KLE at TREC 2008 Blog Track: Blog Post and Feed Retrieval

Yeha Lee, Seung-Hoon Na, Jungi Kim, Sang-Hyob Nam, Hun-young Jung, Jong-Hyeok Lee
Division of Electrical and Computer Engineering
Pohang University of Science and Technology
San 31, Hyoja-Dong, Nam-Gu, Pohang, 790–784, Republic of Korea
{sion,nsh1979,yangpa,namsang,blesshy,jhlee}@postech.ac.kr

Abstract

This paper describes our participation in the TREC 2008 Blog Track. For the opinion task, we made an opinion retrieval model that consists of preprocessing, topic retrieval, opinion finding, and sentiment classification parts. For topic retrieval, our system is based on the passage-based retrieval model and feedback. For the opinion analysis, we created a *pseudo opinionated word* (POW), O, which is representative of all opinion words, and expanded the original query with O.

For the blog distillation task, we integrated the average score of all posts within a feed, and the average score of the most relevant N post scores. We also examined the pseudo-relevance feedback for the distillation task by focusing on various document selection schemes to expand the query terms. The experimental results show a significant improvement over previous results.

1 Introduction

Blog track explores information seeking behavior in the blogosphere. Blog track was first introduced in TREC 2006. In TREC 2008, the Blog track has two main tasks: the opinion finding task and the blog distillation task.

Our approach to the opinion finding task is a three-step process. The first is preprocessing step. HTML tags and non-relevant contents provided by blog providers such as site description and menus are removed. In the topic retrieval step, we select the top $3,000$ documents ordered by topic-relevance. Our topic retrieval system, based on the passage-level retrieval model, estimates the relevance between a document and a given topic. To find documents which express an opinion about a given topic, we first estimate the degree of how opinionated a document is. Finally, we interpolate the topically relevant score and the opinion score of a document, and select the top $1,000$ documents as documents that express an opinion about a given topic. Our opinion analysis system is based on the lexicon-based approach. For opinion analysis, we created a *pseudo opinionated word* (POW), O, which is representative of all opinion words, and expand the original query with O.

Blog distillation is similar to the resource selection problem in distributed information retrieval. We formed two hypotheses to evaluate the degree of relevance between a blog feed and a given topic, and made two models to support these hypotheses. Furthermore, we propose a novel approach to select feedback documents for the pseudo relevance feedback.

2 Preprocessing

TREC Blog06 collection contains permalinks, feed files and blog homepages. We only used the permalink pages for the opinion retrieval task and the feed distillation task. The permalinks are encoded by HTML, and there are many different styles of permalinks. Beside the relevant textual parts, the permalinks contain many non-topical or non-relevant content such as HTML tags, advertisements, site descriptions, and menus, as well as topical contents.

The non-relevant content consist of many different types of blog templates which may be provided from commercial blog service venders to personal users. We propose a simple and effective algorithm, *DiffPost*, to deal with the non-relevant content. Much of the non-relevant content does not change in the same blog feed over a long period of time. DiffPost assumes that unchanged content between blog posts within the same blog feed are non-relevant (non-topical) content, and regards only changed content as relevant (topical) content.

To preprocess corpus, we firstly discarded all HTML tags, and applied DiffPost algorithm to remove non-relevant content. DiffPost segments each document into lines using the carriage return as a separator. DiffPost tries to compare sets of lines, and then regards the intersection of sets as the non-content information.

For example, let P_i and P_j be blog posts within the same blog feed. Let S_i and S_j be the sets of lines correspond to P_i and P_j, respectively.

$$NoisyInformation(P_i, P_j) = S_i \cap S_j \qquad (1)$$

We discard non-relevant contents through the set difference between a document and noisy-information. Finally, we removed stopwords from the content results of the DiffPost algorithm.

