Context-Aware Tourist Trip Recommendations

Christopher Laß
Department of Informatics
Technical University of Munich
Boltzmannstr. 3
85748 Garching bei München, Germany
christopher.lass@tum.de

Daniel Herzog
Department of Informatics
Technical University of Munich
Boltzmannstr. 3
85748 Garching bei München, Germany
herzogd@in.tum.de

Wolfgang Wörndl
Department of Informatics
Technical University of Munich
Boltzmannstr. 3
85748 Garching bei München, Germany
woerndl@in.tum.de

ABSTRACT

Mobile and web-based services solving common tourist trip design problems are available, but only few solutions consider context for the recommendation of point of interest (POI) sequences. In this paper, we present a novel approach to incorporating context into a tourist trip recommendation algorithm. In addition to traditional context factors in tourism, such as location, weather or opening hours, we focus on two context factors that are highly relevant when recommending a sequence of POIs: *time of the day* and *previously visited point of interest*. We conducted an online questionnaire to determine the influence of the context factors on the user's decision of visiting a POI and the ratings of the POIs under these conditions. We integrated our approach into a web application recommending context-aware tourist trips across the world. In a user study, we verified the results of our novel approach as well as the application's usability. The study proves a high usability of our system and shows that our context-aware approach outperforms a baseline algorithm.

CCS CONCEPTS

• **Information systems → Recommender systems**;

KEYWORDS

Context-Aware Recommender System, Tourist Trip Recommendation, Point of Interest, Tourism

1 INTRODUCTION

Recommender Systems (RSs) are commonly known as software systems that suggest certain items to users in a predictive manner [1]. These systems facilitate the presentation of the available information typically by comparing user preferences to some reference attributes. However, RSs can deliver more sophisticated suggestions by adapting to the specific contextual situation of the recommendation. Hence, context-aware recommender systems (CARSs) provide different movie suggestions based on contextual factors like a user's mood or the time of the day.

In the tourism domain, CARSs are also being developed and researched. Several relevant contextual factors (e.g., weather) and their respective contextual conditions (e.g., raining) have already been identified. For example, a CARS reduces the relevance of outdoor activities while it is raining [3].

Most tourism CARSs focus on suggesting single points of interest (POIs). Only few solve route-planning problems for tourists who want to visit multiple interesting POIs consecutively. This problem statement is summarized as the Tourist Trip Design Problem (TTDP) [13]. The TTDP defines the generic problem of personalized tourist trip generation and is commonly seen as an extension of the Orienteering Problem (OP) [23]. The basic idea is to maximize an objective score between an specific start and end point with several POIs in between [11].

In previous works, we aimed to solve the TTDP and introduced a mobile application and web service for tourist trip recommendations around the world. It takes the user's preferences, time and budget into account [16]. However, contextual information was not considered. In this work, we propose a novel, context-aware route recommendation algorithm that enhances our previous and related work. It incorporates various contextual information, including two that are especially relevant for POI sequences.

The rest of the paper is organized as follows. In Section 2 we present context factors that our CARS observes and explain how the respective contextual conditions ratings have been acquired. In Section 3, our novel context-aware route recommendation algorithm is presented. Section 4 describes the application *TourRec*, implementing our algorithm. In Section 5, the introduced CARS is evaluated against the RS from our previous work. Section 6 and 7 list related work and conclude the paper.

2 EVALUATING CONTEXT FACTORS FOR TOURIST TRIPS

In this section we briefly discuss context and explain how context is relevant for our RS. In order to integrate context-aware information into a tourist trip RS, we first have to identify appropriate context factors for the tourism domain and assess the influence of selected context factors. Also, the effects of each context factor under several contextual conditions on the user's route satisfaction have to be investigated. Therefore, we conducted an online study to observe context factors specifically relevant for sequences of POIs and derive information of similar pre-existing research for other context factors.

2.1 Context in Tourism Recommender Systems

A common definition describes context as "any information that can be used to characterize the situation of a [...] person, place, or object." [12]. Since we are only interested in information that is relevant in the tourism domain, we limit and categorize "any information" to physical context. Physical context can be described as the user's immediate physical surroundings. This includes, but is not limited to, *time of the day*, *light*, *weather*, *date*, *season*, and *temperature* [8]. This information could be retrieved by modern smartphones with a GPS sensor or light sensor in conjunction with

the current time. However, this does not work well for predictions. For a route RS this data mostly has to be retrieved by external services such as openweathermap[1]. Furthermore, the system itself has to be aware of the users physical context at each segment of the route recommendation. For example, a recommended route should contain less outdoor activities during the night or while it is raining.

Another relevant type of context which we do not yet consider in this work is social context. Social context can be described as the user's social group composition at the time of taking the recommendation. The user's standing and role in the group is also an important factor [2]. For example, a recommended route should contain no nightlife activities such as going to a club when children are part of the group.

2.2 Acquiring Context Relevance

We designed an online questionnaire to acquire quantitative measures of how selected contextual factors influence a user's decision of going to a POI. The following approach assesses the context relevance and is based on a methodology presented by Baltrunas et al. [3].

For the preliminary questionnaire a set of possible context factors should be selected by domain experts. The questionnaire participants are asked to imagine certain conditions and whether a specific context factor (e.g., weather) has a positive or negative influence on the rating of a particular item [1, 2].

