Topic-specific Retweet Count Ranking for Weibo

Hangyu Mao, Yang Xiao, Yuan Wang, Jiakang Wang, and Zhen Xiao*

Department of Computer Science, Peking University, China
{mhy, xiaoyang, wangyuan, wangjkcs, xiaozhen}@net.pku.edu.cn

Abstract. In this paper, we study *topic-specific* retweet count ranking problem in Weibo. Two challenges make this task nontrivial. Firstly, traditional methods cannot derive effective feature for tweets, because in topic-specific setting, tweets usually have too many shared contents to distinguish them. We propose a LSTM-embedded autoencoder to generate tweet features with the insight that any different prefixes of tweet text is a possible distinctive feature. Secondly, it is critical to fully catch the meaning of topic in topic-specific setting, but Weibo can provide little information about topic. We leverage real-time news information from Toutiao to enrich the meaning of topic, as more than 85% topics are headline news. We evaluate the proposed components based on ablation methods, and compare the overall solution with a recently-proposed tensor factorization model. Extensive experiments on real Weibo data show the effectiveness and flexibility of our methods.

Keywords: Weibo, micro-blog, retweet, retweet count ranking, social network

1 Introduction

With the development of micro-blogging services, Weibo, the biggest micro-blogging service in China, has changed the organization of its three major entities (i.e., topic, tweet and user) as shown in Figure 1. (1) Topics are ranked according to their popularity in the Hot Topic List as shown in the left column. Generally speaking, topic is the group of all tweets sharing the same #topic name#, but it has its own properties such as topic category and topic information. (2) Tweets are divided into common tweets and recommended tweets as show in the right column. Recommended tweets are usually informative and interesting, and they are shown before common tweets in the topic page. (3) Users are encouraged to read tweets in topic pages rather than scattered tweets in their timelines. In a word, topic is becoming the core unit to organize tweets and users in Weibo.

In fact, hot topic is now the main source of page view (PV) and unique visitor (UV) in Weibo. For example, the PV of the topic #Running Man# increases from 13.23 to 42.81 billion after Weibo introduces the "Super Topic" service last year, and the total PV of the top-20 hot topics increases from 127.65 to 361.68 billion. Besides, topic is also beneficial for improving user experience and

* The contact author.

Fig. 1. The organization of topic, user and tweet in Weibo.

increasing advertising revenue. On one hand, users can know the detailed and representative information of one topic more easily, because all relevant tweets are grouped and ranked together by topic. On the other hand, advertisers can easily find the target users who browse the specific topic proactively, which means that the advertisements will be more effective.

However, as we know, most users only look through the recommended tweets in the first few pages of a topic. Under this condition, to attract more PV/UV, and further obtain the above mentioned benefits of topic, we need to find out *popular tweets* [1] and make them as the recommended tweets for each topic. In reality, finding out popular tweets from all tweets sharing the same #topic name# is not easy. Currently, the procedure can only be done manually, and it is easy to miss the best popular tweets.

Naturally, in this paper, we try to find out the popular tweets for topics in Weibo automatically (and objectively relative to some metrics)?

We formulate this task as a *topic-specific* retweet count ranking problem. Specifically, we predict the retweet count ranking order for all tweets *sharing the same #topic name#, i.e., belonging to the same topic*, and further recommend the highest ranked tweets as the recommended tweets to the corresponding topic (rather than to users). This is why we call our problem *topic-specific* instead of *personalized*. [12, 13] have done a lot of work to show that retweet action should be studied at topic level. We refer the readers to their papers for further details.

