Knowl Inf Syst (2019) 60:523–544
https://doi.org/10.1007/s10115-018-1217-7

REGULAR PAPER

Personalized itinerary recommendation with time constraints using GPS datasets

Yu-Ling Hsueh[1] · Hong-Min Huang[1]

Received: 19 June 2017 / Revised: 14 November 2017 / Accepted: 7 May 2018 /
Published online: 11 May 2018

Abstract Planning a personalized itinerary for an unfamiliar region requires much effort to design desirable travel plans. With the rapid development of location-based social network (LBSN) services, data mining techniques are utilized to retrieve useful information such as *geographical features* and *social relationships*. In this paper, we propose a *personalized itinerary recommendation with time constraints* (pirT) framework for the LBSN by exploiting geographical features and social relationships to recommend a personalized itinerary that satisfies user preferences (i.e., travel behaviors). In pirT, we have designed a user-based collaborative filtering with time preference (UTP) to explore user preferences by considering the visiting time of locations which the users have visited in our framework. UTP allows a tourist to find users with similar travel behaviors to those of the service requester in the past and to recommend interesting locations in the itineraries that these similar users have traveled to before. Subsequently, given a beginning location and a destination with a time constraint specified by the tourist, we devise the top-k A^* search-based recommendations and re-ranking itinerary candidate algorithms to efficiently plan the top k personalized itineraries. In the planning process, we simultaneously take account of the visiting time of locations, the transit time between locations, and the order of visiting locations. We conducted our experiments on the Gowalla dataset and demonstrated the effectiveness of our pirT framework comparing it with the personalized trip recommendation (PTR) framework. The results show that our pirT framework is superior to the PTR framework.

Keywords Personalized itinerary recommendation · Recommender systems · Trip planning · Collaborative filtering · Location-based social networks · GPS data

✉ Yu-Ling Hsueh
hsueh@cs.ccu.edu.tw

Hong-Min Huang
hhm100m@cs.ccu.edu.tw

[1] Department of Computer Science and Information Engineering, National Chung Cheng University, Chiayi, Taiwan

1 Introduction

Traveling is one of the leisure activities to relax and enjoy after work. However, not every tour package is satisfactory for everyone. When planning a *personalized* itinerary, tourists put a great deal of time and effort to research their travel plans, since a huge amount of travel information is available on the Internet. Moreover, the smart phones and location-based navigation systems have been gaining dramatic popularity in recent years. Some of the major functionalities allow users to easily record their GPS data (i.e., check-in data and trajectories). Some location-based social network (LBSN) Web sites permit their users to share travel experiences, mark their locations (i.e., latitude and longitude of positions with timestamps), record outdoor activities, and connect with each other. Therefore, we can discover important information that includes *geographical features* (e.g., interesting locations or sights, proper visiting time and order of each location, travel time from one location to another) and *social relationships* (e.g., users whose travel behaviors are similar) in the LBSN. For example, Gowalla,[1] Foursquare,[2] and ShareMyRoutes.com[3] enable users to use mobile APPs installed on their phones to mark where they are and share what outdoor activities they do. Therefore, we can exploit geographical features and social relationships supported by LBSN services to retrieve user preferences and suggest a personalized recommendation for a query user to plan their itineraries.

In recent years, the study of trip planning has apparently moved from considering the shortest distance on a map to researching geographical features and social relationships for location recommendation. Nevertheless, several studies [3,12,15,16] compute the shortest route for improving performance. The related work [2,9,10,13,17,19–21] has proposed to recommend a location for traveling. The existing work does not consider both geographical features and user preferences to suggest a personalized recommendation for a trip itinerary. In other words, the existing work never takes advantage of interesting locations or sights, proper visiting time, or visiting order for each location, transit time from one location to another, or user preferences to recommend a personalized itinerary. In this paper, we propose a novel approach termed *personalized itinerary recommendation with time constraints* (pirT) that integrates geographical features and social relationships to fulfill a personalized itinerary recommendation that satisfies user's preferences when taking a trip to an unfamiliar city. During the trip planning, we consider both time constraints and geographical features consisting of the trip duration, the time to visit a location, the transit time from one location to another, and the visiting order of locations. For example, a tourist wants to plan a trip to New York city, a place where he or she seldom goes. The tourist starts by checking New York city hotels and then arranges a trip with a total duration of 10 h. Hence, we aim to recommend a personalized itinerary provided by similar users who have had similar travel behaviors in the past. The trip duration is set to 10 h, as specified by the tourist.

The results of pirT are mainly affected by geographical features, social relationships, and trip duration. There are three challenges that need to be resolved. *First*, we handle how to recognize different GPS data which belong to the same location. We study the density-based methods for clustering to deal with a large amount of GPS data for data preprocessing. In other words, based on density-based clustering methods [8], we can reduce the size of GPS data. *Second*, we deal with how to retrieve the geographical features from the GPS data. The LBSN services contain a large number of the GPS data. By using the statistical

[1] http://gowalla.com.

[2] https://foursquare.com/.

[3] http://www.sharemyroutes.com/.

models [9], we analyze GPS data to find interesting locations, the proper visiting time and visiting order for each location, and the transit time between two locations. For example, to extract these geographical features, we identify the most popular locations in New York city for a tourist. *Third*, we consider how to determine a location to recommend to the tourist based on similar users. The collaborative filtering approaches have been utilized intensively on recommender systems [1]. We propose a user-based collaborative filtering with time preference (UTP) to find similar users with similar preferences in the past and predict the locations for an itinerary which depends on users similar to the tourist. Likewise, we have designed a metric by integrating geographical features with user preferences to measure the quality of a recommended itinerary.

As mentioned above, we propose pirT to recommend a personalized itinerary that considers user GPS data to retrieve geographical features and UTP to retrieve similar users that considers the visiting time of locations to understand the user travel behaviors of a tourist who plans to travel to an unfamiliar region. The pirT framework comprises off-line and online phases. In the off-line phase (data preprocessing), we first analyze GPS data to discover the geographical features and measure the travel behaviors of a query user to observe user preferences. In the online phase (trip planning), we plan an itinerary by considering geographical features and user preferences for the query user. The goal of this paper is to design an efficient planning algorithm and an effective recommendation service to find the top k personalized itineraries. We summarize the contributions of this paper as follows.

- We propose the *personalized itinerary recommendation with time constraints* (pirT) framework by considering geographical features, user preferences, and trip duration.
- We study the geographical features to analyze user travel behaviors for a query user and propose the user-based collaborative filtering with time preference (UTP) approach to find users similar to the query user.
- We propose the top-k A^* search-based recommendation and re-ranking itinerary candidate algorithms to efficiently plan a personalized itinerary.

The rest of this paper is organized as follows. In Sect. 2, we review the related work. In Sect. 3, we provide an overview of our system architecture. In Sects. 4 and 5, we introduce our methods involving off-line and online phases to compute itineraries by analyzing the geographical features and user preferences. In Sect. 6, we show the experimental results and prove the effectiveness of our framework in detail. Finally, we describe our conclusions in Sect. 7.

2 Background and related work

In this section, we present prior work which is related to this paper. The related work involves the location-based social networks (LBSNs), trip planning, and location recommendation topics. Each topic is described as follows.

2.1 Location-based social networks

In the last few years, the location-based social networks (LBSNs) such as Gowalla, Foursquare, and ShareMyRoutes have been gaining popularity. The LBSN services are a platform where users can create personal profiles, friends, pictures, activities, check-in records, etc. The users share their life experiences and connect with other users on the LBSN platforms, in other words, some people like interacting on the Internet more than in the real world.