With this methodology we observe the context factors *time of the day* and *previously visited POI*. These are especially crucial for sequences of POIs and have not yet been observed in related work. For other context factors like *day of the week*, *weather* and *temperature*, which are also relevant for single POIs, we can rely on [3]. Also *opening hours* is considered, which is a context factor that does not require a preliminary user study. The mentioned context factors are later incorporated within the context-aware recommendation algorithm.

Preliminary, twelve POIs in Munich, Germany have been selected and mapped into six predefined categories: *Arts and Museum*, *Food*, *Music Event*, *Nightlife Spot*, *Outdoors and Recreation* and *Shopping*. It is assumed that categories represent all their corresponding POIs. Figure 1 shows how participants are asked whether they would visit a certain POI just after they have been to a different POI. Additionally, we asked the participants at which times they would go to certain POIs.

The aim of this study was to evaluate the influence of the selected context factors on their decisions to visit a category represented by a selected POI as well as the change of POI popularity precipitated by contextual conditions. In total, we received 324 responses by 27 participants.

The measured relevance (U) for each context factor for all POI categories are computed and listed in Table 1. It is normalized to an interval $[0, 1]$; where $U = 0$ means that the context factor does not have any influence for this POI category. U is also relevant for the actual context-aware route recommendation algorithm and is there being utilized as a weighting factor for the context assessment in Equation 5.

[1]https://openweathermap.org/

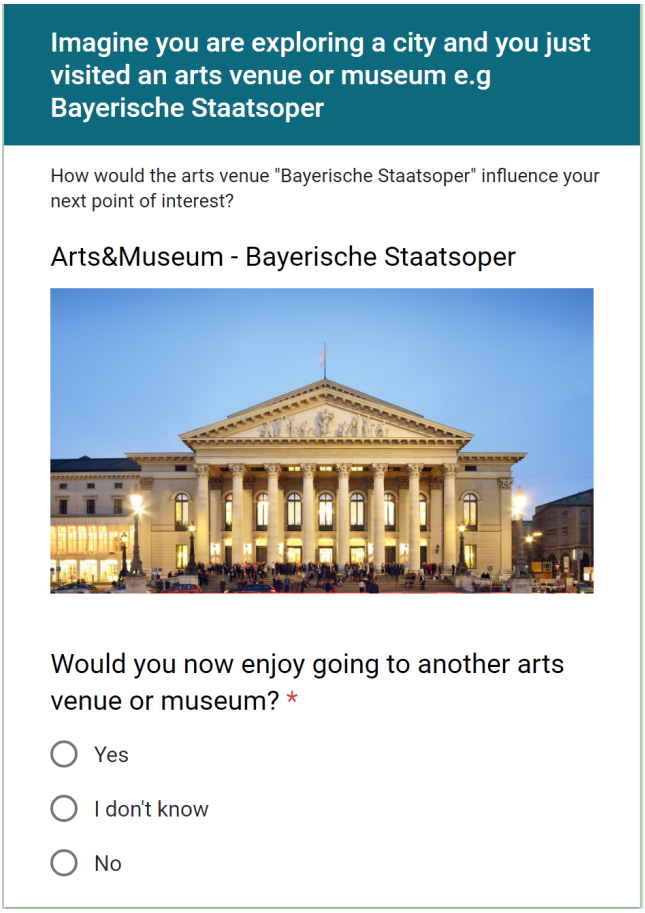


Figure 1: Online questionnaire to acquire context relevance.

In addition to the measured relevance (U) of a context factor, our context-aware approach (cf. subsection 3.3) is also dependent on ratings for POIs under different contextual conditions. The dataset resulting from the previous conducted questionnaire can also be utilized to determine such a rating. To make the responses quantifiable *Yes*, *I don't know* and *No* are mapped to the values $2, 1, 0$. A simple approach would be to use the mathematical expectation value as a rating of a POI category for each contextual condition. However, this does not respect the variation of the rating for a POI when a contextual condition holds or not. Informally speaking, if a POI category is typically very popular, except during *night*, the expectation value would not reflect the real value of the contextual condition *night*. For example, the expectation value for the category *food* is 1.3. However, if one only considers ratings for *food* under the contextual condition *night*, the expectation value is 0.749. Hence, we must present a comparison between the average ratings of POI and ratings of the same items assuming a certain contextual condition holds. We achieve this by dividing the expected value for a specific contextual condition by the expected value over all ratings for this POI category. For the category *Food* during *night* time, the calculation is therefore: $0.749/1.3 = 0.58$. All computed

Table 1: Measured relevance of the contextual factors by POI categories.

Contextual Factor	Arts and Museum	Music Event	Nightlife Spot	Food	Outdoors and Recreation	Shopping
Previously visited POI	0.52	0.31	0.26	0.49	0.33	0.42
Time of the day	0.48	0.69	0.74	0.51	0.67	0.58

ratings for POI categories in different contextual conditions are displayed in Table 2.

3 A NOVEL APPROACH FOR CONTEXT-AWARE ROUTE RECOMMENDATIONS

This section gives a detailed explanation of paradigms for incorporating context into RSs, the baseline path-finding algorithm and our approach for a context-aware path-finding algorithm.

