An Empirical Study on Learning to Rank of Tweets

[1]Yajuan Duan* **[2]Long Jiang** **[2]Tao Qin** **[2]Ming Zhou** **[2]Heung-Yeung Shum**
[1]Department of Computer Science and Technology
University of Science and Technology of China
[2]Microsoft Research Asia
{v-yaduan,longj,taoqin,mingzhou,hshum}@microsoft.com

Abstract

Twitter, as one of the most popular micro-blogging services, provides large quantities of fresh information including real-time news, comments, conversation, pointless babble and advertisements. Twitter presents tweets in chronological order. Recently, Twitter introduced a new ranking strategy that considers popularity of tweets in terms of number of retweets. This ranking method, however, has not taken into account content relevance or the twitter account. Therefore a large amount of pointless tweets inevitably flood the relevant tweets. This paper proposes a new ranking strategy which uses not only the content relevance of a tweet, but also the account authority and tweet-specific features such as whether a URL link is included in the tweet. We employ *learning to rank* algorithms to determine the best set of features with a series of experiments. It is demonstrated that whether a tweet contains URL or not, length of tweet and account authority are the best conjunction.

1 Introduction

Twitter provides a platform to allow users to post text messages known as tweets to update their followers with their findings, thinking and comments on some topics (Java et al., 2007). The searched tweets are presented by Twitter in chronological order except the first three, which are ranked by considering popularity of tweets in terms of the number of retweets.

This ranking method, however, has not taken into account the content relevance and twitter account; inevitably, a large amount of pointless tweets (Pear Analytics, 2009) may flood the relevant tweets. Although this ranking method can provide fresh information to tweet users, users frequently expect to search relevant tweets to the search queries. For example, consider someone researching consumer responses toward the iPad. He or she would like to find tweets with appropriate comments such as *iPad is great* or *you can find many useful features of iPad,* rather than tweets with irrelevant comment, even if they are most recent or popular.

Moreover, neither Twitter's current chronological order based ranking nor the recently introduced popularity based ranking can avoid spam. A developer can accumulate hundreds of thousands of followers in a day or so. At the same time, it is not difficult for spammers to create large quantities of retweets. By contrast, content relevance ranking can effectively prevent spammers from cheating. Different from ranking tweets through chronological order and popularity, a content relevance strategy considers many characteristics of a tweet to determine its ranking level. Thus it is difficult for spammers to break the ranking system by simple methods such as increasing retweet count or number of followers.

In this paper, we propose a method to rank the tweets which outputs the matched tweets based on their content relevance to the query. We

* The work was done when the first author was intern at Microsoft Research Asia

Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010), pages 295–303,
Beijing, August 2010

investigate the effects of content features and non-content features and produce a ranking system by a *learning to rank* approach.

With a series of experiments, we determined the best set of features and analyzed the effects of each of individual feature. We provide empirical evidence supporting the following claims,

- Account authority, length of tweet and whether a tweet contains a URL are the top three effective features for tweet ranking, where containing a URL is the most effective feature.
- We find an effective representation of account authority: the number of times the author was listed by other users. We find through experiments that this representation is better than the widely adopted number of followers.

2 Related Work

2.1 Real-time Search

At present, a number of web sites offer the so-called real-time search service which mainly returns real-time posts or shared links, videos and images obtained from micro-blogging systems or other medium according to the user's query. We investigate the ranking method used by these web sites. From their self-introduction page, we find four main criteria for ranking real-time posts. They are posting time, account authority, topic popularity and content relevance.

Specifically, Twitter maintains a specialized search engine which ranks tweets according to posting time and topic popularity. In addition, Google, Twazzup[2] and Chirrps[3] rank real-time tweets by posting time. While the last one also ranks tweets by popularity, which is measured by retweet count.

Tweefind[4] ranks search result according to authority of authors which depends on how popular, relevant, and active the author is. Additionally, Twitority[5] rank tweets by author authority as well.

Bing and CrowdEye[6] rank tweets by posting time or content relevance. Bing takes authors authority, retweet count and freshness into consideration while measuring the relevance. To determine the relevance of a tweet, CrowdEye considers a number of factors including content relevance and author influence which appears to rely heavily on the number of followers an author has. It turns out that the number of followers is not a very reasonable measure of the influence of an account according to our experimental results.

2.2 Twitter Recommendation

Besides tweet search, recently some researchers have focused on twitter recommendation system.

