 different random samples of topics from the training topic set, for both the external and internal dictionaries. Therefore, we use $X = 100$ in our experiments on the test topics .

After X is fixed, on the 50 training topics, a parameter sweeping is applied to optimise the free parameters a and k in Equations (2) & (4). The sweeping is applied within [0, 1] with an interval of 0.05 for a, and within (0, 1000] with an interval of 50 for k. From the training, we obtain $a = 0.25$ and $k = 250$, which will be applied on the 50 test topics from the TREC 2007 Blog track opinion finding task.

[5]We call the Wilcoxon signed-rank matched-pairs test as the *Wilcoxon test* in the rest of this paper.



(a) Linear combination with proximity



(b) Log. combination with proximity

Figure 6: The combination parameter (a in Equation (2) or k in Equation (4)) against MAP obtained on the test topics using linear or Log. combination. Term proximity is applied in the baseline.

		External		Internal	
Baseline	Measure.	Linear	Log	Linear	Log
InLB	*Entropy*	10.26	9.80	9.74	10.19
InLB	*Spread.*	0.08140	0.04220	0.07370	0.06390
InLB+Prox.	*Entropy*	10.49	9.490	10.47	10.48
InLB+Prox.	*Spread.*	0.07730	0.02870	0.07210	0.05850

Table 5: The *Entropy* and *Spread* values obtained using the linear combination or the Log. combination with External or Internal dictionary.

Baseline		Linear Comb.		Log Comb.	
Baseline	MAP_{bl}	MAP_l	diff.	MAP_{log}	diff.
External dictionary					
InLB	0.2727	0.2991	+9.68*	0.3049	+11.81*
InLB+Prox.	0.3027	0.3362	+12.28*	0.3402	+13.75*
Internal dictionary					
InLB	0.2727	0.2924	+7.22*	0.2981	+9.31*
InLB+Prox.	0.3027	0.3304	+10.08*	0.3377	+12.83*

Table 6: The MAP of the baselines (MAP_{bl}), opinion finding with linear combination (MAP_l), and that with Log. combination (MAP_{log}). diff. is the improvement over the baselines in percentage. Bo1 is used for term weighting. All improvements are statistically significant at 0.01 level as indicated by the stars.

7. EXPERIMENTS: EVALUATION

This section evaluates our proposed method on the test topics. Our experiments on the test topics are summarised in Figures 5 and 6, without and with the use of term proximity in the baseline, respectively. From both figures, we can see that Bo1 results in a much better retrieval performance than KL in all cases. Indeed, KL's resulting MAP values are always statistically significantly lower than the those of Bo1, according to the Wilcoxon test at 0.01 level. This is expected because, as mentioned as Section 5, when KL is used for term weighting, different random samples from the training topic set agree little on the top weighted opinionated terms.

From Figures 5 and 6, we also find that the effectiveness of the Log. combination method (see sub-Figures 5(b) & 6(b)) seems to be less sensitive to the change of its parameter value than the linear combination (see sub-Figures 5(a) & 6(a)). To test this observation, we compute the *Entropy* and the *Thread* measures proposed by Metzler for measuring the parameter sensitivity [14]. *Entropy* measures how much variation of retrieval effectiveness is there over a working range of parameter values, and *Spread* measures the distance between the best and the worst retrieval effectiveness within this working range of parameter values [14]. In our computation, this working range of values of parameters a or k are the same as those used for parameter sweeping introduced in Section 6. Table 5 contains the obtained *Entropy* and *Spread* values for using Bo1. We can see that both combination methods lead to relatively similar Entropy values. However, the Log. combination method provides a smaller *Spread* value than the linear combination in all cases. As stated in [14], it is preferred to have a low *Spread* over a low *Entropy*. Therefore, we conclude that the Log. combination method (Equation (4)) has a lower parameter sensitivity than the linear one (Equation (2)).