3.1 Paradigms for Incorporating Context in Recommender Systems

This section describes how RS and CARS can be modeled and which paradigms exist to integrate context into a traditional, two-dimensional (2D) RS. 2D RSs try to estimate the rating function R by considering only the *User* and *Item* dimensions [2]:

$$R : User \times Item \rightarrow Rating \tag{1}$$

This rating function can be extended to model a three dimensional (3D) recommendation [2]:

$$R : User \times Item \times Context \rightarrow Rating \tag{2}$$

Adding context in the rating function increases the complexity of the recommendation algorithm. This is a non-trivial problem. Adomavicius and Tuzhilin [2] identify three different paradigms how to incorporate context in a traditional, 2D recommendation process:

Contextual pre-filtering *(or contextualization of recommendation input)*: The context is utilized to construct a dataset only with the most relevant data. After that, a traditional RS can generate the actual recommendations.

Contextual post-filtering *(or contextualization of recommendation output)*: In contrast to contextual pre-filtering, the traditional RS is executed on the *entire* data first and afterwards the context is applied on the resulting set. This can be achieved by:

- *Filtering* out recommendations that are irrelevant (in a given context), or
- *Adjusting* the ranking of recommendations on the list (based on a given context).

Contextual modeling *(or contextualization of recommendation function)*: In this paradigm the 2D RS must be modified and directly incorporate context into the recommendation algorithm.

3.2 Baseline Algorithm

We have improved a route recommendation algorithm in previous work [16, 24] which is not context-aware.

The general idea is to combine as many single POIs as possible to maximize the entertainment for the user while still respecting existing constraints like time. The process of generating a path from an origin to a destination while suggesting relevant POIs in between can be generally divided into two subtasks:

- The POI gathering and scoring, and
- executing a path-finding algorithm to find the optimal route consisting of a subset of the gathered POIs.

For the POI gathering, we are using the Foursquare API[2] to search for POIs in the general area between the source and the destination point. The gathered items are classified into six categories, users can give preferences for these categories (see below). Our algorithm then computes a score for each item. The score is based on the total Foursquare rating and the number of votes, and also the user preference for the corresponding category [24].

The algorithm to combine the POIs to a reasonable route is based on the well-known Dijkstra's algorithm to find the shortest path in a graph. Dijkstra's algorithm is an iterative algorithm that provides the shortest path from one particular starting node to all other nodes in a graph with non-negative edge path costs. In our scenario, the nodes are the places with the associated score and the edges represent the distance between the places.

Prior to the graph spanning, each POI is assigned with a value for the time to spend there. Then, a weighted graph is created using an feasible time value to walk the direct physical distance between each vertex as edge weight. Prior to the comparison of a subpath with another path, it is checked whether the subpath exceeds the specified timeframe. If the timeframe is exceeded, the subpath will be rejected. If another valid subpath from the origin to the immediate vertex can be found prior to the current path, they are compared against each other.

To generate not the shortest, but the best path in the POI graph, we maximize the fraction entertainment/distance for each subpath. The entertainment value is the accumulated sum of the scores of all items on the subpath. To adapt the number of items per category, the baseline algorithm uses the following formula Equation 3.

$$S = p_{pref,poiCategoriesInPath} \times entertainment \tag{3}$$

The idea is to maximize the product of entertainment and Pearson's correlation coefficient between the user's preferences and the amount of POIs per category in the observed path. Pearson's coefficient p gives values between -1 (indicating perfect negative correlation) and +1 (perfect correlation), with 0 meaning no correlation exists between the datasets. Using the correlation coefficient

[2]https://developer.foursquare.com/start/search

Table 2: Ratings for points of interest categories in different contextual conditions.

Contextual Condition\POI Category	Arts and Museum	Music Event	Nightlife Spot	Food	Outdoors and Recreation	Shopping
Previously visited POI (category)						
Arts and Museum	1.36	1	1.16	1.43	1.25	0.72
Food	1.4	1.06	1.77	0.19	1.28	1.18
Music Event	0.04	1.32	1.69	1.1	0.6	0.11
Nightlife Spot	0	1.45	1.43	1.04	0.13	0
Outdoors and Recreation	1.63	1.42	0.86	1.37	0.76	1.52
Shopping	0.91	0.52	0.79	1.45	0.97	1.25
Time of the day						
Morning	1.56	0.1	0.19	0.3	1.36	1.82
Midday	1.56	0.19	0.07	1.29	1.41	1.78
Afternoon	1.48	0.68	0.15	0.85	1.41	1.71
Evening	0.64	1.71	0.79	1.4	0.76	0.8
Night	0.42	1.55	2	0.58	1.07	0.11

aims to balance the amount of POIs in each category more appropriate in relation to the user's preferences. The extension of this adapted POI score with context-awareness is explained in the following subsection.

3.3 Incorporating Context into the Baseline Algorithm

The first challenge that arises is to determine how context-awareness can be calculated for a route. Our collected dataset includes two indications that can be utilizes for this task. First, ratings for categories in different contextual conditions as displayed in Table 2. A rating $r_{TC1...Ck}$ indicates the evaluation for the POI category T made in the context $C1, ..., Ck$ and must be in the interval $[0, 2]$. Second, the relevance of contextual factors $U_{C1...Ck}$ of each context $C1, ..., Ck$ on a POI category T as displayed in Table 1. Like illustrated in subsection 2.2 the measured relevance must be in an interval between $[0, 1]$.

