An Exploration of Ranking-based Strategy for Contextual Suggestion

Peilin Yang and Hui Fang

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

Abstract. We describe our efforts in TREC 2012 Contextual Suggestion Track. The task is to provide suggestions to users based on their personal interests as well as their contexts. To tackle the problem, we propose to rank candidate suggestions based on their similarity to the personal profile and that to the contexts (i.e., geographic and temporal information). The ranking function is computed based on the similarity between a suggestion and the places that the user like and the dis-similarity between the suggestion and the places disliked by the user. The similarities are computed based on the either the category or description of the suggestions.

1 Introduction

TREC 2012 Contextual Suggestion Track provides a venue for evaluating the "zero query term problem" [1], which aims to proactively provide user suggestions based on their contexts. Specifically, given a user's profile (i.e, the places that he/she likes and dislikes) and the user's context (i.e., location and time), the task is to return the user with suggestions that are relevant to either the profile or context.

The data set consists of two parts: (1) profiles of 34 users, i.e., their preferences on 49 example suggestions; (2) 50 contexts (i.e., geographical and temporal information). The problem set up is to retrieve 50 suggestions based on each pair of profile and context.

To solve the problem, we need to address three challenges:

- *How to find candidate suggestions?*
- *How to rank the suggestions based on the user profile?*
- *How to rank the suggestions based on the contexts?*

To address the first challenge, we retrieve candidate suggestions from multiple online sources based on the geographical information. For the second challenge, we propose to model the relevance of a suggestion based on its similarities with positive examples in the profile (i.e.,the places liked by the user) and the dis-similarities with the negative examples (i.e., the places disliked by the user). The similarity between a suggestion and a place can be measured based on their categories as well as their descriptions. Finally, to address the last challenge, we propose to post-process the results by filtering out the suggestions that do not satisfy the temporal requirements.

2 Our Method

2.1 Gathering Candidate Suggestions

To collect candidate suggestions, we crawl the information from multiple online sources such as Yelp and Foursquqre based on the geographical information from the 50 contexts. The collected candidate suggestions have different fields such as categories, web sites, business hours, etc. Moreover, we also crawl the web site for each candidate suggestion.

2.2 Ranking based on User Profiles

We now describe how to rank candidate suggestions based on user profiles. The profile of each user consists of the user's preferences for 49 example locations. The locations that a user likes are referred to as "positive examples", and those disliked by the user are referred to as "negative examples". Intuitively, the relevance score of a candidate suggestion should be higher when it is similar to positive examples while different from the negative examples.

Formally, we denote U as a user and C as a candidate suggestion. Moreover, let $P(U)$ denote positive examples, i.e., a set of places that the user likes, and $N(U)$ denote negative examples, i.e., a set of places that the user dislikes. The relevance score of C with respect to U can then be computed as follows:

$$S(U,C) = \varphi \times S_P(P(U),C) + (1-\varphi) \times S_N(N(U),C) \quad (1)$$

$$= \varphi \times \frac{\sum_{p\in P(U)} SIM(p,C)}{|P(U)|} + (1-\varphi) \times \frac{\sum_{p\in N(U)} SIM(n,C)}{|N(U)|}, \quad (2)$$

Table 1. Examples of Categories in Example Suggestions

NAME	Yelp Categories	FourSquare Categories
HoSu Bistro	SushiRestaurant→Restaurants; KoreanRestaurant→Restaurants; JapaneseRestaurant→Restaurants	SushiRestaurant→Food
The Rex	JazzBlues→MusicVenues→Arts	JazzClub → MusicVenue → ArtsEntertainment
St. Lawrence Market	Grocery→Shopping; FarmersMarket→Shopping	FarmersMarket → FoodDrinkShop → ShopService
...	...	...

where $\varphi \in [0,1]$ and it regularize the weights between the positive and negative examples. When $\varphi = 1$, the highly ranked suggestions would be those similar to the locations that the user likes. When $\varphi = 0$, the highly ranked suggestions would be those different from the locations that the user dislikes. $S_P(P(U), C)$ measures the similarity between the positive user profile and the candidate suggestion, and we assume that it can be computed by averaging the similarity scores between each positive example and the candidate suggestion. $|P(U)|$ corresponds to the number of positive examples in the user profile.

Thus, it is clear that the problem of computing the relevance score of a candidate suggestion with respect a user can be boiled down to the problem of computing the relevance score between a candidate suggestion and a place mentioned in the user profile, i.e., $SIM(e, C)$, where e is an example from the user profile. We explore the following two types of information to compute $SIM(e, C)$: (1) the category of a place; and (2) the description of a place.

Category-based similarity: Category is a very important factor that may greatly impact user preferences. The categories of the crawled suggestions are often hierarchical. Here is an example category, i.e., *[History Museum→ Museum→Arts]*. The categories becomes more general from the left to the right. In this example, *Arts* is the most general category while *History Museum* is the most specific category. Note that we represent the hierarchical categories as a set of categories in this paper.