Table 6 compares the retrieval performance of our approach with the baselines. The setting of parameters a and k is obtained on the training topics, which is $a = 0.25$ and $k = 250$. We only report the results obtained using Bo1 in this Table since KL's performance is already shown to be less effective in Figures 5 and 6. Table 6 shows remarkable improvement over the baselines brought by our proposed dictionary-based approach. All improvements are statistically significant according to the Wilcoxon test at 0.01 level. Moreover, although the use of the external dictionary leads to a better performance than the internal one in all cases, the difference is minor and statistically insignificant according to the Wilcoxon test at 0.05 level. This demonstrates that our proposed approach is capable of achieving effective performance while being efficient and practical without the need for any manual effort.

In addition, we also examine if our proposed approach is able to improve the best TREC baseline run, namely uams07topic proposed in [5][6]. This run applies the collection enrichment technique, which expands the queries on the

[6]We would like thank the TREC organisers for making run uams07topic available for our research.

uams07topic baseline		Linear Comb.		Log Comb.	
Dictionary	MAP_{uams}	MAP_l	diff.	MAP_{log}	diff.
External	0.3453	0.3737	+8.22*	0.3749	+8.57*
Internal	0.3453	0.3655	+5.85*	0.3671	+6.31*

Table 7: The MAP of the best TREC baseline run (MAP_{uams}), opinion finding with linear combination (MAP_l), and that with Log. combination (MAP_{log}). diff. is the improvement over the baselines in percentage. Bo1 is used for term weighting. All improvements are statistically significant at 0.01 level as indicated by the stars.

AQUAINT2 collection, and retrieves from the Blog06 collection using the expanded query. Run uams07topics achieved the best topic-relevance baseline run, and also the second best opinion finding run in the TREC 2007 Blog track opinion finding task, despite having no opinion finding features enabled [13]. Table 7 provides the result of applying our proposed approach on top of the best TREC baseline run. From Table 7, we find that our proposed approach significantly improves uams07topics with the use of either external or internal dictionary. When the internal dictionary is used with Log. combination, our proposed approach provides an MAP of 0.3671, which would make it the second best run in the TREC 2007 Blog track opinion finding task. Note that the best run in this task was submitted by the University of Illinois at Chicago, which applies Support Vector Machines for sentiment analysis. Despite the effectiveness of their approach, its application requires extensive training on large amount of data collected from Wikipedia, RateitAll.com, etc. [29, 30]. On the other hand, using our proposed approach, the time spent on training our model, including the dictionary generation, is relatively trivial (approximately 15 minutes with a Pentium III 1GHz processor). Note that in a dynamic environment where the collection grows from time to time, using our proposed method, it is not necessary to repeat the training in responce to the collection growth, unless the growth has a significant impact on the vocabulary.

8. COMPARISON WITH BLOG OPINION RETRIEVAL USING OPINIONFINDER

We also compare our proposed approach with the one proposed in [7, 8], which uses OpinionFinder, a freely available and sophisticated Natural Language Processing (NLP) toolkit [25], to identify subjectivity in text. Applying OpinionFinder was shown to be one of the most effective opinion identification features [20].

For a given document, OpinionFinder is adapted to produce an opinion score for each document, based on the identified opinionated sentences. The opinion score $Score(d, OF)$ of a document d produced by OpinionFinder is defined as follows:

$$Score(d, OF) = sumdiff \cdot \frac{\#subj}{\#sent} \quad (9)$$

where $\#subj$ and $\#sent$ are the number of subjective sentences and the number of sentences in the document, respectively. *sumdiff* is the sum of the *diff* value of each subjective sentence in the document, showing the confidence level of subjectivity estimated by OpinionFinder.

For a given new query, such an opinion score is then combined with the relevance score $Score(d, Q)$ to produce the final relevance score in the same way as described above for the dictionary-based approach. The only difference is to replace $Score(d, Q_{opn})$ with $Score(d, OF)$ in Equations (2) & (4). Moreover, the free parameters of the combination methods are set to $a = 0.25$ and $k = 100$ based on training using the topics of the TREC 2006 opinion finding task.

