Language Modeling Approaches to Blog Post and Feed Finding

Breyten Ernsting **Wouter Weerkamp** **Maarten de Rijke**

ISLA, University of Amsterdam
Kruislaan 403, 1098 SJ Amsterdam
http://ilps.science.uva.nl/

Abstract: We describe our participation in the TREC 2007 Blog track. In the opinion task we looked at the differences in performance between Indri and our mixture model, the influence of external expansion and document priors to improve opinion finding; results show that an out-of-the-box Indri implementation outperforms our mixture model, and that external expansion on a news corpus is very benificial. Opinion finding can be improved using either lexicons or the number of comments as document priors.

Our approach to the feed distillation task is based on aggregating post-level scores to obtain a feed-level ranking. We integrated time-based and persistence aspects into the retrieval model. After correcting bugs in our post-score aggregation module we found that time-based retrieval improves results only marginally, while persistence-based ranking results in substantial improvements under the right circumstances.

1 Introduction

We describe our experiments for this year's edition of the Blog track. Our main aims were (1) to compare our inhouse mixture model against an Indri-based baseline for topical blog post retrieval, and (2) for the distillation task to examine the influence of time and frequency of posting about a given topic on retrieval effectiveness. In two largely independent sections we first discuss our work on the opinion finding task (Section 2) and then our work on the feed distillation task (Section 3). We conclude in Section 4.

2 Opinion Finding

The opinion finding task aims at returning blog posts that contain an opinion regarding a certain topic [7]. The results of last year's opinion finding task indicate that a strong topical retrieval system is the single most important part of opinion finding. In Section 2.1 we present the models we use for topical retrieval and the usage of external expansion to improve topical blog post retrieval is discussed in Section 2.2. Section 2.3 shows the implementation of opinion finding indicators. Finally we present run details (Section 2.5) and results (Section 2.6).

2.1 Topic Retrieval

Our baseline approach to the topical blog post retrieval task uses language models. The models we use in this track are similar to the models we use in the TREC Enterprise track and are described more fully in [1].

2.1.1 Indri

For comparison we use an out-of-the-box implementation of Indri 2.4[1] to process the 2007 topics. As preprocessing steps we strip all HTML tags and remove stopwords; we do not apply stemming.

2.2 External Expansion

Our baseline query model $p(t|q)$ is simply estimated using $p(t|q) = n(t,q) \cdot |q|^{-1}$, with $n(t,q)$ being the number of occurrences of term t in query q, and $|q|$ the query length.

To improve topical retrieval performance we use relevance models [2]; the relevance models are constructed using feedback documents and return feedback terms with an associated weight. Instead of constructing the relevance models based on the top K documents of the blog collection we use insights from [6] stating that many queries issued on blog search engines are in fact news related. We use the (contemporaneous) AQUAINT-2 news corpus to construct relevance models; from the top 40 document, we select the top 10 terms. We normalize their weights so that they sum to 1. The weighted query is issued against the blog collection to retrieve the final set of blog posts.

2.2.1 Query Rewriting

For expansion on the Indri run, we also apply the query rewriting strategies proposed in [5]: individual terms of multiple-term queries are combined into phrases using the Indri query language. A query like "sci fi channel" is rewritten to combinations of *sci fi*, *fi channel*, *sci fi channel*, etc.

[1] http://www.lemurproject.org/indri/

2.3 Document Priors

On top of our topical retrieval method, we implement opinion finding methods using query independent document priors ($p(d)$). We believe blog posts have a degree of opinionatedness regardless of the topic; besides, this approach does not have a negative impact on the speed of the retrieval system.

The main issue, then, is how to estimate the document priors. We compare two document priors for opinionatedness: (1) a lexical approach and (2) a comment-based approach. The lexical approach uses a list of opinionated words from the OpinionFinder system.[2] From this list we extract only the strong positive and strong negative words. The document prior is then estimated as given in Eq. 1:

$$p(d) = \sum_{i=1}^{w} c(w_i, d) \cdot |d|^{-1}, \quad (1)$$

where w is the list of opinionated words and $c(w_i, d)$ is the count of opinion word i in document d and $|d|$ is the document length in words.

Our second, comment-based approach is based on the intuition that opinionated blog posts are more likely to attract discussion. When a post contains an explicit point of view on a topic, readers are more likely to comment (either by agreeing or by expressing their own opinion on the topic). Using this idea, we estimate document priors as follows:

$$p(d) = \log(N_{comments,d}), \quad (2)$$

where $N_{comments,d}$ is the number of comments in document d.

