POSTECH at TREC 2009 Blog Track: Top Stories Identification

Yeha Lee, Hun-young Jung, Woosang Song, and 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,blesshy,woosang,jhlee}@postech.ac.kr

Abstract. This paper describes our participation in the TREC 2009 Blog Track. Our system consists of the query likelihood component and the news headline prior component, based on the language model framework. For the query likelihood, we propose several approaches to estimate the query language model and the news headline language model. We also suggest two approaches to choose the 10 supporting relevant posts: Feed-Based Selection and Cluster-Based Selection. Furthermore, we propose two criteria to estimate the news headline prior for a given day. Experimental results show that using the prior significantly improves the performance of the task.

1 Introduction

Blog track explores information seeking behavior in the blogosphere. Compared with previous Blog track, the Blog track 2009 aims to investigate more refined and complex search scenarios in the blogosphere. In TREC 2009, the Blog track has two main tasks: Faceted Blog Distillation Task and Top Stories Identification Task.

Among two tasks, we participate in the Top Stories Identification Task. The Top Stories Identification Task is a new pilot search task addressing the news dimension in the blogosphere. Query of this task is a date "query". For a given date query, a system should rank news headlines according to their importance on the given day. Furthermore, for each headline, the system should provide 10 supporting blog posts which are relevant to and discuss the news story headline. To achieve this, the system will be provided with news headline corpus and the Blogs08 corpus, and could use external resources.

For the Top Stories Identification Task, our approach consists of two steps: the preprocessing step and the news headline ranking step. In the preprocessing step, HTML tags and non-relevant contents, provided by blog providers such as site description and menus, are removed. In the news headline ranking step, we make two language models, the date query language model and the headline language model, and estimate the probability that each news headline will be the top story.

2 Preprocessing Step

TREC Blog08 collection contains permalinks, feed files and blog homepages. We only used the permalink pages for the top stories identification 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 contents such as HTML tags, advertisements, site descriptions, and menus.

The non-relevant contents consist of many different types of blog templates which may be provided from commercial blog service venders. We used the DiffPost algorithm [5, 7] to deal with the non-relevant contents.

To preprocess the Blog08 corpus, we firstly discarded all HTML tags, and applied DiffPost algorithm to remove non-relevant contents. 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.

$$NoisyInformation(P_i, P_j) = S_i \cap S_j \tag{1}$$

We discarded 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 News Headline Ranking Step

The Top Stories Identification Task aims to find important news headlines for a given date query. Let H be news headline and Q be a date query. To estimate the importance of news headline on a date query, we used a language model framework, widely used for many information retrieval tasks.

$$P(H|Q) \propto \underbrace{P(Q|H)}_{\substack{\text{Query}\\ \text{Likelihood}}} \quad \underbrace{P(H)}_{\substack{\text{Headline}\\ \text{Prior}}} \tag{2}$$

That is, we assume that for a given date query the importance of news headlines can be estimated using the probability that a query language model generates each news headline. We evaluate each component using only Blog08 corpus and news headline corpus without resorting to any external resources.

3.1 The Query Likelihood

For the query likelihood, we should estimate two language models, the Query Language Model (QLM) and the News Headline Language Model (NHLM). Both language models are estimated using the contents of blog posts.

The Query Language Model We gather blog posts between -3 and +7 days for a given query (date query). We believe that if a news headline is important on a given day, many blog posts relevant to the news topic were posted near the day.

The gathered documents contain not only the information relevant to the news topic but also background information or non-relevant topics. We assume that the documents are generated by a mixture model of the QLM and the collection language model. Let $QD = \{d_1, d_2, \cdots, d_n\}$ be the documents between -3 and +7 days for a given query.

$$P(QD) = \prod_i \prod_w ((1-\lambda)P(w|\theta_{QLM}) + \lambda P(w|\theta_C))^{c(w;d_i)} \tag{3}$$

where θ_{QLM} is a query language model, $P(w|\theta_C) = \frac{ctf_w}{|C|}$: $ctfw$ is the number of times term w occurred in the entire collection, $|C|$ is the length of the collection, $c(w;d_i)$ is the count of a word w in a document d_i and λ is a weighting parameter [1].

Then, we can estimate θ_{QLM} using the EM algorithm [1]. The EM updates for $P(w|\lambda_{QLM})$ are as follows:

$$t_w^n = \frac{(1-\lambda)P^n(w|\theta_{QLM})}{(1-\lambda)P^n(w|\theta_{QLM}) + \lambda P^n(w|\theta_C)} \tag{4}$$

$$P^{n+1}(w|\theta_{QLM}) = \frac{\sum_{j=1}^n c(w;d_j)t_w^n}{\sum_i \sum_{j=1}^n c(w;d_j)t_{w_i}^n} \tag{5}$$

The News Headline Language Model To learn the NHLM, for each news headline, we first retrieved blog posts relevant to its topic. We evaluate the relevance between a news headline H and a blog post d using the KL-divergence language model [4] with Dirichlet smoothing [8].

$$Score(H,d) = -\sum_w P(w|H) \log \frac{P(w|H)}{P(w|d)} \tag{6}$$

where $P(w|H)$ is the maximum likelihood estimates of the news headline, and $P(w|d) = \frac{c(w;d)+\mu_d P(w|\theta_C)}{|d|+\mu_d}$: μ_d is a smoothing parameter [2].