3 Topic Retrieval

3.1 Passage Based Ranking

Generally, a blog post consists of several topics, rather than presenting a single topic. Thus, even if a blog post is relevant, it does not mean that all parts of the blog post are relevant. Instead, only some parts of the blog post will be relevant to a query. Therefore, we extract some part or snippet of the blog post known to be relevant to the query, and use it as evidence for the topic retrieval.

To this end, we adopted the passage-based language model for the topic retrieval task. One of the most important issues in passage retrieval is the definition of the passage. We used a completely-arbitrary passage (Na et al., 2008c).

We used the score of the best passage, which indicates the passage that maximizes its relevance to a query, as the passage-level evidence. Our ranking function is as follows:

$$Score_P(Q, D) = \max_{P \in SP(D)} Score(Q, P) \qquad (2)$$

In Eq. 2, Q is a given query, $Score_P(Q, D)$ indicates the passage-level evidence of a document D. $SP(D)$ is the pre-defined set of all possible passages. We can use the interpolation of the document-level evidence, $Score_D(Q, D)$, and the passage-level evidence as follows:

$$Score(Q,D) = (1-\alpha)Score_D(Q,D) + \alpha Score_P(Q,D) \quad (3)$$

where α is the interpolation parameter, controlling the impact of the passage-level evidence on the final similarity score.

Each score is calculated based on the language modeling approach. The relevance scores of document, D, and passage, P, are defined as log-likelihood of query of the document, D, and the passage, P, respectively. For the document language model, we used the modified Dirichlet prior smoothing, $DirV$ (Na et al., 2008a), which improves the precision and handles the low performance of retrieval for verbose type of queries at the same time. Due to its high precision, $DirV$ allow us to obtain a more improved performance after the pseudo-relevance feedback. For the passage language model, we used the Jelinek-Mercer smoothing (Zhai and Lafferty, 2004).

3.2 Pseudo Relevance Feedback

Traditional pseudo-relevance feedback is document-level feedback which assumes that the entire content of a feedback document is relevant to a query (Zhai and Lafferty, 2001). However, normal blog posts are topically diverse so that they may contain non-relevant parts as well as relevant parts about a given topic. To remedy this problem, we adopted the completely arbitrary passage-level feedback (Na et al., 2008b).

In a completely arbitrary passage-level feedback, for top N documents, the best passages are extended by enlarging their context by maximally L length in the forward and backward directions. We use them as feedback context instead of top N documents, and update the query model based on the model-based feedback (Zhai and Lafferty, 2001).

Table 1: The topic and opinion retrieval scores (topic/opinion) for the Baseline Adhoc Retrieval Task runs

run id		MAP	p@10
1	08	0.4483/0.3671	0.6720/0.5560
	Overall	0.4304/0.3252	0.6773/0.5000
2	08	0.4532/0.3684	0.7180/0.5860
	Overall	0.4567/0.3418	0.7640/0.5340
3	08	0.4297/0.3484	0.7200/0.6060
	Overall	0.4330/0.3250	0.7573/0.5527
4	08	**0.4954/0.4052**	**0.7920/0.6440**
	Overall	0.4696/0.3485	0.7560/0.5487
5	08	0.4724/0.3822	0.7440/0.6160
	Overall	**0.4776/0.3543**	**0.7867/0.5580**

3.3 Topic Retrieval Runs

In the baseline adhoc retrieval task, we submitted 5 runs, as follows:

1. **KLEPsgRetT** uses passage based ranking with the title field of the topic
2. **KLEPsgRetTD** is KLEPsgRetT, with the title and the description fields of the topic
3. **KLEPsgRetTDN** is KLEPsgRetT, with the title, the description and the narrative fields of the topic
4. **KLEPsgFeedT** uses passage based ranking and passage based feedback with the title field of the topic
5. **KLEPsgFeedTD** is KLEPsgFeedT, with the title and the description fields of the topic

The results of our Baseline Adhoc Retrieval Task are shown in table 1.

4 Opinion Finding

In this step, we evaluate the degree of how opinionated a blog post is about a given topic. In the previous step, the top $3,000$ documents are returned according to the degree of topical relevance. We evaluate their opinion scores, and interpolate their opinion scores and topically relevant scores. And then, we select the top $1,000$ documents as results of the opinion finding task.