We can compute $SIM(e, C)$ based on the category similarities between e and C as follows:

$$SIM_{\mathcal{C}}(e, C) = \frac{\sum_{c_i \in \mathcal{C}(e)} \sum_{c_j \in \mathcal{C}(C)} \frac{|Intersection(c_i, c_j)|}{max(|c_i|, |c_j|)}}{|\mathcal{C}(e)| \times |\mathcal{C}(C)|}, \tag{3}$$

where $\mathcal{C}(e)$ denotes the set of categories of location e and $|Intersection(c_i, c_j)|$ is the number of common categories between c_i and c_j. Recall that we crawled the candidate suggestions from two online sources. Table 1 shows example categories from both sources. Thus, we combine the similarity scores computed based on the categories from them as follows:

$$SIM_{\mathcal{C}'}(e, C) = \phi \times SIM_C^{Yelp}(e, C) + (1 - \phi) \times SIM_C^{FourSquare}(e, C). \tag{4}$$

In our experiments, we set ϕ as 0.5 which means the importance of category score of Yelp and FourSquare are the same.

Description-based similarity: In example suggestions, each suggestion has its unique description which typically is at a short length. We want to learn how the descriptions can affect people's decision on different places. By comparing the descriptions of training suggestions with textual web sites of testing suggestions we may find some interesting connections. We use textual web sites of testing suggestions because we believe that textual web sites are more reliable than descriptions especially when we rank candidate suggestions. The similarity used function is the F2EXP ranking function [2]. Thus, we compute the similarity scores as follows: $SIM_{\mathcal{D}}(e, C) = F2EXP(DES(e), DES(C))$, where $DES(e)$ is a description of the example place e.

2.3 Ranking the candidates based on the contexts

There are two types of context: geographical and temporal information. All of the candidate suggestions crawled in the first step are related to the geographical requirement because they are retrieved based on it. To make sure the suggestions satisfying the temporal requirement, we collected the business hours from Yelp, and then assign the business hours for each cateogry if the majority suggestions in that category follow the same business hour. Candidate suggestions that do not meet the temporal requirement are then removed from the final ranking list.

3 Submitted Runs and Experiment Results

We submitted two runs: UDInfoCSTc and UDInfoCSTdc.

- UDInfoCSTc: We use only category information to compute the similarities as shown in Equation (3).
- UDInfoCSTdc: We use both category and description information to compute the similarities as follows:

$$SIM_{combine}(e, C) = \alpha \times SIM_{\mathcal{C}}(e, C) + (1 - \alpha) \times SIM_{\mathcal{D}}(e, C) \tag{5}$$

where $\alpha \in [0, 1]$ and it controls the weight for the two similarity methods.

In both runs, we post-process the results to meet the geographic and temporal requirements. All the parameters are trained based on the example suggestions. We tried all combinations of parameters mentioned in previous subsections. For each of them, we pick up their values from [0,1] with pace length 0.1. We get the parameters set that leads the best performance of each fold and average them.

For each suggestion, the assessor judges W (Website), D (Description), G (Geographical), T (Temporal), GT (combine G and T) and WGT (combine WGT). For combined measures, only all of them are appropriate then the score will be "1"; otherwise the score will be "0". Here we only show the P@5 WGT performance:

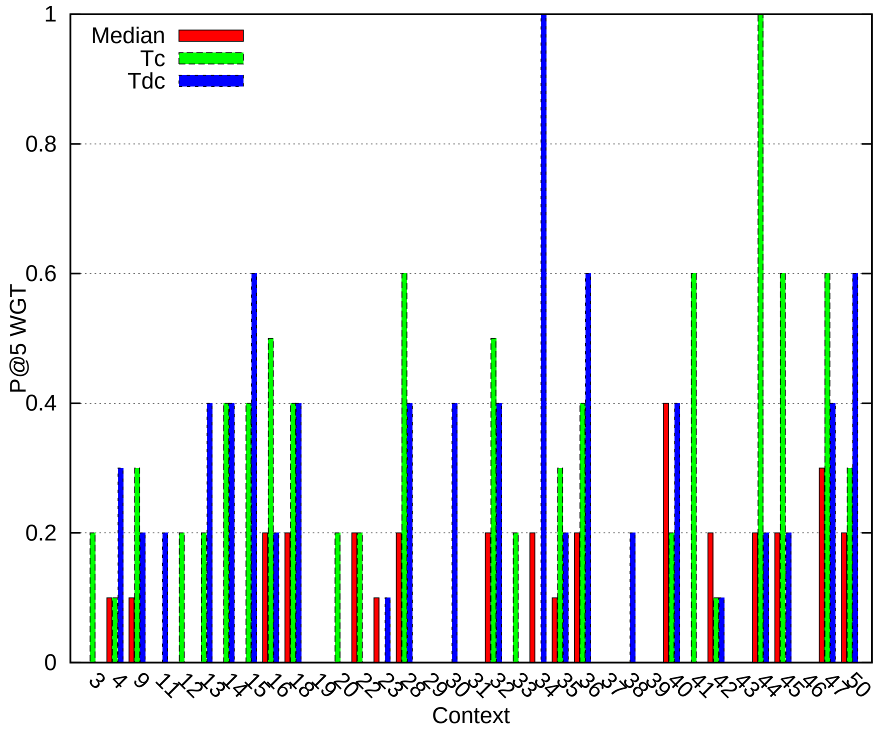

Fig. 1. P@5 WGT Performance of UDInfoCSTc and UDInfoCSTdc (compare with median) on each context by averaging all profiles for that context.

