USC/ISI at TREC 2011: Microblog Track

Donald Metzler and Congxing Cai

Information Sciences Institute
University of Southern California
Marina del Rey, CA, USA

Abstract

This paper describes the search system we developed for the inaugural TREC 2011 Microblog Track. Our system makes use of best-practice ranking techniques, including term, phrase, and proximity-based text matching via the Markov random field model, pseudo-relevance feedback using Latent Concept Expansion, and a feature-based ranking model that uses a simple, but effective learning-to-rank model. We adapted each of these approaches to the specifics of the microblog search task, giving rise to a highly effective end-to-end search system. The official results from the TREC evaluation suggest that pseudo-relevance feedback and learning-to-rank yield significant improvements in precision at early rank under different evaluation scenarios.

1 Introduction

Microtexts represent a relatively new, but nearly ubiquitous communication medium that has begun replacing traditional "long" forms of communications, such as email and blogs. Popular examples of microtexts include SMS text messages, status updates, and microblog messages. Such texts exhibit a number of unique properties that necessitate the development of novel ranking techniques and evaluation methodologies. In order to catalyze research along these directions, a new TREC track, called the Microblog Track, was run as part of TREC 2011.

The Microblog Track search task is technically defined as temporally-biased *ad hoc* search over a stream of microblog (Twitter) messages. That is, information needs are defined in terms of a *keyword query* and a *temporal reference point*. It is assumed that the user issued the keyword at the temporal reference point and is looking for microblog posts that contain information that is both *recent* and *relevant*.

This search task differs from previous TREC *ad hoc* search tasks (e.g., news search, Web search, etc.) in a number of important ways, thereby giving rise to a number of interesting research problems. We specifically focused on the following task-specific research challenges while developing our system:

- **Very short documents.** Microblog messages are short, by their very definition. For example, Twitter messages are limited to 140 characters of content. This hard limit, along with other contextual factors, causes users to make heavy use of abbreviations, phonetically shorten terms, drop vowels, etc [4]. Therefore, microblog messages exhibit a great deal of lexical variation that only exacerbates the vocabulary mismatch problem that plagues information retrieval systems. To help overcome this issue, we use pseudo-relevance feedback to build an expanded, lexically richer query representation.

- **Highly varied document quality.** User generated content, including microblog messages, vary greatly in terms of content quality. While some authors pride themselves in producing high quality content, others create content that is barely decipherable. Since it is unlikely that low quality messages will yield much valuable information, we developed a number of features that quantify the quality of microblog content.

- **Language identification issues.** Microblogs are inherently multi-lingual. However, for the purpose of this track, all non-English messages were considered non-relevant. Therefore, accurate language identification is important, not only for this track, but for real microblog search systems, as well. Instead of imposing a hard filtering of the messages, we used the confidence score output by a highly effective SVM-based English language classifier as a feature within our ranking functions.

- **Temporally-biased queries.** The track required that results satisfy two temporal conditions. First, only tweets that were issued before the query's temporal reference point could be returned. Second, the results must be returned in *reverse chronological order* (i.e., most recent first). To handle these requirements, we made use of a modified version of the Indri search engine [12]. Our modifications also ensured that all collection statistics were computed using only "past" information.

- **Retrieval metrics.** Recency and relevance are both critical factors when evaluating microblog search results. However, given the insufficiency of existing metrics, the official evaluation metric of this year's track was precision at rank 30 (P@30). This metric focuses entirely on relevance and completely ignores recency. Therefore, all of our models were optimized for P@30. However, it is important to note that the learning framework we use to tune our models is flexible enough to easily optimize for other metrics, such as those that combine relevance and recency.

- **Lack of training data.** Since this is the first year that the Microblog Track was run, no existing training data was available to help tune the parameters of our ranking functions. To overcome this issue, we recruited a few volunteers to help us construct a small training set of 15 queries and 346 binary judgments. Although small, the training set was successfully used to learn effective learning-to-rank models.

The remainder of this paper describes the details of our system and how we addressed each of these challenges.

2 Methods

Given that this was the first year that the Microblog Track was run, we focused almost exclusively on establishing a highly effective baseline system that leverages existing state-of-the-art retrieval approaches and upon which more advanced capabilities and improvements can be built moving forward. Along these lines, the following approaches were used:

- The **Markov random field retrieval model** [9] forms the basis of our text-based scoring functions (Section 2.1).