2.4 Polarity

New in this year's opinion task is the polarity subtask: given an opinionated post we need to identify whether it is either negative, positive or neutral. To address this task we experiment with two approaches. The first approach continues on the opinionated words list of the previous section. We use the two opinionated lists separately (positive and negative words) and use Eq. 3 to estimate the polarity of each post:

$$pol(d) = \begin{cases} positive & \text{if } r(d) > 0.01 \\ negative & \text{if } r(d) < -0.01 \\ neutral & \text{otherwise,} \end{cases} \quad (3)$$

where $r(d)$ is defined as

$$r(d) = (\sum_{i=1}^{n} c(n_i, d) - \sum_{i=1}^{p} c(p_i, d)) \cdot |d|^{-1}, \quad (4)$$

in which n is the list of negative words, p the list of positive words, $c(n_i, d)$ the number of times word n_i occurs in document d and $|d|$ the document length in words.

For our second approach we look at expressions of opinion other than words. The idea is that posts containing more expressive language tend to be more negative about the topic discussed in that post. To estimate this negativity, we use the following indicators: exclamation marks, question marks, ellipsis (...), and all caps strings of more than 3 characters. The ratio of these indicators is calculated for each blog post using Eq. 5, where $c(i, d)$ is the total number of occurrences of the above indicators in document d and $|d|$ is the document length in words. Polarity is estimated according Eq. 6:

$$r(d) = c(i, d) \cdot |d|^{-1} \quad (5)$$

$$pol(d) = \begin{cases} positive & \text{if } r(d) < 0.1 \\ negative & \text{if } r(d) \geq 0.1 \end{cases} \quad (6)$$

2.5 Runs

For the runs using the mixture model we use the 2006 topics and assessments to estimate the best mixture weights. Results show that only the title component has a positive influence on retrieval performance and the best mixture is estimated to be 0.15 title, 0.60 body text and 0.25 background.

(A) **`uams07indbl`** the baseline run uses an out-of-the-box Indri implementation.

(B) **`uams07topic`** the topic run also uses Indri out-of-the-box; queries are first rewritten and then expanded using the external corpus (as described in Section 2.2).

(C) **`uams07mmbl`** the baseline mixture model run: the best mixture as tested on the 2006 data without additional features.

(D) **`uams07mmq`** identical to the previous run; instead of the baseline query model relevance models are used based on feedback on a news corpus.

(E) **`uams07mmqcom`** identical to the previous run; to identify opinionated posts document priors based on the number of comments are included as discussed in Section 2.3

(F) **`uams07mmqop`** identical to run uams07mmq; document priors based on the ratio of opinionated words (Section 2.3) are used to estimate opinionatedness.

(G) **`polarity`** runs uams07topic, uams07mmbl and uams07mmqop use the opinionated words ratio as polarity indicator. Runs uams07mmq and uams07mmqcom use the punctuation-based polarity identifier (see Section 2.4).

2.6 Results

We look at the performance of our runs on topical retrieval, opinion retrieval and polarity identification. Table 1 shows the MAP and p@10 scores for all runs on the retrieval tasks and the R-accuracy[3] on the polarity identification [4].

[2] URL: http://www.cs.pitt.edu/mpqa/

[3] The R-accuracy is the fraction of retrieved documents above rank R that are classified correctly, where R is the number of opinion-containing documents for that topic.

Table 1: Blog post retrieval results

	topic		opinion		polarity
run id	MAP	p@10	MAP	p@10	R-acc.
A	0.4342	0.6800	0.3281	0.4920	–
C	0.3105	0.6060	0.2123	0.3840	0.1284
B	**0.4741**	**0.7620**	**0.3453**	**0.5620**	**0.1827**
D	0.1898	0.4500	0.1273	0.2520	0.0711
F	0.1865	0.4720	0.1459	0.3200	0.0840
E	0.1834	0.4620	0.1302	0.2900	0.0677

From Table 1 it is clear that run B (uams07topic, Indri with external expansion) performs best on all tasks.

Surprisingly, the mixture model runs with external expansion perform significantly worse than the baseline run, even though we see an improvement of external expansion on the Indri runs; it appears that the external expansion is not performed correctly, leading to expanded queries that miss original query terms. Topic 902 (*lactose gas*) provides an example: the expanded query contains the terms gas, contamination, water, underground, solution, tce, eaten, whey, study, atoms, but the original query term lactose is missing. The performance decreases dramatically from 0.4127 in the baseline run to 0.003 in the expanded runs. This error occurs in about half of the topics, making the final run scores (both on MAP and precision) much lower. An example of a topic that is expanded correctly, is topic 947 (*sasha cohen*). It achieves an AP score of 0.2897 in the baseline run, which increases to 0.6371 in the expanded runs. Similar effects are noticeable in other correctly expanded queries.