We gather only blog posts between -7 and +28 days for a given date query among the search results. We then choose 10 supporting relevant posts. When selecting 10 supporting posts, we want them to capture diverse aspects or opinions relevant to the news story. To achieve this, we propose two different approaches.

One is the Feed-Based Selection (FBS) that selects supporting relevant posts based on feed information which belongs to them. This approach selects 10 supporting relevant posts from as various blog feeds as it possible. To this end, we choose at most two blog posts from each blog feed according to their relevance scores from Eq. 6.

The other is the Cluster-Based Selection (CBS) which first groups search results into 10 clusters and chooses one representative document from each cluster. To cluster search results, we use K-Medoid clustering algorithm [3], and J-Divergence [6] as the distance function.

$$J(d_i||d_J) = \sum_w p(w|\theta_i) log \frac{p(w|\theta_i)}{p(w|\theta_j)} + \sum_w p(w|\theta_j) log \frac{p(w|\theta_j)}{p(w|\theta_i)} \tag{7}$$

[1] In our runs, we set $\lambda = 0.7$

[2] In our runs, we set $\mu_d = 1000$

We estimate the NHLM using the maximum likelihood estimate of the 10 supporting relevant posts and the Dirichlet smoothing [8].

Let θ_{NHLM} and θ_C be the NHLM and the collection language model, respectively. Let *SD* be a set of the 10 supporting relevant posts.

$$P(w|\theta_{NHLM}) = \frac{c(w;SD) + \mu P(w|\theta_C)}{|SD| + \mu} \tag{8}$$

where $c(w;SD)$ is the count of a word w in the document set *SD* and μ is a smoothing parameter [3].

The Score Function To evaluate the query likelihood, we use KL-divergence language model [4] to rank news headlines in response to a given date query.

Let $Score_{QLH}(Q,H)$ be the relevance score of the news headline H with respect to a given date query Q.

$$Score_{QLH}(Q,H) = \sum_w P(w|\theta_{QLM}) log P(w|\theta_{NHLM}) + const(Q) \tag{9}$$

3.2 The News Headline Prior

We propose two criteria to estimate the news headline prior that it will be a top story for a given day. We consider these criteria as the priors of a news headline in that they are independent of the query language model.

The Temporal Profiling The Temporal Profiling criterion uses time information of blog posts relevant to each news headline. We made a temporal profile of each news headline using a temporal profiling approach that Diaz and Jones [2] proposed. The temporal profile of the news headline H is defined as follows:

$$P(t|H) = \sum_{d \in R} P(t|d) \frac{P(H|d)}{\sum_{d' \in R} P(H|d')} \tag{10}$$

where R is the document set which contains top 500 blog posts among the search results from Eq. 6, and $P(H|d)$ is approximated using $Score(H,d)$, and

$$P(t|d) = \begin{cases} 1 \text{ if } t \text{ is equal to the document date} \\ 0 \text{ otherwise} \end{cases}$$

We then smooth the temporal profile $P(t|H)$ using the background model as follows:

$$P(t|H) = (1-\alpha)P(t|H) + \alpha P(t|C) \tag{11}$$

where α is a smoothing parameter and $P(t|C) = \frac{1}{|D|} \sum_{d \in C} P(t|d)$: $|D|$ is the total number of documents in the entire collection.

[3] We set μ=2000

This temporal profile is defined at each single day. However, the blog posts about the important news story may occur over a period of several days. Therefore, we smooth the temporal profile model with the model for adjacent days. Let $Score_{TP}(H)$ be the value of a news headline estimated using the temporal profile of the news headline.

$$Score_{TP}(Q,H) = \frac{1}{|\phi|}\sum_{\phi_i} P(t+\phi_i|H) \tag{12}$$

where ϕ indicates the period [4].

The Term Importance The Term Importance criterion uses term information of each news headline. We believe that respective terms have a different importance for a given date query. If a headline consists of more important terms, it is more likely to be a top story. For example, a headline consisted of common words or stopwords may not a top story.

To estimate the term-based evidence, we first extracted all NP phrases from each news headline using Stanford Parser [5]. We then gathered all n-gram ($n \leq 3$) from NP phrases. We evaluate the importance of the n-gram terms using two heuristic measures.

One is term frequency measure that is naive, but widely used in the many IR tasks. Intuitively, if a term occurs frequently at a query date, it is more likely to be important. Let nt be n-gram term and $TF(nt,Q)$ be a term frequency measure for a date query Q.

$$TF(nt,Q) = log(1+c(nt;Q)) \tag{13}$$

where $c(nt;Q)$ means the count of a term nt within the news headlines that are issued at the same date with a given date query Q.

