First Place Solution for NLPCC 2018 Shared Task User Profiling and Recommendation

Qiaojing Xie, Yuqian Wang, Zhenjing Xu, Kaidong Yu, Chen Wei(✉), and ZhiChen Yu

Turing Robot, Beijing, China
{xieqiaojing,wangyuqian,weichen}@uzoo.cn

Abstract. Social networking sites have been growing at an unprecedented rate in recent years. User profiling and personalized recommendation plays an important role in social networking, such as targeting advertisement and personalized news feed. For NLPCC Task 8, there are two subtasks. Subtask one is User Tags Prediction (UTP), which is to predict tags related to a user. We consider UTP as a Multi Label Classification (MLC) problem and proposed a CNN-RNN framework to explicitly exploit the label dependencies. The proposed framework employs CNN to get the user profile representation and the RNN module captures the dependencies among labels. Subtask two, User Following Recommendation (UFR), is to recommend friends to the users. There are mainly two approaches: Collaborative Filtering (CF) and Most Popular Friends (MPF), and we adopted a combination of both. Our experiments show that both of our methods yield clear improvements in F1@K compared to other algorithms and achieved first place in both subtasks.

Keywords: User profiling · User tags prediction
Multi label classification · Friend recommendation

1 Introduction

Nowadays, UTP and UFR have attracted great attention due to the popularity of social networks. For example, on social networks where people share information and connect to each other, users present themselves with demographical information as well as tags indicating their specialities or interests. These tags form an important part of user profiling for personalized user/item recommendation. In the absence of tags, user generated data or other information could be used to predict tags for users. Meanwhile, current social relations of users can also be used to recommend new users they would like to follow. Since user behavioral data is heterogeneous, it is still challenging to effectively leverage the heterogeneous information for user profiling and recommendation.

This shared task includes two subtasks. Subtask one is UTP which can be considered as a MLC problem. Given users' other information except tags, we need to predict related tags to the users. Subtask two is UFR, where we predict the users a user would like to follow in the future given users' following relationship and other provided information. We will introduce the two subtasks respectively in the following sections.

1.1 UTP

User Profiling can be defined as user information tagging, which is to assign user tags based on the users' past interests/behavior. Wu et al. [3] conducted unsupervised keywords extraction from Twitter messages to tag Twitter users' interests and concerns, and evaluated the performance by human annotators. Lai et al. [2] created a news recommender system to predict users' interests by analyzing their reading behaviors. Some works consider user profile inference as a supervised classification task. Li et al. [1] extracted user information from social media websites like Twitter, Google Plus or Facebook and make predictions for user attributes based on their tweets. Yin et al. [4] proposed a probabilistic model for personalized tag prediction, which integrates three factors: an egocentric effect, environmental effects and web page content.

In this subtask, we cast UTP as a MLC problem. For MLC problem, Binary Relevance (BR) [7] is a classical method. However it lacks of sufficient ability to discover dependencies among labels. To address this issue, various methods have been proposed. Classifier chains (CC) [8] extends BR by taking label dependencies into account. CC links binary classifiers as a chain and feeds the predictions of the earlier classifiers as features to the latter classifiers. Label Powerset (LP) combines multiple tags as new tags for classification. Condensed Filter Tree (CFT) [5] tries to find the best label sequences to make the best prediction. For multi label classification of images, CNN-RNN [6] extends CC by utilizing RNN to model label dependencies.

1.2 UFR

Friend recommendation is either based on topological structures of a social network, or derived from profile information of users.

Traditional methods use Friend-of-Friend (FOF) to obtain candidate friends set. This set of friends can be then sorted by several criteria including the popularity of friends and the number of shared friends between the target user and the candidate friends. This approach requires the existence of second degree linkage of users in data.

Semantic based methods recommend friends that are similar to the target user. As for similarity measurement, heterogeneous information could be used. Chin et al. [10] recommended friends to the target user using proximity and homophily. Xiao et al. [11] analysed the personality of the users by mining their tweets content and recommended friends with similar personalities. Wang et al. [12] proposed Friendbook system which collects user-centric knowledge from sensors on the smartphone and modeled life styles of users in order to suggest friends who share similar life styles with the target user. Gou et al.