We could understand social relationships and find out interesting locations. Therefore, many researchers consider user preferences and location features to support the recommended services. Four locale-based metrics to analyze the social association were proposed [18]. They observe that people who share more locations receive more attention from other users, and people who are popular follow other popular users. A trajectory-based real-time and on-the-go experience sharing system in a metropolitan area called MetroScope was presented [11]. MetroScope allows people to share their daily life experiences and helps them to find interesting activities. A data model of socio-spatial network that stores frequently traveled routes was introduced [6]. Besides, they presented a query language for implementing social-based route recommendation.

2.2 Trip planning

The traveling salesman problem (TSP) has drawn increasing attention in the last three decades. Researchers have improved TSP performance by using approximation algorithms [5]. Subsequently, the geographical information systems (GIS) have focused considerable attention on the TSP. Several related trip planning query methods [3,12,15,16] for the GIS applications were proposed. In Li et al. [12], a novel query called the trip planning query (TPQ) was presented. Given a beginning location, a destination and a set of locations, each of which belongs to a category specified by a query user, the solution finds the best trip from his or her beginning location to a specified destination which passes through at least one location from each category. A multi-type nearest neighbor (MTNN) query was addressed in Ma et al. [15], and a page-level upper bound (PLUB)-based algorithm was proposed to find the shortest route that starts at the query location and goes through one location from each category without any predefined visiting order of categories. An optimal sequenced route (OSR) query problem was addressed in Sharifzadeh et al. [16], and three optimal solutions were proposed to find the shortest route which begins at a user-specified location and visits one or more locations from each category in a particular order based on each category. In Chen et al. [3], the authors proposed a multi-rule partial sequenced route (MRPSR) query to assist users in planning a trip that involves several destinations that the users want to visit and finding the trip passing through exactly one location from each category. The goal of MRPSR is to efficiently find a route that has the minimal total traveling distance and the predefined traveling order. However, in recent years, new research studies [9,13,19] have appeared to tackle the issue of recommendation systems.

2.3 Location recommendation

For the recommendation research, [20,22] have shown how interesting locations and traveling sequences affect the location recommendations retrieved from a huge number of GPS trajectories. In Zheng et al. [22], the authors presented multiple individuals' location histories with a tree-based hierarchical graph (TBHG) and proposed a HITS-based (Hypertext- Induced Topic Search) inference model for mining interesting locations and traveling sequences. Moreover, in Zheng and Xie, 2011 [20], a traveling recommendation system was performed to support generic recommendations and personalized recommendations. In Ye et al. [17], the authors exploited the social and geographical characteristics of users and locations to understand the relationship between the users and locations for location recommendation. Although these works only recommend interesting locations or locations that are matched to user preferences, the itineraries for tourists are not provided by the system. In Hsieh et al. [9], the authors proposed a time-sensitive route recommendation and proved that geographical features can improve route planning results. In Lu et al. [14], Trip-Mine was proposed to efficiently find an optimal trip, which has been searched for as the most interesting trip which matches a user-specified trip duration. However, neither method observes user preferences that can affect the itinerary recommendation. In Hsieh et al. [13]; Lu et al. [10]; Yoon et al. [19], an itinerary recommendation to recommend itineraries was computed by considering user preferences and trip duration. However, the important information, such as visiting time and transit time, was not considered, resulting in itineraries that are not particularly useful. In contrast to the related work, we recommend the user-preferred itineraries by considering all of the geographical features and analyzing user preferences with geographical features. Finally, we summarize the differences between this paper and related work in location recommendation as shown in Table 1.

3 System overview

In this section, we first describe our system architecture that comprises two phases. Second, we discuss the data update mechanism for refreshing the data preprocessing procedures. Third, we present the terminologies and definitions used in this paper.

3.1 System architecture

We describe the architecture of our work in Fig. 1. We divide the architecture into the off-line phase and the online phase. The off-line phase mainly preprocesses geographical features and user preferences, while the online phase retrieves a personalized itinerary recommendation based on a user-specified query. In the off-line phase, first of all, we reduce a huge amount of GPS data by density-based clustering methods in order to find the similar points within the same cluster (i.e., a location) and reduce points where people seldom visit (i.e., outliers). Next, we perform a statistical model to identify the knowledge of geographical features extracted from clustered GPS data. Then, we take advantage of the geographical features to estimate ratings for the locations that have been visited by a query user. The ratings represent the travel behaviors of the query user. Therefore, to find a set of similar users who have similar travel behaviors for a query user, we calculate the similarity between the query user and each user in the system based on their ratings. Furthermore, when the query user wants to visit a location that he or she has not visited, we take account of similar users to predict whether this location matches the travel behaviors of the query user. In the online phase, we use the aforementioned knowledge to recommend the top k personalized itineraries. The query user specifies a beginning location with a starting time and destination with an ending time before planning a trip. The A^* search-based recommendation algorithm proposed in this paper plans the personalized itinerary candidates by considering a user-preferred location list and proper travel time between locations. Subsequently, we re-rank the personalized itinerary candidates by ranking the scores of all user-preferred locations and considering the proper time to visit a location and the visiting order between locations. Finally, the algorithm returns the top k personalized itineraries to the query user. The details of the procedures for the off-line and online phases are described in Sects. 4 and 5, respectively.

3.2 Data update mechanism

As described above, the data preprocessing procedures depend on GPS data previously recorded by the users in the LBSN services. We need to keep updating the procedures to

Table 1 Differences between this paper and other existing work

	Trip planning	Popularity	Visiting time	Transit time	Visiting order	User preference	Trip duration
Lu et al. [13]	√	√	√			√	√
Hsieh et al. [9]	√	√	√	√	√		√
Hsieh et al. [10]	√				√	√	√
Zheng and Xie [20]		√			√	√	
Lu et al. [14]	√	√					√
Yoon et al. [19]	√	√		√	√	√	√
Ye et al. [17]						√	
Zheng et al. [22]		√			√		
pirT	√	√	√	√	√	√	√

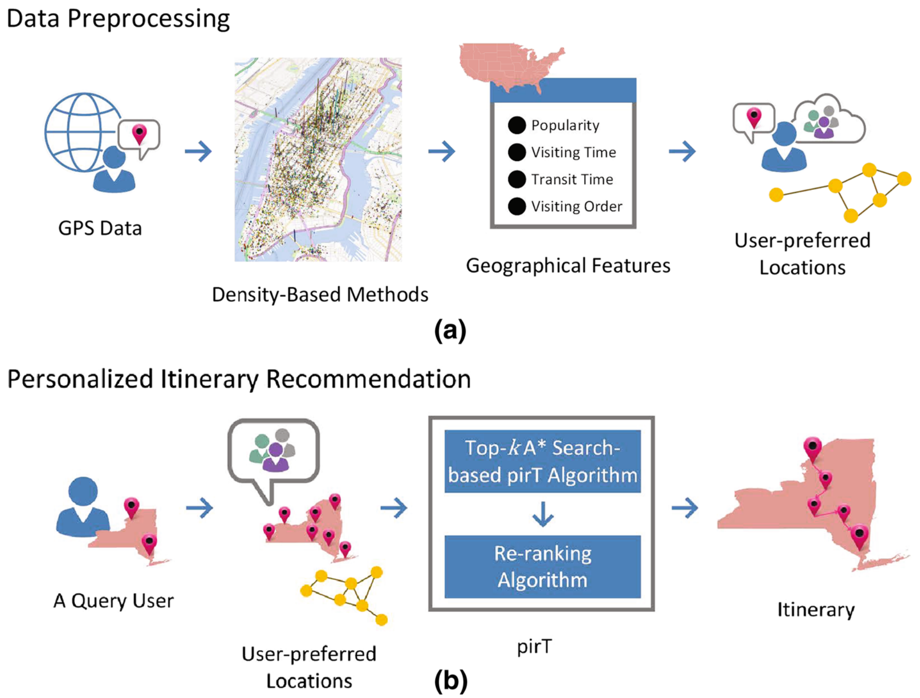


Fig. 1 System architecture. **a** Off-line phase and **b** online phase

retrieve new knowledge for recommending the personalized itineraries that satisfy the user preferences. The update mechanism can be either a period update or an event trigger mechanism for our architecture. We observe that the growth of registered users and check-in locations in the LBSN services such as Gowalla and Foursquare can be available on the periodic trigger. On the growth of registered users, we plan to analyze their preferences once a month and provide them with some personalized itineraries. In addition, considering the growth of check-in locations, we can re-perform data analysis once a week to update user preferences. On the event trigger, for new users, we plan the popular itineraries proposed by existing work because the users have too little GPS data such that analyzing their preferences for recommending their personalized itineraries is impossible. Moreover, when there are special events such as festivals, people may travel to more sights; therefore, such an event also triggers our update mechanism.