4.1 Opinion Scoring

For the opinion scoring module, most previous approaches can be divided into two types, *classification-based approach* and *lexicon-based approach* (Ounis et al., 2006; Macdonald et al., 2007). Our approach is based on the lexical-based approach.

To determine the opinionatedness of a blog post, we create a *pseudo opinionated word* (POW), O, which is a representative of all opinionated words, and make a new query, *POW-annotated query*, $Q^{'}$, by adding O to the original query. For the opinion finding task, we calculate the relevant score between a document and a POW-annotated query, $Q^{'}$, as follows:

$$Score(Q^{'},D) = (1-\alpha)S_{rel}(Q^{'},D) + \alpha S_{op}(Q^{'},D) \quad (4)$$

where α is the interpolation parameter. $S_{op}(Q^{'},D)$ and $S_{rel}(Q^{'},D)$ are an opinion score and a topically relevant score of a document, respectively.

A topically relevant score of a document does not have any relation with opinionated words. Therefore, we can rewrite Eq. 4 as follows:

$$Score(Q^{'},D) = (1-\alpha)S_{rel}(Q,D) + \alpha S_{op}(Q^{'},D) \quad (5)$$

where $S_{rel}(Q,D)$ is a topically relevant score obtained from the previous step, Section 3.

To estimate the opinion score of a document, $S_{op}(Q^{'},D)$, it is a simple and effective approach to add up the opinion scores of all words within a document. Because POW represents all opinionated words, $S_{op}(Q^{'},D)$ is calculated as follows:

$$\begin{aligned} S_{op}(Q^{'},D) &= \sum_{w} P(Sub|w)tf(w;D) \\ &= \sum_{w \in O} P(Sub|w)tf(w;D) \quad (6) \end{aligned}$$

where $P(Sub|w)$ represents the subjectivity of a word, w, that is, an opinion score of a word.

However, there are some problems with Eq. 6. Each blog post has a different number of words. A long blog post has more chance to contain opinionated words than a short blog post. Therefore, blog posts should be normalized according to their length. There are many studies about document length normalization in IR communities. We use the Okapi framework as the method of document length normalization. When using the Okapi model, $S_{op}(Q', D)$ of Eq. 6 is re-written as follows:

$$S_{op}(Q',D) \propto \frac{tf(O;D)}{tf(O;D)+k_1\left\{(1-b)+b\frac{LEN(D)}{avgLEN}\right\}} \quad (7)$$

where $LEN(D)$ is the length function of the blog post, and $avgLEN$ indicates the average length function value of all posts. We used unique term count of a post as the length function, and estimated the term frequency of Eq. 7, $tf(O;D)$, using Eq. 6.

4.2 Constructing the Opinion Lexicon

In the lexicon-based approach, one of the most important problems is to define the subjectivity of a word. We use different types of lexicons for defining the subjectivity of a word - SentiWordNet, automatically learned model from the review corpus.

4.2.1 SentiWordNet

SentiWordNet (Esuli and Sebastiani, 2006) is a lexical resource for opinion mining. SentiWordNet assigns to each synset of WordNet three sentiment scores: positivity, negativity, objectivity. We use the maximum value of the positive scores and the negative scores for all senses of a word as the degree of subjectivity of a word. Let $synset(w)$ be the set of synsets of a word, w.

$$P(Sub|w) = \max_{s \in synset(w)} (\max(P(pos|s), P(neg|s)))$$

where $P(pos|s)$ and $P(neg|s)$ represent the positive and negative score of the synset of the word, respectively.

That is, $P(Sub|w)$ means the subjectivity of w when w is maximally opinionated.