Given this data we can calculate a context-awareness factor $\overline{C}$ with a simple weighted arithmetic mean:

$$\overline{C} = \frac{\sum_{i=1}^{k} U_{Ci} r_{TCi}}{\sum_{i=1}^{k} U_{Ci}} \tag{4}$$

$\overline{C}$ can now be used to extend the 2D recommender baseline algorithm by scaling the result of its comparison function:

$$S = p_{pref, poiCategoriesInPath} \times entertainment \times \overline{C} \tag{5}$$

According to the given constraints, also $\overline{C}$ is in the interval $[0, 2]$ with 0 essentially nulling the score S while 2 would double its value.

To better explain the methodology we can illustrate it with an example comparison considering the two context factors *time of the day* and *previously visited POI*:

It is 5 pm and the user has just been to a restaurant. The CARS should now calculate the score for another restaurant. In this scenario the 2D comparison algorithm would calculate a score of 9.5. The context-aware comparison algorithm (Equation 5) extends the 2D comparison algorithm (Equation 3) by the context-awareness factor $\overline{C}$. To calculate $\overline{C}$, we use the values 0.49 and 0.51 for relevance of the context factors from Table 1 and the values 0.19 and 0.85 for ratings for the category *food* in the current contextual condition from Table 2. After calculating $\overline{C}$, the 2D score of 9.5 is downscaled to 5.

$$\overline{C} = 0.526 = \frac{0.19 \times 0.49 + 0.85 \times 0.51}{0.49 + 0.51} \tag{6}$$

$$S = 5 = 9.5 \times 0.526 \tag{7}$$

According to 3.1, one could assume that this algorithm adheres to *contextual post-filtering*. However, the definition explicitly states, that the traditional RS must be executed on the *entire* data first. Since this is not the case, the paradigm *contextual modeling* was utilized to incorporate context into the baseline.

The full set of context factors considered in the current implementation and their values (contextual conditions) are:

- Previously visited POI (category): arts and museum, food, music event, nightlife spot, outdoors and recreation, shopping
- Time of the day: morning (8am - 12pm), midday (12pm - 2pm), afternoon (2pm - 6pm), evening (6pm - 10pm), night (past 10pm)
- Day of the week: working day, weekend
- Weather: sunny, cloudy, clear sky, rainy, snowing
- Temperature: hot, warm, cold
- Opening hours: open, closed

One benefit of the weighted arithmetic mean is the independence of the number of context factors. This list can easily be extended. Also the amount of context factors applied on POIs within a path can vary. For example, designing context factors only known for a specific POI category, e.g. *nightlife spot*, is not a concern. On the other hand, one disadvantage resulting from considering multiple context factors for $\overline{C}$ is that a supposedly drastic condition, e.g. the POI is closed, can be balanced out by a different condition such as *sunshine*.

4 THE TOURREC WEB APPLICATION

This section firstly presents the multi-tiered and service-oriented system architecture we introduced in (removed for review process). One of the main goals was to facilitate the integration of additional data sources, path-finding algorithms and clients. We then present the user interfaces of our web application *TourRec* [3].

4.1 Architecture

The system is distributed across multiple physical devices offloading application logic, computation and storage onto multiple web services running in the cloud. The applications multi-tier architecture can be partitioned into the presentation tier, application logic tier and data tier, while the application logic tier itself is also partitioned. This architectural style decomposes an application into loosely coupled services, functionality can thereby be easily reused. For example, clients do not have to re-implement the whole application logic but rely on the application logic tier. Other advantages are an improved modularity and the fact that each segment is easier to understand, develop and test. It also simplifies further development since components can be assigned more precisely to experts in their respective fields. An iOS developer gets to develop the iOS app, the android developer the android application and the data scientists can improve the algorithm. Also multiple people can work parallel and independently on different components.

Additionally, the whole application has been containerized with Docker [4] containers. Containers consist of a complete and isolated run time environment: the software including all its dependencies, libraries and other binaries, and configuration files needed to run it. By containerizing our application, the differences in OS distributions and underlying infrastructure are abstracted away. This makes it fairly easy for possible new contributors to run the whole project on their local environment. In addition, it enables continuous delivery and deployment.

In the following, the three tiers are presented.

4.1.1 Presentation Tier. The web application being demonstrated as well as the mobile application we have developed in our previous work is representative for the presentation tier. It is end user facing, aimed to provide high user satisfaction and is responsible to handle user input and display computed information. It utilizes services from the underlying application logic tier and external services like Google Maps.

4.1.2 Application Logic Tier. The two main functionalities of this tier are the gathering of POIs and executing path-finding algorithms. First, POIs are gathered from multiple external service providers such as Foursquare[5] and normalized. Then, this data is passed over to the path-finding algorithms that are executed afterwards. Each path-finding algorithm is extracted into its own dedicated microservice and communicates with the application logic tier via HTTP in order to return a ordered list of POIs to visit. The path-finding microservices can be implemented using arbitrary programming languages, databases, hardware and software environment.

