Exploration of Opinion-aware Approach to Contextual Suggestion

Peilin Yang and Hui Fang

Department of Electrical and Computer Engineering
University of Delaware, USA
{franklyn,hfang}@udel.edu

Abstract. In this paper we describe our effort on TREC Contextual Suggestion Track. Using resources such as description or websites of example suggestions to build user profile has been proven to be effective on last year's TREC. This year we also leverage the power of using user profile. Real world opinions of the suggestions are used in our method to build both user profile and candidate suggestion profile. Two ranking method are investigated to rank the candidate suggestions: linear interpolation and learning to rank. For description generation, we apply the similar method as used in the last year. The structured description combines the category information of the suggestion, meta-description of the website, reviews of the suggestion and the similar example suggestions that the user liked. Official results of our submitted runs show the effectiveness of the proposed method.

1 Introduction

TREC 1014 Contextual Suggestion Track gives researchers the chance to test their methods on providing better personalized suggestions as well as concise and informative description of the suggestions to travellers who are travelling a new city and planning to have some fun in the city. This year is the third year of Contextual Suggestion Track in a row. There is basically one notable change comparing to Contextual Suggestion Track 2013 - the number of example suggestions in Contextual Suggestion Track 2014 is 100 (which was 50 in Contextual Suggestion Track 2013) and the example suggestion are from two cities (was from single city in Contextual Suggestion Track 2012 and Contextual Suggestion Track 2013). There are several groups (including us) that utilized different sources to model the user profile [2] in Contextual Suggestion Track 2013. Based on our last year's experience [7] and the increasing number of example suggestions, we still rely on building user profiles to help ranking candidate suggestions. When building user profile, we utilize the opinions from online resource to enrich the profiles as what we did in the last year. This year two different methods are tested to rank the candidate suggestions: linear interpolation and learning to rank. Linear interpolation is the method we used in last year but learning to rank is complete new. We would like to try learning to rank as we have bunch of features available. For description generation, we stick to our structured description generation method of last year because of its effectiveness [7]. Basically, we emphasize on learning to rank method to rank the candidate suggestions and other parts are pretty much similar to last year.

2 Our Method

2.1 System Framework

Our method consists of the following parts:

- Useful information gathering,
- Profile modeling,
- Candidate suggestion ranking,
- Description generation.

We will describe them in details in the following sub-sections.

2.2 Useful Information Gathering

As similar to our submitted runs in Contextual Suggestion Track 2013 [7], this year we also crawled the candidate suggestions from open web Yelp [1]. 16 high-level categories like restaurant, shopping are selected since they cover

[1] http://www.yelp.com

the most appropriate categories used in Contextual Suggestion Track. Approximately 1000 candidate suggestions are crawled for each category. Information including the name, average rating, address, business hour, all ratings and the associated text reviews of the candidate suggestion are crawled. Approximately 60,442 candidate suggestions are crawled for all contexts, resulting in average 1208 candidate suggestions per context. The websites of the candidate suggestions are also crawled (limit to 100 pages). The above mentioned information of example suggestions is also manually identified and then automatically crawled. The crawled data is stored mainly in JSON format and is used in both candidate suggestion ranking and description generation.

2.3 Profile Modeling

We use opinion to model user profile as well as candidate suggestion profile due to its richness. Specifically, we use positive opinions of the example suggestions that the user likes (positive example suggestions) to build her positive user profile, and use negative opinions of example suggestions that the user dislikes (negative example suggestions) to build negative user profile. The intuition is that users with similar ratings of a suggestions share something in common of why they like or dislike the suggestion.

Formally, the user profiles are estimated as [7]:

$$\mathcal{U}_{pos} = \bigcup_{es_i \in ES(U) \bigcap R_U(es_i)=POS} REP(O_{pos}(es_i)) \quad (1)$$

$$\mathcal{U}_{neg} = \bigcup_{es_i \in ES(U) \bigcap R_U(es_i)=NEG} REP(O_{neg}(es_i)) \quad (2)$$

where es_i is example suggestion i. $O_{pos}(es_i)$ represents all positive text reviews about es_i, $O_{neg}(es_i)$ represents all negative text reviews, and $REP(O(es_i))$ denotes how to represent text reviews $O(es_i)$ in the profile. $R_U(es_i)$ is the rating of example suggestion es_i given by user U. The original value of $R_U(es_i)$ could be numerical, and we map these values into either POS or NEG. We build the positive and negative profile for a candidate suggestion similar to what we did in building user profile as follows:

$$\mathcal{CS}_{pos} = REP(O_{pos}(CS))$$
$$\mathcal{CS}_{neg} = REP(O_{neg}(CS)).$$

For user ratings in Contextual Suggestion Track collection, we map the example suggestions with rating $\{3, 4\}$ from the user as positive and opinions with rating $\{0, 1\}$ from the user as negative. Example suggestions with rating $\{2\}$ are simply ignored since they are hard to categorize into either positive or negative. For the opinions crawled from Yelp (either for example or candidate), opinions with rating $\{4, 5\}$ are viewed as positive and opinions with rating $\{1, 2\}$ are treated as negative. Opinions with rating $\{3\}$ are simply ignored and are not used for building the profiles since they often contain mixed polarities and thus are hard to categorize.

