Beijing University of Posts and Telecommunications(BUPT) at TREC 2016:A Rating Model Based on Tags for Contextual Suggestion

Danping Yin, Siyuan Gao, Zonghui Peng, Yameng Li, Ruifang Liu

PRIS Lab, Beijing University of Posts and Telecommunications, China
xiaoyin0110@bupt.edu.cn

ABSTRACT
In this paper we focus on the effort of Beijing University of Posts and Telecommunications (BUPT) on the TREC 2016's Contextual Suggestion Track. The problem we are supposed to tackle is how to make suggestions for a particular person with the provided context as well as its preferences. Basically we regard tags as the most important factor, and get ratings for different attractions with the ratings of tags. Also we use Collaborative Filtering to fill the missing ratings. A ranked list is generated after calculating users' ratings for candidate attractions.

Keywords: Tags, Ratings, Collaborative Filtering

1. INTRODUCTION

The Contextual Suggestion Track has been engaged on searching techniques for proposals about a trip destination based on context and user interests since 2012. Participants are given a set of contextual information including a city the user is located in, a trip type, a trip duration, a type of group the user is travelling with, the season the trip will occur in, and attractions' details which the user prefers to. Besides, a list of 50 candidates is provided in the Phase2 experiment for participants to reorder.

Like the TREC 2015 Contextual Suggestion Track, participants are allowed to return result either in the Phase 1 experiment or in the Phase 2 experiment. The Phase1 experiment is a collection based task which requires suggestions consisting of a ranked list of up to 50 attractions based on the dataset either from the open web or the TREC CS Web Corpus released as an additional data. The Phase 2 experiment is a reranking task which aims to rerank the 50 candidates provided by the file "Phase2_requests.json". We participated both Phase 1 and Phase 2 experiments.

In this paper we present the detailed work of Beijing University of Posts and Telecommunications (BUPT) in the TREC 2016 Contextual Suggestion Track. The problem we aimed to solve is How to get the ratings of attractions depending on user's preferences and related context. In consideration of abundant tags given in the profile, we decided to take up the task in perspective of tags. Thus the detailed problems we figured out are as follows:

(1) How do we calculate the rating for a particular tag?

(2) How do we get the rating for a particular attraction with the rating of tags?

In Section 2 we detail our method of Contextual Suggestion. Section 3 is the description of the experimental result. Moreover, conclusions and future work is illustrated in Section 4.

2. OUR METHOD

The procedures of our work are listed:

(1) Crawling data from open web to obtain more comprehensive corpus,

(2) Getting the candidates,
(3) Predicting the missing ratings,
(4) Generating the ranked list,
(5) Adjusting the scores.

The following sub-sections are the descriptions of the processes above.

2.1 Data Gathering

From the profiles given by the Phase 1, we get useful information of attractions such as a city id indicating where a certain attraction lies in, a URL link containing more details of the attraction in the open web, and a particular title. However these are obviously not enough, as tags and ratings of attractions are particularly necessary for our method. Thus we tried to crawl more detailed items for attractions with the help of Yelp API and Foursquare API. The detailed items, which are picked from the crawled text, contain rating, a tag list, and the count of reviews for a certain attraction. We differed the crawling source by attractions' URL. For URLs possessed by Yelp domain, we searched information by Yelp API; For URLs belonging to Foursquare domain, we crawled details through Foursquare. We crawled data for Phase 1 experiment as much as possible, so that abundant data are available. For Phase 2 experiment, it is necessary to make sure that the candidates in lack of tags are supplied detailed items through crawling from the web.

After crawling information on Yelp and Foursquare, we extracted ratings, tags and count of reviews for further use. Finally, 14991 candidate suggestions are crawled out from 18755 candidate suggestions given in Phase 2.

2.2 Getting the candidates

We now gain a big dataset of plentiful attractions in different cities.

In Phase 1, we considered the attractions in the dataset located in the same city with which mentioned in the context of a particular user. After throwing away those with the count of reviews less than 300, we regarded the collection of attractions as the original candidates.

The problem we aim to solve in the Phase 2 experiment is generally the same as that in Phase 1, namely rating the candidate attractions and getting the ranked list. The difference is that candidates are given in the file with a set of suggestions requests that were made during the Phase 1 experiment. Because the candidates provided lack items like ratings and tags, we supplement the candidates' records with tags and other information crawled from the open web.