4.1.3 Data Tier. User feedback is being stored and handled within this tier. As it might not seem urgent to dedicate an own tier for this data, the need increases when user accounts are introduced, for example.

4.2 Interfaces

The web application is built with help of the javascript framework Vuejs[6] and the CSS framework Bulma[7]. It is mainly structured into three segments *search*, *recommendation*, *feedback*.

In the *search* segment, as seen in Figure 2, users can enter their preferences for all six predefined categories, the origin and destination as well as the time frame for the route. The input is validated both client side as well as server side. For example, the maximum distance between origin and destination is 7 kilometers and the maximum time frame is 12 hours.

The *recommendation* segment, as seen in Figure 3, is structured as follows. A map with the suggested POIs and a rendered walking path on it is located on the left-hand side. On the top left-hand side some contextual information that has been acquired by the system and is relevant for the user for this situation is displayed. Finally an ordered list of POIs and their respective estimated time of arrival and departure can be found beneath the contextual information.

The *feedback* segment gives the user a short introduction into the feedback process. In essence, it is a simple table with multiple statements which can each be answered with radio buttons on a five-point Likert scale ranging from *strongly disagree* to *strongly agree*. In section 5 we describe the feedback setup in greater detail. Note, that only one route per request gets displayed. The application logic tier randomly selects either the baseline or context-aware path-finding algorithm. The algorithm name is stored in conjunction with the feedback a user provides.

5 USER STUDY

The research question of this paper is whether a context-aware algorithm distinguishing between several contextual conditions can improve the trip recommendations generated by a baseline. We conducted a user study to evaluate the performances of both algorithms. We spread the link to the *TourRec* application via mail and added questionnaires to the interfaces which the users were asked to complete. The participants could access the application on any device with an internet connection since it is publicly available.

5.1 System Usability Score

The System Usability Scale (SUS) is a questionnaire for measuring how people perceive the usability of a computer system [6]. The questionnaire is composed of ten usability statements with five possible response options on a scale ranging from strongly disagree to strongly agree. SUS is technology independent and can be used for hardware, software, websites, mobile applications and more. The key benefits of SUS are reliability, validity, no need a baseline and the fact that it is an industry standard [7].

19 participants completed the SUS questionnaire after using *TourRec*, the average score was 84,167. With the help of Sauro's graph, the score approximately converts to a percentile rank of

[3] https://tourrec.arubacao.com/
[4] https://www.docker.com/
[5] https://foursquare.com/
[6] https://vuejs.org/
[7] http://bulma.io/

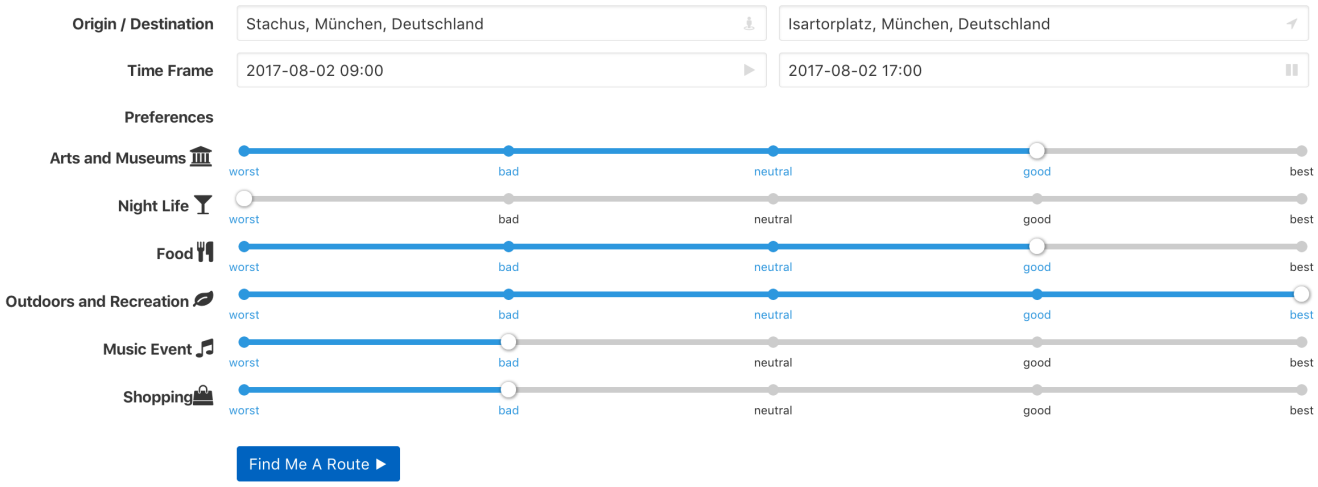
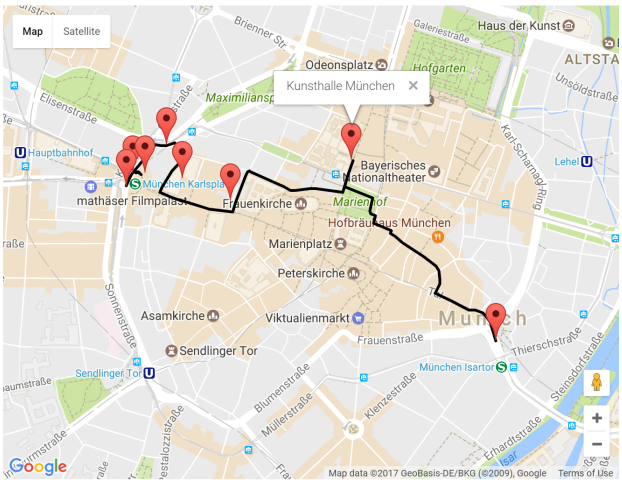