Chen et al. (2010) presented an approach to recommend URLs on Twitter as a means to better direct user attention in information streams. They designed the recommender taking three separate dimensions into consideration: content source, topic interest and social voting.

Sun et al. (2009) proposed a diffusion-based micro-blogging recommendation framework aiming to recommend micro-blogs during critical events via optimizing story coverage, reading effort and delay time of a story. The key point of this method is to construct an exact diffusion graph for micro-blogging, which is difficult due to the presence of extensive irrelevant personal messages and spam.

2.3 Blog Search and Forum Search

Another related topic is blog search and forum search. Recently, many approaches for blog search and forum search have been developed, which include *learning to rank* methods and link-based method.

Learning to rank approach

Xi et al. (2004) used features from the thread trees of forums, authors, and lexical distribution within a message thread and then applied Linear Regression and Support Vector Machine (SVM) to train the ranking function. Fujimura et al. (2005) exploited provisioning link and evaluation link between bloggers and blog entries, and scored each blog entry by weighting the hub and authority scores of the bloggers.

Link-Based approach

[2] Twazzup: http://www.twazzup.com/
[3] Chirrps: http://chirrps.com/
[4] Tweefind: http://www.tweefind.com/
[5] Twitority: http://www.twitority.com/
[6] CrowdEye: http://www.crowdeye.com/

Kritikopoulos et al. (2006) introduced similarities among bloggers and blogs into blog ranking. This method enabled the assignment of a higher score to the blog entry published by a blogger who has already accepted a lot of attention. Xu and Ma (2006) built a topic hierarchy structure through content similarity. Liu et al. (2007) presented a newsgroup structure-based approach PostRank which built posting trees according to response relationship between postings.

Chen et al. (2008) proposed a posting rank algorithm which built link graphs according to co-replier relationships. This kind of method exploits different types of structures among postings and improved the performance of traditional link-based ranking algorithm for forum search. However, it is difficult to rank postings which only have a few words simply based on content by using FGRank algorithm. And PostingRank approach relies too much on reply relations which are more likely to suffer from topic excursion.

Although approaches proposed above perform effectively in forum search and blog search, they are not appropriate for twitter search because tweets are usually shorter and more informal than blogs. Furthermore, it does not have the explicit hierarchy structure of newsgroup messages on forums. In addition, tweets possess many particular characteristics that blog and forum do not have.

3 Overview of Our Approach

To generate a good ranking function which provides relevant search results and prevents spammers' cheating activities, we analyze both content features and authority features of tweets and determine effective features. We adopt *learning to rank* algorithms which have demonstrated excellent power in addressing various ranking problems of search engines.

3.1 Learning to Rank Framework

Learning to Rank is a data-driven approach which integrates a bag of features in the model effectively. Figure 1 shows the paradigm of learning for tweet ranking.

At the first step, we prepare the training and test corpus as described in Section 5. Then we extract features from the training corpus. RankSVM algorithm (Joachims Thorsten, 1999) is used to train a ranking model from the training corpus. Finally, the model is evaluated by the test corpus.

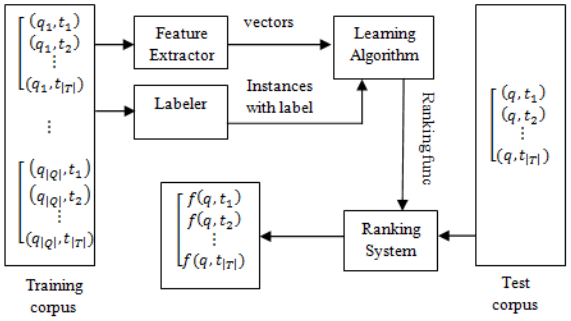

Figure 1. General Paradigm of Learning for Tweets Ranking

3.2 Features for Tweets Ranking

One of the most important tasks of a *learning to rank* system is the selection of a feature set. We exploit three types of features for tweet ranking.

1) *Content relevance features* refer to those features which describe the content relevance between queries and tweets.
2) *Twitter specific features* refer to those features which represent the particular characteristics of tweets, such as retweet count and URLs shared in tweet.
3) *Account authority features* refer to those features which represent the influence of authors of the tweets in Twitter (Leavitt et al., 2009).

In the next section, we will describe these three types of features in detail.

4 Feature Description

4.1 Content Relevance Features

We used three content relevance features, Okapi BM25 (Robertson et al., 1998), similarity of contents and length of tweet.

