Microblog TRACK 2011 of FDU

Bingqing Wang
School of Computer Science
Fudan University
Shanghai, China, 200433
wbq@fudan.edu.cn

Xuanjing Huang
School of Computer Science
Fudan University
Shanghai, China, 200433
xjhuang@fudan.edu.cn

Abstract

Twitter provides huge amount of short messages, raises challenge problems to the research community. The Microblog Track of TREC detects the special behavior of the twitter dataset in the "real-time" retrieval task. This paper reports our participation in the Microblog Track task. Given the query topics, each participants are required to conduct a "real-time" retrieval task, which seeks for the most recent and interesting tweets for each query topic. Our focus in this task includes two aspects: (1)data preprocessing to remove non-English tweets, and (2)feature extraction for clustering the tweets into two categories. Given the huge interest in the microblog, there is lot of work to apply different linguist analysis techniques and data analysis methods to explore the behavior and special features in the Microblog sphere.

1 Introduction

As a social network basically relied on text messages, Twitter, the microblog platform, produces millions of short messages and delivers those messages among the network interweaved by friends, colleagues, family members and many others. These short messages (or sometimes referred as status) provide updated news, offer useful information from coupon to job opportunities, express the blog holders' opinion and etc. Some special features such as "hash tag", "retweet" has attracted lots of attention from the research community. Research work are focused on different aspects such as how to normalize the tweet-language, label POS-tags for each word, identify Named Entities, analyze the opinions, explore the user graph, use the virtual social network data to predicate real-world events, etc.

With a strong interest in the microblog and social network, we took part in the TREC 2011 Microblog TRACK, so that we can explore and exploit some special features of the twitter.

This year, the Microblog Track proposed a "real-time" retrieval task, which requires the participants to seek for the most recent and interesting tweets for a query. The dataset Twitter2011 covers two-weeks tweets from Jan 23rd to Feb. 8th 2011. This task is aimed to stimulate the scenario when the users look for some tweets for the given query within that two-week. Then, each test query topic is given a timestamp indicating what exact time that query is supposed to be submitted in that two weeks. The participants should only return the tweets posted before the query timestamp. The participants can use the external resources such as Wikipedia, the webpages, and the url in the tweets, but future information is not allowed in the "real-time" task, which including the whole dataset information such as IDF and other future evidence such as the Wikipedia explaining what the query is, but posted after the query timestamp.

This paper reports our participation into the Microblog track. To accomplish this task, we should face some problems such as identifying the language of the tweets, normalizing the tweets-language, and filter out irrelevant tweets and finding the "interesting" tweets.

We used multiple virtual machine instance to crawl all the dataset with the HTML crawler provided by TREC. The job was accomplished within one day, which brings us a lot of benefit. Since the Twitter website would change their links to the tweets, it is better to collect the dataset in an early age. To identify the language of the tweets, we extracted the language tag from the HTML web pages, and used opensource language identification tool "TextCat" to identify the language. Normalizing the tweet-language is a lot of work. We tried some opensource dictionary-based tool such as "Jazzy"[1], but got a limited normalization effect, which droves us to empirically normalize the tweets by rules. In the "real-time" scenario, we first find out all the tweets before the query timestamp, then generate two features from each tweets. One feature is the BM25(Robertson, et al., 1999) weight formula, but in the "real-time" scenario, we calculate the "real-time" IDF value for each query word. The other feature is calculated by the SVD decomposition as the conventional collaborative filtering. Finally use kmeans++ to cluster these tweets, and retain those tweets that have higher BM25 score. In this way, we filter out some more tweets.

This paper is structured as follows. Section 2 gives an overview of the twitter dataset and data download process . Section 3 introduces the language identification process and the preprocessing steps. Section 4 introduces the method to find the relevant tweets. Section 5 reports the experiment results and give some discussion. And the conclusion is made in Section 6.

