# A Personalised Recommendation System for Context-Aware Suggestions

Andrei Rikitianskii, Morgan Harvey, and Fabio Crestani

Faculty of Informatics, University of Lugano (USI), Switzerland
{andrei.rikitianskii, morgan.harvey, fabio.crestani}@usi.ch

**Abstract.** The recently introduced TREC Contextual Suggestion track proposes the problem of suggesting contextually relevant places to a user visiting a new city based on his/her preferences and the location of the new city. In this paper we introduce a more sophisticated approach to this problem which very carefully constructs user profiles in order to provide more accurate and relevant recommendations. Based on the track evaluations we demonstrate that our system not only significantly outperforms a baseline method but also performs very well in comparison to other runs submitted to the track, managing to achieve the best results in nearly half of all test contexts.

## 1 Introduction

Modern web services such as search engines and online mapping tools provide a means for people to find interesting things to do and places to go when visiting a new city. However in many cases, due to the sheer number of possibilities and perhaps a lack of time for planning, it is not possible or desirable to manually search through all of the available options. In such instances, recommender systems can provide users with a personalised set of suggestions of places to visit that they are likely to enjoy, recommendations which are often made based on the user's previous interactions with the system. Recommender systems are becoming an ever more frequently investigated area of research and are used by many thousands of web sites and services - including the likes of Amazon, Netflix and Trip Advisor - to improve the experience for their users and are especially useful on mobile devices [9].

The vast majority of work in the area has considered only 2 dimensions of interest - the users and the items to be recommended - however there has been a recent surge of interest in the inclusion of a third dimension: context [1]. Rather than simply recommending items that the user will like based on a profile, context-sensitive recommender systems attempt to also include various other sources of information to make the resulting recommendations more accurate and relevant. Examples of useful contextual information are, for example, the time of day or current location of the user. Use of context is clearly of significant benefit when suggesting places to visit since factors such as location will dictate which places can feasibly be visited. The increased awareness of the importance of context when making recommendations is demonstrated by the introduction of a new TREC track dedicated to this problem in 2012 which has continued into 2013 [4].

In this work we present a new approach to recommending places to users incorporating geographical information as context and exploiting data from multiple sources. Via analysis of results from TREC evaluations performed by a large group of users we demonstrate the high level of performance delivered by this method, showing that it is able to significantly outperform the Contextual Track baseline and all other track entrants in nearly half of all cases. Over all metrics our system performs considerably better than the median result. We conclude the paper with a more detailed analysis of the results, indicating potential avenues for future work.

## 2 Related Work

Recommendation and personalisation are 2 very common themes in modern Information Retrieval research. Much early work was conducted in the 90s and the field has seen a resurgence of interest lately, partially due to the Netflix prize [7] and also due to the realisation that context can play an important part in generating truly accurate and useful results [1, 12]. In this work we focus solely on the problem of recommendation in which a potentially massive set of candidate items are filtered down such that the subset of remaining items will be of interest to the user based on their profile information. Ideally these final suggestions should be ranked such that the first item has the greatest probability of being liked by the user with probability of interest decreasing as rank increases [8].

Personalisation has been frequently applied to mobile search and recommendation problems such as tourist guides [2] and event navigation tools [9]. In many of these approaches various forms of context are considered. Ardissono et al. [2] present a system designed to help tourists navigate their way around the Italian city of Turin and make recommendations to users by assigning each to one of several tourist "groups" with certain requirements. Recommendations are made by taking various contextual factors into account such as opening hours, entrance prices and determine user preferences based on features such as the historical period of the attractions.

Schaller at al. [9] attempt to provide recommendations to visitors of a large-scale event composed of many smaller sub-events. Their system uses a variety of forms of profile information and constructs tours around sets of recommended events in an effort to maximise the amount of time visitors spend at attractions, rather than travelling between them. They demonstrate the importance of considering contextual information when constructing tours and recommendations such as visitor and event locations, start and end times and durations of events.

Many of the approaches submitted to last year's track approached the problem in a similar manner [4]. They first obtain a list of suitable venues by querying a search API using the geo context data, then construct profiles using the terms in the descriptions of places already given positive ratings by users and finally rank the list of potential candidates based on their descriptions' similarities to the positive user profile. In this work we expand on this simple framework in a number of ways. We consider both positive and negative ratings separately, are far more selective when choosing terms to build profiles, make use of term expansions techniques to mitigate matching issues caused by varied vocabulary use and use machine learning methods to build a classifier and ranker.