Okapi BM25 score measures the content relevance between query Q and tweet T. The standard BM25 weighting function is:

$$\mathrm{BM25}(T,Q)=\sum_{q_i\in Q}\frac{\mathrm{IDF}(q_i)\cdot \mathrm{tf}(q_i,T)\cdot(k_1+1)}{\mathrm{tf}(q_i,T)+k_1\left(1-b+b\frac{\mathrm{Length}(T)}{\mathrm{avg}_{\mathrm{length}}}\right)} \quad (1)$$

where Length(T) denotes the length of T and avg_{length} represents average length of tweet in corpus. IDF(q_i) is Inverse Document Frequency.

Similarity of contents estimates the popularity of documents in the corpus (Song et al., 2008). In our case, it measures how many tweets of the query are similar in content with the current tweet. We calculate a cosine similarity score for every pair of tweets, and the final similarity score for tweet T_i in T_{Q_k} is computed by the following formula:

$$\text{Similarity}(T_i) = \frac{1}{|T_{Q_k}|-1}\sum_{T_j \in T_{Q_k}, j \neq i} \frac{TV_i \cdot TV_j}{|TV_i| \cdot |TV_j|} \quad (2)$$

Where TV_i represents the TFIDF vector of T_i and T_{Q_k} refers to tweets collection of query Q_k.

Length is measured by the number of words that a tweet contains. Intuitively, a long sentence is apt to contain more information than a short one. We use length of tweet as a measure of the information richness of a tweet.

4.2 Twitter's Specific Features

Tweets have many special characteristics. We exploit these characteristics and extract six twitter specific features as listed in Table 1.

Feature	Description
URL	Whether the tweet contains a URL
URL Count	Frequency of URLs in corpus
Retweet Count	How many times has this tweet been retweeted
Hash tag Score	Sum of frequencies of the top-n hash tags appeared in the tweet
Reply	Is the current tweet a reply tweet
OOV	Ratio of words out of vocabulary

Table 1. Twitter Specific Features

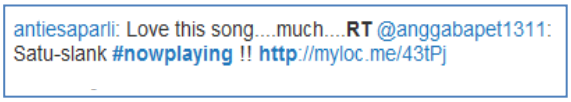

Figure 2. A Tweet Example

URL & URL Count: Twitter allows users to include URL as a supplement in their tweets. The tweet in Figure 2 contains URL http://myloc.me/43tPj which leads to a map indicating where the publisher located.

URL is a binary feature. It is assigned 1 when a tweet contains at least one URL, otherwise 0. URL Count estimates the number of times that the URL appears in the tweet corpus.

Retweet Count: Twitter users can forward a tweet to his or her followers with or without modification on the forwarded tweets, which is called retweet on Twitter. A retweeted tweet usually includes an *RT* tag. Generally, sentences before *RT* are comments of the retweeter and sentences after *RT* are the original content, perhaps with some modifications. Here we only consider tweets including *RT* with the original content unmodified. Retweet count is defined as the number of times a tweet is retweeted. In Figure 2, original tweet *Satu-slank #nowplaying !! http://myloc.me/43tPj* is retweeted once.

Hash Tag Score: Publishers are allowed to insert hash tags into their tweets to indicate the topic. In Figure 2, *#nowplaying* is a hash tag. We collect hash tags appearing in the tweets of every query and sort them in descending order according to frequency. Tag frequency for tweet T_i of query Q_k is computed from normalized frequency of top-n tags.

$$\text{TagScore}(T_i) = \frac{1}{z_k}\sum_{\substack{j=1, tag_j \in Tag_{Q_k} \\ tag_j \in T_i}}^{n} \text{freq}(tag_j) \quad (3)$$

Where z_k is the normalization factor. $\text{freq}(tag_j)$ represents the frequent of tag_j in corpus. And Tag_{Q_k} denotes the tag collection extracted from T_{Q_k}.

Reply: This is a binary feature. It is 1 when the tweet is a reply and 0 otherwise. A tweet starting with a twitter account is regarded as a reply tweet in our experiment. Figure 3 shows an example.