2 Related Work

Research work on the Microblog dataset are versatile, which we would like to classify them into the following categories. (1)Data Collection, which collect the data for a long period and gives an overview on the dataset, such as (Perovic,et al., 2010)(Yang, et al., 2011) (2) Elemetary linguistic analysis, such as lexical form normalization(Han and Baldwin, 2011)(Brody and Diakopoulos, 2011), part-of-speech tagging(Gimpel, et al., 2011), keyphrase extraction(Zhao, et al., 2011). One obvious feature of Twitter is all kinds of out-of-vocabulary words such as initials, abbreviation, and other irregular form, to help the user express their meaning within the 140 character limit. That is why the current research work focus on these elementary linguistic analysis work. (3) Linguistic application: such as sentiment analysis(Jiang, et al., 2011)(Gonzalez-lbanez, et al., 2011), classification(Qazvinian, et al., 2011)(Zanzotto, et al., 2011). Twitter contains lots of comments on the news, products, tv shows, movies, which provides a rich source for the researcher to analyze the users' opinions. (4) Social Graph(Meeder, et al., 2011)(Wu, et al., 2011): researchers focus on the following and followed relation among the users. (5) Prediction: people use the data in the virtual microblog sphere to predict what will happen in the real world, for example, predicting the epidemics(Aramaki, et al., 2011), predicting the gender of the user(Burger, et al., 2011), predicting the investors(Bar-Haim, et al., 2011), predicting the information credibility(Catillo, et al., 2011).

[1]http://jazzy.sourceforge.net/

3 Twitter2011 Dataset

The dataset Twitter2011 is collected by each participant. The organizer provides the links in "Seed File", the participants restore these dataset from these seed files. One thing important for the participant is to download the dataset as soon as possible since the links in the website will not exist permanently. We collected the dataset using parallel virtual machine instances.

Seed File: each seed file contains about 10 thousand lines, each line contains the twitter users' name, **the encoded tweet id**, and the hashcode to verify that tweet. A url pointing to a real tweet can be recovered from these information, which can be used to download the tweet. There are 1670 seed files scattering in 17 days from Jan.23rd 2011 to Feb. 8th 2011.

Download Tool: The download tools provided by TREC have two versions, the JSCON version and the HTML version. The JSCON version require the user to have a registered twitter API on Twitter's whitepaper list, which allow the registered user to crawl 20000 tweets each hour. The JSCON version can provide complete information for each tweet, including the tweet content, the language, the following and followed information for that tweet, which is surely a good benefit for the researchers.

The HTML version could only extract the content and the posted timestamp for each tweet. However, the HTML version did not have a limit on the crawling speed. And the crawled html version tweets have the same html structure which makes it quite easy to clean the html tags and extract the meta data according to your needs. Another point should be noted is that the TREC download tool has a function to check the downloaded tweets and re-download those missing ones, which is quite helpful for us. We chose the HTML version of the download tool.

Crawler Setup: We setup 9 virtual instances to download the whole Twitter dataset, and let them work parallelly. Totally, for the 17 day's tweets and 9 virtual instances, we deployed 2 days tweets download work for 1 virtual instance, which would cost about 20 minutes to download the tweets for one seed file. Each instance would work about 24 hours. The original html downloaded data is kept in hadoop's sequence file format, which can be extracted easily with the TREC's download tools.

Data Statistics: After downloading and amending the dataset, we got 16,141,812 Tweets, with 14081863 tweets for html response code 200, 1123076 tweets for response code 302, 243979 for response code 403, and 692894 tweets for response code 404. Finally, 15204939 tweets contain the real content, 936873 tweets are missing. The missing rate is 5.8%. This crawling job was done on **June. 1st.**

4 Language identification and Preprocessing

4.1 Language Identification

Since we used the HTML crawling tool, we can not get the language information of each tweet. However, in the HTML web page, there is an html element which contains a feature indicating the language of this tweet. The element has the pattern "¡class=ˋtweet-url screen-name" hreflang=", where the hreflang feature will show the language of the tweet user. This feature is not a precise one, but it can filter out some obvious non-English tweets, especially for Asia language such as Japanese, Korean. The statistics of the language tag is shown in Tab 1. English tweets is the majority of the Twitter2011 dataset. The second place goes to the Japanese language, followed by Spanish, whose amount seems to be less than half of the Japanese. But we doubt that Spanish's ratio should be higher, since many Spanish users would use English as the default language in the Twitter platform.