More concretely, we propose a Topic-Specific reTweet count Ranking (T-STR) framework to dig this important yet challenging task more deeply. Three main components make up of TSTR framework, making it more applicable in real systems. Firstly, it is impractical to directly deal with the large number of newly-generated tweets in a short time. We propose a Candidate Tweet Filter to filter out unpopular tweets. Secondly, traditional methods cannot derive effective features for tweets, because tweets belonging to the same topic usually have

[1] We measure the *popularity* of a tweet by its retweet count. As pointed out by [15, 25, 27, 28], retweet is the key mechanism for information diffusion on micro-blogging services. A larger retweet count usually means that more users have seen, and will see, the corresponding tweet and topic, and that we will further get more benefits. In fact, researchers often use popular level as the synonym of retweet count [12, 14].

too many shared contents to distinguish them. We propose a LSTM-embedded autoencoder (LSTM-AE) to generate tweet features. This LSTM-AE takes any different prefixes of tweet text as a possible distinctive feature, which makes it very suitable for tweet feature generation. Finally and most importantly, it is critical to fully catch the meaning of topic in topic-specific setting, but we can get little information about topic from Weibo. For example, as shown in Figure 1, there is only one word (i.e., pay attention) about the most important "Two Sessions" (i.e., NPC and CPPCC) in China. We leverage external real-time news information from Toutiao, the most popular news recommendation platform in China, to enrich the meaning of topic, as we find that more than 85% topics are headline news. We also propose a denoising autoencoder (DAE) to extract topic features from those headline news.

In summary, our contributions are two-fold. (1) This work advances the study of *topic-specific* retweet prediction problem, which has not been well studied like traditional retweet prediction tasks as pointed out by [12, 13]. (2) We evaluate the proposed components based on ablation methods, and compare the overall solution with a recently-proposed tensor factorization model. Extensive experiments on real Weibo data show the effectiveness and flexibility of our methods.

2 Related Work

Retweet studies can be roughly divided into retweet analysis and retweet prediction. Retweet analysis aims at understanding why people retweet and which factors impact retweet [1, 18, 19, 26]. Retweet prediction tries to figure out who will retweet a specific tweet or how many times a specific tweet will be retweeted. Our work is an instance of retweet prediction.

Most retweet prediction models formulate retweet count prediction as classification or regression problems [6, 14, 25, 27, 28], and only a few researchers study retweet count prediction from a ranking perspective. [17] want to figure out *who* will retweet messages using a Learning-to-Rank framework. They explore a lot of factors, such as retweet history, followers status, followers active time and followers interests, and find that followers who have common interests are more likely to be retweeters. [9] try to answer *who* should share *what*, and extend this problem into two information retrieval scenarios: user ranking and tweet ranking. They propose a Hybrid Factor Non-Negative Matrix Factorization model to estimate each entry of user-tweet matrix. [22] also study both user ranking and tweet ranking. They train a coordinate ascent Learning-to-Rank algorithm to rank the incoming tweets as well as users, and find that tweet-based features have a better predictive ability. [11] study *personalized* tweet ranking according to their probability of being retweeted so that users can find interesting tweets in a short time. They build a user-publisher-tweet graph to re-rank the tweets.

All the above studies focus on user level retweet prediction. Perhaps, [12] and [13] are the most relevant studies to ours. The authors also investigate retweet prediction at topic level. They propose a tensor factorization model named V2S to model a set of observed retweet data as a result of three topic-specific factors, i.e., topic virality, user virality and user susceptibility. In their work, they use LDA [4] to generate the latten topics for each tweet. In contrast, topic in our work is directly tagged by users using #topic name#, and all tweets going to be ranked share a same topic.

3 TSTR Framework

3.1 Consideration and Design

There are three major considerations and designs in our system.

The first consideration is that how to deal with the large number of tweets. As we know, a lot of new tweets will be generated in one minute. It is impractical to directly extract features for all those tweets in a short time. Considering that we only care about the popular tweets when we do recommendation, it is natural to propose a Candidate Tweet Filter to filter out unpopular tweets.

The second consideration is that how to derive effective features for tweet. On one hand, tweet is short text with a random length. On the other hand, tweets belonging to the same topic usually have many shared words. Traditional bag-of-word methods and topic model methods such as LDA [4] are not suitable for this task, because those methods suffer from either sparsity or inefficiency for short texts [23]. The recurrent neural network (RNN) may be a better choice, because it can summarize and generate word sequence of arbitrary length and distinguish sequences that have same words but in different orders [8]. So we extend traditional RNN-based encoder-decoder structures and propose a LSTM-AE model with attention mechanism for tweet feature generation.