3.3 Preliminaries

Definition 1 (*GPS points*) $D = \{d_1, d_2, \ldots, d_n\}$ is a set of GPS points, where each $d_i = (u, t, p)$ in D indicates that a user u marked a point p at a timestamp t in the LBSN services. A point $p = (lat, lng)$ consists of the latitude and longitude of a position.

Please note that we use a dot notation to specify a specific attribute of a data tuple. For example, $d_i.t$ represents the timestamp t, when a GPS point d_i was checked in. Otherwise, u, t, or p is a general attribute for a user, a timestamp, or a GPS point.

Definition 2 (*Locations*) A set of locations $L = \{\ell_1, \ell_2, \ldots, \ell_m\}$ is discovered by a clustering method from the GPS data D, where each ℓ_j in L is essentially a GPS point p in the space. Each ℓ is associated with a timestamp t. We say ℓ_i is a location checked in earlier than ℓ_j, when the timestamp t_i of ℓ_i is smaller than the timestamp t_j of ℓ_j.

Definition 3 (*Visited and unvisited locations*) $L_{\text{visited}} = \{\ell_1, \ell_2, \ldots, \ell_w\}$ is a subset of L, and each ℓ_i in L_{visited} has been visited by a user u in the past. Likewise, $L_{\text{unvisited}} = \{L - L_{\text{visited}}\}$ is a subset of L and each ℓ_i in $L_{\text{unvisited}}$ has never been visited by u.

Definition 4 (*User-preferred locations*) Let $RL_u = \{\ell_1, \ell_2, \ldots, \ell_k\}$ denote a list of top-k recommended locations retrieved from $L_{\text{unvisited}}$ of a user u. Each recommended location in RL_u is ranked by the user-preferred location score (see Sect. 4.3.3) of user u in descending order.

Definition 5 (*Itinerary*) A user's itinerary $I = \{r_1, r_2, \ldots, r_z\}$ is a sequence of recommended locations, where each $r_i = (\ell, t)$ in I consists of a location ℓ in RL_u and a suggested arrival time t for ℓ. I is ordered by the arrival time of each r_i such that $r_1.t < r_2.t < \cdots < r_z.t$.

Definition 6 (*Travel time*) Given a user's itinerary $I = \{r_1, r_2, \ldots, r_z\}$, the travel time $r_z.t - r_1.t$ is less than or equal to the user-specified travel time.

Definition 7 (*Itinerary score*) Given a user's itinerary $I = \{r_1, r_2, \ldots, r_z\}$, the itinerary score $\text{IPS}(I)$ denotes the satisfaction of I for a user u.

$$\text{IPS}(I) = \sum_{i=1}^{z} \text{PS}(u, r_i)$$

$PS(u, r_i)$ is the user-preferred location scores function to measure the satisfaction of unvisited locations $L_{\text{unvisited}}$ for a user u. The details of how to compute the user-preferred location score are described in Sect. 4.3.3.

4 The off-line phase: data preprocessing

4.1 Density-based clustering methods

In this paper, we first reduce a large amount of GPS data D by extracting locations L, in which the points reside in groups, and by detecting the infrequent check-in points. The density-based clustering methods have been developed to find arbitrary shapes of clusters. The methods discover the clusters based on the notion of density. In other words, such methods discover a cluster if the points in the cluster contain more than a minimum number of points (MinPts) that are within a specified range (ε-neighborhood). Therefore, we use DBSCAN [7] to discover clusters of arbitrary shapes and filter out outliers. For example, we show that some points in the GPS data are marked near the central park in New York city in Fig. 2a. The GPS points near Central Park are added to the same cluster as Central Park as illustrated in Fig. 2b. We cluster these GPS points that were possibly marked for Central Park and define the cluster as a location ℓ. In addition, we consider the locations that only appear for a certain period of time (e.g., a week). However, in some cases, for example, many people were at the Apple store on Fifth Avenue to buy iPhones and checked in their positions just within a day. Those points could become a location. Therefore, a cluster is not considered as a location if the difference between the earliest and the latest timestamps of two GPS points in the cluster is less than one week.

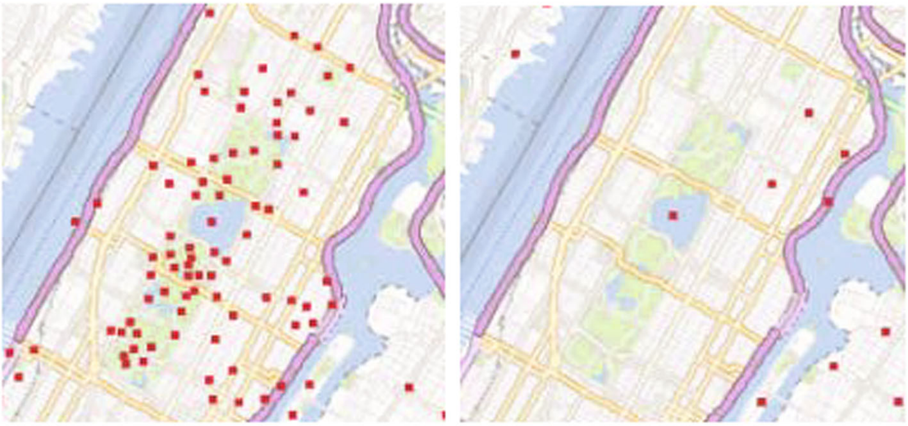


Fig. 2 Density-based clustering methods. **a** Check-in GPS points near Central Park, **b** extracted locations for Central Park

4.2 Geographical features

We assume a user's travel behaviors and a proper personalized itinerary should consider geographical features before analyzing user preferences and planning personalized itineraries. Accordingly, we use a statistical model to measure the following factors over locations ℓ after the clustering analysis:

1. The popularity of locations.
   Generally speaking, a popular location should attract tourists to visit. The number of check-in data represents the popularity of a location. The relative popularity of a location L is defined in the following formula where $N(\ell)$ is the number of check-ins and $N_{\max}$ is the total number of check-ins of all users.
$$f_{\mathrm{pop}}(\ell) = \frac{N(\ell)}{N_{\max}} \tag{1}$$
2. The visiting time of a location.
   In general, tourists usually visit a location within a certain time period. The certain time period may be a popular or a non-popular time period for visiting the location. The popular time period means that tourists have visited this location at that time period more often than they visited the location at other time periods. Conversely, a non-popular time period means that tourists have visited this location at a certain time period less often. Therefore, we analyze the 24-h time periods in a day for each location and observe the time-dependent popularity of each location. A Temporal Visiting Distribution was defined in Hsieh et al. [9] for a location ℓ denoted as $TVD_\ell(t_i)$. $TVD_\ell(t_i)$ indicates that the probability distribution of a location ℓ that checks in at time t_i. Using the symmetric Kullback–Leibler (KL) divergence between TVD and a thin Gaussian distribution $G(t;\mu,\sigma^2)$ can determine the proper time of visiting a location at a given time t. The proper visiting time is defined as follows:
$$f_{\mathrm{visit}}(\ell) = D_{KL}(G(t;\mu,\sigma^2)||TVD_\ell(t_i)) \tag{2}$$
3. The transit time between locations.