Figure 2: *TourRec* Search Interface



Day of Week: **Saturday**

Weather Forecast: **sky is clear**

Temperature Range: between **-3.13°C** and **8.45°C**

Name	Category	ET Arrival	ET Departure
start		09:00:00	09:00:00
Gloria Palast		09:00:00	10:40:00
Karlsplatz (Stachus)		10:41:00	12:26:00
L'Osteria		12:28:00	13:23:00
Oberpollinger		13:24:00	13:59:00
St. Michael		14:01:00	15:01:00
Kunsthalle München		15:06:00	16:46:00
destination		16:56:00	16:56:00
Name	Category	ET Arrival	ET Departure

Figure 3: *TourRec* Response Interface

94% [21]. This means *TourRec* performs better than about 94% of systems tested in terms of perceived usability. Everything about 90% can be interpreted as an A in school grades. However, since *TourRec* is accessible from virtually any device, the actual systems usability varies for different screen sizes, operating systems and browser vendors.

5.2 Algorithm Performance

In addition to the SUS, we conducted an A/B test to measure the effect of the novel approach on the user's route satisfaction compared to the baseline system that does not exploit context at all. Hence, only one tourist trip recommendation is displayed after every request. Apart from the route, users are not able to distinguish between them. The recommendation screen shown in Figure 3 displays contextual information (e.g. the weather) whether or not the context is actually considered.

After every recommendation, a questionnaire for this part of the evaluation is presented to the user. It is composed of the following six statements and also with five possible response options on a scale ranging from *strongly disagree* (1) to *strongly agree* (5):

(1) Overall, I am satisfied with the recommended tour
(2) The number of places in my route is well chosen
(3) The selection of different categories in the trip is satisfying
(4) Places are suggested at the right times during the tour
(5) The tour is feasible for a walking tourist
(6) I consider taking this route myself

In total, 15 forms were completed for the baseline algorithm and 9 for the context-aware approach. Figure 4 illustrates the performance of both algorithms for each of the six questions in subsection 5.2. Our novel approach for context-aware route recommendation performs somewhat better in the *overall satisfaction* ($\varnothing : 3,67, \sigma : 1,41$)

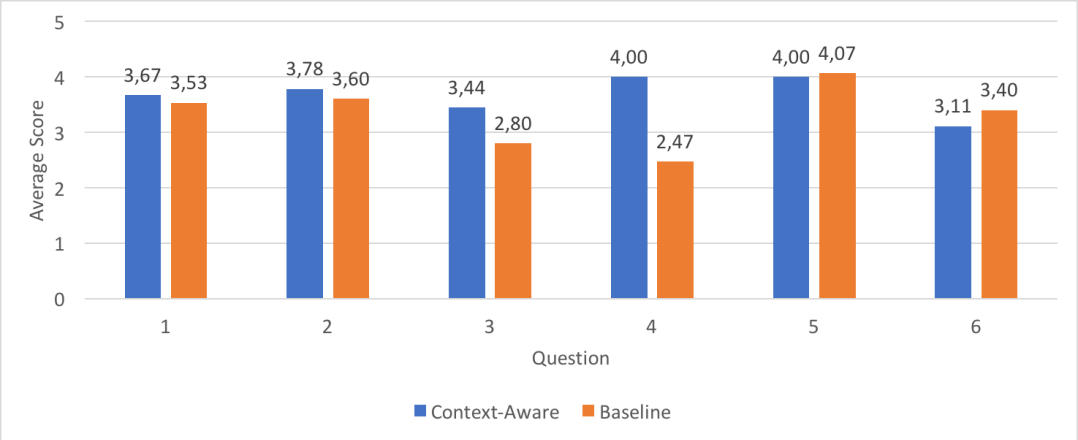


Figure 4: Algorithm Performance Results

and *number of places* ($\varnothing : 3,78, \sigma : 1,39$). In terms of *feasible walking route* and *consider taking the route* the context-aware algorithm is rated slightly lower than the baseline. However, for these four mentioned questions the actual difference is almost neglectable. The biggest difference can be seen for *diversity of categories* and especially *right times* where the context-aware algorithm outperforms the baseline.

A Mann-Whitney U test shows that the difference in *right times* is significant for $\alpha = 0.01$ while the other results we obtained are not yet significant. We conclude that our novel approach leads to improved recommendations but we have to conduct a larger user study in the future to verify our results.

6 RELATED WORK

Some applications solving the TTDP have been developed [9, 23] and numerous publications on this topic ranging from the avoidance of trafficked walking paths [20] to randomizing these [19] exist. In this section, we highlight some important related work based on a general overview of [18].

Gionis et al.[14] and Lim [17] consolidated POIs into categories to enforce a predefined visiting order. In [17] the visiting order was defined by user preferences, and time and budget constraints. Instead of enforcing a specific order, Vansteenwegen et al. [22] recommended tours comprising POI categories that best match user preferences while adhering to these trip constraints.