[13] designed SFViz system that helps people seek friends with similar interests. Feng et al. [14] linearly combined multiple similarity measurements to find friends similar to the target user. This approach can provide with precise personalized recommendation based on carefully selected features. However a great effort is needed to collect appropriate information from users as well as their friends.

Classic recommendation approaches can also be applied to friend recommendation. In this case, a person may have the roles of user and friend at the same time. Collaborative filtering is amongst the most popular recommendation algorithms [15,16]. Memory-based collaborative filtering contains user-based and item-based algorithms, which make recommendations according to the similarity of user-item behaviors. Another category of collaborative filtering algorithm is matrix factorization, which learns a dense distributed representation for each user and item. Different variations of matrix factorization have been explored, such as SVD, LDA and ALS. The two previous algorithms are usually applied to explicit data such as movie ratings, and ALS could be more adapted to implicit datasets [17].

Recent works apply deep neural networks to friend recommendation. Liu et al. [18] combined deep learning techniques and collaborative information to explore the user representations latent behind the topology and content. Ding et al. [19] extracted deep features of users using CNN and performed recommendation using Bayesian Personalized Ranking.

2 Proposed Methods

2.1 UTP

The task of UTP is to predict tags which are related to a user. We cast UTP as a MLC problem. Compared to the multi-class classification, MLC is different: multi-class refers to classifying instances into one or more classes which does not take the label dependencies into account, while MLC explicitly models the dependencies among labels. Motivated by CNN-RNN [6], we utilize CNN to capture rich representations of users. We employ RNN to model dependencies among labels. Unlike CNN-RNN [6], which only considers the previous prediction of the maximum probability, we think and prove that the previous prediction of multiple labels will have an effect on later predictions. The illustration of the CNN-RNN framework is shown in Fig. 1.

CNN Module. We employ CNN to get the representation of each user profile. We use $c_{i,j}$ to represent the feature map element of i-th row and j-th column:

$$c_{i,j} = f\Big(\sum_{d=0}^{D-1}\sum_{m=0}^{F-1}\sum_{n=0}^{F-1} w_{d,m,n}x_{d,i+m,j+n} + w_b\Big) \tag{1}$$

where $D, F, w_{d,m,n}, x_{d,i+m,j+n}$ indicate the depth, filter size, the filter and pixel in the image. d, m, n represent d-th layer, m-th row and n-th column.

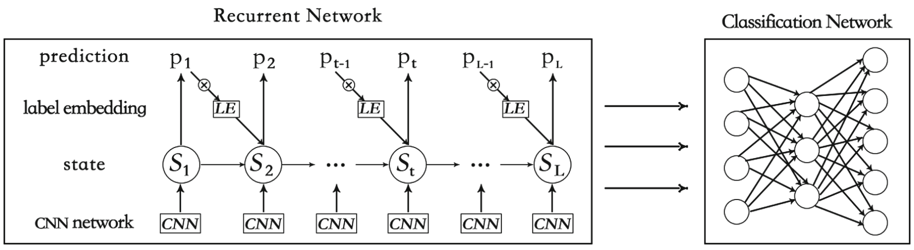

Fig. 1. The architecture of the proposed CNN-RNN model. Our proposed framework consists of three key modules: the output of CNN is employed as the user profile representation; the RNN captures the dependencies among labels, and the classification network combines the CNN output and the output of the recurrent layer as features to compute label probability.

Suppose n filters are used and the n resulting feature maps are $C^{(1)}, ..., C^{(n)}$. Then a pooling operation Pool$(\cdot)$ is applied to each of these n feature maps to produce n p-dimensional vectors Pool$(c^{(1)}), \cdots,$ Pool$(c^{(n)})$. The output of CNN is denoted by o_{CNN}:

$$o_{CNN} = \text{Flatten}(\text{Concat}(\text{Pool}(C^{(1)}), \cdots, \text{Pool}(C^{(n)}))) \tag{2}$$

RNN Module. The label prediction of time step t is represented as a vector p_t, the prediction label embedding e_t can be obtained by multiplying the predicted vector with a label embedding matrix U,

$$e_t = Up_t \tag{3}$$

We use the output of CNN and the previous label prediction information as combined features to feed to the recurrent layer:

$$x_t = \text{Concat}(o_{CNN}, e_{t-1}) \tag{4}$$

$$h_t = f(W_x x_{t-1} + W_h h_{t-1} + W_c c_t) \tag{5}$$

$$p_t = W_p h_t + b_p \tag{6}$$

where h_t, x_t, c_t indicate the hidden state of RNN, the RNN input and the cell state at time step t respectively. W_x, W_h, W_c, W_p, b_p are the parameters to be learned during the training process. f is an activation function.