- **Latent Concept Expansion** [10] is used to help overcome the vocabulary mismatch problem (Section 2.2).

- A simple, but effective **learning-to-rank model** [6] is used to combine evidence from multiple features (Section 2.3).

Our four official runs represent different combinations of these basic approaches. We now provide a brief description of each approach.

2.1 Text Scoring

Our text-based scoring function makes use of the Markov random field retrieval model [9]. The model can capture dependencies between terms and provides a formal and highly effective framework for combining scores from term, phrase, and proximity-based text matching features.

Specifically, in this work, we use the *full dependence variant* of the MRF model, which assumes that all terms are dependent on each other [9]. Rather than describe the technical details of the model, we provide an example of how the model is applied to the query *moscow airport bombing* (Topic 36). It can be shown the following query, expressed in the Indri query language [12], ranks documents according to the full dependence model [9]:

```
#weight(
  0.8 #combine(moscow airport bombing)
  0.1 #combine(#1(airport bombing)
                  #1(moscow airport)
                  #1(moscow airport bombing))
  0.1 #combine(#uw8(airport bombing)
#uw8(moscow bombing)
#uw8(moscow airport)
#uw12(moscow airport bombing)))
```

where #weight and #combine are Indri query language operators that combine scores from term matches, phrase matches (#1), and proximity matches (#uwN). It has been empirically shown that this particular way of combining text matching evidence is highly effective for a variety of tasks [8].

Readers who may be familiar with the MRF model will note that most previous studies have used the *sequential dependence variant* of the model, since it typically achieves comparable effectiveness to the full dependence variant, but is substantially more efficient at runtime. However, our preliminary experiments suggested that the full dependence model yielded superior results compared to the sequential dependence model. We hypothesize this is the case because the full dependence model promotes tweets that match as many query terms and as many exact matching subsequences of query terms as possible, whereas the sequential dependence model is less aggressive since it only looks at phrase and proximity features defined over adjacent query terms. While the sequential dependence model may suffice for long documents, our findings suggest that the full dependence model is better for ranking short noisy documents. In the future, we plan to undertake a more detailed empirical evaluation to develop a better understanding of this phenomenon.

2.2 Query Expansion

As we described in the introduction, the fact that Twitter messages are so short only exacerbates the so-called *vocabulary mismatch problem*. The most common approaches for overcoming the lexical gap between queries and documents (tweets) are query expansion and document expansion. In this work, we focus our attention on query expansion. To expand queries, we make use of Latent Concept Expansion (LCE), an effective pseudo-relevance feedback technique developed specifically for the MRF retrieval framework [10]. LCE is a generalization of relevance-based language models [5] that permits dependencies between terms to be modeled and arbitrary features to be used for pseudo-relevance feedback.

To illustrate the potential benefits of query expansion within the microblog domain, consider the query *oprah winfrey half sister* (Topic 13), which concerns the revelation that media mogul Oprah Winfrey had a half-sister she never knew about. For this query, the top 10 expansion terms returned by LCE are:

> “oprah”, “winfrey”, “she”, “secret”, “AP” (Associated Press), “family”, “harvey”, “reveal”, “watching”, “announce”

To understand why these terms are chosen by LCE, consider what a typical tweet on this subject may look like. For example, it is likely to have the form “Oprah Winfrey [**announced** | **revealed**] that **she** has a half-sister (**AP** news)”. Therefore, these terms are capturing the most salient terms (named entities, pronouns, news sources, verbs, etc.) that are used when describing this particular topic. By expanding the original query with these terms, it is possible to identify tweets that may not contain all (or any) of the original query terms, but are still relevant.

All of our LCE-based runs use 100 feedback tweets, 10 expansion terms, and weight the contribution of the original query and expansion terms equally. Based on a preliminary analysis of our results, we believe that the LCE parameters used were far from optimal, which likely limited the potential gains of the technique. As part of future work we will undertake a more detailed tuning of the parameters to unlock the full potential of the approach.