We retrieve the transit time between locations based on the cluster analysis performed on GPS data and provide a proper transit time to a query user in planning an itinerary. The transit time affects the choice of the next location that the query user decides to visit. Depending on two consecutive locations, we implicitly obtain the transit time between locations as "staying + transit" time. This staying and transit time is calculated by subtracting the check-in time of the first location ℓ_i from that of the second location ℓ_j. Hsieh et al. defined a duration distribution (DD) between location ℓ_i and ℓ_j denoted as $DD(\ell_i, \ell_j)$ as the probability distribution over time duration $t_{i,j} = t_j - t_i$, which is similar to *TVD*. Consequently, using the symmetric Kullback–Leibler (KL) divergence finds the proper transit time duration between locations. The proper transit time is defined as follows:

$$f_{\text{duration}}(\ell_i, \ell_j) = D_{KL}(G(t; \Delta_{i,j}; \sigma^2)||DD(\ell_i, \ell_j)) \tag{3}$$

4. The visiting order of locations.
   Considering the visiting order of locations is the last factor that reveals realistic traveling order in real life. We adopt a *n*-gram language model in Hsieh et al. [9] to measure a personalized itinerary I, where I is a sequence of locations consecutively visited, as an optimal visiting order. Adopting the average of the probabilities of uni-gram, bi-gram, and trigram denoted as P_{uni}, P_{bi}, and P_{tri}, respectively, estimates the visiting order.

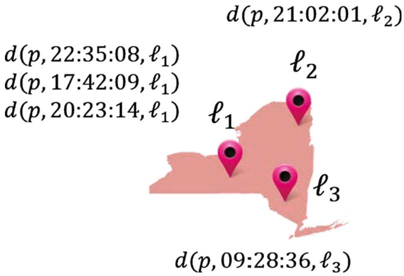
$$f_{\text{order}}(I) = \frac{P_{uni}(I) + P_{bi}(I) + P_{tri}(I)}{3} \tag{4}$$

4.3 User-preferred locations

4.3.1 User ratings by counting time period

We consider user travel behaviors to estimate the ratings for a subset of locations L, L_{visited}, which have been visited by a query user q. We count the number of each check-in location ℓ which means the number of times a user q visited a location ℓ. The number of location ℓ is defined as (5) and (6), i.e., *count*. A higher *count* of a location represents that the user q prefers the location to others. We define the relative check-in location to normalize each location. The relative check-in location divides the *count* of each location by MAX, and MAX is the maximum *count* among all locations which the user q visited. Subsequently, we present two types of user ratings to estimate each location based on the time period pattern. We discuss the time period pattern of the visiting time when the user q liked traveling. The first approach adopts the 12-h clock defined as (5). The 12-h clock means that the 24 h of the day are divided into two periods, a.m. and p.m.. We calculate the total number of people visiting a location in the a.m. and p.m., respectively. Likewise, the second approach chooses the 24-h clock defined as (6) and the 24-h clock divides the day into 24 h. We count the number of each hour period when user q has visited the location in the past.

$$r(q, \ell) = \left(\frac{count}{\text{MAX}}, |\text{am}(d.t)|, |\text{pm}(d.t)|\right)$$
$$\text{where } \forall \ell \in L_{\text{visited}}, \forall d \in D, d.u = q, d.p \in \ell \tag{5}$$
$$r(q, \ell) = \left(\frac{count}{\text{MAX}}, |0\text{hr}(\ell.t)|, |1\text{hr}(\ell.t)|, \ldots, |23\text{hr}(\ell.t)|\right)$$
$$\text{where } \forall \ell \in L_{\text{visited}}, \forall d \in D, d.u = q, d.p \in \ell \tag{6}$$



	q
ℓ_1	$(1.0, 0, 3)$
ℓ_2	$(0.333, 0, 1)$
ℓ_3	$(0.333, 1, 0)$

Fig. 3 An example of user ratings by counting the time period with a 12-h clock

The user rating of query user q is a set of user ratings R for which the user q has visited the locations ($\forall \ell \in L_{\text{visited}}$) in the past. The user rating is denoted as shown in Eq. (7).

$$R_q = \{r(q, \ell) | \forall \ell \in L_{\text{visited}}\} \tag{7}$$

Given a 12-h clock as an example in Fig. 3, we assume a query user q has visited ℓ_1, ℓ_2, and ℓ_3, $L_{\text{visited}} = \{\ell_1, \ell_2, \ell_3\}$. The user q visited ℓ_1 three times, and the visiting times were 22:35:08, 17:42:09, and 20:23:14. Visiting ℓ_2 and ℓ_3 are once, and their arrival times are 21:02:01 and 09:28:36, respectively. Hence, $r(q, \ell_1)$ is (1.0, 0, 3), $r(q, \ell_2)$ is (0.333, 0, 1), and $r(q, \ell_3)$ is (0.333, 1, 0). The user rating of user q is $R_q = \{r(q, \ell_1), r(q, \ell_2), r(q, \ell_3)\} = \{(1.0, 0, 3), (0.333, 0, 1), (0.333, 1, 0)\}$.

4.3.2 *User-based collaborative filtering with time preference (UTP)*

The collaborative filtering is a well-known recommendation method for recommender systems [1]. A user is recommended a location that users with similar tastes and preferences (i.e., travel behaviors) have visited in the past.

In this paper, we propose the user-based collaborative filtering with time preference (UTP) strategy to retrieve a set of users U_q, who are similar to a query user q. Therefore, we utilize user ratings by counting time period R_q, which is a set of location ratings of the user q estimated by time period pattern to analyze user travel behaviors. Moreover, we use the ratings to calculate UTP for finding users similar to user q. The similarity(q, u) of the query user q with respect to the other $u \in U_q$ is defined as Eq. (8).

$$\begin{aligned} &\text{similarity}(q, u) \\ &= \frac{\sum_{\ell \in L'} \text{TIME}(r(q, \ell), r(u, \ell)) \times (1 - \text{DIFFERENCE}(r(q, \ell), r(u, \ell)))}{|L'|} \times \frac{|L'|}{\text{MAX}} \\ &= \frac{\sum_{\ell \in L'} \text{TIME}(r(q, \ell), r(u, \ell)) \times (1 - \text{DIFFERENCE}(r(q, \ell), r(u, \ell)))}{\text{MAX}}, \end{aligned} \tag{8}$$

where L' is a set of locations that the query user p and user u have visited and $|L'|$ is the number of L'. MAX is the maximum $|L'|$ among all users in U_q. We divide $|L'|$ by MAX as weight to distinguish a user whose number of common visited locations is higher than that of others. The higher number of common visited locations of user u implies that the user q is more likely to visit a location if user u visited it before.