Baseline	Linear Comb.		Log Comb.	
	MAP_{OF}	MAP_{int}	MAP_{OF}	MAP_{int}
InLB	0.2870	0.2924	0.3064	0.2981
InLB+Prox.	0.3155	0.3304	0.3450	0.3377
uams07topic	0.3627	0.3655	0.3655	0.3671

Table 8: The MAP obtained by using OpinionFinder (MAP_{OF}) and by our opinion finding method using internal dictionary (MAP_{int}) when applied to three different baselines. No statistically significant difference is found.

Table 8 compares the retrieval performance of our proposed approach with the one using OpinionFinder, with two different combination methods, and three different baselines. Bo1 is used for weighting the terms in the internal dictionary. From Table 8, we find that both the OpinionFinder-based approach and our dictionary-based approach provide comparable retrieval performance. According to the Wilcoxon test, there is no statistically significant difference between their resulting MAP values at the 0.05 level.

Our dictionary-based approach is light-weight because the opinion scoring of the documents are performed during indexing, and the overall process has negligible computational overheads. In contrast, using OpinionFinder to process the blog posts, it took approximately 19,370 CPU hours of a Pentium III 1GHz processor to process the 3.2 million documents in the Blog06 collection, and the processing time is expected to increase if the collection grows, even if the growth is insignificant. Such figures make the OpinionFinder-based approach difficult to use in an operational setting, despite its effectiveness in mining subjectivity.

9. CONCLUSIONS AND FUTURE WORK

In this paper, we have proposed an effective and practical approach to retrieving opinionated blog posts without the need for manual effort. The proposed approach is practical in the sense that the opinion scores are computed during indexing, and the involved computational cost is neglegible compared to other state-of-the-art approaches. Through extensive experiments on the large-scale Blog06 test collection, our proposed approach has shown marked and statistically significant improvements over strong and robust baselines, including the best TREC baseline run. Despite the simplicity of our proposed approach, it is effective and is capable of achieving the second best TREC run. The use of the automatically generated internal dictionary provides a retrieval performance that is as good as the use of an external dictionary manually compiled from various linguistic resources. In addition, our proposed approach provides a comparable retrieval performance to the approach using OpinionFinder, a toolkit for mining subjectivity based on NLP techniques, while being relatively less computationally expensive.

Moreover, in this paper, we have shown that the detection of opinionated blog documents can be effectively done in a statistical way, if appropriate statistics are applied. We have shown that different random samples from the collection reach a high concensus on the opinionated terms if the Bose-Einstein statistics given by the geometric distribution

are applied. We have also explained the reason why the commonly used Kullback-Leibler divergence measure sometimes fails in selecting opinionated terms. Such an explanation was confirmed by our experiments.

In the future, we plan to investigate further applications of our proposed approach. For example, we plan to extend the work to detecting the polarity or the orientation of the retrieved opinionated documents [13]. We also plan to study the connection of the opinion finding task to question answering, for example, by extracting the opinionated sentences within a blog post about a given target.

10. REFERENCES

[1] G. Amati. *Probabilistic models for information retrieval based on Divergence from Randomness*. PhD thesis, University of Glasgow, 2003.

[2] G. Amati, E. Ambrosi, M. Bianchi, C. Gaibisso, and G. Gambosi. FUB, IASI-CNR and University of Tor Vergata at TREC 2007 Blog Track. In *Proceedings of TREC 2007*.

[3] G. Amati, E. Ambrosi, M. Bianchi, C. Gaibisso, and G. Gambosi. Automatic Construction of an Opinion-Term Vocabulary for Ad Hoc Retrieval. In *Proceedings of ECIR 2008*.

[4] F. Cacheda, V. Plachouras, and I. Ounis. A Case Study of Distributed Information Retrieval Architectures to Index One Terabyte of Text. *Information Processing and Management*, 41(5), 2005.

[5] B. Ernsting, W. Weerkamp, and M. de Rijke. Language Modeling Approaches to Blog Post and Feed Finding. In *Proceedings of TREC 2007*.

[6] N. Godbole, M. Srinivasaiah, and S. Skiena. Large-Scale Sentiment Analysis for News and Blogs. In *Proceedings of ICWSM 2006*.