We also hypothesize that external sources of expansion evidence, such as Wikipedia, query logs, and temporally-aligned news corpora, would also be useful for constructing expanded query representations. We also believe that temporally-biased expansion models, such as the one proposed by Massoudi et al., could prove to be effective [7]. These are areas of potentially fruitful future work.

2.3 Learning-to-Rank

Up until this point, we have focused exclusively on approaches for computing highly effective text matching scores for Twitter messages. However, as we described in the introduction, tweets exhibit a high variance in content quality and are written in a variety of languages beyond English. Therefore, it is necessary to combine multiple sources of evidence (e.g., text scores, content quality scores, language identification confidences, etc.) to facilitate effective microblog ranking.

There are many different ways to combine evidence, including result set fusion [3], inference networks [13], and learning-to-rank approaches [6]. In this work, we use a simple, but effective learning-to-rank approach for combining evidence from multiple features. Another reason for choosing this particular paradigm is because learning-to-rank approaches have been shown to be effective for Twitter search in the past [2].

The remainder of this section describes the learning-to-rank model we used, our feature set, and how the model parameters are estimated, respectively.

2.3.1 Model

We utilize a simple linear learning-to-rank model. Given a query Q and a tweet D, the model computes a relevance score $s(Q, D)$ according to:

$$s(Q, D) = \sum_{i}^{N} \lambda_i f_i(Q, D)$$

where N is the total number of features, $f_i(Q, D)$ is a feature function, and λ_i is a model parameter. Given a query Q, tweets D are ranked in descending order of their relevance score $s(Q, D)$.

To instantiate the model, we must define a set of features ($f(Q, D)$) and estimate the model parameters (λ).

2.3.2 Features

As we will describe in more detail in the next section, the Microblog Track required participants to crawl their own data set. Two versions of the crawler were available. One version required special access to theTwitter APIs and provided a feature-rich JSON representation of each tweet, while the other crawler downloaded a barebones HTML representation of the tweets. Since we did not have special access to the Twitter API, we download the basic HTML version of the data, which limited the types of features we could use within our model.

We used the following set of features, some of which were inspired by the work of Duan et al. [2], while others are novel:

- `text_score(Q,D)` - Text matching feature. For runs that do not use LCE, this is the MRF model's full dependence score computed for the query Q and tweet D. For runs that use LCE, this is the score of the expanded query with respect to tweet D.
- `tdiff(Q,D)` - Time difference feature. The difference in time, as measure in seconds, between the query Q's temporal reference point and tweet D's timestamp.
- `has_hashtag(D)` - Does the tweet D contain a hashtag? (binary valued)
- `has_url(D)` - Does the tweet D contain a URL? (binary valued)
- `length(D)` - The length (number of terms) of the tweet D.
- `oov_pct(D)` - The percentage of terms in tweet D that are out-of-vocabulary (OOV). We use the English Aspell dictionary as our vocabulary.
- `is_reply(D)` - Is the the tweet D a reply to another tweet? (binary valued)
- `english_prob(D)` - The confidence score of our English language classifier. The classifier is a Support Vector Machine (SVM) model trained using a small set of manually labeled tweets. It uses character trigrams and average word length as features, and achieves an accuracy of around 93%.

Due to time restrictions, we did not use the number of times a tweet was re-tweeted or any user-specific information (e.g., authority scores) as features within our model. As part of future work, we are interested in expanding our feature set and developing a better understanding of the relative importance of different types of features.

2.3.3 Parameter Estimation

Since our model only has 8 features, there is no need to resort to overly sophisticated learning-to-rank parameter estimation strategies. Indeed, for such simple models, it is likely best to keep things as simple as possible. In the spirit of simplicity, all of our learning-to-rank models are learned using using a simple coordinate-level ascent approach that directly optimizes P@30, the official Microblog Track metric [11].

Feature	Weight
text_score	0.5549
tdiff	0
has_hashtag	0.0203
has_url	0.1218
length	0
oov_pct	-0.1218
is_reply	-0.1218
english_prob	0.0593

Table 1: Learning-to-rank model feature weights.