The third consideration is that how to fully catch the meaning of topic in topic-specific setting. As mentioned in the Introduction, Weibo can provide little information for topics. Fortunately, [16, 20] point out that micro-blogging service is more than social network but news media, and over 85% topics are headline news in real world. We also find a similar conclusion in our dataset. So we leverage real-time news information from Toutiao to enrich the meaning of topic. We also propose a DAE model to translate those news information into topic feature.

LSTM-AE and DAE can generate features for tweet and topic, respectively. As for user features, we can crawl them from user database directly. After all features about tweet, topic and user have been generated, we use Tweet Ranker to learn the desired ranking function.

The TSTR framework shown in Figure 2 summarizes our designs.

3.2 Candidate Tweet Filter

We train a random forest with a dynamic filtering threshold as the Candidate Tweet Filter [2]. Tweet having low popular level with high probability will be filtered out, and others are kept as candidates.

[2] Due to space limitation, we move the features used for building Candidate Tweet Filter into the supplemental material.

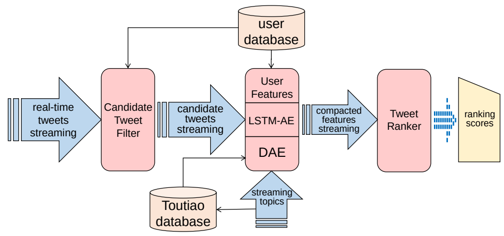


Fig. 2. An illustration of TSTR framework.

3.3 LSTM-AE

A simplified LSTM-AE structure is shown in Figure 3(a). LSTM-AE can deal with the random-length property of one tweet and the similar content property of tweets belonging to the same topic, because it is a special RNN. At the same time, it can avoid overfitting because of the added Dropout Layer. Specially, the attention mechanism makes the model focus on more useful parts of the tweet.

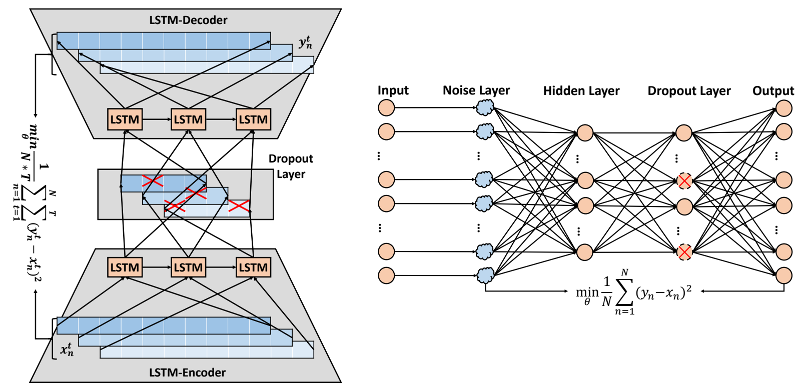
The inputs are word embeddings of each tweet. LSTM-AE tries to reconstruct those inputs by minimize the defined loss function in Equation (1). After training is finished, we extract the outputs of Dropout Layer as tweet features.

Similar RNN-based encoder-decoder structures have been used in other NLP tasks [8, 21]. LSTM-AE extends those models in two ways. During training, we extend the loss function. The classical models try to maximize a conditional log-likelihood. Differently, LSTM-AE tries to minimize the mean squared error between (x_n^t, y_n^t) as follows

$$\min_{\theta} \frac{1}{N * T} \sum_{n=1}^{N} \sum_{t=1}^{T} (y_n^t - x_n^t)^2 \tag{1}$$

where x_n^t is the n-th input embedding at time step t, and y_n^t is a function of x_n^t conditioned on model parameter θ. *This difference reflects our insight about how to drive distinctive tweet features in this special topic-specific setting: any different prefixes of tweet text is a possible distinctive feature.* After training is finished, we extend the usage method. The classical models are usually used for generating a target sequence given an input sequence. They care more about the outputs of LSTM-Decoder. Differently, LSTM-AE is used for generating tweet features. We care more about the outputs of Dropout Layer. Specifically, at each timestep t, LSTM-Encoder generates a new state s^t for current input x^t, considering the latest state s^{t-1} and the latest memory cell C^{t-1}. The final tweet feature $\overline{X}$ is a concatenation of the last hidden state and the attention-weighted state of all intermediate hidden states after dropout. In this way, we can potentially get both the global and the local information of tweet [21].