Figure 3. Reply Tweet

OOV: This feature is used to roughly approximate the language quality of tweets. Words out of vocabulary in Twitter include spelling errors and named entities. According to a small-scale investigation, spelling errors account for more than 90% of OOVs excluding capitalized words, tags, mentions of users and

URLs. We use a dictionary with 0.5 million entries to compute the ratio of OOVs in a tweet.

$$\text{Quality(T)} = \frac{\#\text{ of OOVs in T}}{\text{Length(T)}} \qquad (4)$$

4.3 Account Authority Features

There are three important relations between users in Twitter: follow, retweet, and mention. Additionally, users are allowed to classify their followings into several lists based on topics. We measured the influence of users' authorities on tweets based on the following assumptions:

- Users who have more followers and have been mentioned in more tweets, listed in more lists and retweeted by more important users are thought to be more authoritative.
- A tweet is more likely to be an informative tweet rather than pointless babble if it is posted or retweeted by authoritative users.

PageRank algorithm for calculating popularity score for users.
Input: Directed Graph G of retweet relationship
Damping factor e.
Output: popularity score for each user
Procedure:
Step 1: popularity score of all users are initialized as $1-e$.
Step 2: update the popularity score for users.

$$PScore_{t+1}(v_i) = 1 - e + e \cdot \sum_{v_j \in R_{v_i}} \frac{PScore_t(v_j) RN_{ij}}{N_j}$$

R_{v_i} denotes the collection of users who retweeted v_i's tweet.
RN_{ij} is the number of times v_i has been retweeted by v_j.
N_j is the number of users whose tweets v_j has retweeted.
Step 3: Repeat the second step until all popularity scores will never change.

Figure 4. PageRank Algorithm for Calculating Popularity Score for Users

In order to distinguish the effect of the three relations, we computed four scores for each user representing the authority independently.

- Follower Score: number of followers a user has.
- Mention Score: number of times a user is referred to in tweets.
- List Score: number of lists a user appears in.
- Popularity Score: computed by PageRank algorithm (Page et al., 1999) based on retweet relations.

Following the retweet relationship among users, we construct a directed graph G (V, E). In our experiments, G is built from a tweet collection including about 1.1 million tweets. V denotes twitter users that appear in training examples. E is a set of directed edges. If author v_i published the tweet t_k, and author v_j retweeted t_k after v_i, there exists an edge from v_j to v_i. We call v_i original author and v_j retweeter. Figure 4 shows the PageRank algorithm for calculating popularity scores for twitter users. In our experiment, damping factor e was set to 0.8. Like Dong et al. (2010) did, we define three subtypes for each account authority score. Table 2 presents features of account authority we use.

Feature	Description
Sum_follower	Sum of follower scores of users who published or retweeted the tweet
Sum_popularity	Sum of popularity scores of users who published or retweeted the tweet
Sum_mention	Sum of mention scores of users who published or retweeted the tweet
Sum_list	Sum of list scores of users who published or retweeted the tweet
First_follower	Follower score of the user who published the tweet
First_popularity	Popularity score of the user who published the tweet
First_mention	Mention score of the user who published the tweet
First_list	List score of the user who published the tweet
Important_follower	The highest follower score of the user who published or retweeted the tweet
Important_popularity	The highest popularity score of the user who published or retweeted the tweet
Important_mention	The highest mention score of the user who published or retweeted the tweet
Important_list	The highest list score of the user who published or retweeted the tweet

Table 2. Account Authority Features for tweet

5 Experiment Data and Evaluation

We introduce the data we used in experiment and the evaluation metrics in this section.

5.1 Data

We analyze 140 hot searches on CrowdEye within a week. They consist of big events,

famous persons, new products, festivals, movies and so on. The most frequent types of hot searches, which account for more than 81% of all hot searches, are as follows:

- News: news about public figures and news related to some places.
- Products: character description, promotion information and comments about products.
- Entertainment: mainly about movies, including film reviews and introductions about plots.

We select 20 query terms as shown in Table 3, including 5 persons, 5 locations, 5 products and 5 movie names. Specifically, Locations are sampled from a list of American cities. Person names come from the hot search and hot trends provided by Twitter and CrowdEye. Products are sampled from the popular searches of 35 product categories on eBay. And movies are selected from a collection of recommended movies from 2005 to 2010. We crawl 162,626 English tweets for the selected queries between March 25, 2010 and April 2, 2010 from Twitter Search. After removing the repeated ones, 159,298 tweets remained.

Query type	Query terms
Locations	New York, Nashville, Denver, Raleigh, Lufkin
Person Names	Obama, Bill Clinton, James Cameron, Sandra Bullock, LeBron James
products	Corvette, iPad, Barbie, Harry Potter, Windows 7
Movies	The Dark Knight, up in the air, the hurt locker, Batman Begins, Wall E

Table 3. 20 Query Terms

Retweets are forwardings of corresponding original tweets, sometimes with comments of retweeters. They are supposed to contain no more information than the original tweets, therefore they drops out of ranking in this paper.