## 3 Dataset and Tasks

The TREC Contextual Suggestion Track investigates search techniques for complex information needs that are highly dependent on context and user interests. In this track the goal is to suggest personalised attractions to an individual, given a specific geographic context. The track imagines a traveler in a new city. Given a set of the traveler's preferences for places and activities in their home city, the system should suggest places and activities in a new city that the person may enjoy. In this paper we use the terms "attraction," "place" and "venue" interchangeably.

As input to the task, participants were provided with a set of 635 profiles, a set of 50 example suggestions, and a set of 50 geo contexts in CSV/JSON format. Example suggestions can represent attractions of different types, for example: bars, restaurants, museums, etc. All the attractions are from the Philadelphia area. Each profile corresponds to a single user, and indicates that user's preference with respect to each example suggestion. Each training suggestion includes a title, description, and an associated URL. Each context corresponds to the GPS coordinate of the centre of a number of cities in the United States. The set of cities is quite diverse in terms of population: starting from small cities such as Beckley, WV (with a population of 17,606) up to much larger cities such as Atlanta, GA and Wichita, KS (with populations in the hundreds of thousands or even millions). Profiles consist of two ratings for a series of attractions, one rating for the attraction's title and description and another for the attraction's website. The ratings are given on a five-point scale, ranging from "strongly disinterested" to "strongly interested", based on how interested the user would be in going to the venue if they were visiting the particular city it is located in.

As output to the task, for each profile/context pairing, the participant should return a ranked list of up to 50 ranked suggestions. Each suggestion should be appropriate to the profile (based on the user's preferences) and the context (according to the location), contains a title, description and attraction's URL. The description of the suggestion may be tailored to reflect the preferences of that user. Profiles correspond to the stated preferences of real individuals, who will return to judge proposed suggestions. Users were recruited through crowdsourcing sites or are university undergraduate and graduate students. For the purposes of this experiment, you can assume users are of legal drinking age at the location specified by the context.

## 4 A New Approach for Context Suggestion

To generate ranked lists of appropriate recommendations for each user profile and geographical context we developed a geo context-aware system. The system can be broken down into the following 4 steps:

1. processing geo contexts;
2. inferring user term preferences;
3. building a personal ranking model;
4. ranking suggestions

In the following section we describe these individual steps in more detail.

### 4.1 Processing Geographical Contexts

Before we can apply any user profile-based personalisation we first need a set of appropriate candidate attractions located within a small radius of the geo context specified. We used the Google Places API [1] to obtain a list of potential suggestions, retrieved based on a query consisting of GPS coordinates and attraction types. We considered only types of venues, as defined by the Google Places API, which were present within the training set. In doing so we retrieved 27 different types, such as: night clubs, amusement parks, libraries, movie theatres, shopping malls, etc. On average, for each geo context, we collected about 350 suggestions.

Google Places only provides a short *title* and a web site *URL* for each suggestion. So that users can evaluate the quality of each suggestion a description of each venue should be provided. To generate these brief descriptions we first queried the Yandex Rich Content API [2] which, given a URL as a query, returns a short static textual description of the page's content. While the Yandex API has generally quite good coverage, there were instances where it was unable to return any information and in these cases we instead queried the Google Custom Search API and used the web site snippet it returned.

### 4.2 Inferring User Term Preferences

In order to make personalised suggestions for each user we need to be able to compare a new venue with that user's profile to determine how likely it is that the user will like it. We therefore need to have some representation of the user's likes and dislikes based on the training data made available to us, i.e. the venues each user has already rated. In line with previous work on recommender systems [6], we chose to maintain separation between positive and negative preferences using descriptive terms from already rated venues. We used Natural Language Toolkit (NLTK) software [3] to extract only nouns, adjectives, adverbs and verbs from each

[1] Google Places API - https://developers.google.com/places/documentation/
[2] Yandex Rich Content API - http://api.yandex.com/rca/
[3] Toolkit version 1.0 used, available from http://nltk.org/

description and represented these as binary vectors, indicating the presence or absence of each term. For each user, we extracted positive and negative terms, using them to build separate positive and negative profiles. A positive term is one derived from a positively rated venue, a negative term is one from a venue that was given a negative rating. Venues with a *title and description* rating of more than 2 was considered to be positive, while a venue was considered to be negatively rated when it was allocated a rating of less than 2. Terms from neutral suggestions were ignored.