4.2.2 Corpus-Based Lexicon

In addition to SentiWordNet, we use Amazon's product review corpus and product specification corpus to create the opinionated lexical resource. Users of Amazon represent their opinion about products through reviews and product ratings. Therefore, the reviews contain opinionated words which the users use to express their opinion. The product specification is an objective document that consists of objective information such as product name, properties of the product, and product manual. We regard the review corpus as a subjective corpus, and the product specification corpus as an objective corpus. Then, we set the generative model of a word by two mixture models, $P(w|Sub)$, $P(w|Obj)$.

Let $R = w_1, \cdots, w_n$ be a large review document, which concatenates all reviews in the review corpus. The generative process of R is defined by the mixture:

$$P(w) = \lambda P(w|Sub) + (1-\lambda)P(w|Obj) \quad (8)$$

$$logP(R) = \sum_{i=1}^{n} logP(w) \quad (9)$$

To maximize the log-likelihood, we iteratively update $P(w|Sub)$ by applying the EM algorithm. After updating through iterations, we can obtain the probability that a word is generated from the subjective model, $P(w|Sub)$. We also obtain the subjectivity of a word, $P(Sub|w)$, using the Bayesian rule.

Finally, to combine the subjectivities defined in SentiWordNet and learned from the corpus, we use the term-specific mixture model where the prior probability of term w for opinion class is assumed to be $P(Sub|w)$ of SentiWordNet. In other words, a generative model for word w in a review is modeled by

$$P(w) = \lambda_w P(w|Sub) + (1-\lambda_w)P(w|Obj) \quad (10)$$

where λ_w is $P(Sub|w)$ of SentiWordNet.

4.3 Query-Specific Opinion Lexicon

4.3.1 Feedback-Style Learning

In this section, we address the query-specific opinion lexicon. There are differences in the words used to express user's opinion, according to each query (domain). Therefore, we should deal with opinionated lexicon according to query. To this end, we propose a feedback-style learning method to make a query-specific opinionated lexicon.

Let $TopN$ be the set of top-retrieved documents in response to a given query. For each document, we assign document-level subjectivity, $P(Sub|D)$, from the initial lexicon model. Then, the new model of the opinion lexicon, $P'(Sub|w)$, is estimated from the initial model of the opinion lexicon and the document-level subjectivities as follows:

$$P'(Sub|w) = \sum_{D \in TopN} P(Sub|D)P(D|w) \quad (11)$$

where $P(D|w)$ indicates how dominated a word w in document D is. Let $TopN(w)$ be the subset of feedback documents which contain w. We assume that $P(D|w)$ is uniformly distributed on $TopN(w)$. To estimate $P(Sub|D)$, we introduce the following simple expectation of the subjectivity for each opinionated word w in document D:

$$E_w(Sub|D) = \frac{\sum_{w \in D} P(Sub|w)}{|D|} \quad (12)$$

$$P(Sub|D) \approx \frac{E_w(Sub|D)}{\max_{D \in TopN} E_w(Sub|D)} \quad (13)$$

4.3.2 Using Passage Context

As mentioned above, a blog post consists of several topics rather than presenting a single topic. Therefore, we propose passage-level learning for query-specific lexicon, instead of the whole document. We firstly extract a best passage using complete-arbitrary passage, and expand its context by maximally L length in the forward and backward directions. After extracting the extended passage, we calculate $P(Sub|D)$ and $P(D|w)$ for Eq. 11 based on the passage.

4.4 Opinion Finding Runs

In the opinion finding task, we submitted 4 runs, as follows:

1. **KLEDocOpinT** uses corpus-based lexicon for opinionated lexicon resource, with the title field of a topic

2. **KLEDocOpinTD** is KLEDocOpinT, with the title and the description fields of a topic

3. **KLEPsgOpinT** uses query-specific opinion lexicon for opinionated lexicon resource, with the title field of a topic

4. **KLEPsgOpinTD** is KLEPsgOpinT, with the title and the description fields of a topic

Table 2 shows our results of the Opinion Finding Task.