We sample 500 tweets for each query from its original tweets collection and ask a human editor to label them with a relevance grade. In order to ensure the annotation is reasonable, we set multiple search intentions for each query referring to the topics arising in the tweets about the query in the corpus. Specifically, for Locations, tweets describing news related to the location are relevant. For people, what they have done and the comments about them are regarded as relevant information. For products, tweets including feature description, promotion and comments are considered relevant. And for movies, tweets about comment on the movies, show time and tickets information are preferred. We apply four judgment grades on query-tweet pairs: excellent, good, fair and bad. According to the statistics, about half of the tweets in the experiment data are labeled as bad. Table 4 presents the distribution for all grades.

Grade	Excellent	Good	Fair	Bad
Percentage	20.9%	10.9%	16.9%	51.3%
Min	2.4%	1.8%	4.0%	8.0%
Max	69.8%	23.2%	54.4%	81.0%

Table 4. Tweet Distribution of Each Grade

5.2 Evaluation Metrics

There are several metrics that are often used to measure the quality of rankings. In this paper, we use Normalized Discount Cumulative Gain (NDCG) which can handle multiple levels of relevance as the evaluation metrics (Jarvelin and Kekalainen, 2002).

6 Results

Five-fold cross-validation was used in our experiments. We choose tweets of sixteen queries (four from each query type) as the training data. The remaining tweets are divided into evaluation data and validation data equally.

6.1 Learning to Rank for Tweet Ranking

We learn a ranking model by using a RankSVM algorithm based on all features we extracted, which is denoted as RankSVM_Full. In the experiment, a toolkit named svm^{struct} [7] implemented by Thorsten Joachims is used. Figure 5 shows the comparison between our method which integrates three types of features and ranking through chronological order, account authority, and content relevance individually.

In this experiment, Content Relevance is measured by BM25 score. And Account

[7] SVM^{struct}: http://svmlight.joachims.org/svm_struct.html

Authority is approximated by the number of followers of the user. Figure 5 illustrates that ranking through content relevance is not as effective as other methods. This is because our work is essentially re-ranking on the result of Twitter Search. Hence almost all tweets include the query term which makes it difficult to distinguish them by BM25 score. Figure 5 also reveals that account authority is useful for ranking tweet relevance; it outperforms ranking through chronological order and is competitive to our model trained from all features. This agrees with the assumption we made about the influence of user authorities on tweets.

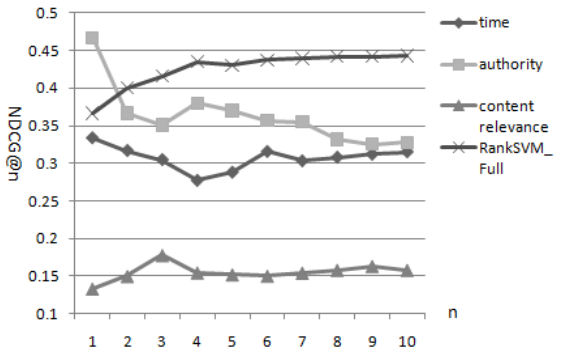

Figure 5. Performance of Four Ranking Methods

6.2 Feature Selection

As the RankSVM_Full underperforms against some models trained from subsets of features, we use an advanced greedy feature selection method and find the best feature conjunction to improve the performance of RankSVM_full. Figure 6 shows the feature selection approach.

Although greedy feature selection approach is commonly used in many problems, it does not work efficiently in addressing this problem partly for data sparseness. It is always blocked by a local optimum feature set. In order to resolve this problem, we first generate several feature sets randomly and run the greedy selection algorithm based the best one among them. Finally, we find the best feature conjunction composed by *URL, Sum_mention, First_List, Length,* and *Important_follower*, from which a model is learnt denoted as RankSVM_Best. Figure 7 illustrates that this model outperforms RankSVM_Full by about 15.3% on NDCG@10.