Tour recommendation can also be formulated as a Generalized Maximum Coverage Problem [4]. The objective here is to find an optimal set of POIs considering its rating and the user's preferences. Brilhante et al. then extended their algorithm later by incorporating a variation of the Travelling Salesman Problem to find the shortest path within an optimal set of POIs [5]. To get from POI to POI within a route, Kurashima et al. also consider different transport modes utilizing the Markov model depending on user preferences and frequently traveled routes [15]. With patterns derived from taxi GPS traces, Chen et al. developed a more context-aware solution considered traveling times based on different traffic conditions [10].

7 CONCLUSION

In this paper, we presented the web application *TourRec* that allows context-aware tour recommendations in arbitrary locations across the world. The system takes a starting point, a destination, a timeframe and user preferences for six predefined categories into account. It solves a variant of the OP applied to the tourism domain. In a preliminary questionnaire, the influence of the context factors *time of the day* and *previously visited POI* were measured as well as ratings for POI categories in different contextual conditions. The results are utilized within the context-aware algorithm.

Context-awareness is incorporated into the baseline algorithm as a scaling factor altering a POI's score depending on the immediate contextual condition. The focus and innovation of our work is on recommending sequences of items. Therefore the influence of an already visited POI on the score of additional items based on their category is important. Users may not be interested in another restaurant if they just had lunch or dinner, for example. In a user study we evaluated that the incorporation of context-awareness leads to a slightly improved user satisfaction and a significant improvement of recommending POIs at the right time.

One disadvantage of our approach is the possible equalizing of two or more extreme contextual conditions due to the weighted arithmetic mean. A modified version could solely consider the context factor that has the largest negative or positive effect on a POI. Using one single factor could potentially characterize the POI's score in a more user satisfying way. Another improvement can be achieved by increasing the number of predefined categories or introduce subcategories. The current limitation of only six categories has led to issues in the route recommendation scenario.

We have designed our framework and architecture for easy extension. We are working on a mobile solution [16] because context-aware route planning seems especially promising in a scenario of mobile users with smartphones visiting a city. In this case, the recommended items should be adapted to the current position and other contextual conditions of the user. We also plan to implement more path-finding algorithms and compare them with the approach presented in this paper in larger user studies. Additional future work includes recommending for groups of visitors.

REFERENCES

[1] Gediminas Adomavicius and Alexander Tuzhilin. 2005. Toward the Next Generation of Recommender Systems: A Survey of the State-of-the-Art and Possible Extensions. *IEEE Trans. on Knowl. and Data Eng.* 17, 6 (June 2005), 734–749. DOI: https://doi.org/10.1109/TKDE.2005.99

[2] Gediminas Adomavicius and Alexander Tuzhilin. 2008. Context-aware Recommender Systems. In *Proceedings of the 2008 ACM Conference on Recommender Systems (RecSys '08)*. ACM, New York, NY, USA, 335–336. DOI: https://doi.org/10.1145/1454008.1454068

[3] Linas Baltrunas, Bernd Ludwig, Stefan Peer, and Francesco Ricci. 2012. Context Relevance Assessment and Exploitation in Mobile Recommender Systems. *Personal Ubiquitous Comput.* 16, 5 (June 2012), 507–526. DOI: https://doi.org/10.1007/s00779-011-0417-x

[4] Igo Brilhante, Jose Antonio Macedo, Franco Maria Nardini, Raffaele Perego, and Chiara Renso. 2013. Where Shall We Go Today?: Planning Touristic Tours with Tripbuilder. In *Proceedings of the 22Nd ACM International Conference on Information & Knowledge Management (CIKM '13)*. ACM, New York, NY, USA, 757–762. DOI: https://doi.org/10.1145/2505515.2505643

[5] Igo Ramalho Brilhante, Jose Antonio Macedo, Franco Maria Nardini, Raffaele Perego, and Chiara Renso. 2015. On planning sightseeing tours with TripBuilder. *Information Processing and Management* 51, 2 (2015), 1 – 15. DOI: https://doi.org/10.1016/j.ipm.2014.10.003

[6] John Brooke. 1996. SUS-A quick and dirty usability scale. *Usability evaluation in industry* 189, 194 (1996), 4–7.

[7] John Brooke. 2013. SUS: A Retrospective. *J. Usability Studies* 8, 2 (Feb. 2013), 29–40. http://dl.acm.org/citation.cfm?id=2817912.2817913

[8] P. J. Brown, J. D. Bovey, and Xian Chen. 1997. Context-aware applications: from the laboratory to the marketplace. *IEEE Personal Communications* 4, 5 (Oct 1997), 58–64. DOI: https://doi.org/10.1109/98.626984

[9] Luis Castillo, Eva Armengol, Eva OnaindÃŋa, Laura SebastiÃą, JesÃžs GonzÃąlez-Boticario, Antonio RodrÃŋguez, Susana FernÃąndez, Juan D. Arias, and Daniel Borrajo. 2008. samap: An user-oriented adaptive system for planning tourist visits. *Expert Systems with Applications* 34, 2 (2008), 1318 – 1332. DOI: https://doi.org/10.1016/j.eswa.2006.12.029