Table 1: Statistics of the Language Tag in HTML

hreflang	language	amount	ratio
en	English	10894469	67.492%
ja	Japanese	2753207	17.056%
es	Spanish	1205837	7.470%
ko	Korean	84273	0.522%
fr	French	83432	0.517%
de	German	80551	0.499%
it	Italian	37670	0.233%
ru	Russia	19087	0.118%
tr	Turkish	10045	0.062%
NA	NA	973240	6.029%
total		16141812	100%

About 67% twitter user's screen-name language is English, which we noted as **Twitter2011Eng**. In this filtered dataset, there are still a lot of non-English tweets. We used TextCat[2] to identify the language of each tweet. TextCat has been reported showing good performance on the conventional news dataset(Cavnar and Trenkle, 1994). It builds the character-level language model for each word, and calculate the score of each sentence by heauristic method. The user can control how many languages would this tool guess for one sentence. Since it is based on the language model, the user can select the models empirically or even train his own model. We used the default model attached in the toolkit and tuned the language the tool can output.

In order to evaluate the performance of this tool, we sampled some sentences from the filtered Twitter2011Eng dataset. 5000 tweets were sampled, and 250 tweets were annotated as English and non-English. The result is shown in Table. 2, which shows the balancing of the precision value and the recall value. Finally we set the parameter to be "a=30" and "u=1.07". And according to the 250 annotated tweets, 131 tweets are labeled as English, the ratio is 52.4%, which shows that there are still lots of non-English tweets after the first round

[2]http://www.let.rug.nl/vannoord/TextCat/

Table 2: Performance of TextCat on Twitter

a	u	Precision	Recall	F1-Measure
10	1.05	0.95	0.725	0.822
20	1.05	0.918	0.77	0.838
30	1.05	0.87	0.82	0.844
30	1.07	0.85	0.85	0.85

coarse filtering. After the second round filtering by TextCat, 5304401 tweets are left, which are noted as "Twitter2011EngClean".

4.2 Heauristic Preprocessing

Preprocessing of the Tweets include some problems: (1) how to handle the user's tag such as "@Sophia", and the hash tag such as "#job", "#SocialMedia". (2) how to recognize the internet slang in the twitter language such as "lol"(laugh it loudly), "lmao"(laughing my ass off). (3) how to cope with those irregular long word such as "ccccoooooolll", ".......aha", "hahahaha".

We converted the user tag "@Sophia" to "-user-", remove the "#" symbol of the hashtag, recognize the url in tweets and converted it to "-url-". For those irregular words, at first we tried some dictionary-based tools to clean the data, for example Jazzy[3], Some simple spell corrector[4]. For each OOV, Jazzy would rely on the morphology and the pronounciation of the OOV to generate top k most possible candidate word. However, such method is heavily relied on the dictionary you used, and the normalizing effect is far from satisfaction. For example, common words such as "haha" would be normalized to "here", "I" is normalized to "ii", "wknds"(weekends) is normalized to "winds", "iTumes" is normalized to "tunes", people's name such as "Louis" would be normalized to "Lousy", internet initials such as "u"(you) would be recognized as "us". proper name such as "INXS"(an Australia band) would be normalized to "ins". All these drove us to normalize the twitter dataset heauristically. We cleaned those sequential dots ".....", break the long connected words, and keep the rest unchanged.

[3]http://jazzy.sourceforge.net/

[4]http://norvig.com/spell-correct.html

5 Retrieving Tweets

We build our system based on the Lucene[5] toolkit. The whole process is given in Fig. 5. After identifying the English tweets, empirically normalizing the tweets, each tweet is given two features for clustering. The first feature real-time "BM25" represents the tweets score using the BM25 with the real-time inverse document frequency information, the second feature comes from SVD decomposition. Both the short query and the long tweets are casted onto the same reduced space, so that we can get the similarity of the query and the tweet on this reduced space. Both features are extracted in the strictly "real-time" scenario.