An advanced greedy feature selection algorithm.
Input: All features we extracted.
Output: the best feature conjunction *BFC*
Procedure:
Step1: Randomly generate 80 feature set *F*.
Step 2: Evaluate every feature set in *F* and select the best one denoted by *RBF*.
Features excluded those in *RBF* are denoted as *EX_RBF*
Step 3: t = 0,*BFC*(t)=*RBF*;
  Repeat
    Foreach feature in *EX_RBF*
      If Evaluation(*BFC*)
        < Evaluation(*BFC*, feature)
        BFC(t+1) = {*BFC*(t), feature}
        EX_RBF(t+1) = *EX_RBF*(t) – {feature}
  While *BFC*(t+1) ≠ *BFC*(t)
Note: Evaluation(*BFC*) refers to the performance of ranking function trained from features in *BFC* on validation data.

Figure 6. Advanced Greedy Feature Selection Algorithm

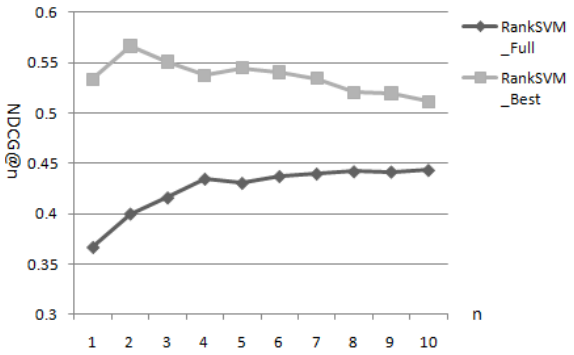

Figure 7. Comparison between RankSVM_Full and RankSVM_Best

We conduct a paired t-test between RankSVM_Best and each of other four ranking methods on NDCG@10 of ten test queries. The results demonstrate that RankSVM_Best outperforms ranking through time, account authority and content relevance respectively with a significance level of 0.01, and RankSVM_Full with a level of 0.05.

6.3 Feature Analysis

We are interested in which features in particular are highly valued by our model for tweet ranking. We evaluate the importance of each feature by the decrement of performance when removing the feature measured from RankSVM_Best. Figure 8 reveals the importance of each feature in our model.

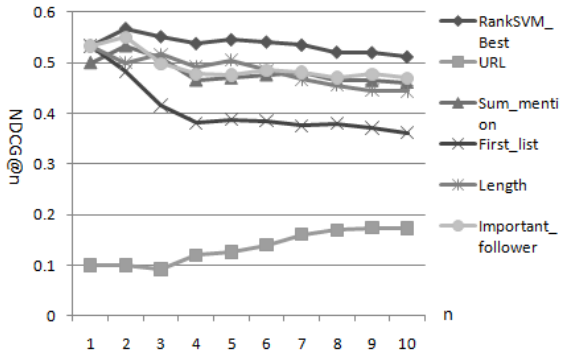

Figure 8. Importance of Each Feature

We observe from Figure 8 that URL is very important for our model; without it the performance declines seriously (with a significance level of 0.001). The reason may be that URLs shared in tweets, which provide more detailed information beyond the tweet's 140 characters, may be relevant to the query at a high probability.

Another useful feature is the number of lists that the author of the tweet has been listed in. The performance of ranking decreases with a significance level of 0.05 when removing it from the best feature combination. However, other features do not show significant contribution.

7 Discussion

Our experiment in section 6.2 demonstrates that features such as Hash tag Score and Retweet Count are not as effective as expected. This may be due to the small size of training data. We present an approach to learn an effective tweets ranker in a small dataset through feature selection. However, 20 queries are not sufficient to train a powerful ranker for Twitter.

In this study, to minimize the annotation effort, for each test query, we only annotate the tweets containing the query (returned by Twitter Search) and then used them for evaluation. With this kind of evaluation, it is hard to completely evaluate the significance of some features, such as content relevance features. In the future, we will select more queries including both hot searches and long tail searches, and select tweets for annotation directly from the twitter firehose.

There is also an opportunity for more accurate retweet relation detection in our work. At present, we just identify the retweet whose original tweet has not been modified, which leaves out a fair amount of retweet information. We would need to develop a more precise retweet relation detection method.

8 Conclusion

In this paper, we study three types of tweet features and propose a tweet ranking strategy by applying learning to rank algorithm. We find a set of most effective features for tweet ranking. The results of experiments demonstrate that the system using *Sum_mention*, *First_list*, *Important_follower*, *length* and *URL* performs best. In particular, whether a tweet contains a URL is the most effective feature. Additionally, we find in the experiments that the number of times the account is listed by other users is an effective representation of account authority and performs better than the number of followers that is widely used in previous work.