Due to the relative brevity of the descriptions, this approach of using only the terms present is unlikely to result in many exact term matches and will therefore deliver quite unreliable similarity scores. Consider, for example, if the negative profile for a user contains the word "sushi" and this is matched against a description containing the terms "raw" and "fish." Without performing any kind of term expansion these concepts, despite their obvious similarity, would not make any contribution to the similarity score. However, by expanding existing raw terms using similar words we can (at least partially) overcome this vocabulary mismatch. When comparing profiles with venue descriptions, instead of simply using the raw terms, we checked for matches between the synonym lists returned for each term in WordNet [4]. Given a list of synonyms for a term $a$ and those for a term $b$, we consider the terms to be matching if the two lists share at least one component (i.e. if there is some overlap).

Using both the positive and negative models, we can estimate what the user's opinion might be about a potential suggestion based on its description. To estimate how positive the description is for user $u$, we can calculate the cosine distance between vector $\overrightarrow{D_i}$ representing the description of venue $i$ and positive user profile $\overrightarrow{M_u^+}$ as follows:

$$cos^+(\overrightarrow{D_i}, \overrightarrow{M_u^+}) = \frac{\overrightarrow{D_i} \cdot \overrightarrow{M_u^+}}{\|\overrightarrow{D_i}\| \cdot \|\overrightarrow{M_u^+}\|}$$

The same formula was applied to estimate how negative the description is, by using the negative user profile. $cos^+$ and $cos^-$ scores were used in the final ranking model as described in section 4.3.

### 4.3 Building a Personal Ranking Model

After obtaining a set of potential suggestions for a given geographical context (described in section 4.1), we need to rank these potential candidates according to the user's preferences. Because of the variability of individual preferences among users it is nearly impossible to build an accurate global ranking model and given that the task is to provide *personalised* suggestions this would not be suitable anyway. To investigate how varied user preferences were, we measured the level of agreement between all judgments from user profiles by using a standard statistical *overlap* metric [10] where the overlap between two sets of items $A$ and $B$ is defined as:

[4] WordNet - http://wordnet.princeton.edu

$$Overlap(A, B) = \frac{|A \cap B|}{min(|A|, |B|)} \tag{1}$$

The mean pairwise overlap between *title and description* ratings is 0.38 and between *website* ratings is 0.39. Both of these overlaps are quite small, suggesting that users have different preferences. Therefore, we decided to build a personal ranking model for each user in order to more precisely adapt suggestion to their own preferences.

**Model and Training data.** We consider the choice of suitable candidates as a binary classification problem. We separate relevant and non-relevant suggestions for each individual user, and then rank those classed as relevant based on confidence score estimated by the classifier. To generate training set data for each profile we used example suggestion weight as a linear combination of *title and description* and *website* ratings:

$$Weight(S) = \lambda R_{desc,s} + (1 - \lambda) R_{url,s}$$

with the condition $\lambda \in [0, 1]$ and $Weight \in [0, 4]$. In the formula , $S$ indicates the example suggestion from a particular profile, $R_{desc,s} \in [0, 4]$ is the suggestion *title and description* rating, $R_{url,s} \in [0, 4]$ is the suggestion *website* rating. We then assigned a positive label to suggestions with a combined weight of more than a threshold $T^+$ and negative label to the suggestion with weight less than threshold $T^-$.

These thresholds $T^+$ and $T^-$ were tuned to try to balance the number of positive and negative samples in the training set. The degree of imbalance is represented by the ratio of sample size of the small class to that of the large class. We considered a training set to be imbalanced if the ratio was less than 1:5, i.e. if the largest class was more than 4 times greater than the smallest. $T^+$ and $T^-$ were turned for each profile by using a simple iterative algorithm. If the algorithm was unable to converge (i.e. sufficiently balance the 2 classes) after 3 iterations we consider this particular profile to be unsuitable for classification and use an alternate approach for making recommendations which we outline later.