Table 2: The topic and opinion retrieval scores (topic/opinion) for the Opinion Finding Task runs

run id		MAP	p@10
1	08	**0.5190/0.4569**	0.8020/**0.7200**
	Overall	**0.4937/0.4061**	0.7767/0.6287
2	08	0.4809/0.4160	0.7680/0.6740
	Overall	0.4896/0.3937	0.8093/0.6273
3	08	0.5190/0.4515	**0.8240**/0.7100
	Overall	0.4855/0.4024	0.7767/0.6200
4	08	0.4951/0.4219	0.7860/0.6640
	Overall	0.4896/0.3984	**0.8307/0.6467**

5 Polarity Task

5.1 Polarity Classification

The polarity task is similar to the opinion finding task. The difference is that the polarity task classifies the semantic orientation of a blog post into positive and negative orientations. Therefore, the approaches which are applied to the opinion finding task are almost applied to the polarity task, too. In the polarity task, we have to estimate the semantic orientations (polarities) of a word, $P(Pos|w)$ and $P(Neg|w)$, instead of the subjectivity of a word, $P(Sub|w)$. After estimating the semantic orientations of a word, we again applied Eq. 7 to calculate positive and negative scores of blog posts by using $P(Pos|w)$ and $P(Neg|w)$, respectively.

$$pol(D) = \begin{cases} positive & \text{if } S_{polarity}(D) > \lambda \\ negative & \text{if } S_{polarity}(D) < -\lambda \\ neutral & \text{otherwise} \end{cases} \quad (14)$$

where $S_{polarity}(D)$ is the difference between positive and negative scores, and λ is the threshold for classifying the polarity.

Similar to the opinion finding task, we use the Amazon review corpus to estimate the semantic orientations of words. All reviews have user ratings, and thus user ratings are used to decide the semantic orientations of the reviews. That is, we regard the reviews whose user rating is 4 and 5 as the positive review corpus, and the reviews whose user rating is 1 and 2 as the negative. We estimate $P(Pos|w)$ and $P(Neg|w)$ using the EM training with each review corpus.

run id		MAP	p@10	R-prec
1_pos	08	0.1865	0.2408	0.2130
	Overall	0.1772	0.2262	0.2004
1_neg	08	0.1491	0.1687	0.1614
	Overall	0.1218	0.1380	0.1330

Table 3: The performance of the Polarity Task

5.2 Polarity Runs

We submitted 1 run in the polarity task, as follows:

1. **KLEPolarity** uses corpus-based lexicon for polarity-tagged lexicon resource, with the title field of a topic.

Table 3 shows the results of the Polarity Task.

6 Feed Distillation

Blog distillation task aims to find the blog feeds which are recurrently interested in a given topic. Thus, in the blog distillation system, the retrieval unit is the blog feed rather than the blog post. We form two hypotheses to estimate the degree of relevance between a blog feed and a given topic.

- *Global Evidence Model (GEM)* : We believe that a blog feed which has more relevant posts at a higher rate is more relevant. For example, a blog feed that has 10 relevant posts out of total 20 posts is more relevant than a blog feed that has 10 relevant posts out of total 100 posts.

- *Local Evidence Model (LEM)* : We believe that a few posts that are highly relevant with a given topic represent the blog feed. Typically a blog feed addresses several topics. Therefore, the relevance of the blog feed about a given topic depends on some posts related to a given topic, rather than all posts within the blog feed.

To estimate the degree of the relevance between a blog feed and a given topic, we made two models to satisfy these hypotheses. Furthermore, to update the original query model, we proposed a novel approach to select feedback documents.

6.1 Distillation Retrieval Model

To find blog feeds devoted to a given topic, we integrate two models, Global Evidence Model (GEM) and Local Evidence Model (LEM). Let $S(Q,F)$ be the final relevant score between a feed and a given query, and let $S_{GEM}(Q,F)$ and $S_{LEM}(Q,F)$ be relevant scores obtained from GEM and LEM, respectively.

$$S(Q,F) = (1-\lambda)S_{GEM}(Q,F) + \lambda S_{LEM}(Q,F) \quad (15)$$

where λ is the interpolation parameter, which controls the weight of the GEM score and the LEM score.