Classification Network. We combine the output of CNN and the previous label prediction embedding as features to feed the classification network and calculate the predicted distribution over labels o_{out}:

$$o_{out} = W_o\text{Concat}(o_{RNN}, o_{CNN}) + b_o \tag{7}$$

where o_{RNN}, o_{CNN} indicate the last hidden state of RNN and the CNN output. W_o, b_o are parameters to be learned during training.

2.2 UFR

We assume that on social platform, users may follow friends because they know their friends in the real world, or they are interested in the tweets posted by their friends, or because the friends are celebrities or the friend accounts are popular.

Based on these hypothesis, the social and other types of information about all users and friends could be useful for recommending friends to users. However, neither sufficient tweets nor tags for friends are available in this task, we can not rely much on the features of friends. And thus, we decided to fully leverage social relations and user features.

Firstly we tried ALS, an efficient matrix factorization algorithm, which decomposes the sparse user-item matrix into low-dimensional user factors p and item factors q:

$$r'_{u,i} = p_u^T q_i \tag{8}$$

where $r'_{u,i}$ is 1 if user u follows item i, otherwise is 0.

Different from SVD which is optimized using Stochastic Gradient Descent (SGD), ALS optimizes the two types of factors alternatively by fixing one type while optimizing the other. Thus, ALS can be easily parallelized, and for implicit datasets where the user-item matrix is less sparse than explicit datasets, ALS is usually more efficient.

Apart from social following information, we also tried to incorporate other types of user information in the Collaborative Filtering algorithm, therefore we decided to implement user-based CF. For simplicity of implementation, given M users and N friends, we formulate user-based CF as following:

$$P = S \times C \tag{9}$$

where $C \in \mathbb{R}^{M \times N}$ is the user-friend matrix and each element $c_{i,j}$ is 1 if the user i follows friend j, otherwise is 0; $S \in \mathbb{R}^{M \times M}$ and each element $s_{u,v}$ represents the similarity between user u and user v; and $P \in \mathbb{R}^{M \times N}$ is the prediction result matrix and each element $p_{i,j}$ represents the score for user i and friend j. Using P, we filter out friend ids that are already friends of each user, and then sort the rest of the friends by prediction scores.

For this task, we calculated different types of similarity values between users, and the similarity matrix S is formulated as a weighted sum of different similarity matrices:

$$S = \sum \alpha_k S_k \tag{10}$$

$$S_k = \mathrm{Norm}(G_k) \times \mathrm{Norm}(G_k)^T \tag{11}$$

where α_k is a scalar weight, and $G_k \in \mathbb{R}^{M \times F}$ is the user-feature matrix with each element $g_{i,j}$ equals 1 if user i has feature j, otherwise equals 0. As for features, we used users' social relations, tags, check-in categories, tweets and profile information which includes gender, province and city. For social relation similarity, features are the set of friends. For tags, check-in and profile similarities, features are the total set of values of the corresponding information. Using tweets information, we compared two types of similarities: one uses the 10,000 most frequent words tokenized with LTP [21]; the other uses the 10,000 most frequent topics including hashtags surrounded by "#", user accounts initialized with "@", and emoji tags surrounded by "[" and "]". Our motivation behind topic extraction is that we assume that these topics represent users' emotions, their preferences on hot topics and popular accounts. We compared several combinations of these similarities and the results will be discussed in the next section.

3 Experiments

3.1 Data Analysis

In this task, there are several data sets given, including users social information, users tags, users check-in and tweets, as well as users profile information including their gender, city and province.

In users' social information, there are 56,217 users and 2,242,334 friends in total. 20% of the users (11,602) have only 1 friend and 82% of friends (1,860,094) are followed by only 1 user. The intersection of user and friend set is 12,674. All these statistics show that we have a quite sparse social network in question. As for user information, although all the users have profile and check-in information, only 11,714 users have tags, and only 9,343 users have tweets. Friends have even less related information. Lacking of useful user and friend information makes it hard to leverage similarities between users and their friends. Detailed statistics for users and different types of information are described in Table 1.