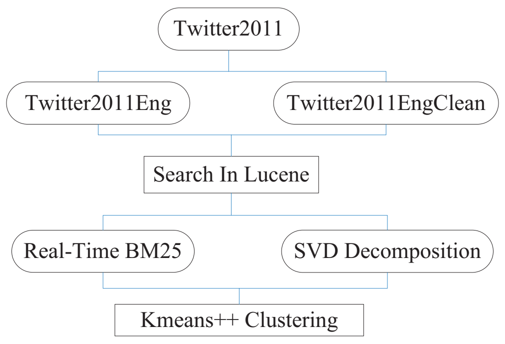


Figure 1: Process Overview

5.1 Query Processing

The high efficiency of Lucene allowed us to retrieve and collect the document for many query words. Filtering those documents that is behind the query timestamp, we can get the "real-time" document frequency for the input query word, and input query phrase. TREC also gives a queryTweetTime element for each query topic, which actually is the latest tweet id for that query topic. This can be used to collect the "real-time" corpus size. After collecting the document frequency and the corpus size, it is easy to use the conventional scoring function to evaluate each tweet.

Empirical method is used to segment the input query, by observing the "real-time" document frequency of the query phrase. Given a long query, in each iteration, we would remove one query word and check the rest query phrase's document frequency. If

[5]http://lucene.apache.org/java/docs/index.html

the document frequency could meet the requirement, a query phrase would be cut off from the original query, and the process would continue with the rest of the query words. For example, the query "mexico drug war" would be segmented as "mexico drug", "mexico", "drug war", "drug", "war".

5.2 Real-Time BM25

As stated above, the "real-time" BM25 is easily implemented, when we get the necessary document frequency with the help of Lucene's high running efficiency. The conventional BM25 formula is given as follows,

$$\text{BM25}(Q,D) = \sum_{i=1}^{n} IDF_{rt}(q_i) \cdot \frac{(k_1+1)\cdot tf(q_i,D)}{k_1\cdot(1-b+b\frac{Len(D)}{avgdl})} \quad (1)$$

$$IDF_{rt}(q_i) = \log\frac{N_{rt}-df_{rt}(q_i)+0.5}{df_{rt}(q_i)+0.5} \quad (2)$$

Suppose the query Q has n query word, noted as q_i, where k_1 and b are parameters, typically set as $k_1 = 1$ and $b = 0.75$. $tf(q_i, D)$ is the raw term frequency of query word q_i in the document(tweet) D, $df(q_i, D)$ is the "real-time" document frequency of query word q_i, N is the "real-time" corpus size.

Lucene is famous for its high performance. However, the ranking modular in Lucene is a little complicated, and some information such as the average document length is not directedly stored in Lucene, and the document length in Lucene is encoded in the Lucene system which is not so straightforward as term frequency. Although there are existing BM25 implementaion on Lucene, we used our own implementation.

So each tweet could be ranked according to this "real-time" BM25 score. However, the "real-time" retrieval is a set retrieval task, which requires the participants to remove those irrelevant documents. That is why we resort to the SVD decomposition and finally clustering on the retrieved tweets.

5.3 SVD Decomposition

SVD(Singular Value Decomposition) is a common technique used in collaborative filtering and the recommendation system. Given the Word-Tweet Matrix built from the retrieved "real-time" tweet collection, this technique fits the problem, and it provides another view on the tweet other than the BM25 formula.

Given the Word-Tweet Matrix M, where each row represents a word, each column represents a tweet. The SVD tries to decompose this matrix into

$$M = U\Sigma V^T \quad (3)$$

where Σ is a diagonal matrix with non-negative singular values on the diagonal, the left matrix U is composed of the eigenvectors of MM^T, the right matrix V is composed of the eigenvectors of M^TM

We used the SVD implementation in LingPipe[6], which also offer a comprehensive and hands-on tutorial on the SVD. The SVD decomposition process follows the work of (Gorrell and Webb, 2005), which is a Generalized Hebbian Algorithm(GHA) for singular value decomposition in natural language processing. Since in NLP, we would face with huge sparse matrix, the GHA algorithm would calculate the eigen vectors based on the single observation of the input matrix, which also use limited space.

The parameters used in the SVD based on GHA algorithm include the number of the singular value, which is set to 20 in our experiment, the learning rate and the iteration steps which control the decomposition speed. A practical issue in the SVD is to fight with the sparse Word-Tweet Matrix. We empirically remove those words that only occur once in the returned "real-time" collection for the input query topic.