There are many aspects we would like to explore in the future. First, this research is based on the search results returned from Twitter which contains the input query. The tweets not containing the queries are not returned. We will explore query expansion approaches to improve the recall of the search results. We did not consider spam issues in the ranking process. However, spam filtering is important to all types of search engines. We will explore the impacts of spam and work out a spam filtering approach.

References

Chen Jilin, Rowan Nairn, Les Nelson, Michael Bernstein, and Ed H. Chi. 2010. Short and Tweet: Experiments on Recommending Content from Information Streams. *In the Proceedings of the 28th International conference on Human Factors in Computing Systems*, Pages: 1185-1194.

Chen Zhi, Li Zhang, Weihua Wang. 2008. PostingRank: Bringing Order to Web Forum Postings. *In the proceedings of the 4th Asia Information Retrieval Symposium*, Pages: 377-384.

Dong Anlei, Ruiqiang Zhang, Pranam Kolari, Jing Bai, Fernando Diaz, Yi Chang, Zhaohui Zheng, and Hongyuan Zha. 2010. Time of the essence: improving recency ranking using Twitter data. *In the proceedings of the 19th International Conference on World Wide Web*, Pages: 331-340.

Fujimura Ko, Takafumi Inoue, and Masayuki Sugisaki. 2005. The EigenRumor Algorithm for Ranking Blogs. *In the proceedings of the 2nd Annual Workshop on the Weblogging Ecosystem: Aggregation, Analysis and Dynamics, World Wide Web*.

Jarvelin Kalervo, and Jaana Kekalainen. 2002. Cumulated gain-based evaluation of IR techniques. *ACM Transactions on Information Systems*, Volume 20, Pages: 422-446.

Java Akshay, Xiaodan Song, Tim Finin, and Belle Tseng. 2007. Why we twitter: Understanding Microblogging Usage and Communities. *In the proceedings of the 9th International Workshop on Knowledge Discovery on the Web and the 1st International Workshop on Social Networks Analysis*. Pages: 118-138.

Joachims Thorsten. 1999. Making Large-Scale SVM Learning Practical. *Advances in Kernel Methods: Support Vector Learning*, Pages: 169-184.

Pear Analytics. 2009. Twitter Study-August 2009.

Kritikopoulos Apostolos, Martha Sideri, and Iraklis Varlamis. 2006. BlogRank: Ranking Weblogs Based on Connectivity and Similarity Features. *In the proceedings of the 2nd International Workshop on Advanced Architectures and Algorithms for Internet Delivery and Applications*.

Leavitt Alex, Evan Burchard, David Fisher, and Sam Gilbert. 2009. The Influentials: New Approaches for Analyzing Influence on Twitter. *A publication of the Web Ecology Project*.

Liu Hongbo, Jiahai Yang, Jiaxin Wang, Yu Zhang. 2007. A Link-Based Rank of Postings in Newsgroup. *In the proceedings of the 5th International Conference on Machine Learning and Data Mining in Pattern Recognition*, Pages: 392-403.

Page Lawrence, Sergey Brin, Rajeev Motwani, and Terry Winograd. 1999. The PageRank Citation Ranking: Bring Order to the Web. *Technical report*, Stanford University.

Robertson Stephen E., Steve Walker, and Micheline Hancock-Beaulieu. 1998. Okapi at TREC-7: Automatic Ad Hoc, Filtering, VLC and Interactive. *In the Proceedings of the 7th Text Retrieval Conference*. Pages: 199-210

Song Young-In, Chin-Yew Lin, Yunbo Cao, and Hae-Chang Rim. 2008. Question Utility: A Novel Static Ranking of Question Search. *In the Proceedings of the 23rd AAAI Conference on Artificial Intelligence*. Pages: 1231-1236

Sun Aaron R., Jiesi Cheng, and Daniel D. Zeng. 2009. A Novel Recommendation Framework for Micro-blogging based on Information Diffusion. *In the proceedings of the 19th Workshop on Information Technologies and Systems*.

Xi Wensi, Jesper Lind, and Eric Brill. 2004. Learning effective ranking functions for newsgroup search. *In the proceedings of the 27th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, Pages: 394-401

Xu Gu, and Ma Wei-Ying. 2006. Building Implicit Links from Content for Forum Search. *In the proceedings of the 29th International ACM SIGIR Conference on Research and Development in Information Retrieval*. Pages: 300-307.