Since this is the first time that the TREC Microblog Track was run, we did not have access to any data from which we could train our models. To alleviate this problem, we recruited a few volunteers from within our organization to help us construct a small set of training data. Each volunteer was asked to issue one or more queries (without any knowledge of the TREC Microblog test queries) to a prototype microblog search engine, which ranked tweets based on the text score alone. The volunteers were then asked to annotate the relevance of the top 30 results returned for each query. All relevance judgments were binary and users were allowed to skip (i.e., not judge) results that they were unsure of. This yielded a total of 15 training queries and 346 judgments. Although this training data set is extremely small, it contained enough signal to distill a relatively effective learning-to-rank model, as we will show in the next section.

Using this small training set, we learned the linear ranking function presented in Table 1. We use this ranking function for all of our learning-to-rank runs. By inspecting the weights of the model, we see that the text_score feature provides the most positive evidence in favor of relevance. Other positive indicators of relevance include the has_url (which was also observed by Duan et al. [2]), english_prob, and has_hashtag. On the other hand, the learned model suggests that oov_pct and is_reply are negative indicators of relevance. We found these weights to be intuitive and to match our expectations.

It is worth noting that two features, tdiff and length were assigned a weight of 0, which suggests that they do not provide strong evidence in favor of or against relevance. We suspect that if the metric being optimized for included a recency component, then the tdiff feature would play a more important role in the model. As for the length feature, it is likely that most tweets are more or less the same length, and hence length on its own is not a strong relevance signal.

Tweet Type	Count
200 (OK)	14,579,587
302 (Found)	1,156,771
404 (Not found)	289,886
403 (Forbidden)	115,568
Total (searchable)	15,736,358

Table 2: Tweets2011 corpus summary statistics (HTML version, crawled in May 2011).

3 Experiments

This section describes our experiences with downloading the Twitter corpus, our experimental methodology, and an overview of our results.

3.1 Data and Methodology

As mentioned earlier, the Microblog Track was unique because it required participants to crawl/download the data set on their own. This was necessary to abide by Twitter's terms of service. The track organizers provided participants with a list of user and tweet IDs that make up the data set. They also provided helper scripts for downloading the data set. If a participating group had special access to Twitter's APIs, then they could download the data in a feature-rich JSON format. All other participants had to download the data in a bare-bones HTML format that contains less information than the JSON format. Since we do not have elevated access to Twitter's APIs, we downloaded an HTML version of the data set.

Our crawl was performed in late May 2011 on an Intel i7 processor, 16GB of RAM, running Fedora 12 with a 10GBps ethernet connection. We used the provided HTML crawler, which downloaded tweets at a rate of approximately 1 block of 10k tweets every 8 minutes.

Figure 1 shows the distribution of HTTP response codes returned during crawling. The response codes are summarized as follows:

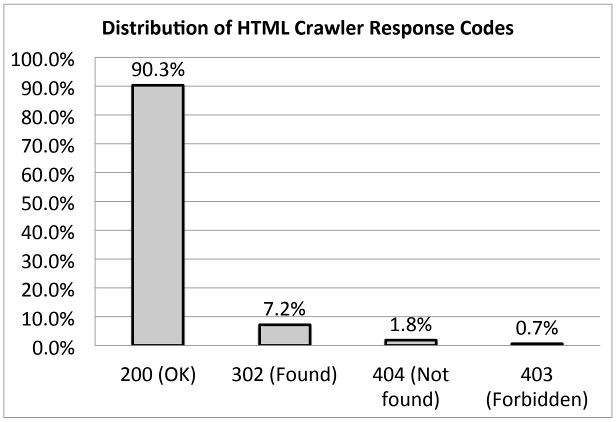

Figure 1: Distribution of HTML crawler response codes.

- 200 (OK) - a successfully downloaded tweet.
- 302 (Found) - a successfully downloaded re-tweet (via a redirect).
- 403 (Forbidden) - the user has disabled public sharing of their tweets.
- 404 (Not found) - the user account no longer exists.

Based on the download statistics, we see that a vast majority (over 97%) of the tweets were successfully downloaded, while only a small fraction were no longer accessible. Table 2 shows a detailed count of each type, as well as the total number of searchable tweets (15,736,358) in our version of the corpus.

A set of 50 test topics was released for evaluation purposes. Each topic consists of a keyword query and a temporal reference point, which acts as a query timestamp. NIST employed a pooling strategy to obtain ground truth for evaluation purposes. There were no relevant items for Topic 50 and it was therefore eliminated from the topic pool. Furthermore, only 33 topics had tweets that were judged to be highly relevant.