Table 1. Analysis of all the provided information (Coverage of users/friends is the number of users/friends having the corresponding types of information)

Information type	Number of values	Coverage of users	Coverage of friends	Total ids
Tweet	-	9,343	4,284	9,743
Tag	22,211	11,714	4,061	11,995
Check-in	260	56,217	12,936	60,000
Gender	2	56,217	12,923	59,320
Province	37	56,217	12,923	59,320
City	55	56,217	12,923	59,320

For submission of subtask one, we need to predict tags for 2,776 user ids. 2,720 of these ids have social relations, but only 769 of them have tweets. None of the ids has tag information in the provided data.

For subtask two, 33,857 user ids need to be recommended of potential friends based on their existent social relations. Among them, 6,091 user ids have no more than 5 friends in all the provided data, which would probably be involved in cold start problems.

3.2 Evaluation Metrics

The evaluation of both subtasks will use F1@K as defined below:

$$P_i@K = \frac{|H_i|}{K} \tag{12}$$

$$R_i@K = \frac{|H_i|}{|V_i|} \tag{13}$$

$$F1_i@K = \frac{P_i@K * R_i@K}{P_i@K + R_i@K} \tag{14}$$

$$F1@K = \frac{1}{N}\sum_{i=1}^{N} F1_i@K \tag{15}$$

where $|H_i|$ is the correctly predicted item set (item refers to tag in UTP and friend in UFR) for user i's top prediction, $|V_i|$ is the ground truth item set for user i. $P_i@K, R_i@K$ and $F1_i@K$ are the precision,recall and F1 for user i. In UTP, we set $K = 3$. In UFR, we set $K = 10$.

3.3 Experimental Results and Analysis

UTP. As aforementioned, for UTP submission task, only 769 users have tweets. We combined users' tweets, social links, check-ins and profile information as features for training and prediction. In this task, we chose as baselines four advanced approaches solving MLC problems: BR [7], CC [8], LP and Adaptation Algorithm (AA). The first three approaches use Decision Tree (DT), Naive Bayes (NB) and Random Forest (RF) methods, while AA is implemented with MLKNN. In addition, in order to verify the importance of label dependencies, we employ CNN [9] as a strong baseline. For tweets information, Deep learning(DL) approaches use 300-dimensional word2vec[1] vectors for training, while others use bag-of-words features.

We compared with 11 methods on the given dataset, and the results for $P@K$, $R@K$, $F1@K(K{=}3)$ are shown in Table 2. From Table 2, we can see that our proposed CNN-RNN model achieved the best performance on all metrics and our model outperformed the second best method by 39.59% in $P@K$.

It is obvious that DL approaches outperformed other traditional methods. We notice that the BOW model is not enough to represent the user profile since the BOW model does not consider the spatial correlation among features, while CNN uses convolution and pooling operation to capture richer information from different regions of the feature maps.

[1] https://code.google.com/p/word2vec.

Table 2. Results in $P@K$,$R@K$,$F1@K$, bold numbers indicate the best results each line.

Approaches	Methods	Metrics		
		P@K	R@K	F1@K
BR	DT	5.68%	3.31%	2.09%
	NB	8.07%	5.56%	3.29%
	RF	5.23%	3.29%	2.02%
CC	DT	5.41%	3.19	2.00%
	NB	9.13%	6.40%	3.76%
	RF	5.50%	3.41%	2.10%
LP	DT	5.79%	3.74%	2.27%
	NB	6.13%	4.31%	2.53%
	RF	5.43%	3.54%	2.14%
AA	MLKNN	5.39%	3.24%	2.02%
DL	CNN	19.06%	10.91%	6.94%
	CNN-RNN	**58.65%**	**31.97%**	**20.69%**

Regarding label dependencies, our model and LP outperformed BR. Compared with CNN, our method takes label dependencies into account and outperformed CNN by 13.75% in $F1@K$. We assume that users' tags in social networks are usually relevant. Users who are interested in a certain area are usually interested in related fields.