To be more understandable, we give a simple example based on tweet text "A like B". The input embeddings are $x^1 = IE_{(A)}$, $x^2 = IE_{(like)}$ and $x^3 = IE_{(B)}$. Because of the memory mechanism of LSTM cell, the hidden state of LSTM-AE can represent the feature embeddings $s^1 = FE_{(A)}$, $s^2 = FE_{(A,like)}$ and $s^3 = FE_{(A,like,B)}$ in some extent. During training, LSTM-AE try to generate output embedding y^t as similar as x^t based on s^t. After training, the concatenation $\overline{X}$=$[FE_{(A,like,B)}, \alpha_1 FE_{(A)} + \alpha_2 FE_{(A,like)} + \alpha_3 FE_{(A,like,B)}]$ is used as the final tweet feature. As we can see, any prefixes (A), (A,like) and (A,like,B) of tweet text "A like B" is used to generate different features $FE_{(A)}$, $FE_{(A,like)}$ and $FE_{(A,like,B)}$. This is why we say "any different prefixes of tweet text is a possible distinctive feature". This method is useful for our topic-specific application. In contrast, if we only use $FE_{(A,b,C,D,E)}$ as the final tweet feature, it will be too similar with $FE_{(A,B,C,D,E)}$ to distinguish each other, especially when the tweet text is long. Please note that this is very common in topic-specific setting.



(a) Simplified LSTM-AE. (b) Simplified DAE.

Fig. 3. Simplified LSTM-AE and DAE structures.

3.4 DAE

As mentioned before, we leverage information from Toutiao to enrich the meaning of topic. Specifically, we use topic as keyword to search Toutiao, and only the returned news headlines are processed. News titles frame the interpretation of the article content and provide the most important information for readers, which have been confirmed by many researchers [2, 10]. Note that this method requires a strict time-consistence between topic and the external news.

After we get those news titles, we need to translate them into effective topic features. Our preliminary experiments show that bag-of-word features and topic model features cannot cooperate well with the embedded tweet features. A DAE is proposed to learn embedding features for topic so that both topic features and tweet features have a similar semantic space.

A simplified DAE structure is illustrated in Figure 3(b). We extract a fixed number of verb and noun phrases with largest *tf-idf* values from news title so that the main entity and emotional inclination can be captured on the whole. After that, the corresponding embedding concatenation is used as the input of

DAE. As with LSTM-AE, we extract the outputs of Dropout Layer as topic features after training is finished.

Although this structure is common, we have some insights for this design. Firstly, the number of topic is much less than the number of tweet, so a Noise Layer with different noise variances can create more training data. Besides, we also observe that the combination of Noise Layer and Dropout Layer has a lower loss than Noise Layer alone during training. As this simple structure is enough for good results, we leave other novel models for future work.

3.5 Tweet Ranker

In this paper, we use a pair-wise method to train an ensemble of Multiple Additive Regression Tree (MART) as Tweet Ranker, which is similar to LambdaMART [5, 7]. The final feature vector is the concatenation of topic features, tweet features and user features [3]. With the supervision of actual retweet count, Tweet Ranker will learn the desired ranking function.

3.6 Chinese Word Embedding

LSTM-AE and DAE assume that we have got the embedding representations of all the Chinese words in topics and tweets. Specifically, we use Neural Probabilistic Language Model [3] to generate those embeddings. Our method is supported by the open source PaddlePaddle[4] deep learning platform, which has more than three million Chinese words as the original corpus.

4 Experiments

Due to space limitation, we move the detailed analyses of experimental data and Candidate Tweet Filter into the supplemental material. However, we would like to highlight some conclusions: (1) the experimental data and the data of the whole Weibo system have a consistent topic distribution; (2) more than 93% unpopular tweets can be filtered out by Candidate Tweet Filter.