All of our submissions were run using the Indri search engine [12]. Indri was used because it was particularly suitable for the specifics of the task. In particular, we used Indri's numeric field support to annotate each tweet with its timestamp. At query time, we could then issue queries of the form:

```
#filreq(
  #less(time 1297191104)
  #weight(0.8 #combine(2022 fifa soccer)
          0.1 #combine(#1(fifa soccer)
                       #1(2022 fifa)
                       #1(2022 fifa soccer))
          0.1 #combine(#uw8(fifa soccer)
                       #uw8(2022 soccer)
                       #uw8(2022 fifa)
                       #uw12(2022 fifa soccer))))
```

We utilized the following procedure to rank tweets in response to a query according to the Microblog Track guidelines. First, we retrieve the 1000 most relevant tweets according to the text_score feature. Second, we filter out all re-tweets (i.e., tweets with HTML status code 302 or those that begin with the string "RT"). Then, if this is a learning-to-rank run, we re-order the non-filtered tweets based on the learned model (Table 1). Next, we truncate the (relevance-ordered) ranked and retain only the top 30 results. Finally, we return the top 30 results in reverse chronological order.

It is very important to note that our system only returns at most 30 results per query. This was part of our strategy to optimize every aspect of our system for the official evaluation metric (P@30). Hence, it makes little-to-no sense to evaluate our runs based on measures like MAP. We chose this particular strategy because it is optimal for precision at rank 30. Indeed, if our system had returned more than 30 results per query, it would face the risk of introducing non-relevant (or less relevant) documents into the top 30 during the process of sorting the tweets in reverse chronological order. If future evaluations use a different metric of interest, such as one that combines relevance and recency, then this particular strategy would unlikely yield satisfactory results.

Table 3 summarizes our official runs and the approaches used by each. If a run ID contains "FD" then it makes use of the MRF full dependence model, if the ID contains "RM" then it makes use of Latent Concept Expansion, and if it contains "L" then it uses learning-to-rank.

Finally, we note that none of our runs made use of external or future data. The only data that may be construed as "external", depending on your point of view, are the

Run ID	Approaches Used
isiFD	MRF
isiFDL	MRF + learning-to-rank
isiFDRM	MRF + LCE
isiFDRML	MRF + LCE + learning-to-rank

Table 3: Summary of the approaches used by our four official runs.

Criteria	isiFD	isiFDL	isiFDRM	isiFDRML
AllRel	.4361	**.4551**	.4476	.4442
HighRel	.1384	.1434	**.1566**	.1556

Table 4: Precision at rank 30 for each official run under the two relevance criteria.

relevance judgments obtained for training our learning-to-rank models and the Aspell dictionary for detecting out of vocabulary terms. However, we view these more as "basic" resources than "external" ones.

3.2 Results

We now describe the results from our four official runs. The results are summarized in Table 4. The evaluation is broken down into two separate sets of metrics. The first considers all queries that had at least one tweet judged relevant (denoted "AllRel."). The second only considers highly relevant tweets as relevant, and is averaged over the 33 topics that had such judgments (denoted "HighRel"). The metric reported is precision at 30.

If we consider all queries, then we see that the isiFDL run, which makes use of the MRF retrieval model and learning-to-rank performs better than all of the other runs. Interestingly, while pseudo-relevance feedback showed some improvements, it did not perform as well as the isiFDL run. The difference between isiFDL and isiFD represents the only statistically significant difference amongst all pairs of runs.

When we only consider highly relevant judgments, our findings are substantially different. Under this scenario, the isiFDRM run, which combines the MRF and LCE outperforms all other methods, including those that make use of learning-to-rank. Here, only the difference between isiFDRM and isiFD is statistically significant amongst all pairs of runs.

Therefore, the results are mixed and not altogether conclusive. We had originally hypothesized that both learning-to-rank and pseudo-relevance feedback would be effective on their own, which does indeed turn out to be true. However, we had also hypothesized that the gains would be additive and that isiFDRML would consistently outperform all of the other runs. However, this was simply not the case.