UFR. For our off-line experiments, we split social data to form off-line training and test set. For each user, we randomly choose 80% of his friends as training set and the rest as test set. In this way we assure that all the test users appear in training set. We then filter out friends set in the test set to make sure they appear in users or friends set of the training data. Since we already have the set of user ids to predict for the shared task, we only keep those user ids in our test data. It provides with the information of 56,217 users and 1,861,408 friends in training set, and 28,129 users and 167,679 friends in test set. In order to efficiently do the computation and ensure cover as much user ids in test set as possible, we filter out friends set and only keep friends with 2 or more followers. As a result of this filtering process, we have 47,957 users and 293,254 friends in training set and 24,688 users in test set. We can observe that nearly 20% of the user ids to be predicted have no more than 5 friends in all the provided data. And thus we split out our test set into two partitions: users with more than 10 friends in training data, and users with 10 or less than 10 friends. In total, the first test partition contains 10,438 users, and the second contains 14,250 users. At submission stage, we took all the provided data as training set to conduct predictions.

Firstly, we compare user-based methods (UB) using different information sources as similarity features. From Table 3, it is clear that using social relations as features is the best, and adding any other type of information will decrease the performance. The reason is possibly that it is straightforward to use users' past social behavior as features rather than other irrelevant ones when to predict users' future social relations.

Table 3. Results of user-based methods using different features on the two test set partitions. ("Coverage of users" means the number of users having recommendation results using each method; "$F1@K$ cov" refers to the F1 score within users covered by the corresponding method; "Added to UB-social" refers to the results of the user-based method on all the users, combining social and the corresponding information with equal weights as similarity features using formula(8))

methods	Test partition 1			Test partition 2		
	Coverage of users	$F1@K$ cov ($F1@K$ total)	Added to UB-social (F1@K)	Coverage of users	$F1@K$ cov ($F1@K$ total)	Added to UB-social (F1@K)
UB-social	10438	**2.59% (2.59%)**	-	14212	**1.35% (1.34%)**	-
UB-tag	1816	1.43% (0.25%)	−0.05%	3874	0.14% (0.03%)	−0.26%
UB-vocab	1573	1.41% (0.21%)	−0.07%	4048	0.09% (0.02%)	−0.34%
UB-topic	1572	1.71% (0.25%)	−0.04%	4039	0.14% (0.04%)	−0.30%
UB-gender	10438	2.19% (2.19%)	−0.40%	14250	0.48% (0.48%)	−0.86%
UB-province	10438	2.09% (2.09%)	−0.49%	14250	0.46% (0.46%)	−0.87%
UB-city	10436	2.00% (2.00%)	−0.58%	14249	0.45% (0.45%)	-0.89%

We also tried ALS and Most Popular Friends(MPF). MPF is an intuitive method simply recommending K most popular friends to each user after removing those who are already friends of the user. We set 20 dimensions for user and friend factors, and conduct 50 iterations while training ALS. Table 4 shows the performance comparisons of the three methods.

It can be seen from Table 4 that ALS achieves the best performance on the test partition 1, while UB-social performs the best on test partition 2.

4 Conclusion

In this paper, we compared our algorithms with other mainstream advanced methods in user profiling and recommendation, and our proposed methods

Table 4. Results using different methods on the two test set partitions

Methods	Test partition 1			Test partition 2		
	$P@K$	$R@K$	$F1@K$	$P@K$	$R@K$	$F1@K$
MPF	7.34%	3.01%	2.00%	0.75%	1.18%	0.44%
UB-social	9.39%	3.93%	2.59%	**1.98%**	**6.37%**	**1.34%**
ALS	**12.48%**	**5.13%**	**3.40%**	1.24%	1.99%	0.72%

achieved the best performance in both subtasks. For UTP subtask, we combined tweets, socials, check-ins and profiles as feature for training and prediction through data analysis. We proposed deep learning framework CNN-RNN, which employ CNN to get the user profile representation and model the dependencies among labels by RNN. Experiment results show that utilizing RNN mechanism to model label dependencies is effective for this MLC problem.

For subtask two, we combine collaborative filtering methods with MPF to conduct friend recommendations. Our submission results are based on ALS for most of the users, for the rest of the users who do not have ALS recommendation results due to lack of social relations, we simply propose K most popular friends after removing their existent friends. However we found that for users with no more than 10 friends in the training data, user-based CF is much more efficient than MPF and ALS. In the future, the ensemble of different recommendation methods is worth investigating deeply. Ranking methods like wide & deep [20] could also be tried to re-rank friend candidates set generated by collaborative filtering methods. Another possible way of studying this subtask is to consider friend recommendation as a MLC problem and leverage heterogeneous information in the process.