The other is distribution within the news headline corpus. We believe that if a term occurrence is concentrated at a query date, it is more likely to be important. Let $DI(nt,Q)$ be the measure of the term distribution for a date query Q. We define $DI(nt,Q)$ as follows:

$$DI(nt,Q) = \frac{c(nt;Q)}{\sum_{t\in T} c(nt;t)} \tag{14}$$

where T indicate a set of days corresponding to timespan covered by the news headline corpus.

Let $Score_{TI}(Q,H)$ be the value of a news headline evaluated based on the term-based evidence.

$$Score_{TI}(Q,H) = \max_{nt}(TF(nt,Q) \times DI(nt,Q)) \tag{15}$$

[4] In our runs, the period ϕ is between -3 and +7 days from the query day Q

[5] http://nlp.stanford.edu/software/lex-parser.shtml

Run	MAP	P@10
KLEFeed	0.0132	0.0345
KLECluster	0.0182	0.0600
KLEFeedPrior	0.1548	0.2800
KLEClusPrior	0.1605	0.2964

Table 1. The performances based on the headlines relevance judgments only

Run	NDCG-alpha@10	intent-aware P@10
KLEFeed	0.066	0.023
KLECluster	0.098	0.037
KLEFeedPrior	0.504	0.162
KLEClusPrior	0.409	0.117

Table 2. The performances based on the blog-post level evaluation

3.3 The Ranking Fuction

From above steps, we can get three scores related to the importance of the news headline. To rank the news headlines, we should combine the score of the query-likelihood with two measures for the headline prior. However, each score has different scale. Therefore, a value of each score is scaled from 0 to 1. Finally, our ranking function is as follows:

$$\begin{aligned} Score(H,Q) &= (1-\beta_1)Score_{QLH}(Q,H) \\ &+ \beta_1\{(1-\beta_2)Score_{TP}(Q,H)+\beta_2 Score_{TI}(Q,H)\} \end{aligned} \quad (16)$$

where β_1 is the weighting parameter that controls the importance between the query likelihood and the headline prior components, and β_2 controls the weight between two evidences for the headline prior.

4 Runs

In the Top Stories Identification Task, we submitted 4 runs as follows:

1. **KLEFeed** chooses 10 supporting blog posts using the FBS to estimate the NHLM, and does not use the news headline prior, that is, $\beta_1 = 0$.
2. **KLECluster** chooses 10 supporting blog posts using the CBS to estimate the NHLM, and does not use the news headline prior.
3. **KLEFeedPrior** chooses 10 supporting blog posts using the FBS to learn the NHLM, and uses the news headline prior with $\beta_1 = 0.7, \beta_2 = 0.2$.
4. **KLEClusPrior** chooses 10 supporting blog posts using the CBS to learn the NHLM, and uses the news headline prior with $\beta_1 = 0.7, \beta_2 = 0.2$.

Table 1 and 2 shows the performances of our runs based on the headlines relevance judgments only and based on the blog-post level evaluation, respectively.

5 Conclusions

We have described our participation in the TREC 2009 Blog Track. For Top Stories Identification Task, we presented two components: the query likelihood and the news headline prior, based on the language model framework. To evaluate the query likelihood, we estimated the query language model and the news headline language model using the contents of blog posts. We also proposed two methods to choose the 10 supporting blog posts for each news headline: FBS and CBS. Furthermore, we suggested two criteria to estimate the news headline prior. One is Temporal Profiling criterion that uses time information of blog posts relevant to each headline. The other is Term Importance criterion that uses term information of a news headline. Experimental results show that using the news headline prior significantly improves the performance.

6 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 2009.

References

1. Dempster, A.P., Laird, N., Rubin, D.: Maximum likelihood from incomplete data via the em algorithm. In: Journal of the Royal Statistical Society. (1977) 39(B):1–38
2. Diaz, F., Jones, R.: Using temporal profiles of queries for precision prediction. In: SIGIR '04: Proceedings of the 27th annual international ACM SIGIR conference on Research and development in information retrieval, ACM (2004) 18–24
3. Kaufman, L., Rousseeuw, P.J.: Finding groups in data: an introduction to cluster analysis. John Wiley and Sons, New York (1990)
4. Lafferty, J., Zhai, C.: 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, ACM (2001) 111–119
5. Lee, Y., Na, S.H., Kim, J., Nam, S.H., young Jung, H., Lee, J.H.: Kle at trec 2008 blog track: Blog post and feed retrieval. In: Proceedings of TREC 2008. (2008)
6. Lin, J.: Divergence measures based on the shannon entropy. IEEE Transactions on Information theory **37** (1991) 145–151
7. Nam, S.H., Na, S.H., Lee, Y., Lee, J.H.: Diffpost: Filtering non-relevant content based on content difference between two consecutive blog posts. In: ECIR. Volume 5478 of Lecture Notes in Computer Science., Springer (2009) 791–795
8. Zhai, C., Lafferty, J.: A study of smoothing methods for language models applied to information retrieval. ACM Trans. Inf. Syst. **22**(2) (2004) 179–214