2.3 Predicting the miss ratings by CF(Collaborative Filtering)

In view that attractions in the preference list are partly marked with ratings, an approach to filling the missing ratings is needed. After deeply diving into the data, we used CF (Collaborative Filtering) to predict the missing ratings.

There are 442 users along with 106 tags in the provided request file. Matrix $W_t = [v_1, v_2, v_3 \ldots v_n]$ is constructed by us to represent the data. The dimension of W_t is 442×106, where v_n is a vector with a dimension of 442×1, and n is 106. The matrix W_t is partly filled with the ratings of each user. Tag-based CF method predicts the missing ratings by making use of the tag-vector similarity.

The similarity is calculated by the formula (2-1).

$$Sim(v_e, v_p) = 0.5 + 0.5 \times \frac{v_e \cdot v_p}{|v_e||v_p|} \quad (2-1)$$

v_e represents the existed rating vector and v_p represents the to-be-predicted rating.

The miss rating is calculated by the formula (2-2).

$$Rating += rating_e \times Sim(v_e, v_p) \quad (2-2)$$

$rating_e$ is the existed ratings, while $Rating$ is accumulated gradually. After this process, the matrix W_t is supposed to be all filled by ratings.

Also, we noticed that some tags of candidates which crawled from the Internet shared the same meaning with tags of the aforementioned 106 tags but shown as different words or phrases. After consideration, we used Word2Vector tools by Google for mapping the two tags with the same meaning but from various sources. For tags that do not exist in the Word2Vector training data, we map them manually.

2.4 *The Model of Generating the ranked list*

With the help of the profile and context, we are to obtain a candidates' ranked list of attractions by utilizing the following three pieces of information:

• the user's preferences,

• the average online rating r_a,

• the set of category tags C_a associated with each attraction.

With the candidate attractions, we model our rating method as follows.

First, we collect every tag marked by a user as tag-user pairs, then calculate the rating for each tag in the tag-user pairs in terms of the user's preferences. For each pair, the rating ranges from 0 to 5.The formula (2-3) is as follow:

$$R(user, C_a) = \frac{\sum_{attractions\_marked\_C_a} R(user, attraction)}{N_{attraction}} \quad (2-3)$$

$R(user, attraction)$ represents the rating for an attraction marked by an user, $N_{attraction}$ represents the count of attractions in the preference list, $R\big((user, C_a)\big)$ represents user's rating about the tag.

Next, we computed a rating for the candidates with user's ratings about tags. The formula (2-4) is as follow:

$$R(user, candidate) = \frac{\sum_{C_a\_of\_candidate} R(user, C_a)}{N_{C_a}} \quad (2-4)$$

N_{C_a} represents the count of tags marked on a candidate, $R(user, candidate)$ represents the user's rating of a candidate.

The final ranking method of Phase 1 and Phase 2 is not the same.

In phase 1 experiment, the rating $R(user, candidate)$ is divided into 10 levels in steps of 0.5 length. Higher the score is, higher the level will be. We get a ranked list of attractions for each user through the attractions' level. Then we take the quality of the attractions into account, which is measured by the average online rating r_a. When attractions are at the same level (as mentioned above), the one owning higher r_a is ranked higher. Finally we get the final suggestion ranked list.

The ranking way of the Phase 2 is stated in the next sub-section 2.5.

2.5 *The Adjustment of Scoring*

We score each candidate mainly from two aspects: One is how much a user may prefer the kind of an attraction, namely $R(user, candidate)$. Since ratings of tags to every user are generated using collaborative filtering, we can measure a user's preference to an attraction by using a mean or a max function on the all of its tags (Formula 2-4 only showed the mean function). For example, an attraction has three tags: sports, fishing and entertainment, with scores of 3, 4 and 5 respectively by a user; then the score of this situation will be 4 if we use a mean function or 5 if we use a max function. The other aspect considered is the rating of an attraction. As ratings of attractions have been crawled on Yelp and Foursquare, we can take advantage of the ratings from the two sources. However, what we should make clear is the ratio about the two sources. If it shows the rating on Yelp is more believable, more weights on ratings of attractions from Yelp should be put, otherwise we put more weights on ratings from

Foursquare. Another factor of the weights is the count of reviews of the rating. This comes from the intuition that if a rating from Yelp or Foursquare has very few review counts, it is likely to have large variance, which is consequently unbelievable. Thus we put less weight on ratings if they have few review counts. And the final score is a combination of the three parts of scores mentioned above, i.e. users' rating of attractions, rating from Yelp and rating from Foursquare.