Figure 1 and 2 show the performance of our runs compared with median performance of the track based on context and profile, respectively. It is clear that our runs are better than the median performance in most cases.

Table 2 shows the overall performance of two runs, i.e., the performance is for the corresponding measures over all judged profiles and contexts. It is clear that using only category information is more effective for WGT (i.e., website, geographical and temporal) while using description can help for using GT only.

4 Conclusion

In TREC 2012 Contextual Suggestion Track, we submitted two runs. Our goal is to evaluate (1) an information gathering strategy that crawls and integrates the information about candidate suggestions from multiple sources,

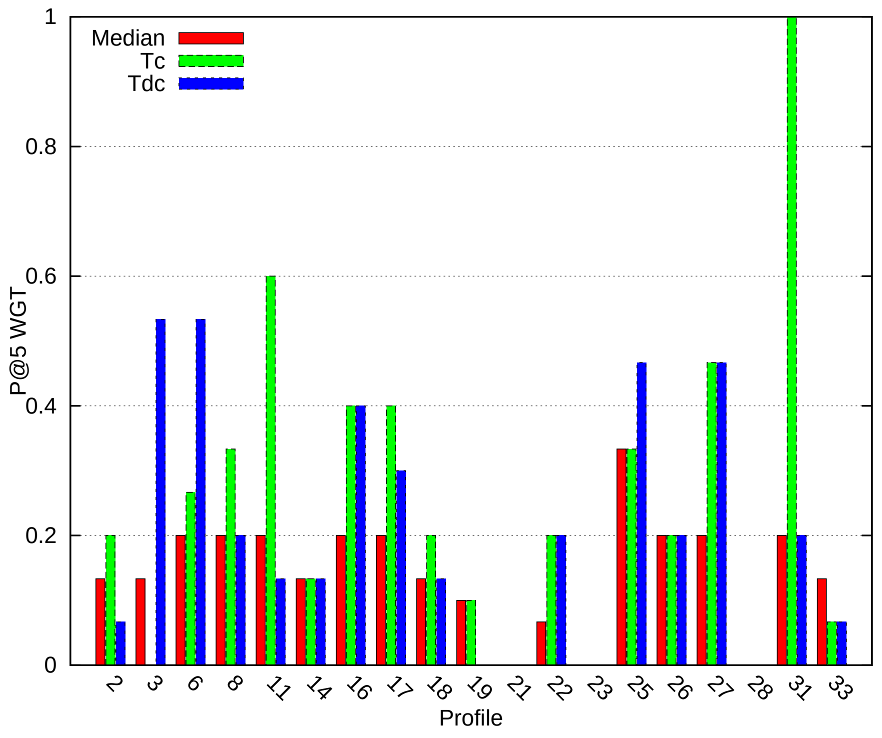

Fig. 2. P@5 WGT Performance of UDInfoCSTc and UDInfoCSTdc (compare with median) on each profile by averaging all contexts for that profile.

Table 2. Overall Performance of Two Runs

	P@5 WGT	**P@5 W**	**P@5 GT**
UDInfoCSTc	**0.2481**	0.3500	0.4950
UDInfoCSTdc	0.2210	0.3500	**0.5442**

such as Yelp, Foursquare etc.; and (2) a ranking-based method that computes the similarity between a candidate suggestion and a user profile based on both descriptions and categories of the candidate and example suggestions. In particular, the two runs are UDInfoCSTc and UDInfoCSTdc. In UDInfoCSTc, we use categories only to compute the similarities between each ⟨candidate suggestion/example⟩ suggestion pair. For each user, positive and negative similarities are combined using optimal coefficients learned from examples. Final ranking is based on combined similarities scores. In UDInfoCSTdc, we use the similar strategy with UDInfoCSTc but when computing similarities we use categories together with descriptions. For both runs, we post-process the results to meet the geographic and temporal requirements. The performance of our two submitted runs are in general better than the median performance. Moreover, using category only as the similarity function we get a overall better performance than using combined category and description scores.

References

1. J. Allan, B. Croft, A. Moffat, and M. Sanderson. Frontiers, challenges, and opportunities for information retrieval: Report from swirl 2012 the second strategic workshop on information retrieval in lorne. *SIGIR Forum*, 46(1):2–32, May 2012.
2. 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.