By default we set uniform weights to the 2 different ratings for each example ($\lambda = 0.5$), meaning that the importance of *title and description* is the same as *website*. It is possible that the true influence of these two factors may not be equal and this may depend on the kind of venue under consideration. For example, for many cafs the description may provide sufficient information upon which to base a decision, whereas for a restaurant the user may wish to browse the website first, perhaps to look at the menu before making a decision. These type-dependent values for $\lambda$ could perhaps be learnt from the training data, however we leave this for future work.

**Learning Algorithm and Features.** We chose a Naïve Bayes classifier as our learning algorithm. This is a simple probabilistic classifier based on applying

Bayes' theorem and making the assumption that each feature's weight (or in the binary case, presence or absence) is independent of the weights of other features, given the class variable. Although this assumption is unlikely to be entirely true in many cases, it greatly simplifies the model - making it tractable - and does not significantly degrade performance in practice. The motivation for choosing such a simple classifier was that it generally performs better on small data sets than more sophisticated machine learning techniques [3] and does not require any complex parameter tuning or additional learning. In our case, the size of the training data set is never greater than 50 examples; the number of examples for each profile lies in the range 30-49. We use the Weka implementation of the classifier [11] for all of our experiments.

Each suggestion in the training set is represented by a feature vector, consisting of two different types of features: boolean and real-valued. The boolean features were derived based on attraction types, representing the user's preferences with regard to the kind of venue suggested. As described in section 4.1, the Google Places API returns a simple type for each place, indicating what kind of venue it is. Each place can be assigned to multiple types and as such our binary feature vector encodes the types each suggestion has been assigned to: 1 if it is assigned to that type, 0 otherwise.

The 3 real-valued features were based on the cosine distance between the suggestion description and both user profiles (positive and negative), reflecting user term preferences, and the description length. For a given user $u$ and suggestion $i$, we calculated two features $cos^+(\overrightarrow{D_i}, \overrightarrow{M_u^+})$ and $cos^-(\overrightarrow{D_i}, \overrightarrow{M_u^-})$, where $D_i$ is a description of suggestion $S$ and $M_u^+$ and $M_u^-$ are the positive and negative profiles for user $u$. The description length feature is simply the length of the description in characters. We believe that the length of description may be an important factor when the user explores the suggestions for an attraction as a longer description may provide more detailed information, allowing the user to be more sure of their rating.

## 4.4 Ranking Suggestions

To rank the potential suggestions for each user, we use their individual personal ranking model as described in Section 4.3. The personal ranking model was first used to determine the 50 most relevant suggestions for each geographical context, in descending order of confidence score as estimated by the classifier. The confidence score, in the case of a Naïve Bayes classifier, is simply the posterior probability that the suggestion belongs to class "relevant" and therefore encodes, in some sense, how likely it is that the user will like the candidate venue. This approach has been demonstrated to work well for ranking problems [13].

As mentioned in the previous section, there were a few profiles for which it wasn't sensible to build a classifier due to the level of imbalance between the 2 classes in the training data. In this case, potential suggestions were ranked by using only the user term preferences. We ordered the suggestions in descending order of their scores, which were calculated as the difference between $cos^+$ and

$cos^-$. This is a reasonable, if slightly simplified approach, since it will return a positive value if the similarity between the candidate venue and the positive profile is greater than its similarity compared with the negative profile and vice-versa.

## 5 Results

In this section we present an overview of the performance of our system. Output suggestions were judged both by the original group of users who supplied the training data and NIST assessors. The user corresponding to each profile judged suggestions in the same manner as for the training examples, assigning a value of 0-4 for each *title and description* and *url*. NIST assessors judged suggestions in terms of geographical appropriateness. In total, 223 of the potential 31750 profile/context pairs were judged, i.e. not all pairs were used for evaluation. The top 5 suggestions for each profile/context pair were taken into account for evaluation.

To evaluate the performance of the model for the problem of Contextual Suggestion, three different measures were used: Precision at Rank 5 (P@5), Mean Reciprocal Rank (MRR) and Time-Biased Gain (TBG). P@5 and MRR are traditional evaluation metrics to measure the overall effectiveness of an IR system in terms of its ability to return a good ranked list. The TBG metric, on the other hand, was developed specially for the contextual suggestion task [5]. As the basis for evaluation, a suggestion was counted as "relevant" if the user liked both the description and the geographically appropriate document. All other suggestions were counted as "non-relevant". P@5 and MRR are calculated by using these definitions for relevant and non-relevant. The TBG metric is more complex and takes into account the impact of descriptions and disliked suggestions, which are ignored by P@5 and MRR. All the metrics were computed for each profile/context pair, and then averaged across all pairs.