[7] D. Hannah, C. Macdonald, J. Peng, B. He, and I. Ounis. Experiments in Blog and Enterprise Tracks with Terrier. In *Proceedings of TREC 2007*.

[8] B. He, C. Macdonald, and I. Ounis. Ranking Opinionated Blog Posts using OpinionFinder. In *Proceedings of SIGIR 2008*.

[9] A. Java, P. Kolari, T. Finin, A. Joshi, and J. Martineau. The BlogVox Opinion Retrieval System. In *Proceedings of TREC 2006*.

[10] A. Lenhart, and S. Fox. Bloggers : a portrait of the Internet's new storytellers. *Pew Internet & American Life Project*, 2006.

[11] C. Lioma, C. Macdonald, V. Plachouras, J. Peng, B. He,I. Ounis. University of Glasgow at TREC 2006: Experiments in Terabyte and Enterprise Tracks with Terrier. In *Proceedings of TREC 2006*.

[12] C. Macdonald, and I. Ounis. The TREC Blog06 Collection : Creating and Analysing a Blog Test Collection *DCS Technical Report TR-2006-224*. University of Glasgow. 2006.

[13] C. Macdonald, I. Ounis, and I. Soboroff. Overview of the TREC 2007 Blog Track. In *Proceedings of TREC 2007*.

[14] D. Metzler. Estimation, sensitivity, and generalization in parameterized retrieval models. In *Proceedings of CIKM 2006*.

[15] G. Mishne, and M. de Rijke. A Study of Blog Search. In *Proceedings of ECIR 2006*.

[16] G. Mishne. Multiple Ranking Strategies for Opinion Retrieval in Blogs. In *Proceedings of TREC 2006*.

[17] G. Mishne. Using Blog Properties to Improve Retrieval. In *Proceedings of ICWSM 2006*.

[18] I. Ounis, M. de Rijke, C. Macdonald, G. Mishne, and I. Soboroff. Overview of the TREC 2006 Blog Track. In *Proceedings of TREC 2006*.

[19] I. Ounis, G. Amati, V. Plachouras, B. He, C. Macdonald, and C. Lioma. Terrier: A High Performance and Scalable Information Retrieval Platform. In *Proceedings of OSIR 2006 Workshop*.

[20] I. Ounis, C. Macdonald, and I. Soboroff. On the TREC Blog track. In *Proceedings of ISWSM 2008*.

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

[22] S. E. Robertson, S. Walker, M. Hancock-Beaulieu, M. Gatford, and A. Payne. Okapi at TREC-4. In *Proceedings of TREC 4*.

[23] O. Vechtomova. Using Subjective Adjectives in Opinion Retrieval from Blogs. In *Proceedings of TREC 2007*.

[24] E. Voorhees. *TREC: Experiment and Evaluation in Information Retrieval*. The MIT Press, 2005.

[25] T. Wilson, P. Hoffmann, S. Somasundaran, J. Kessler, J. Wiebe, Y. Choi, C. Cardie, E. Riloff, and S. Patwardhan. OpinionFinder: a system for subjectivity analysis. In *Proceedings of HLT/EMNLP on Interactive Demonstrations*, 2005.

[26] H. Yang, J. Callan, and L. Si. Knowledge Transfer and Opinion Detection in the TREC 2006 Blog Track. In *Proceedings of TREC 2006*.

[27] K. Yang, N. Yu, A. Valerio, H. Zhang, and W. Ke. Fusion Approach to Finding opinions in Blogosphere. In *Proceedings of ICWSM 2006*.

[28] K. Yang, N. Yu, and H. Zhang. WIDIT in TREC 2007 Blog Track: Combining Lexicon-Based Methods to Detect Opinionated Blogs. In *Proceedings of TREC 2007*.

[29] W. Zhang, W. Meng, and C. Yu. Opinion retrieval from blogs. In *Proceedings of CIKM 2007*.

[30] W. Zhang, and C. Yu. UIC at TREC 2007 Blog Track. In *Proceedings of TREC 2007*.

[31] G. Zhou, H. Joshi, and C. Bayrak. Topic Categorization for Relevance and Opinion Detection. In *Proceedings of TREC 2007*.