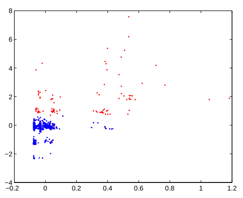
We first return the top 1000 tweets ranked by the "real-time" BM25 formula, then build the word-Tweet matrix, after decompose the matrix, we would get the left singular matrix which represents the projected vector for each word, and the right singular matrix which represents the projected vector for each tweet. Given the input query, each query word q_i corresponds to a vector v_{q_i} in the projected vector space. Each tweet is projected as v_t. So the similarity of the input query and the tweet would be

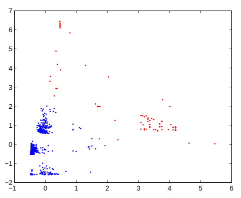
$$SimSVD(Q,D) = (\sum_{i=1}^{n} v_{q_i}) \cdot v_t \quad (4)$$

[6]http://alias-i.com/lingpipe/demos/tutorial/svd/read-me.html

5.4 Clustering

After generating the features for each tweet, we would use KMeans++ to cluster on the returned result, which choose the initials to enhance the robust of the clustering algorithm. Some examples of the clustering result for the query topic is shown in Fig. 5.4.


(a) MB001


(b) MB002

We use the first 2 topics as an example to show the effect of the clustering. The x-axis represents the BM25 feature, while the y-axis represents the SVD feature.

6 Experiment Result

We submitted 2 runs. Tab 5.4 show the evaluation result provided by TREC, which has removed the retweets by empirical method. Given the query judgement files, we re-evaluate the four runs, and report the result in Tab 5.4. All the tweets were evaluated by as "relevant" or "highly-relevant". Given the 50 testing query topics, totally, 49 topics have tweets judged as relevant, 33 topics have tweets judged as "highly-relevant". We both report the evaluation result by "relevance" and "high relevance".

From the table above, we could find that the "high relevance" tweets are scare, and there is a lot of space to improve it, which could be a research focus in the future.

Comparing the result, we could find that the English identification tool which could work fine in the newswire field, would not work so well as in the microblog sphere. In the Twitter dataset, removing the English tweets would make the MAP drop. We think that is because the MAP value is strongly affected by the "recall" value of the retrieval result. Removing the non-English tweets would surely decrease the "recall" value a lot. But the precision would not be affected too much, which can be observed by comparing the P30 and P10 value of these different runs. We should use more sophisticated techniques to handle the Term-Tweet Matrix, which could have improved the overall performance.

7 Conclusion

We report our method and evaluation result for participating the TREC 2011, Microblog Track. We used virtual machine instances to download the dataset parallelly in order to speed up the download speed, removed the non-English tweets, extracted features from each tweet and used clustering method to remove the uninteresting tweets. The result shows that there is still a lot of work in the future to improve the recall value of the retrieval performance.

Acknowledgment

The author wishes to thank the anonymous reviewers for their helpful comments. This work was partially funded by National Natural Science Foundation of China (61003092, 61073069), Shanghai Science and Technology Development Funds(10dz1500104), Doctoral Fund of Ministry of Education of China (200802460066), and Key Projects in the National Science & Technology Pillar Program(2009BAH40B04).

References

Sasa Petrovic, Miles Osborne and Victor Lavrenko. Streaming First Story Detection with application to Twitter. NAACL, Los Angeles, USA. 2010.

Jaewon Yang and Jure Leskovec. Temporal Variation in Online Media. ACM International Conference on Web Search and Data Mining (WSDM '11), 2011.

Bo Han and Timothy Baldwin, Lexical Normalisation of Short Text Messages: Makn Sens a #twitter, ACL 2011

Samuel Brody and Nicholas Diakopoulos, Cooooooooooooooollllllllllllll!!!!!!!!!!!!!! Using Word Lengthening to Detect Sentiment in Microblogs, EMNLP 2011

Kevin Gimpel, Nathan Schneider, Brendan O'Connor, Dipanjan Das, Daniel Mills, Jacob Eisenstein, Michael Heilman, Dani Yogatama, Jeffrey Flanigan and Noah A. Smith, PART-OF-SPEECH TAGGING FOR TWITTER: ANNOTATION, FEATURES, AND EXPERIMENTS. ACL 2011