6.1.1 Global Evidence Model

GEM is originated from the assumption that the blog feed which has relevant posts at higher rates is more relevant.

GEM views a blog feed as a collection of blog posts. In this regard, we can view the blog distillation problem as a resource selection problem. Our GEM is similar to the Small Document Model (Elsas et al., 2008). When regarding a blog feed as a collection of blog posts, there are some considerations. One is the problem when a small number of very long posts represent the blog feed. Each post has a different length. Therefore, a few very long posts may represent the blog feed. The other is the problem related to the size of the blog feed. Each blog feed has a different number of posts. Therefore, feeds with more posts have a higher probability of being relevant. To remedy this problem, we should normalize the blog feeds using the number of posts within them.

To handle these problems, we made a scoring function for GEM using the average score of all documents in a blog feed:

$$S_{GEM}(Q,F) = \sum_{D \in F} Score(Q,D)P(D|F) \quad (16)$$

where F represents the blog feed, and $Score(Q,D)$ represents relevant score of between a given query and a document, and $P(D|F)$ indicates the probability that the document, D, is generated from the blog feed, F. In GEM, the usage of the average score resolves the second problem.

To remedy the first problem, we regard the generative probability of the blog post, $P(D|F)$, as the uniform distribution.

$$P(D|F) = \frac{1}{|F|} \quad (17)$$

where $|F|$ is the number of total blog posts within a blog feed. The usage of the uniform distribution can remedy the first problem since it assigns the same weight to each blog post.

We used the KL-divergence framework (Lafferty and Zhai, 2001) to measure the relevance between a document and a given query, $Score(Q, D)$.

6.1.2 Local Evidence Model

LEM is originated from the belief that a few highly relevant posts represent the relevance of the blog feed about a given topic.

LEM uses the top N most relevant posts to estimate the relevance between a feed and a given topic. The view of the LEM is similar to Pseudo Cluster Selection (Seo and Croft, 2008). That is, a blog feed addresses several topics. Therefore, the relevance of the blog feed about a given topic should be estimated by using the relevant posts about a given topic, not all posts within the blog feed. In this regard, we can regard top N relevant posts as posts which are devoted about a given topic.

The ranking function of LEM is formed as follows:

$$S_{LEM}(Q,F) = \sum_{D \in TopN} Score(Q,D) P(D|TopN) \quad (18)$$

where $TopN$ is the set of top N relevant posts that are regarded as the same topic.

In Eq. 18, $Score(Q, D)$ and $p(D|TopN)$ are the same as in Eq. 16.

6.2 Feedback Model

Relevance feedback is known to be effective for improving retrieval performance. Performance of the relevance feedback depends on how documents for relevance feedback are chosen so that the system can learn most from the feedback information (Shen and Zhai, 2005). The traditional pseudo relevance feedback (PRF) assumes top N documents as relevant documents, and uses them as feedback documents. In this approach, the feedback information can be under a bias toward dominant information within the feedback documents. The biased information will be harmful for improving entire system performance through PRF. As the diversity of the feedback information increases, we can expect a more robust feedback search.

Table 4: The performance of the Blog Distillation Task

run id	**MAP**	**p@10**	**R-prec**
KLEDistLMT	0.3015	0.4480	**0.3601**
KLEDistLMB	0.2852	0.4380	0.3428
KLEDistFBT	**0.3031**	0.4260	0.3454
KLEDistFBB	0.2994	**0.4560**	0.3508

To obtain diverse information about a given topic, we propose a feed-based approach to select feedback documents. We empirically show that our approach improves the performance of the blog distillation task. We used the model-based feedback as feedback method.

Each blog feed consists of many posts which have various viewpoints and represent various properties (sub-topics) about a given topic. Therefore, when we select feedback documents from various feeds, we can obtain diverse information about a given topic.