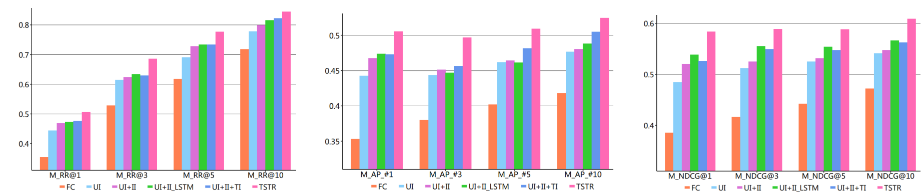
To evaluate the ranking results, we adopt 5 widely used metrics following [11, 24], i.e., Reciprocal Rank (RR), Precision at k (P@k), Average Precision (AP), Normalized Discounted Cumulative Gain at k (NDCG@k), Spearman's Rank Correlation Coefficient (Spearman's ρ). The reported results below contain a "M_" to represent the mean performance on all topics. Some results may also contain a "_#n" to represent that the first n tweets with largest retweet count are relevant. Besides, there are usually 3 to 10 recommended tweets for each topic in Weibo, so we set k and n to 1, 3, 5 and 10 for each group of experiments.

Besides comparing with the recently proposed V2S model [12, 13], we also use the following feature sets and their combinations to train ablation models so

[3] Due to space limitation, we move the user features into the supplemental material.
[4] http://www.paddlepaddle.org/

that we can know the effect of each component in TSTR framework: FC, follower count as feature; UI (User_Info), user features as feature; II (Tweet_Info), original tweet embeddings as feature; TI (Topic_Info), original topic embeddings as feature; II_LSTM, tweet embeddings generated by LSTM-AE as feature; TI_DAE, topic embeddings generated by DAE as feature.

The main experimental results are shown in Figure 4 and Table 1. We analyze those results from the following five aspects.



(a) Reciprocal Rank. (b) Average Precision. (c) NDCG.

Fig. 4. Results of Reciprocal Rank, Average Precision and NDCG.

Table 1. Results of Precision and Spearman's ρ.

	FC	**UI**	**UI+II**	**UI+II_LSTM**	**UI+II+TI**	**TSTR**
M_P@1_#1	0.25568	0.35795	0.40341	0.40120	0.39205	**0.43713**
M_P@1_#3	0.42614	0.52272	0.55114	0.55689	0.53977	**0.61677**
M_P@1_#5	0.50568	0.59091	0.65909	0.66467	0.65909	**0.71856**
M_P@1_#10	0.61364	0.68750	0.73864	0.76647	0.75000	**0.79042**
M_P@3_#3	0.32765	0.39962	0.39773	0.41517	0.41098	**0.44711**
M_P@3_#5	0.43750	0.53409	0.52083	0.52894	0.55682	**0.58084**
M_P@3_#10	0.55114	0.66856	0.64205	0.65269	0.68561	**0.70259**
M_P@5_#5	0.37273	0.42386	0.41364	0.40838	0.42273	**0.45269**
M_P@5_#10	0.50568	0.59318	0.58068	0.56048	0.59773	**0.62874**
M_P@10_#10	0.38295	0.43750	0.42727	0.41617	0.45114	**0.49042**
Spearman's ρ	0.32757	0.33011	0.37825	0.38225	0.37660	**0.38359**

Ablation models. (1) FC: all results of FC are much better than random guess. We also use other single feature to do experiments, and we find that the number of follower has a big impact on the prediction. This has been proved by [6, 14, 15]. (2) UI: we find that ranking according to user features is mostly consistent with ground truth when we only consider information of one entity. In contrast, perhaps due to topic-specific setting, the learned ranking model II is even worse than FC. Detailed results can be found in the supplemental material. [6, 9, 17, 22] use similar baselines, and confirm the importance of user features in their work too. (3) UI+II: we use both user features and tweet features to create a more powerful model. As we can see, most results of UI+II are better than results of UI. In our opinion, feature interaction is the key reason: user features often offer valuable information for tweets. For example, without knowing the user is a gourmet, the II model usually ranks tweets about tourism before tweets about delicacy because tourism is more popular than delicacy in Weibo; and now, the UI+II model can correctly rank tweets about tourism behind tweets about delicacy if those tweets come from a gourmet.