The function TIME$(r(q, \ell), r(u, \ell))$ denoted as Eq. (9) describes how well the users q and u match the visiting time of a location ℓ. For the query user q, we count the number of times visiting a location ℓ in the same time period for the two users q and u, denoted as Matched$(r(q,\ell), r(u,\ell))$. Next, we divide Matched$(r(q,\ell), r(u,\ell))$ by $r_q.count$. The $r_q.count$ is the total number of a location ℓ visited by the user q. For the user u, we compute Matched$(r(u,\ell), r(q,\ell))$ and divide by $r_u.count$ in the same way.

$$\text{TIME}(r(q,\ell), r(u,\ell)) = \frac{\text{Matched}(r(q,\ell), r(u,\ell))}{r_q.count} \times \frac{\text{Matched}(r(u,\ell), r(q,\ell))}{r_u.count} \quad (9)$$

Besides, the function DIFFERENCE$(r(q,\ell), r(u,\ell))$ is a distance metric to calculate the difference between the query user q and the user u. We can determine whether the query user q likes to visit a location ℓ as well as user u. The difference value is between 0 to 1. A smaller value indicates that users q and u are similar. We convert a smaller value to a bigger one and use 1 minus the value. For example, we change the difference value from 0.1 to 0.9. Finally, an arithmetic mean of the total multiplying function TIME and 1 − DIFFERENCE of the ratings by the users q and u for every location $\ell \in L'$ ($L' = L_{q_{\text{visited}}} \cap L_{u_{\text{visited}}}$) is calculated the similarity measure.

$$\text{DIFFERENCE}(r(q,\ell), r(u,\ell)) = \left|\frac{r_q.count}{r_q.\text{MAX}} - \frac{r_u.count}{r_u.\text{MAX}}\right| \quad (10)$$

Furthermore, a set of similar users S_q ($\subset U_q$) for the query user q is the top m users whose similarity scores similarity(q, u_i), $1 \le i \le m$, are sorted in ascending order. The similar users to user q are defined as follows:

$$S_q = \{u_1, u_2, \ldots, u_m\} \quad (11)$$

For example, we compute the ratings of a query user q in Fig. 3. Table 2 lists the ratings of the user q and others. Next, we can measure who the user q is similar to and determine the top m similar users whose similarity scores are top m. Given the users q and u_3 as a instance, both users visited the locations ℓ_1 and ℓ_2. The $r(q,\ell_1)$ of query user q is $(1.0, 0, 3)$ which indicates $\frac{count}{\text{MAX}} = 1.0$, $|\text{am}(d.t)| = 0$, and $|\text{pm}(d.t)| = 3$. Moreover, the $r(u_3,\ell_1)$ of user u_3 is $\frac{count}{\text{MAX}} = 0.666$, $|\text{am}(d.t)| = 0$, and $|\text{pm}(d.t)| = 2$. The number of common visited locations of two users is $|L'| = 2$. The MAX denotes the maximum $|L'|$ among all users; namely, the maximum number of common visited locations with the users q and u_1 is 3. Next, we compute the output of function TIME() for a given input $r(q,\ell_1), r(u_3,\ell_1)$. The user q visited the location ℓ_1 three times and another u_3 visited one twice. Moreover, both users visited at the same time period. As a result, the output of function TIME() is ${}^3/_3 \times {}^2/_2$. We subsequently determine the function DIFFERENCE() between the users q and u_3. The DIFFERENCE$(r(q,\ell_1), r(u_3,\ell_1))$ is $|1.0 - 0.666|$. Likewise, we compute the functions TIME() and DIFFERENCE() of location ℓ_2. To summarize, the similarity(q, u_3) is $\frac{({}^3/_3 \times {}^2/_2)\times(1-|1.0-0.666|)+({}^1/_1 \times {}^2/_3)\times(1-|0.333-1.0|)}{2} \times \frac{2}{3} = \frac{1\times 0.666 + 0.666\times 0.333}{3} = \frac{0.666+0.222}{3} = 0.296$. In the following list, the similar users $U_q = \{u_1, u_2, u_3, \ldots\}$ are in similarity-score order.

4.3.3 User-preferred location scores

We predict the unvisited locations $L_{\text{unvisited}}$ for a query user q based on the locations that are previously visited by similar users S_q. Hence, we propose user-preferred location scores calculated by Eq. (12) to evaluate the unvisited locations $L_{\text{unvisited}}$ of user q. Given the similarity scores set of S_q, we calculate the ratio of each similar user s of query user q who has visited location ℓ. The function Visited(s,ℓ) presents whether the similar user s visited a location ℓ. Therefore, the function Visited(s,ℓ) returns 1 if the user s visited the location ℓ.

Table 2 An example of user-based collaborative filtering with time preference

	L'			Similarity(q, u_i)
	ℓ_1	ℓ_2	ℓ_3	
q	(1.0, 0, 3)	(0.333, 0, 1)	(0.333, 1, 0)	1.0
u_1	(1.0, 4, 6)	(0.5, 2, 3)	(0.2, 1, 1)	0.510
u_2	(0.4, 1, 1)		(1.0, 4, 1)	0.443
u_3	(0.666, 0, 2)	(1.0, 1, 2)		0.296
⋮	⋮	⋮	⋮	⋮

Table 3 An example of user-preferred location scores

	L'			$L_{\text{unvisited}}$			Similarity(q, u_i)
	ℓ_1	ℓ_2	ℓ_3	ℓ_4	ℓ_5	…	
q	(1.0, 0, 3)	(0.333, 0, 1)	(0.333, 1, 0)			…	1.0
u_1	(1.0, 4, 6)	(0.5, 2, 3)	(0.2, 1, 1)		(1.0, 2, 8)	…	0.510
u_2	(0.4, 1, 1)		(1.0, 4, 1)	(0.8, 3, 1)	(0.2, 1, 0)	…	0.443
u_3	(0.666, 0, 2)	(1.0, 1, 2)				…	0.296
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮

Otherwise, the function Visited(s, ℓ) returns 0. We take PS(q, ℓ) as a user-preferred location score as the basis of recommending a location ℓ for the user q when planning a personalized itinerary.

$$\mathrm{PS}(q, \ell) = \frac{\sum_{s \in S_q} \mathrm{Visited}(s, \ell) \times \mathrm{similarity}(q, s)}{\sum_{s \in S_q} \mathrm{similarity}(q, s)} \tag{12}$$

As mentioned in Table 2, we show an example to compute the user-preferred location scores based on someone's similar users. In Table 3, a query user q never visits a location ℓ_4. We measure the satisfaction of location ℓ_4 for the user q. We can obtain a set of similar users $S_q = \{u_1, u_2\}$ and $m = 2$. The user u_1 does not visit the location ℓ_4 and similarity$(q, u_1) = 0.510$. Nevertheless, the user u_2 visited the location ℓ_4 and similarity$(q, u_2) = 0.443$. The user-preferred location scores of query user qPS(q, ℓ_4) equal ${}^{0\times0.510+1\times0.443}/_{0.510+0.443} = 0.464$. Similarly, PS$(q, \ell_5)$ equals 1.0. In consequence, we suppose the location ℓ_5 is better than ℓ_4 to recommend to user q.

5 The online phase: trip planning

We propose a pirT algorithm to recommend the top k itineraries which satisfy the travel behaviors of users based on geographical features, user preferences, and travel duration in this section. The pirT consists of the top-k A^* search-based pirT and re-ranking itinerary candidate algorithms. As described in Sect. 3, a graph is constructed and traversed by depth-first search (short for DFS) before planning the personalized itinerary recommendation. First, we use the user-preferred locations with scores from the data preprocessing phase to provide the nodes and the scores of nodes in the graph. Note that those user-preferred locations have never been traveled by the query user. Next, we use the f_{duration} from the geographical features analysis in Sect. 4 and assign the f_{duration} as a weight (i.e., staying + transit time) for each edge to find the proper transit duration in the graph. Finally, we use a beginning location and a destination with a starting and ending time specified by the query user to schedule a travel path from the beginning location to the destination.