Both the lexicon and comment-based approach have a positive influence on opinion retrieval, with the lexicon approach outperforming the comment-based approach. Finally, polarity detection based on the difference between positive word ratio and negative word ratio performs only slightly better than the punctuation approach, even though the latter distinguishes only positive and negative posts and ignores neutral posts.

3 Feed distillation

The feed distillation task is a new task and it aims at finding feeds that are devoted to a given topic. The general idea is that a user can be presented with a suggested list of feeds that are worth reading, given the topic. Although the task is new, it resembles the Expert Search task in the Enterprise track, and the Topic Distillation task in the Web track.

Below, we present the models we used for topic distillation, and for incorporating aspects of time and persistence in the retrieval model, to improve the accuracy of our feed distillation method.

3.1 Topic retrieval

The method we use to address topic retrieval for the feed distillation task is based on ranking individual posts contained in the feed; it is akin to the method in Section 2.1 (Eq. 1–5). However, $p(t|\theta_d)$ is estimated simply as follows:

$$p(t|\theta_d) = p(t|d), \tag{7}$$

where $p(t)$ is the maximum likelihood-estimate of the term t in the document collection.

3.2 Time-based Reweighing

To improve the accuracy of our topical retrieval system, we incorporate query independent document priors which are based on the creation date of the documents. More recent documents are assumed to better reflect the current interests of a feed (blogger), and that these should therefore rank higher. We model this intuition using a time-based language model [3]:

$$p(d|q) \propto p(q|d)p(d|T_d), \tag{8}$$

where $p(d|T_d)$ is a time-based prior in the model. Since we are interest in *recency* and not a specific *event*, we use an exponential distribution to calculate the priors. This distribution is defined as follows:

$$p(d|T_d) = P(T_d) = \lambda e^{-\lambda(T_C - T_d)} \tag{9}$$

The optimal value of the parameter λ is determined in training experiments. T_C and T_d are measured in days; T_C signifies the most recent date of the documents in the collection, and T_d refers to the document being considered.

Using time-based document priors only works when using blog posts as the unit of retrieval. In order to derive a ranked list of feeds (as required) from a ranked list of blog posts, we take the score of the highest ranked n blog posts of a feed as follows:

$$score(bl|q) = \sum_{r=1}^{r=n} \sum_{d} R(r,d) \cdot p(d|q) \cdot n^{-1}, \tag{10}$$

where bl is a feed, and d ranges over blog posts in bl. $R(r,d)$ is equal to 1 if the rank of document d is equal to r, 0 otherwise.

3.3 Persistence-based Reweighing

Another way to improve the accuracy of our retrieval system is to consider the frequency of on-topic blog posts for feeds. We assume that the number of matching blog posts is proportional to the interest of a blogger in the topic. Consequently, we can derive a ranking of feeds according to the frequency of blog posts matching the topic in question. However, this ignores the fact that some feeds may contain a disproportionate number of blog posts, in comparison to other feeds.

Table 2: Feed distillation results

run id	MAP	p@10	p@30
uams07bdtop	0.1589	**0.3111**	0.2141
uams07bdtblm	**0.1605**	0.3067	0.2156
uams07bdfreq	0.1248	0.2467	**0.2170**

We therefore consider the number of on-topic blog posts in the feed versus the number of blog posts, either matching the topic or not, to score the feeds.

We incorporate this "persistence score" in our retrieval model using a linear combination of both the topic-based score and the persistence score. The topic-based score per blog post is calculated as detailed in Section 3.1 and the score per feed is calculated similar to Eq. 10 (i.e., we take the highest scoring blog post per feed). This leads to:

$$p(bl|q) = \lambda \cdot p_c(bl|q) + (1 - \lambda) \cdot p_f(bl|q), \quad (11)$$

where $p_c(bl|q)$ denotes the topic-based score for the feed given the topic, and $p_f(bl|q)$ denotes the persistence score for the feed given the topic; note that $p_c(bl|q)$ is the same as in Eq. 10, while $p_f(bl|q)$ is defined as follows:

$$p_f(bl|q) = \sum_{d \in bl} R(d|q) \cdot |bl|^{-1} \quad (12)$$

Here, $R(d|q) = 1$ if $p(d|q) > 0$, and $R(d|q) = 0$ otherwise; $|bl|$ denotes the number of blog posts in the blog bl.

3.4 Runs

The following runs were submitted:

uams07bdtop uses the topical blog post retrieval model as described in Section 3.1 and aggregates blog posts to a feed according to Eq. 10.

uams07tblm uses the time-based retrieval model as described in Section 3.2. Training experiments showed the $\lambda = 0.04$ was the optimal setting for the exponential function.

uams07bdfreq uses the persistence retrieval model as described in Section 3.3. $\lambda = 0.5$ was the setting used in this run.