Since we submitted our system as an entrant to the TREC Contextual Suggestion track we can compare our results with those of two baselines other competing systems from other institutions. The baselines use the Google Places API to retrieve context-relevant candidates and rank them based on their overall popularity (a common baseline measure in recommender systems). BaselineA uses all candidates whereas baselineB uses only candidate places which are present in the ClueWeb12 collection. For the description, Google Places provided a description, review, or a blurb from the meta-description tag on the website. Personalisation was not attempted for either baseline runs. BaselineB is used as a baseline for participants who gathered suggestions from ClueWeb12 datasets only. Table 1 shows evaluation results for our proposed model, baselineA system and two best systems. It also shows the median score, which is calculated based on the results from all 34 systems submitted to the TREC track. The column *P@5 Rank* presents a rank of system result among all 34 systems based on P@5 metric.

The results show that our system greatly outperforms baselineA, however the baseline result does appear quite weak in comparison to the submitted runs

| | P@5 Rank | P@5 | MRR | TBG |
|---|---|---|---|---|
| UDInfoCS1 | 1 | 0.5094 | 0.6320 | 2.4474 |
| UDInfoCS2 | 2 | 0.4969 | 0.6300 | 2.4310 |
| **our method** | **3** | **0.4332** | **0.5871** | **1.8374** |
| baselineA | 25 | 0.1372 | 0.2316 | 0.5234 |
| median | | 0.2368 | 0.3415 | 0.8593 |

**Table 1.** Results for our method compared with baselineA run, two best runs and median scores.

(based on the median score). This means that most participants overcame baselineA result. Nevertheless, our run performs significantly better than median score: P@5 +45%, MRR +41%, TBG +53%. More detailed analysis of our results shows that, according to the MRR metric, our system was able to return the best result over all entrants for 48.43% of user/context pairs. When considering P@5, our system returned the best result in 23% of cases and was better than the median score 61% of the time. Moreover, according to P@5 metric, our system placed 3rd among all 34 systems and 2nd among all 19 groups that participated in the track. We found that about 1.5% of all suggestions had a website which could not be loaded during the assessment procedure. Removing these suggestions from the top 5 leads to performance improvements of: P@5: +0.5%, MRR: +1% and TBG: +1.5%.

## 6 Analysis

Besides final/max/min/median results for each profile/context pair, the organisers also provided all judgments (description, website, geographical relevance) for each suggestion from the 223 profile/context pairs which were judged. Using an evaluation script provided for the track and these judgments we performed a more detailed analysis of our results. Table 2 shows how the geographical relevance(G), description(D) and the website(W) ratings contributed to the P@5 and MRR scores. According to this statistic, almost all documents retrieved by our system were geographically appropriate to the context, suggesting that the approach of pre-filtering candidates is effective. The website of each suggestion and its description appear to contribute equally to final result quality. We found that there was a small correlation (0.271) between the number of candidates returned by the Google Places API for a city and the performance of the system, suggesting that it is easier to make good recommendations when there is a wide variety of possible candidates. The 4 context cities with the smallest population also have the worst performance in terms of P@5 and, unsurprisingly, there is a strong correlation between the population of a city and the number of candidates returned for it by the API (0.693).

In general, assessors judged 1115 suggestions from the Top 5 suggestions for 136 different profiles . These suggestions represent 772 unique venues, i.e.