Algorithm 1 Top-k A^* search-based recommendation algorithm

Require: (a) *graph*: planning the personalized itinerary in graph, (b) top-k: the first top k personalized itinerary that a query user specifies, (c) s: a beginning location with starting time, and (d) e: a destination location with ending time.

Ensure: The top k personalized itineraries having the top k highest scores

```
1: candidate_set = Ø;
2: traversal_stack.push(path(s));
3: trip_time = e.t - s.t;
4: s.g = 0;
5: s.h = heuristic_estimate(s, e);
6: while traversal_stack ≠ empty do
7:    path = traversal_stack.pop();
8:    current = get the last node in path;
9:    for all neighbor connecting with current in graph do
10:      if neighbor in path then
11:         continue;
12:      end if
13:      neighbor.g = current.g + f_duration(current, neighbor);
14:      neighbor.h = heuristic_estimate(neighbor, e);
15:      if neighbor.f > travel_time then
16:         continue;
17:      end if
18:      Add neighbor to path;
19:      if neighbor = e then
20:         Add path to candidate_set;
21:      else
22:         traversal_stack.push(path);
23:      end if
24:   end for
25: end while
26: return Re-ranking(top-k, candidate_set);
```

5.1 Top-k A^* search-based recommendation algorithm

According to the heuristic function of the A^* algorithm, we have developed the top-k A^* search-based recommendation algorithm. We utilize the heuristic function to reduce the number of times to traverse the graph for the trip planning. In the heuristic function, we use the trip duration instead of the distance. In Lines 9-24 of Algorithm 1, we calculate the heuristic value from the current traversal node to the destination. The heuristic value is the trip duration calculated by the driving distance using the Haversine formula by the average speed from the current traversal node to the destination. Based on the above steps, the minimum trip duration of each path from the beginning location to the destination is calculated. If the trip duration is larger than the user-specified travel time, we can prune the current traversal node. Algorithm 1 terminates when the system traverses all the paths called itinerary candidates to the destination and invokes the re-ranking itinerary candidate function to return the first top k personalized itinerary as a result.

Algorithm 2 Re-ranking Itinerary Candidate Algorithm

Require: (a) top-k: the first top k personalized itinerary that a query user specifies and (b) candidate_set: a set of personalized itineraries.
Ensure: The first top k personalized itineraries having the first top k highest scores
1: top-k_set = ∅;
2: **for all** $path$ in candidate_set **do**
3: $path.scores = \text{IPS}(path) + f_{visit}(path) + f_{order}(path)$;
4: Add the $path$ to top-k_set;
5: **end for**
6: **return** The first top k $path$ in the top-k_set having the first top k highest scores;

5.2 Re-ranking itinerary candidate algorithm

After finishing the top-k A^* search-based recommendation algorithm, the next step is to conduct the re-ranking itinerary candidate algorithm. In Line 3 of Algorithm 2, we mark the scores for each itinerary candidate by IPS(), $f_{\text{order}}()$, and $f_{\text{visit}}()$ metrics which are calculated in user-preferred location scores and geographical features analysis in Sect. 4, respectively. Finally, we select the first top k itineraries to be the personalized itinerary and return it to the query user.

6 Experiments

In this section, we evaluate the effectiveness of the proposed *pirT* framework and compare them with the *PTR* [13] framework. We use the Gowalla dataset [4] of the USA as illustrated in Fig. 4. This US dataset contains 3,669,330 check-in data, 674,056 locations, and 54,552 users. We first describe the experiment settings constructed in a series of experimental evaluations. Compared with the PTR framework, the experimental results of our pirT framework are reported in this section. Second, we show the effectiveness of the personalized itinerary in terms of our user-based collaborative filtering with time preference (UTP) and similarity-based collaborative filtering (SCF) proposed in the PTR framework. Third, we evaluate the efficiency of planning personalized itineraries with our proposed pirT algorithm and compare it with Trip-Mine proposed in PTR. In our experiments, all algorithms are implemented in Java.

6.1 Experiment settings

In the experimental evaluations, we utilize the US dataset to avoid the sparsity problems when calculating user similarity. Therefore, we randomly pick 100 users as testing users whose number of check-in data was greater than 500 and who had visited New York, which is major city offering a large number of attractions, and which attracts many tourists. There are therefore many check-in data available for the training and testing purpose. We remove the GPS data of testing users who are in New York as a training dataset and the removed GPS data as a testing dataset. As described in Sect. 3, we use the training dataset as GPS data and perform each step of the off-line phase that contains density-based methods, geographical

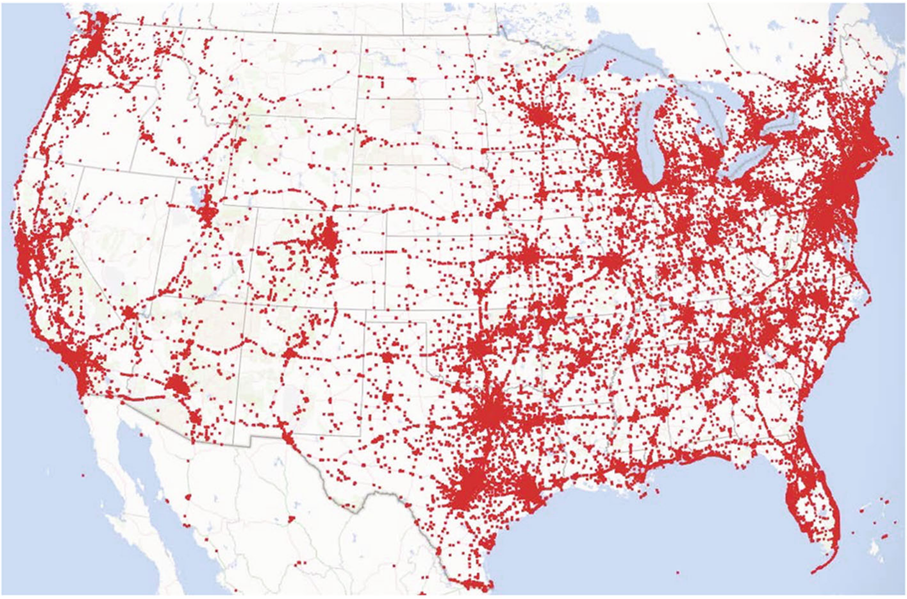


Fig. 4 Distribution of user check-in data in the USA in the Gowalla dataset

features, and user-preferred locations. We use the testing dataset to process the personalized itinerary recommendation steps of the online phase after data preprocessing. For planning the personalized itinerary, we retrieve a sequence of check-in locations of each testing user within one day as an itinerary. We pick the beginning location with the starting time and the destination with the ending time of the itinerary of the user as a query input to plan a personalized itinerary through the personalized itinerary recommendation steps.

Table 4 summarizes the default parameter settings in the following experimental evaluations. In addition, the parameters of MinPts, ε-neighborhood, and $PeriodDays$ are used in DBSCAN clustering method. Based on the historical statistics, we set $MinPts$ to 6, which is the minimum number of check-in points in a cluster to be tagged as a location.

Compared with the user-based collaborative filtering with time preference (UTP) of our paper and the similarity-based collaborative filtering (SCF) of PTR (short for PTR), we measure the effectiveness of itinerary prediction by employing $precision$ and $recall$. The measurements are defined in the confusion matrix as shown in Table 5. True positive (TP) is the number of correct positive predictions for instances which are in fact positive; false positive (FP) is the number of incorrect positive predictions for instances which are actually negative; false negative (FN) is the number of incorrect negative predictions for instances which are actually positive; and true negative (TN) is the number of correct negative predictions for instances which are in fact negative. Precision is a fraction of the number of the predicted locations that are visited by a user (true positives) among all recommended locations (true positives plus false positives) in the personalized itinerary, while recall is a fraction of the number of the predicted locations that are visited by a user among all locations that are actually visited by a user.