References

1. Li, J., Ritter, A., Hovy, E.: Weakly supervised user profile extraction from twitter. In: Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics, (Volume 1: Long Papers), vol. 1, pp. 165–174 (2014)
2. Lai, Y., Xu, X., Yang, Z., et al.: User interest prediction based on behaviors analysis. Int. J. Digit. Content Technol. Appl, 6(13) (2012)
3. Wu, W., Zhang, B., Ostendorf, M.: Automatic generation of personalized annotation tags for twitter users. In: Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics. Association for Computational Linguistics 2010, pp. 689–692 (2010)
4. Yin, D., Xue, Z., Hong, L., et al.: A probabilistic model for personalized tag prediction. In: Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. ACM, pp. 959–968 (2010)
5. Li, C.L., Lin, H.T.: Condensed filter tree for cost-sensitive multi-label classification. In: International Conference on Machine Learning, pp. 423–431 (2014)
6. Wang, J., Yang, Y., Mao, J., et al.: Cnn-rnn: a unified framework for multi-label image classification. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 2285–2294. IEEE (2016)

7. Tsoumakas, G., Katakis, I., Vlahavas, I.: Mining multi-label data. Data mining and knowledge discovery handbook, pp. 667–685. Springer, Boston (2009)
8. Read, J., Pfahringer, B., Holmes, G.: Classifier chains for multi-label classification. Mach. Learn. **85**(3), 333 (2011)
9. Kim, Y.: Convolutional neural networks for sentence classification (2014). arXiv preprint arXiv:1408.5882
10. Chin, A., Xu, B., Wang, H.: Who should I add as a "friend"?: a study of friend recommendations using proximity and homophily. In: International Workshop on Modeling Social Media, pp. 1–7 (2013)
11. Xiao, P., Fan, Y.Q., Du, Y.J.: A personality-aware followee recommendation model based on text semantics and sentiment analysis. In: Huang, X., Jiang, J., Zhao, D., Feng, Y., Hong, Y. (eds.) NLPCC 2017. LNCS (LNAI), vol. 10619, pp. 503–514. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-73618-1_42
12. Wang, Z., Liao, J., Cao, Q.: Friendbook: a semantic-based friend recommendation system for social networks. IEEE Trans. Mob. Comput. **14**(3), 538–551 (2016)
13. Gou, L., You, F., Guo, J., et al.: SFViz:interest-based friends exploration and recommendation in social networks. In: International Symposium on Visual Information Communication, pp. 1–10. ACM (2011)
14. Feng, S., Zhang, L., Wang, D.: A Unified Microblog User Similarity Model for Online Friend Recommendation. Commun. Comput. Inf. Sci. **496**, 286–298 (2014)
15. Hannon, J., Bennett, M., Smyth, B.: Recommending Twitter users to follow using content and collaborative filtering approaches. ACM Conference on Recommender Systems, pp. 199–206. ACM (2010)
16. Chen, T., Tang, L., Liu, Q., et al.: Combining factorization model and additive forest for collaborative followee recommendation. In: KDD-Cup Workshop (2012)
17. Hu, Y., Koren, Y., Volinsky, C.: Collaborative filtering for implicit feedback datasets. In: Eighth IEEE International Conference on Data Mining, pp. 263–272. IEEE (2009)
18. Liu, Y., Chen, X., Li, S., Wang, L.: A user adaptive model for followee recommendation on Twitter. In: Lin, C.-Y., Xue, N., Zhao, D., Huang, X., Feng, Y. (eds.) ICCPOL/NLPCC -2016. LNCS (LNAI), vol. 10102, pp. 425–436. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-50496-4_35
19. Ding, D., Zhang, M., Li, S.Y., et al.: BayDNN: Friend Recommendation with Bayesian Personalized Ranking Deep Neural Network. In; Proceedings of the 2017 ACM on Conference on Information and Knowledge Management, pp. 1479–1488. ACM (2017)
20. Cheng, H.T., Koc, L., Harmsen, J., et al.: Wide & deep learning for recommender systems. In: The Workshop on Deep Learning for Recommender Systems. ACM, pp. 7–10 (2016)
21. Che, W., Li, Z., Liu, T.: LTP: A Chinese language technology platform. In: Proceedings of the Coling 2010, Demonstrations, Beijing, China, pp. 13–16, August 2010