The proposed LSTM-AE. To test the feature extraction ability of LSTM-AE, we use features from UI+II_LSTM to do an experiment. We find that UI+II_LSTM is better than UI+II and even better than UI+II+TI for metrics such as NDCG. To figure out whether the improvements are statistical generalization, we also apply Student's *t*-test[5] to the results and find that the corresponding improvements are significant at the level of 0.05. Those results prove that LSTM-AE can generate effective feature for short tweet text with random length, even though tweets belonging to the same topic have similar contents in topic-specific setting.

Hypothesis testing. To test our hypothesis that real-time news information from Toutiao is potential to boost this retweet count ranking task, we conduct an experiment using features from UI+II+TI and compare it with UI+II. We see that the improvements are marginal, but please note that the corresponding model is only used to test our hypothesis. Considering that the metrics cover a wide range of ranking evaluations and almost all of the results are improved indeed, we conclude that our assumption is reasonable.

The overall TSTR model. The TSTR model is build on features from all UI+II_LSTM+TI_DAE. The *improvements* of TSTR compared to other models for different metrics are shown in Table 2. Firstly, all the entries in this table are positive, which indicates that TSTR is a flexible framework to fit different ranking evaluation metrics simultaneously. Those good results may be related to the unbiased feature generation ability of LSTM-AE and DAE. To prove this inference, we train a TSTR model based on NDCG@5 and find that it can achieve similar good results on other metrics. Quantitatively, the average improvements (excluding V2S column) are bigger than 6% for all metrics, and the significant level for metrics such as NDCG and AP can achieve 0.01. Compared to UI+II_LSTM, we can say that TI_DAE is necessary for good results. Compared to UI+II+TI, we can say that the combination of LSTM-AE and DAE is necessary for good results. Interestingly, the overall ranking improvements (i.e., Spearman's ρ) are much smaller than the top tweets ranking improvements (i.e., other metrics). One possible reason may be that there are many tweets with small retweet count, and it is not easy to fully distinguish them using features generated by LSTM-AE and DAE. This phenomenon means that the proposed TSTR framework is more suitable for applications such as recommendation and hot events detection. In those applications, we care more about the *higher* ranked tweets rather than the ranking of *all* tweets.

The baseline V2S model. From the average improvement rows in Table 2, we can see that V2S performs better than UI+II_LSTM and UI+II+TI but worse than our TSTR model on the whole. As reported in the original papers, V2S outperforms other state-of-the-art content-based and LDA-based models, so we expect that TSTR could have the same ability too. Fine-grained analyses

[5] It can be used to determine if two sets of data are significantly different from each other: https://en.wikipedia.org/wiki/Student%27s_t-test.

Table 2. *Improvements* of TSTR for different metrics compared to UI+II_LSTM, UI+II+TI and V2S models.

	UI+II_LSTM	**UI+II+TI**	**V2S**
M_P@1_#1	8.96%	11.50%	18.23%
M_P@1_#3	10.75%	14.27%	8.87%
M_P@1_#5	8.11%	9.02%	5.02%
M_P@1_#10	3.12%	5.39%	3.49%
M_P@3_#3	7.69%	8.79%	9.73%
M_P@3_#5	9.81%	4.31%	6.86%
M_P@3_#10	7.65%	2.48%	3.95%
M_P@5_#5	10.85%	7.09%	6.24%
M_P@5_#10	12.18%	5.19%	3.69%
M_P@10_#10	17.84%	14.24%	3.59%
Ave. Improv.	**9.70%**	**8.23%**	**6.97%**
M_AP_#1	6.72%	7.00%	7.70%
M_AP_#3	11.17%	8.86%	6.11%
M_AP_#5	10.47%	5.76%	4.87%
M_AP_#10	7.53%	3.95%	3.40%
Ave. Improv.	**8.97%**	**6.39%**	**5.52%**
M_RR@1	7.02%	6.47%	7.70%
M_RR@3	8.13%	9.00%	4.26%
M_RR@5	5.95%	5.96%	2.90%
M_RR@10	3.55%	2.66%	1.92%
Ave. Improv.	**6.16%**	**6.02%**	**4.20%**
M_NDCG@1	8.51%	10.95%	5.98%
M_NDCG@3	6.09%	7.18%	5.88%
M_NDCG@5	6.14%	7.43%	5.56%
M_NDCG@10	7.55%	8.33%	4.56%
Ave. Improv.	**7.07%**	**8.47%**	**5.50%**
Spearman's ρ	0.35%	1.86%	0.66%