Table 4 Parameter settings in the experimental evaluations

Parameters	Default	Range	Description
$MinPts$	6	6 (Mean: 5.444)	The points in the cluster contain more than a minimum number of points ($MinPts$) that are within a specified range (ε-neighborhood)
ε-Neighborhood	100	50, 100, …, 300	
$PeriodDays$	7	7	A group becomes a location if $\exists p_i.t - p_j.t$ in the group is greater than $PeriodDays$ day(s)
m	60	50, 60, …, 100	The number of similar users
n	10	10	The number of personalized itineraries

Table 5 Precision and recall

		Actual results	
		True	False
Predicted results	True	True positive (TP)	False positive (FP)
	False	False negative (FN)	True negative (TN)

The precision at each location of the kth recommended personalized itinerary is calculated as $precision_k$ in the following equations:

$$precision_k = \frac{TP_k}{TP_k + FP_k}, \tag{13}$$

where TP_k represents the number of the predicted locations that are actually visited by the user and FP_k represents the number of the predicted locations that are not visited by the user in the kth recommended personalized itinerary. Moreover, we adopt the arithmetic mean denoted as (14) to determine the precision of n itineraries planned by the pirT and PTR frameworks.

$$\text{mean} = \frac{1}{n} \times \sum_{k=1}^{n} precision_k \tag{14}$$

In the same way, we calculate $recall_k$ and the arithmetic mean of recall denoted Eq. (15), where TP_k represents the number of the predicted locations that are actually visited by the user in the kth recommended personalized itinerary and FN_k represents the number of locations that are actually visited by the user but are not predicted by the system.

$$recall_k = \frac{TP_k}{TP_k + FN_k}$$
$$\text{mean} = \frac{1}{n} \times \sum_{k=1}^{n} recall_k \tag{15}$$

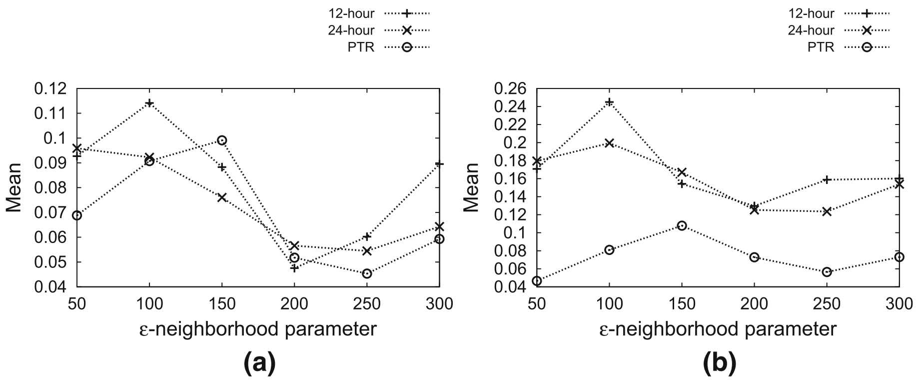


Fig. 5 The ε-neighborhood parameter. **a** Precision, **b** recall

In the following sections, we provided two experiments for verifying the effectiveness of our proposed user-based collaborative filtering with time preference (UTP) in pirT framework (pirT for short) and compared the outcomes with the similarity-based collaborative filtering (SCF) of PTR framework (short for PTR).

6.2 Impact of the ε-neighborhood parameter

The experiment shows that the effectiveness of pirT and PTR by varying the ε-neighborhood parameter to 50, 100, 150, 200, 250, and 300 meters, respectively. In addition, we plan the top ten personalized itineraries recommended by the top 60 similar users. Figure 5a presents that our pirT outperforms PTR except 150-neighborhood parameter on precision. Furthermore, as shown in Fig. 5b, pirT is better than PTR on recall. The reason is that we consider the number of a locations marked by the query users and time preference factors visited by them at a certain time period. However, PTR does not take account of the time preference factors. Finally, the means of precision and recall is summarized in Table 6a, b.

6.3 Impact of the number of similar users

The experiment reports the effectiveness of our pirT and PTR when the number of top similar users is assigned as 50, 60, 70, 80, 90, and 100, respectively. We also find the top ten personalized itineraries on GPS data performed by DBSCAN clustering method with 100-neighborhood parameter. We verify whether the number of top similar users affects the accuracy of the recommendations.

Figure 6a, b depicts the result of precision and recall. The results show that our pirT can almost higher than PTR because PTR does not support time preference factors. Moreover, Table 7a, b gives the data of means of precision and recall.

6.4 Efficiency of trip planning

We conducted an evaluation to compare the efficiency of pirT with PTR. In Table 8, we observe that the CPU time decreases significantly when the ε-neighborhood increases for both approaches. The reason is that the number of locations in New York is less when ε-neighborhood is large when performing the density-based cluster analysis. In summary, when ε-neighborhood is small, pirT achieves better results than PTR because pirT prunes more

Table 6 The ε-neighborhood parameter

	pirT		PTR
	12-h	24-h	
(a) Precision			
50	0.092696	0.095896	0.064316
100	0.114146	0.092229	0.090751
150	0.088286	0.076026	0.099088
200	0.047529	0.05661	0.051752
250	0.060217	0.054491	0.045344
300	0.089513	0.064316	0.059307
(b) Recall			
50	0.170806	0.179641	0.046562
100	0.244987	0.19941	0.080957
150	0.154237	0.167059	0.107837
200	0.129649	0.125238	0.072762
250	0.158779	0.123558	0.056456
300	0.160155	0.153836	0.073025

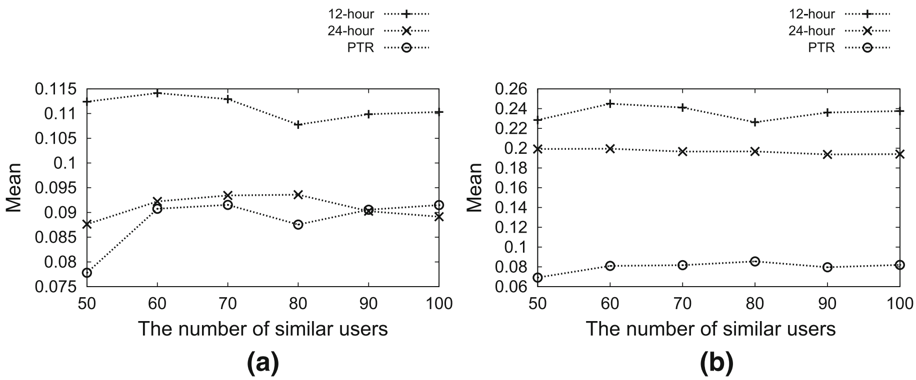


Fig. 6 The number of similar users. **a** Precision, **b** recall

itineraries than PTR when planning the personalized itinerary recommendation. When ε-neighborhood is larger than 200, the performances of both approaches are not significantly different.