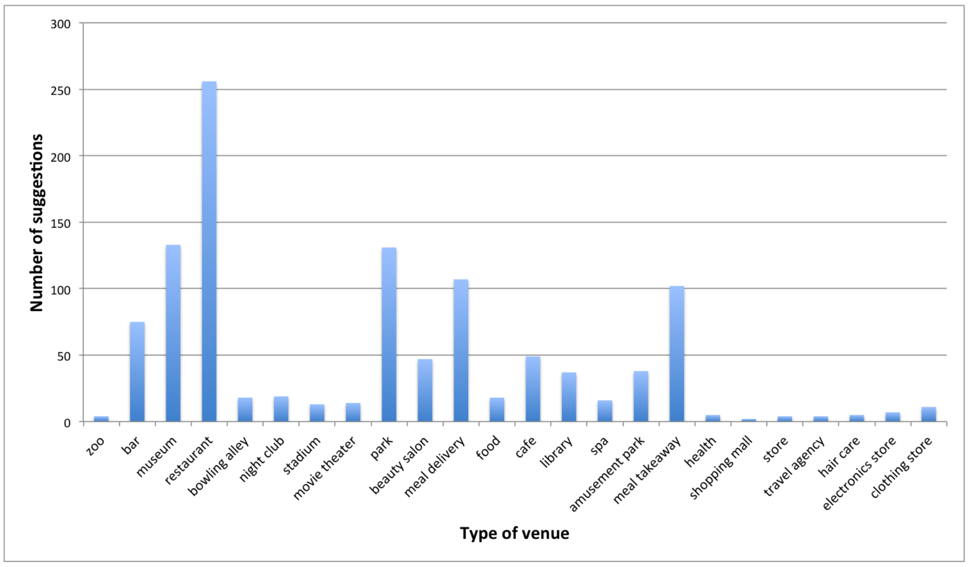

**Fig. 1.** Distribution of numbers of suggestion for 24 different types of venues.

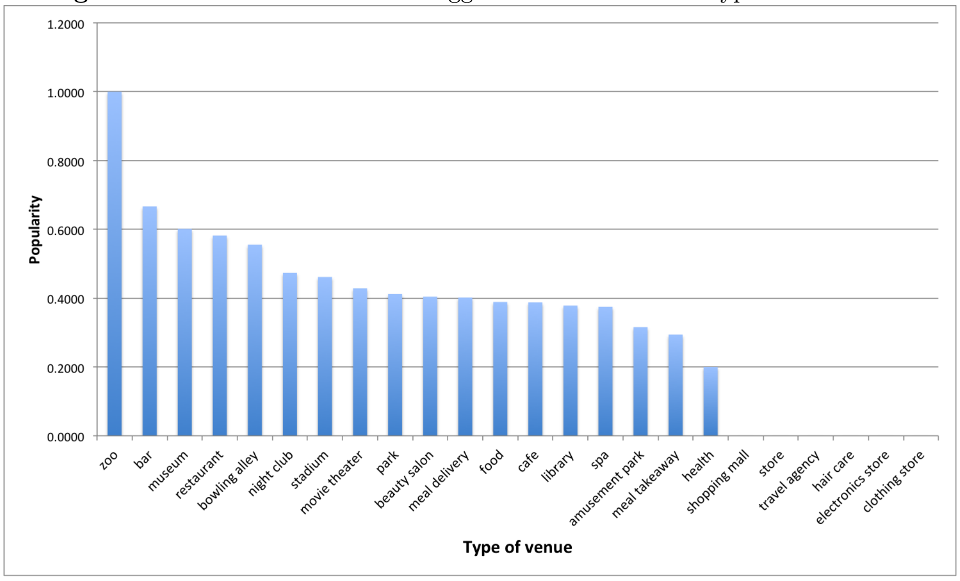

**Fig. 2.** Popularity of 24 different types of venues.

| | **G** | **W** | **D** | **Final(WDG)** |
|---|---|---|---|---|
| P@5 | 0.9363 | 0.5776 | 0.5381 | 0.4332 |
| MRR | 0.9675 | 0.7149 | 0.6700 | 0.5871 |

**Table 2.** Geographical relevance(G), description(D) and website(W) contributions into P@5 and MRR metrics.

some venues were recommended for different profiles at the same time. We explored types of venues which were represented by these suggestions, amounting to a total of 24. Figure 1 presents a distribution over different types of venues. Restaurants (23%), museums (12%) and parks (11.7%) are the most common types of venues. For each venue type we calculated a popularity score, which is the fraction of "relevant" suggestions over all of the suggestions made for that type. Figure 2 demonstrates that the most popular venues for assessor were zoos, bars, museums and restaurants. This can perhaps be explained by the fact that there were few zoos recommended to users, and all of them were counted as "relevant". Restaurants, bars and museums are often suggested and are highly popular because they are very common tourist attractions and their overall popularity is perhaps not strongly affected by a visitor's interests. The popularity of venues such as travel agencies, shopping malls and electronics stores is 0, likely because these types of places are not especially attractive to tourists and are more likely to be frequented regularly by people who live in the area.