We tried several strategies of how to represent the profiles.

- *Use full reviews (FR)*: use full text in the review.
- *Use unique terms from full reviews (UFR)*: only the unique terms in the review.
- *Use selective reviews (SR)*: most frequent terms in the review.
- *Use unique terms from selective reviews (USR)*: only consider the unique terms in SR.
- *Use nouns (NR)*: nouns in the review.
- *Use review summaries (RS)*: summary of the review. RS are the terms extracted from the reviews using Opinosis Algorithm[5].

Simple pre-processing are performed on the original text reviews: 1) Terms are lower cased; 2) Stop words are removed.

2.4 Candidate Suggestion Ranking

Given a user U and a candidate suggestion CS, the similarity score can be computed as follows:

1. Build a positive and negative user profile, i.e., $\mathcal{U}_{pos}$ and $\mathcal{U}_{neg}$, based on the information about example suggestions that the user has rated, i.e, $ES(U)$;
2. Build the positive and negative profile for the candidate suggestion CS, i.e., $\mathcal{CS}_{pos}$ and $\mathcal{CS}_{neg}$;
3. Predict the score of CS, i.e., $S(U, CS)$, based on the similarities between $\mathcal{U}_{pos}$, $\mathcal{U}_{neg}$, $\mathcal{CS}_{pos}$ and $\mathcal{CS}_{neg}$.

In order to compute $S(U, CS)$, we investigate two categories of methods: *linear interpolation* and *learning-to-rank*.

Linear Interpolation The main idea of linear interpolation is to linearly combine the similarity scores between user profile $\mathcal{U}_{pos}$, $\mathcal{U}_{neg}$ and the candidate profile $\mathcal{CS}_{pos}$ and $\mathcal{CS}_{neg}$. We use the following function to estimate the similarity between a user and a candidate suggestion:

$$\begin{aligned} S(U, CS) = \alpha \times SIM(\mathcal{U}_{pos}, \mathcal{CS}_{pos}) - \beta \times SIM(\mathcal{U}_{pos}, \mathcal{CS}_{neg}) \\ - \gamma \times SIM(\mathcal{U}_{neg}, \mathcal{CS}_{pos}) + \eta \times SIM(\mathcal{U}_{neg}, \mathcal{CS}_{neg}) \end{aligned} \tag{3}$$

where $SIM(a, b)$ is text similarity measurement between any two profiles. α, β, γ, $\eta \in [0, 1]$ are parameters that balance the impact of the different similarities to the final similarity score. We use an axiomatic similarity function F2EXP [3] when computing $SIM(a, b)$. A preliminary training process was done on example suggestion using 5-fold cross validation. Different text review representations were tested. Finally, RS (review summary) was chosen to be used in ranking candidate suggestions because it achieves the best performance on training data. The optimal parameter set is $\alpha = 0.7, \beta = 0.0, \gamma = 0.4, \eta = 0.0$.

Learning to Rank The other ranking method we use is learning to rank method. The process is as follows: we first compute the similarity scores $SIM(\mathcal{U}_{pos}, \mathcal{CS}_{pos})$, $SIM(\mathcal{U}_{neg}, \mathcal{CS}_{neg})$, $SIM(\mathcal{U}_{pos}, \mathcal{CS}_{neg})$, $SIM(\mathcal{U}_{neg}, \mathcal{CS}_{pos})$ which is exactly the same as what we do in the linear interpolation method. Here we compute the similarities scores for all text review representations mentioned in 2.3. We then use these scores together with category similarity score and description similarity score which we used in Contextual Suggestion Track 2012 [6] as possible features and utilize the power of learning to rank to compute the score for each candidate suggestion.

Several learning to rank approaches with different feature combinations were tried in the training process. The approaches we tried including Multiple Additive Regression Trees (MART) [4], LambdaMART [1], AdaRank, RankNet, ListNet, Support Vector Machine (SVM) and etc. We also tried different feature combinations: using single opinion representation, e.g. using all four similarities scores generated by applying full review when computing the similarities scores; using combined similarities scores generated by different opinion representations, e.g. combine all similarities scores generated by FR (Full Review), SR (Short Review), NR (Noun Review); using combined similarities scores of opinion similarities scores and category and/or description similarities scores. Similar to the linear interpolation method, we apply 5-fold cross validation training on the example suggestions.

The result on example suggestions shows that LambdaMART together with combining all available similarities scores (category, description and all representations of reviews) as features achieves best performance. We then use all example data as the training data to rank candidate suggestions.