[10] C. Chen, D. Zhang, B. Guo, X. Ma, G. Pan, and Z. Wu. 2015. TripPlanner: Personalized Trip Planning Leveraging Heterogeneous Crowdsourced Digital Footprints. *IEEE Transactions on Intelligent Transportation Systems* 16, 3 (June 2015), 1259–1273. DOI: https://doi.org/10.1109/TITS.2014.2357835

[11] Munmun De Choudhury, Moran Feldman, Sihem Amer-Yahia, Nadav Golbandi, Ronny Lempel, and Cong Yu. 2010. Automatic Construction of Travel Itineraries Using Social Breadcrumbs. In *Proceedings of the 21st ACM Conference on Hypertext and Hypermedia (HT '10)*. ACM, New York, NY, USA, 35–44. DOI: https://doi.org/10.1145/1810617.1810626

[12] Anind K. Dey. 2001. Understanding and Using Context. *Personal Ubiquitous Comput.* 5, 1 (Jan. 2001), 4–7. DOI: https://doi.org/10.1007/s007790170019

[13] Damianos Gavalas, Charalampos Konstantopoulos, Konstantinos Mastakas, and Grammati Pantziou. 2014. A Survey on Algorithmic Approaches for Solving Tourist Trip Design Problems. *Journal of Heuristics* 20, 3 (June 2014), 291–328. DOI: https://doi.org/10.1007/s10732-014-9242-5

[14] Aristides Gionis, Theodoros Lappas, Konstantinos Pelechrinis, and Evimaria Terzi. 2014. Customized Tour Recommendations in Urban Areas. In *Proceedings of the 7th ACM International Conference on Web Search and Data Mining (WSDM '14)*. ACM, New York, NY, USA, 313–322. DOI: https://doi.org/10.1145/2556195.2559893

[15] Takeshi Kurashima, Tomoharu Iwata, Go Irie, and Ko Fujimura. 2010. Travel Route Recommendation Using Geotags in Photo Sharing Sites. In *Proceedings of the 19th ACM International Conference on Information and Knowledge Management (CIKM '10)*. ACM, New York, NY, USA, 579–588. DOI: https://doi.org/10.1145/1871437.1871513

[16] Christopher Laß, Wolfgang Wörndl, and Daniel Herzog. 2016. A Multi-Tier Web Service and Mobile Client for City Trip Recommendations. In *The 8th EAI International Conference on Mobile Computing, Applications and Services (MobiCASE)*. ACM. DOI: https://doi.org/10.4108/eai.30-11-2016.2267194

[17] Kwan Hui Lim. 2015. Recommending Tours and Places-of-Interest Based on User Interests from Geo-tagged Photos. In *Proceedings of the 2015 ACM SIGMOD on PhD Symposium (SIGMOD '15 PhD Symposium)*. ACM, New York, NY, USA, 33–38. DOI: https://doi.org/10.1145/2744680.2744693

[18] Kwan Hui Lim, Jeffrey Chan, Christopher Leckie, and Shanika Karunasekera. 2015. Personalized Tour Recommendation Based on User Interests and Points of Interest Visit Durations. In *Proceedings of the 24th International Conference on Artificial Intelligence (IJCAI'15)*. AAAI Press, 1778–1784. http://dl.acm.org/citation.cfm?id=2832415.2832496

[19] Claudio Lucchese, Raffaele Perego, Fabrizio Silvestri, Hossein Vahabi, and Rossano Venturini. 2012. How Random Walks Can Help Tourism. In *Proceedings of the 34th European Conference on Advances in Information Retrieval (ECIR'12)*. Springer-Verlag, Berlin, Heidelberg, 195–206. DOI: https://doi.org/10.1007/978-3-642-28997-2_17

[20] Daniele Quercia, Rossano Schifanella, and Luca Maria Aiello. 2014. The Shortest Path to Happiness: Recommending Beautiful, Quiet, and Happy Routes in the City. In *Proceedings of the 25th ACM Conference on Hypertext and Social Media (HT '14)*. ACM, New York, NY, USA, 116–125. DOI: https://doi.org/10.1145/2631775.2631799

[21] J. Sauro. 2011. *A Practical Guide to the System Usability Scale: Background, Benchmarks & Best Practices*. CreateSpace Independent Publishing Platform. https://books.google.co.uk/books?id=BL0kKQEACAAJ

[22] Pieter Vansteenwegen, Wouter Souffriau, Greet Vanden Berghe, and Dirk Van Oudheusden. 2011. The City Trip Planner. *Expert Syst. Appl.* 38, 6 (June 2011), 6540–6546. DOI: https://doi.org/10.1016/j.eswa.2010.11.085

[23] Pieter Vansteenwegen and Dirk Van Oudheusden. 2007. The Mobile Tourist Guide: An OR Opportunity. *OR Insight* 20, 3 (2007), 21–27. DOI: https://doi.org/10.1057/ori.2007.17

[24] Wolfgang Wörndl, Alexander Hefele, and Daniel Herzog. 2017. Recommending a sequence of interesting places for tourist trips. *Information Technology & Tourism* 17, 1 (2017), 31–54. DOI: https://doi.org/10.1007/s40558-017-0076-5