show that V2S cannot perform well on metrics such as M_P@1_#1, M_P@3_#3, M_AP#1 and M_RR@1, which means that V2S is not suitable for applications such as recommendation and hot events detection where our TSTR model is a better choice as analysed before.

5 Conclusion

In this paper, we leverage real-time news information from Toutiao to improve the topic-specific retweet count ranking task in Weibo. A TSTR framework is proposed to address this important yet challenging problem. A LSTM-AE and a DAE make up of the core part of TSTR framework. LSTM-AE extends traditional RNN-based encoder-decoder models in two ways for generating tweet features. DAE is designed for translating news information into topic features.

Extensive experiments on real Weibo data show the effectiveness and flexibility of TSTR framework.

We also provide some useful conclusions. (1) User features are more suitable for this topic-specific ranking task than tweet features. (2) Real-time news information from Toutiao is potential to boost applications in Weibo. (3) The proposed TSTR framework is suitable for applications (e.g., recommendation) caring more about the higher ranked tweets rather than the ranking of all tweets.

We will study the following problems in the future: (1) How to leverage other suitable data for the remaining 15% topics that are not reported in Toutiao. (2) How to use network structure information properly in topic-specific setting.

Acknowledgement The authors thank the anonymous reviewers for their comments. This work was supported by the National Natural Science Foundation of China under Grant No.61572044. The contact author is Zhen Xiao.

References

1. Abdullah, N.A., Nishioka, D., Tanaka, Y., Murayama, Y.: User's action and decision making of retweet messages towards reducing misinformation spread during disaster. Journal of Information Processing **23**(1) (2015) 31–40
2. Andrew, B.C.: Media-generated shortcuts: Do newspaper headlines present another roadblock for low-information rationality? The Harvard International Journal of Press/Politics **12**(2) (2007) 24–43
3. Bengio, Y., Ducharme, R., Vincent, P., Janvin, C.: A neural probabilistic language model. J. Mach. Learn. Res. **3** (March 2003) 1137–1155
4. Blei, D.M., Ng, A.Y., Jordan, M.I.: Latent dirichlet allocation. Journal of machine Learning research **3**(Jan) (2003) 993–1022
5. Burges, C.J.: From ranknet to lambdarank to lambdamart: An overview. Learning **11** (2010) 23–581
6. Can, E.F., Oktay, H., Manmatha, R.: Predicting retweet count using visual cues. In: Proceedings of the 22Nd ACM International Conference on Information & Knowledge Management. CIKM '13, New York, NY, USA, ACM (2013) 1481–1484
7. Cao, Z., Qin, T., Liu, T.Y., Tsai, M.F., Li, H.: Learning to rank: From pairwise approach to listwise approach. In: Proceedings of the 24th International Conference on Machine Learning. ICML '07, New York, NY, USA, ACM (2007) 129–136
8. Cho, K., Van Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., Bengio, Y.: Learning phrase representations using rnn encoder-decoder for statistical machine translation. arXiv preprint arXiv:1406.1078 (2014)
9. Cui, P., Wang, F., , Sun, L.: Who should share what?: item-level social influence prediction for users and posts ranking. In: International ACM SIGIR Conference on Research and Development in Information Retrieval. (2011) 185–194
10. Dor, D.: On newspaper headlines as relevance optimizers. Journal of Pragmatics **35**(5) (2003) 695–721
11. Feng, W., Wang, J.: Retweet or not?: Personalized tweet re-ranking. In: Proceedings of the Sixth ACM International Conference on Web Search and Data Mining. WSDM '13, New York, NY, USA, ACM (2013) 577–586
12. Hoang, T.A., Lim, E.P.: Retweeting: An act of viral users, susceptible users, or viral topics? In: SIAM International Conference on Data Mining. (2013)