There are a number of possible explanations for these findings. First, the relevance judgments were obtained using isiFD as a base ranking function. Therefore, the judgments may not be suitable for training pseudo-relevance feedback-based models. Second, we may not have had enough judgments to adequately train a learning-to-rank model in the first place. Third, it may be that pseudo-relevance feedback is not useful for finding "somewhat relevant" tweets, but is actually quite useful for finding "highly relevant" tweets. It would be valuable to understand this better from a risk/reward tradeoff point of view [1]. Finally, the "Highly Relevant" results are averaged over 33 queries, which is a very small sample size, and hence it is inappropriate to draw strong conclusions from the results. We intend to dig deeper into these issues in the future to develop a better understanding the pros and cons of the various approaches employed.

4 Conclusions

Our experiments at the inaugural TREC Microblog Track focused on developing a strong baseline system, based on best-practice approaches, upon which we can build novel, effective approaches in the future. In particular, we made use of the Markov random field model (full dependence variant) for text scoring, Latent Concept Expansion for pseudo-relevance feedback, and learning-to-rank to combine evidence a variety of features (text, content quality, language identification, etc.).

Our experimental results showed that both learning-to-rank and pseudo-relevance feedback approaches were effective. However, we did not observe additive gains across the two approaches. As future work, we plan to understand the relationship between pseudo-relevance feedback and learning-to-rank, to develop novel microblog-specific features, and to investigate unsupervised methods for automatically training microblog ranking functions.

Acknowledgments

We would like to thank Dirk Hovy for providing access to his Twitter English language classifier. We would also like to thank the volunteers that helped us construct a small set of training data.

References

[1] K. Collins-thompson. Accounting for stability of retrieval algorithms using risk-reward curves. In *In Proceedings of the SIGIR 2009 Workshop on the Future of IR Evaluation*, pages 27–28, 2009.

[2] Y. Duan, L. Jiang, T. Qin, M. Zhou, and H.-Y. Shum. An empirical study on learning to rank of tweets. In *Proc. 23rd Intl. Conf. on Computational Linguistics*, pages 295–303, 2010.

[3] E. A. Fox and J. A. Shaw. Combination of multiple searches. In *Proc. 2nd Text REtrieval Conference*, pages 243–252, 1993.

[4] S. Gouws, D. Metzler, C. Cai, and E. Hovy. Contextual bearing on linguistic variation in social media. In *Proceedings of the Workshop on Languages in Social Media*, LSM '11, pages 20–29, Stroudsburg, PA, USA, 2011. Association for Computational Linguistics.

[5] V. Lavrenko and W. B. Croft. Relevance-based language models. In *Proc. 24th Ann. Intl. ACM SIGIR Conf. on Research and Development in Information Retrieval*, pages 120–127, 2001.

[6] T.-Y. Liu. Learning to rank for information retrieval. *Foundations and Trends in Information Retrieval*, 3(3), 2009.

[7] K. Massoudi, M. Tsagkias, M. de Rijke, and W. Weerkamp. Incorporating query expansion and quality indicators in searching microblog posts. In *Proc. 33rd European Conf. on Information Retrieval*, ECIR'11, pages 362–367, Berlin, Heidelberg, 2011. Springer-Verlag.

[8] D. Metzler. *A Feature-Centric View of Information Retrieval*. Information Retrieval Series. Springer, 2011.

[9] D. Metzler and W. B. Croft. A Markov random field model for term dependencies. In *Proc. 28th Ann. Intl. ACM SIGIR Conf. on Research and Development in Information Retrieval*, pages 472–479, 2005.

[10] D. Metzler and W. B. Croft. Latent concept expansion using Markov random fields. In *Proc. 30th Ann. Intl. ACM SIGIR Conf. on Research and Development in Information Retrieval*, 2007.

[11] D. Metzler and W. B. Croft. Linear feature-based models for information retrieval. *Information Retrieval*, 10(3):257–274, 2007.

[12] T. Strohman, D. Metzler, H. Turtle, and W. B. Croft. Indri: A language model-based serach engine for complex queries. In *Proceedings of the International Conference on Intelligence Analysis*, 2004.

[13] H. Turtle and W. B. Croft. Evaluation of an inference network-based retrieval model. *ACM Transactions on Information Systems*, 9(3):187–222, 1991.