## 7 Conclusions and Future Work

In this paper we have described a new system, designed to take part in the TREC Contextual Suggestion track, for making context-sensitive recommendations to tourists visiting a new city. Based on analysis of results obtained from the same users who contributed the training data we have shown that the method is very effective for this problem and was able to significantly outperform the baseline system and, when compared to the 34 other competing system in the track, delivered the 3rd best result in the TREC 2013 Contextual Suggestion track. In nearly half of all contexts, our approach was able to deliver the best set of results, confirming that the choices made during the development of the system were sensible and beneficial. Our method is based on quite a simple strategy of using the descriptions of previously rated places to build user profiles, however we introduce a number of novel additions which have clearly lead to improved performance.

There are several directions for future work. Our ranking model could be easily extended by adding new features to the classifier. For example in the current ranking model we don't use information about the distance between the geolocation specified for each context and the venue, the venue rating (provided by content system) or the cuisine type of restaurants, cafs and bars. We believe that new features based on this information will allow ranking model to reflect user preferences more precisely and that subtly weighting suggestions by their distance from the user's location will lead to better acceptance of the recommendations made. It would also be interesting to make use of other content systems (such as Foursquare and TripAdvisor) to expand the list of potential candidates and the brief descriptions could perhaps be improved or tailed to the user's interests by also considering user reviews or comments from social networks. Finally, instead of assuming that the description and web site of a venue are of equal importance, we could learn specifics weights for the $\lambda$ parameter in our model which could even be conditioned on the type of venue being considered.

## References

1. Gediminas Adomavicius and Alexander Tuzhilin. Context-aware recommender systems. In *RecSys*, pages 335–336, 2008.
2. Liliana Ardissono, Anna Goy, Giovanna Petrone, Marino Segnan, and Pietro Torasso. Intrigue: Personalized recommendation of tourist attractions for desktop and handset devices. In *Applied Artificial Intelligence*, pages 687–714. Taylor and Francis, 2003.
3. D. Brain and G.I. Webb. On the effect of data set size on bias and variance in classification learning. In *4th Australian Knowledge Acquisition Workshop (AKAW '99)*, Sydney, Australia, 1999.
4. Adriel Dean-Hall, Charles L. A. Clarke, Jaap Kamps, Paul Thomas, and Ellen Voorhees. Overview of the trec 2012 contextual suggestion track. In *Text REtrieval Conference (TREC)*, 2012.
5. Kamps J. Dean-Hall A., Clarke CLA. and Thomas P. Evaluating contextual suggestion. In *5th International Workshop on Evaluating Information Access (EVIA)*, Tokyo, Japan, 2013.
6. Morgan Harvey, Bernd Ludwig, and David Elsweiler. You are what you eat: Learning user tastes for rating prediction. In Oren Kurland, Moshe Lewenstein, and Ely Porat, editors, *String Processing and Information Retrieval*, volume 8214 of *Lecture Notes in Computer Science*, pages 153–164. Springer, 2013.
7. Yehuda Koren. Factorization meets the neighborhood: a multifaceted collaborative filtering model. *14th ACM SIGKDD*, pages 426–434, 2008.
8. Francesco Ricci, Lior Rokach, Bracha Shapira, and Paul B. Kantor, editors. *Recommender Systems Handbook*. Springer, 2011.
9. Richard Schaller, Morgan Harvey, and David Elsweiler. Recsys for distributed events: investigating the influence of recommendations on visitor plans. In *36th ACM SIGIR Conference*, pages 953–956. ACM, 2013.
10. Ellen M. Voorhees. Variations in relevance judgments and the measurement of retrieval effectiveness. In *21st ACM SIGIR Conference*, 1998.
11. I.H. Witten and Frank E. *Data Mining: Practical Machine Learning Tools and Techniques*. Morgan and Kaufmann, second edition, 2005.
12. Biao Xiang, Daxin Jiang, Jian Pei, Xiaohui Sun, Enhong Chen, and Hang Li. Context-aware ranking in web search. In *33rd ACM SIGIR Conference*, SIGIR '10, pages 451–458. ACM, 2010.
13. H. Zhang and J. Su. Naive bayesian classifiers for ranking. In *15th ECML2004 Conference*, Pisa, Italy, 2004.