3. EXPERIMENT RESULTS AND ANALYSIS

We submitted one run for Phase 1 experiment and three runs varying the function of calculating the users' rating of attractions and weights of users' rating of attractions, rating from Yelp and rating from Foursquare in the Phase 2 experiment. In runA we put a higher weight on ratings from Yelp (0.4), a lower weight on Foursquare (0.2), and use a max function when calculating the score of preference; In runB we put a higher weight on Foursquare(0.4), a lower weight on Yelp(0.2), and also use a max function; In runC we put average weight on the two sources(0.3) and use a mean function.

Table 1.Performance in Phase1

	ndcg@5	Reciprocal Rank	P@5
bupt_runA	0.2395	0.5366	0.3475
TREC Median	0.2133	0.5041	0.3508
TREC Worst	0.0367	0.1113	0.0721
TREC Best	0.5876	0.9809	0.7213

Table 1 shows the comparison between our run bupt_runA with the comprehensive performance of 15 Phase 1 runs. It is clear that our result performs better than the TREC median score by ndcg@5 and Reciprocal Rank.

Table 2.Overall Performances of Phase 2

	ndcg@5	Reciprocal Rank	P@5
runA(cs.4_.2_max)	0.2758	0.5932	0.4238
runB(cs.2_.4_max)	0.2934	0.6250	0.4479
runC(cs.3_.3_avg)	0.2469	0.6009	0.3790
TREC Median	0.2562	0.6015	0.3931
TREC Worst	0.0266	0.1065	0.0517
TREC Best	0.6165	0.9914	0.8103

Table 2 shows the overall performances of our three runs as well as the median, worst and best performance of 30 Phase 2 runs. We can see from the table that runB outperforms the TREC median score in terms of all evaluating measures, while the other two runs float up or down around the TREC median score. This confirms that our model performs well towards the task. We also learn two tips from the result of our model:

(1) A max function is more suitable for task than a mean function while calculating the users' ratings for attractions.

(2) The rating derived from Foursquare should be put more weights than that from Yelp.

Table 3.Comparison among all the runs

	ndcg@5	Reciprocal Rank	P@5
bupt_runA	0.2395	0.5366	0.3475
runA(cs.4_.2_max)	0.2758	0.5932	0.4238
runB(cs.2_.4_max)	0.2934	0.6250	0.4479
runC(cs.3_.3_avg)	0.2469	0.6009	0.3790

It is apparently indicated that runs in Phase 2 performed better than the run in Phase 1 in the table 3.

Generally, all runs utilized our rating model to generate a ranked list, but there are differences between the Phase 1 experiment and the Phase 2 experiment. The differences are as follows:

(1) We filled the missing ratings for Phase 2 while ignoring them in Phase 1 experiment.

(2) We only used mean function to calculate the users' ratings for candidates in Phase 1 experiment.

(3) The candidates for Phase 2 experiment are more precise than those of Phase 1.

Through the comparison, we find out that our approach to filling the missing ratings by Collaborative Filtering is effective for the task. It also gives evidence to the conclusion we made from table 2, that a max function outperforms a mean function for our model.

4. CONCLUSION

In the TREC 2016's Contextual Suggestion Track, we built a corpus with plentiful attractions and their detailed information. Meanwhile we designed a model to generate the users' ratings for attractions mostly making use of tags. By Collaborative Filtering, we supplement the ratings for certain attractions in the profile. With patient adjustment of the function and parameters, we figured out an available ranking method for the suggesting task this year. Furthermore, if a lot of submitted runs and evaluations are available, a machine learning method is considerable to improve the adjustment of our model's parameters.

REFERENCES

1. P. Yang and H. Fang. An opinion-aware approach to contextual suggestion. In *Proceedings of TREC'13*, 2013.
2. A. Dean-Hall, C. Clarke, J. Kamps, P. Thomas, N. Simone, and E. Voorhees. Overview of the trec 2015 contextual suggestion track. In *Proceedings of TREC'15*,2015.