3.5 Results

We now consider the performance of our runs for the feed distillation task. The results are displayed in Table 2, which shows, for each run, the MAP scores, as well as the p@10 and p@30 scores.

Unfortunately, well after the submission, it emerged that there was a bug in the aggregation module that implements Eq. 10; essentially, it equated the score of a feed with that of its single best scoring post (instead of aggregating the score from the top n posts). In Table 3 we report on results with the corrected aggregation module. We find that taking an aggregate of the top 8 posts to compute the score of the feed tends to yield the best performance, for all measures considered. A topic-level analysis revealed that 8 is optimal, not just on average, but for nearly all individual topics.

n	MAP	p@10	p@30
1	0.1849	0.3267	0.2274
2	0.2740	0.4467	0.3037
3	0.3061	0.4844	0.3363
4	0.3198	0.4978	0.3459
5	0.3289	0.5133	0.3578
6	0.3328	0.5156	0.3615
7	0.3345	0.5178	0.3689
8	**0.3354**	**0.5222**	**0.3733**
9	0.3344	0.5089	0.3711
10	0.3329	0.4978	0.3681
15	0.3259	0.5000	0.3719

Table 3: Evaluation results obtained by ranking feeds based only on aggregating different numbers of posts; n is the number of posts considered. Based on a corrected implementation of Eq. 10.

The time-based prior improved only slightly over the best performing relevance-only baseline (based on aggregating $n = 8$ posts). When we further integrate the output of the (corrected) aggregation module with persistence-based scoring, i.e., recreating the run labeled `uams07bdfreq`, but now based on the corrected aggregation module, the best scores we are able to achieve are 0.3629 (MAP; +8.2% over the best scoring relevance-only baseline in Table 3), 0.5289 (p@10; +1.3%), and 0.4022 (p@30; +7.7%); the improvements in MAP and p@30 are significant.

In sum, then, after implementing our bug fixes we found that feeds can be ranked effectively by considering a small set of posts only, that the time-based prior leads to minor improvements, but that the persistence-based score leads to substantial gains in effectiveness.

4 Conclusions

In this paper we described our participation in the TREC 2007 Blog track. Our aim for the opinion finding task was to experiment with Indri and a mixture model. Result show that Indri significantly outperforms the mixture model. External expansion using a news corpus leads to improvement over the Indri baseline run, although bugs in the implementation caused decreased performance in the mixture model. Opinion finding by means of document priors shows beneficial, especially in case of lexicons. Overall we can conclude that opinion finding is highly dependent on topical retrieval and that focus still should be on this aspect: opinion detection can be done using lexicons, but non-lexical features also show promising results.

As to the feed distillation task, our (corrected) results

show that using time-based document priors improved slightly over the baseline run. Incorporating a persistence score based on the relative frequency with which a blogger posts about a given topic, led to further significant improvements.

Acknowledgments

Maarten de Rijke was supported by NWO under project numbers 017.001.190, 220-80-001, 264-70-050, 354-20-005, 600.065.120, 612-13-001, 612.000.106, 612.066.302, 612.069.006, 640.001.501, 640.002.501, and by the E.U. IST programme of the 6th FP for RTD under project MultiMATCH contract IST-033104.

References

[1] K. Balog, K. Hofmann, W. Weerkamp, and M. de Rijke. Query and document models for enterprise search. In *This Volume*, 2008.

[2] V. Lavrenko and W. B. Croft. Relevance-based language models. In *Proceedings of the 24th Annual international ACM SIGIR Conference on Research and Development in information Retrieval (SIGIR '01)*, pages 120–127. ACM Press, 2001.

[3] X. Li and W. Croft. Time-based language models. *Proceedings of the 12th International Conference on Information and Knowledge Managment (CIKM)*, pages 469–475, 2003.

[4] C. Macdonald, I. Ounis, and I. Soboroff. Overview of the trec 2007 blog track. In *This Volume*, 2007.

[5] D. Metzler and W. B. Croft. A markov random feld model for term dependencies. In *SIGIR '05: Proceedings of the 28th annual international ACM SIGIR conference on research and development in information retrieval*, pages 472–479, New York, NY, USA, 2005. ACM Press.

[6] G. Mishne. *Applied Text Analytics for Blogs*. PhD thesis, University of Amsterdam, 2007.

[7] I. Ounis, M. de Rijke, C. Macdonald, G. Mishne, and I. Soboroff. Overview of the TREC-2006 Blog Track. In *The Fifteenth Text REtrieval Conference (TREC 2006) Proceedings*. NIST, 2007.