13. Hoang, T.A., Lim, E.P.: Microblogging content propagation modeling using topic-specific behavioral factors. IEEE Transactions on Knowledge & Data Engineering **28**(9) (2016) 2407–2422
14. Hong, L., Dan, O., Davison, B.D.: Predicting popular messages in twitter. In: Proceedings of the 20th International Conference Companion on World Wide Web. WWW '11, New York, NY, USA, ACM (2011) 57–58
15. Jenders, M., Kasneci, G., Naumann, F.: Analyzing and predicting viral tweets. In: Proceedings of the 22Nd International Conference on World Wide Web. WWW '13 Companion, New York, NY, USA, ACM (2013) 657–664
16. Kwak, H., Lee, C., Park, H., Moon, S.: What is twitter, a social network or a news media? In: Proceedings of the 19th International Conference on World Wide Web. WWW '10, New York, NY, USA, ACM (2010) 591–600
17. Luo, Z., Osborne, M., Tang, J., Wang, T.: Who will retweet me?: Finding retweeters in twitter. In: International ACM SIGIR Conference on Research and Development in Information Retrieval. SIGIR '13, New York, NY, USA, ACM (2013) 869–872
18. Macskassy, S.A., Michelson, M.: Why do people retweet? anti-homophily wins the day! In: ICWSM. (2011) 209–216
19. Meier, F., Elsweiler, D., Wilson, M.L.: More than liking and bookmarking? towards understanding twitter favouriting behaviour. In: ICWSM. (2014)
20. Myers, S.A., Zhu, C., Leskovec, J.: Information diffusion and external influence in networks. In: Proceedings of ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. KDD '12, New York, USA, ACM (2012) 33–41
21. Shang, L., Lu, Z., Li, H.: Neural responding machine for short-text conversation. arXiv preprint arXiv:1503.02364 (2015)
22. Uysal, I., Croft, W.B.: User oriented tweet ranking: A filtering approach to microblogs. In: Proceedings of the ACM International Conference on Information and Knowledge Management. CIKM '11, New York, USA, ACM (2011) 2261–2264
23. Wang, Y., Xiao, Y., Ma, C., Xiao, Z.: Improving users' demographic prediction via the videos they talk about. In: Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, Austin, Texas, Association for Computational Linguistics (November 2016) 1359–1368
24. Xiao, Y., Zhao, W.X., Wang, K., Xiao, Z.: Knowledge sharing via social login: Exploiting microblogging service for warming up social question answering websites. In: COLING, Citeseer (2014) 656–666
25. Yang, Z., Guo, J., Cai, K., Tang, J., Li, J., Zhang, L., Su, Z.: Understanding retweeting behaviors in social networks. In: ACM International Conference on Information and Knowledge Management. (2010) 1633–1636
26. Yuan, N.J., Zhong, Y., Zhang, F., Xie, X., Lin, C.Y., Rui, Y.: Who will reply to/retweet this tweet?: The dynamics of intimacy from online social interactions. In: Proceedings of the Ninth ACM International Conference on Web Search and Data Mining. WSDM '16, New York, NY, USA, ACM (2016) 3–12
27. Zhang, Q., Gong, Y., Guo, Y., Huang, X.: Retweet behavior prediction using hierarchical dirichlet process. In: Twenty-Ninth AAAI Conference on Artificial Intelligence. (2015) 403–409
28. Zhang, Q., Gong, Y., Wu, J., Huang, H., Huang, X.: Retweet prediction with attention-based deep neural network. In: ACM International on Conference on Information and Knowledge Management. (2016) 75–84