7 Conclusions

In this paper, we have proposed a novel framework termed personalized itinerary recommendation with time constraints (pirT) to plan a personalized itinerary for tourists. We consider the trip duration, the geographical features, and the user travel behaviors in planning personalized itineraries. In pirT, we designed a user-based collaborative filtering with time preference (UTP) and the top-k A^* search-based recommendation with a re-ranking algorithm. The UTP for users aims to find other users whose travel behaviors are similar to theirs, and the top-k A^* search-based recommendation with re-ranking algorithm focuses on planning the person-

Table 7 The number of similar users

	pirT		PTR
	12-h	24-h	
(a) Precision			
50	0.112406	0.087628	0.077807
60	0.114146	0.092229	0.090751
70	0.11292	0.093446	0.091536
80	0.107759	0.09359	0.087547
90	0.109871	0.090252	0.090556
100	0.110321	0.089142	0.09149
(b) Recall			
50	0.228524	0.199194	0.069201
60	0.244987	0.19941	0.080957
70	0.241087	0.19655	0.081692
80	0.226231	0.196713	0.085513
90	0.236044	0.193621	0.079592
100	0.237576	0.193989	0.081971

Table 8 My caption

	pirT (ms)	PTR (ms)
20	61,679	320,298
100	1504	2506
200	31.19	31

alized itinerary. Through comprehensive experimentation, we have validated the proposed pirT, comparing with the PTR framework in terms of effectiveness.

Acknowledgements Funding was provided by Ministry of Science and Technology, Taiwan (Grant No. MOST 106-2221-E-194 -047 -MY2).

References

1. Adomavicius G, Tuzhilin A (2005) Toward the next generation of recommender systems: a survey of the state-of-the-art and possible extensions. IEEE Trans Knowl Data Eng 17(6):734–749
2. Bao J, Zheng Y, Mokbel MF (2012) Location-based and preference-aware recommendation using sparse geo-social networking data. In: Proceedings of the 20th international conference on advances in geographic information systems. SIGSPATIAL '12. ACM, New York, NY, USA, pp 199–208. https://doi.org/10.1145/2424321.2424348
3. Chen H, Ku W-S, Sun M-T, Zimmermann R (2008) The multi-rule partial sequenced route query. In: Proceedings of the 16th ACM SIGSPATIAL international conference on Advances in geographic information systems. GIS '08. ACM, New York, NY, USA, pp. 10:1–10:10. https://doi.org/10.1145/1463434.1463448
4. Cho E, Myers SA, Leskovec J (2011) Friendship and mobility: user movement in location-based social networks. In: Proceedings of the 17th ACM SIGKDD international conference on knowledge discovery and data mining. KDD '11. ACM, New York, NY, USA, pp. 1082–1090. https://doi.org/10.1145/2020408.2020579
5. Cormen TH, Leiserson CE, Rivest RL, Stein C (2009) Introduction to algorithms, 3rd edn. The MIT Press, Cambridge

6. Doytsher Y, Galon B, Kanza Y (2011) Storing routes in socio-spatial networks and supporting social-based route recommendation. In: Proceedings of the 3rd ACM SIGSPATIAL international workshop on location-based social networks. LBSN '11. ACM, New York, NY, USA, pp. 49–56. https://doi.org/10.1145/2063212.2063219
7. Ester M, Kriegel H-P, Sander J, Xu X (1996) A density-based algorithm for discovering clusters in large spatial databases with noise. In: Proceedings of the second international conference on knowledge discovery and data mining, pp 226–231
8. Han J, Kamber M, Pei J (2011) Data mining: concepts and techniques, 3rd edn. Morgan Kaufmann Publishers Inc., San Francisco
9. Hsieh H-P, Li C-T, Lin S-D (2012a) Exploiting large-scale check-in data to recommend time-sensitive routes. In: Proceedings of the ACM SIGKDD international workshop on urban computing. UrbComp '12. ACM, New York, NY, USA, pp 55–62. https://doi.org/10.1145/2346496.2346506
10. Hsieh H-P, Li C-T, Lin S-D (2012b) Triprec: recommending trip routes from large scale check-in data. In: Proceedings of the 21st international conference companion on World Wide Web. WWW '12 Companion. ACM, New York, NY, USA, pp 529–530. https://doi.org/10.1145/2187980.2188111
11. Kim B, Lee Y, Lee S, Rhee Y, Song J (2011) Towards trajectory-based experience sharing in a city. In: Proceedings of the 3rd ACM SIGSPATIAL International workshop on location-based social networks. LBSN '11. ACM, New York, NY, USA, pp 85–88. https://doi.org/10.1145/2063212.2063221
12. Li F, Cheng D, Hadjieleftheriou M, Kollios G, Teng S-H (2005) On trip planning queries in spatial databases. In: Proceedings of the 9th international conference on advances in spatial and temporal databases. SSTD'05, pp 273–290
13. Lu EH-C, Chen C-Y, Tseng VS (2012) Personalized trip recommendation with multiple constraints by mining user check-in behaviors. In: Proceedings of the 20th international conference on advances in geographic information systems. SIGSPATIAL '12. ACM, New York, NY, USA, pp 209–218. https://doi.org/10.1145/2424321.2424349
14. Lu EH-C, Lin C-Y, Tseng VS (2011) Trip-mine: an efficient trip planning approach with travel time constraints. In: Proceedings of the 2011 IEEE 12th international conference on mobile data management—volume 01. MDM '11. IEEE Computer Society, Washington, DC, USA, pp 152–161. https://doi.org/10.1109/MDM.2011.13
15. Ma X, Shekhar S, Xiong H, Zhang P (2006) Exploiting a page-level upper bound for multi-type nearest neighbor queries. In: Proceedings of the 14th annual ACM international symposium on advances in geographic information systems. GIS '06. ACM, New York, NY, USA, pp 179–186. https://doi.org/10.1145/1183471.1183501
16. Sharifzadeh M, Kolahdouzan M, Shahabi C (2008) The optimal sequenced route query. VLDB J 17(4):765–787. https://doi.org/10.1007/s00778-006-0038-6
17. Ye M, Yin P, Lee W-C (2010) Location recommendation for location-based social networks. In: Proceedings of the 18th SIGSPATIAL international conference on advances in geographic information systems. GIS '10. ACM, New York, NY, USA, pp 458–461. https://doi.org/10.1145/1869790.1869861
18. Ying JJ-C, Lee W-C, Ye M, Chen C-Y, Tseng VS (2011) User association analysis of locales on location based social networks. In: Proceedings of the 3rd ACM SIGSPATIAL international workshop on location-based social networks. LBSN '11. ACM, New York, NY, USA, pp 69–76. https://doi.org/10.1145/2063212.2063214
19. Yoon H, Zheng Y, Xie X, Woo W (2010) Smart itinerary recommendation based on user-generated gps trajectories. In: Proceedings of the 7th international conference on Ubiquitous intelligence and computing. UIC'10. Springer, Berlin, pp 19–34. http://dl.acm.org/citation.cfm?id=1929661.1929669
20. Zheng Y, Xie X (2011) Learning travel recommendations from user-generated gps traces. ACM Trans Intell Syst Technol 2(1):2:1–2:29. https://doi.org/10.1145/1889681.1889683
21. Zheng Y, Zhang L, Ma Z, Xie X, Ma W-Y (2011) Recommending friends and locations based on individual location history. ACM Trans Web 5(1):5:1–5:44. https://doi.org/10.1145/1921591.1921596
22. Zheng Y, Zhang L, Xie X, Ma W-Y (2009) Mining interesting locations and travel sequences from GPS trajectories. In: Proceedings of the 18th international conference on World wide web. WWW '09. ACM, New York, NY, USA, pp 791–800. https://doi.org/10.1145/1526709.1526816

Yu-Ling Hsueh is an Associate Professor with the Department of Computer Science and Information Engineering at the National Chung Cheng University, Taiwan, since 2011. Hsueh received her M.S. and Ph.D. degrees in computer science from the University of Southern California in 2003 and 2009, respectively. Her research interests are data analysis, data mining, spatiotemporal databases, mobile data management, scalable continuous query processing, and spatial data indexing.

Hong-Min Huang received his master degree in Computer Science and Information Engineering from the National Chung Cheng University, Taiwan, in 2013. His research interests include data analysis, data mining, and spatiotemporal databases.