2.5 Description Generation

In last year's Contextual Suggestion Track we used a template to generate the descriptions of candidate suggestions [7]. Our description generation method ranked among the top in terms of all measurements, namely P@5, MRR and TBG [2]. This year we apply the same idea when generating the short description for candidate suggestion. We use all four parts: the opening sentence, "official" introduction, highlighted reviews and the concluding sentence of our structured description for both of our runs.

3 Submitted Runs and Experiment Results

We submitted two runs: UDInfoCS2014_1 and UDInfoCS2014_2. UDInfoCS2014_1 uses learning-to-rank method with LambdaMART as ranking method. All kinds of similarity scores with all kinds of text review representations are used as features. Example suggestions and user profiles are used to build the training data. UDInfoCS2014_2 uses linear interpolation method with RS text review representation. For both runs the description combines the opening sentence, meta-description and the picked sentences from the web site, highlighted reviews, and the concluding sentence.

Table 1 shows the overall mean performances of our runs in terms of all evaluation measures. We can see from the table that both of our runs outperform the TREC median score in terms of all measures. This confirms the effectiveness of using opinions to rank candidate suggestions and to generate the short descriptions. UDInfoCS2014_2, which uses linear interpolation as the candidate suggestion ranking method, achieves better performance. UDInfoCS2014_1, which uses learning to rank as the ranking method, also achieves promising performance. However, the learning to rank may be overfitted as the performance is not as good as its performance

Table 1. Overall Mean Performances

Runs	P@5	MRR	TBG
UDInfoCS2014_1	0.4074	0.5482	1.6271
UDInfoCS2014_2	**0.5572**	**0.7439**	**2.7043**
TREC Median	0.3492	0.5350	1.3685

Table 2. More analysis of submitted runs

runid	accuary	precision	recall
UDInfoCS2014_1	0.8375(1252/1495)	0.844(660/782)	0.8451(660/781)
UDInfoCS2014_2	0.8187(1224/1495)	0.9232(890/964)	0.8188(890/1087)

on training data. Moreover, using all kinds of similarity scores using different text review representations may contain noise data.

We also analyze other metrics mainly defined by ourselves:

1. Accuracy: number of suggestions of which the judgements of description and website are consistent divided by the total number of judged suggestions;
2. Precision: number of suggestions of which both description and website are relevant divided by the number of suggestions of which description are relevant;
3. Recall: number of suggestions of which the judgements of description and website are both relevant divided by the number of suggestions of which website are relevant;

Accuracy tests the overall performance. It also reflects the quality of the generated description. With high quality description, user either does not need to go to the website of the suggestion or get a better understanding of the suggestion. Precision is defined in this way since user may not go to the website of the suggestion if the description is not interesting. Recall mainly tests the quality of ranking candidate suggestion method. Table 2 shows the corresponding results of our submitted runs. We can see that UDInfoCS2014_2 has higher precision which could be the reason of its better performance over UDInfoCS2014_1.

4 Conclusion

In Contextual Suggestion Track 2014, we apply profile modeling in ranking the candidate suggestions and apply a structured description generation method. The similarity between user profile and candidate profile are used to feed two methods - linear interpolation and learning to rank. These two methods are used to rank the candidate suggestions. We apply the same idea when generating the description as what we did in Contextual Suggestion Track 2013. We submit two runs to Contextual Suggestion Track 2014. The performance of our two submitted runs are in general better than the TREC median performance which indicating the effectiveness of our proposed method.

References

1. C. J. C. Burges. From RankNet to LambdaRank to LambdaMART: An overview. Technical report, Microsoft Research, 2010.
2. A. Dean-Hall, C. Clarke, J. Kamps, P. Thomas, N. Simone, and E. Voorhees. Overview of the trec 2013 contextual suggestion track. In *Proceedings of TREC'13*, 2013.
3. H. Fang and C. Zhai. An exploration of axiomatic approaches to information retrieval. In *Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '05, pages 480–487, New York, NY, USA, 2005. ACM.
4. J. H. Friedman. Greedy function approximation: A gradient boosting machine. *Annals of Statistics*, 29:1189–1232, 2000.
5. K. Ganesan, C. Zhai, and J. Han. Opinosis: a graph-based approach to abstractive summarization of highly redundant opinions. In *Proceedings of the 23rd International Conference on Computational Linguistics*, COLING '10, pages 340–348, Stroudsburg, PA, USA, 2010. Association for Computational Linguistics.
6. P. Yang and H. Fang. An exploration of ranking-based strategy for contextual suggestion. In *Proceedings of 21st Text REtrieval Conference (TREC 2012)*. NIST, November 2012.
7. P. Yang and H. Fang. An opinion-aware approach to contextual suggestion. In *Proceedings of TREC'13*, 2013.