Xin Zhao, Jing Jiang, Jing He, Yang Song, Palakorn Achanauparp, Ee-Peng Lim and Xiaoming Li, Topical Keyphrase Extraction from Twitter. ACL 2011

Table 3: Evaluation Result: Evaluated by TREC, remove the duplicated retweet

"Relevant"	MAP	R-prec	bpref	P10	P30
filter	0.1425	0.1821	0.2094	0.1857	0.1510
clean.filter	0.1000	0.1410	0.1702	0.1673	0.1340
"high Relevant"	MAP	R-prec	bpref	P10	P30
filter	0.1707	0.1740	0.5024	0.0879	0.0677
clean.filter	0.1017	0.1249	0.3403	0.0606	0.0545

Table 4: Evaluation Result: not remove the retweet

"Relevant"	MAP	R-prec	bpref	P10	P30
rtBM25	0.2458	0.3063	0.3250	0.3367	0.2925
rtBM25.clean	0.2127	0.2737	0.2961	0.3204	0.2789
rtBM25.filter	0.1990	0.2718	0.2794	0.2959	0.2367
rtBM25.clean.filter	0.1298	0.1881	0.1958	0.2755	0.2245
"high Relevant"	MAP	R-prec	bpref	P10	P30
rtBM25	0.1358	0.1422	0.1557	0.0735	0.0544
rtBM25.clean	0.1110	0.1233	0.1321	0.0571	0.0503
rtBM25.filter	0.1265	0.1423	0.1480	0.0612	0.0517
rtBM25.clean.filter	0.0674	0.0784	0.0921	0.0469	0.0463

Long Jiang, Mo Yu, Ming Zhou, Xiaohua Liu and Tiejun Zhao, Target-dependent Twitter Sentiment Classification, ACL 2011

Roberto Gonzalez-lbanez, Smaranda Muresan and Nina Wacholder, IDENTIFYING SARCASM IN TWITTER: A CLOSER LOOK. ACL 2011

Vahed Qazvinian, Emily Rosengren, Dragomir R. Radev and Qiaozhu Mei, Rumor has it: Identifying Misinformation in Microblogs. EMNLP 2011

Fabio Massimo Zanzotto, Marco Pennacchiotti and Kostas Tsioutsiouliklis, Linguistic Redundancy in Twitter. EMNLP 2011

Brendan Meeder, Brian Karrer, Amin Sayedi, R Ravi, Christian Borgs, Jennifer Chayes, We Know Who You Followed Last Summer: Inferring Social Link Creation Times In Twitter. WWW 2011

Shaomei Wu, Jake M. Hofman, Winter Mason, Duncan J. Watts, Who Says What to Whom on Twitter. WWW 2011

Eiji Aramaki, Sachiko Maskawa and Mizuki Morita, Twitter Catches The Flu: Detecting Influenza Epidemics using Twitter. EMNLP 2011

John Burger, John Henderson, George Kim and Guido Zarrella, Discriminating Gender on Twitter. EMNLP 2011

Roy Bar-Haim, Elad Dinur, Ronen Feldman, Moshe Fresko and Guy Goldstein. Identifying and Following Expert Investors on Twitter. EMNLP 2011

Carlos Castillo, Marcelo Mendoza, Barbara Poblete, Information Credibility on Twitter. WWW 2011

Kyung Soon Lee and W. Bruce, Croft and Allan, James: A cluster-based resampling method for pseudo-relevance feedback. In SIGIR '08, pp: 235–242, (2008)

Christopher M. Bishop: Pattern Recognition and Machine Learning. Singapore: Springer 2006, pp: 179-224, (2006)

SE., Robertson, S., Walker and M., Beaulieu: Okapi at TREC7: automatic ad hoc, filtering, VLC and interactive track. NIST SPECIAL PUBLICATION SP, Issue 500, Number 242, pp: 253-264, (1999)

Cavnar, W. B. and J. M. Trenkle, N-Gram-Based Text Categorization, In Proceedings of Third Annual Symposium on Document Analysis and Information Retrieval, Las Vegas, pp. 161-175, 11-13 April 1994

Gorrell, G. and Webb, B., Generalized Hebbian Algorithm for Latent Semantic Analysis. In Proceedings of Interspeech 2005.