We propose two approaches for selecting feedback documents, as follows:

- **Document-Based Selection** uses top K documents as feedback documents as most PRFs do.
- **Feed-Based Selection** uses top N feeds to feedback. For each feed, top K documents are selected as the pseudo relevance documents.

6.3 Distillation Runs

In the distillation task, we submitted 4 runs, as follows:

1. **KLEDistLMT** uses GEM and LEM, with the title field of a topic
2. **KLEDistLMB** is KLEDistLMT, with the title and the description fields of a topic
3. **KLEDistFBT** uses GEM, LEM and Feed-Based selection method to feedback, with the title field of a topic
4. **KLEDistFBB** is KLEDistFBB, with the title and the description fields of a topic

Table 4 shows the results of the Blog Distillation Task.

7 Conclusions

We have described our participation in the TREC 2008 Blog track. We developed an opinion finding system based on the lexicon-based approach and Okapi model. Our opinion finding system uses SentiWordNet and Amazon's review corpus to create an opinionated lexicon resource. We also proposed a novel approach to make a query (domain)-specific opinionated lexicon resource. The experimental results show that our approach significantly improves previous performances in the opinion finding task.

For the blog distillation task, we made two assumptions about relevance between a blog feed and a given topic. The experimental results have empirically shown that our hypotheses is effective at capturing the relevance between the topic and the blog feeds. Furthermore, we developed a novel approach to select feedback documents. This approach increased the diversity of the feedback documents, and led to significantly improve the distillation performance.

8 Acknowledgement

This work was supported in part by MKE & IITA through IT Leading R&D Support Project and also in part by the BK 21 Project in 2008.

References

Jonathan L. Elsas, Jaime Arguello, Jamie Callan, and Jaime G. Carbonell. 2008. Retrieval and feedback models for blog feed search. In *SIGIR '08: Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*, pages 347–354, New York, NY, USA. ACM.

Andrea Esuli and Fabrizio Sebastiani. 2006. Sentwordnet: A publicly available lexical resource for opinion mining. In *Proceedings of LREC 2006*, pages 417–422.

John Lafferty and Chengxiang Zhai. 2001. Document language models, query models, and risk minimization for information retrieval. In *SIGIR '01: Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 111–119, New York, NY, USA. ACM.

Craig Macdonald, Iadh Ounis, and Ian Soboroff. 2007. Overview of the trec-2007 blog track. In *TREC '07*.

Seung-Hoon Na, In-Su Kang, and Jong-Hyeok Lee. 2008a. Improving term frequency normalization for multi-topical documents and application to language modeling approaches. In *ECIR '08*, pages 382–393.

Seung-Hoon Na, In-Su Kang, Yeha Lee, and Jong-Hyeok Lee. 2008b. Applying complete-arbitrary passage for pseudo-relevance feedback in language modeling approach. In *AIRS '08*, pages 626–631.

Seung-Hoon Na, In-Su Kang, Yeha Lee, and Jong-Hyeok Lee. 2008c. Completely-arbitrary passage retrieval in language modeling approach. In *AIRS '08*, pages 22–33.

Iadh Ounis, Maarten de Rijke, Craig Macdonald, Gilad Mishne, and Ian Soboroff. 2006. Overview of the trec-2006 blog track. In *TREC '06*.

Jangwon Seo and W. Bruce Croft. 2008. Blog site search using resource selection. In *CIKM '08: Proceedings of the tenth international conference on Information and knowledge management*. ACM.

Xuehua Shen and ChengXiang Zhai. 2005. Active feedback in ad hoc information retrieval. In *SIGIR '05: Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 59–66, New York, NY, USA. ACM.

Chengxiang Zhai and John Lafferty. 2001. Model-based feedback in the language modeling approach to information retrieval. In *CIKM '01: Proceedings of the tenth international conference on Information and knowledge management*, pages 403–410, New York, NY, USA. ACM.

Chengxiang Zhai and John Lafferty. 2004. A study of smoothing methods for language models applied to information retrieval. *ACM Trans. Inf. Syst.*, 22(2):179–214.