Chapter 9
Location-Based Social Networks: Locations

Yu Zheng and Xing Xie

Abstract While chapter 8 studies the research philosophy behind a location-based social network (LBSN) from the point of view of users, this chapter gradually explores the research into LBSNs from the perspective of locations. A series of research topics are presented, with respect to mining the collective social knowledge from many users' GPS trajectories to facilitate travel. On the one hand, the generic travel recommendations provide a user with the most interesting locations, travel sequences, and travel experts in a region, as well as an effective itinerary conditioned by a user's starting location and an available time length. On the other hand, the personalized travel recommendations find the locations matching an individual's interests, which can be learned from the individual's historical data.

9.1 Introduction

The increasing availability of location-acquisition technologies and Internet access in mobile devices is fostering a variety of location-based services generating a myriad of spatio-temporal data, especially in the form of trajectories [45, 42, 41, 6]. These trajectories reflect the behavior and interests of users, thereby enabling us to better understand an individual and the similarity between different individuals [20, 34, 7, 15, 36]. Research and applications were introduced in Chapter 8 in which the users are the focus and locations are employed as enhanced information for better understanding them. Instead, this chapter discusses the research topics that aim at understanding locations based upon the collective social knowledge of users (e.g., the knowledge contained in their GPS trajectories) starting with generic travel recommendations [48, 44, 37, 38, 40] and then looking at personalized recommendations [46, 43, 39, 13, 33].

Yu Zheng · Xing Xie
Microsoft Research Asia, China
e-mail: {yuzheng,xing.xie}@microsoft.com

Regardless of an individual's preferences,the generic travel recommender systems mine a vast number of trajectories (generated by multiple users) and provide an individual with travel recommendations following a paradigm of "trajectories → interesting locations → popular travel sequences → itinerary planning → activities recommendation." Specifically, these recommender systems first infer the most interesting locations in a region from the given trajectories, and then detect the popular travel sequences among these locations [48]. An interesting location is defined as a culturally important place, such as Tiananmen Square in Beijing or the Statue of Liberty in New York (i.e., popular tourist destinations), and commonly frequented public areas, such as shopping malls/streets, restaurants, cinemas, and bars. With these interesting locations and travel sequences, an ideal itinerary can be planned for a user according to her departure location, destination, and available time [37, 38]. Finally, the generic travel recommendations provide users with some popular activities, e.g., dinning and shopping, that could be performed in a location [40]. All these recommendations mentioned above facilitate a user to travel to an unfamiliar place and plan a journey with minimal effort.

However, the personalized recommender systems learn an individual's interests from her personal location data (e.g., GPS trajectories) and suggest locations to the individual matching her preferences. Specifically, the personalized recommender uses the times that a particular individual has visited a location as her implicit ratings on that location, and estimates an individual's interests in unvisited places by considering her location history and those of other users [46, 44]. As a result, some locations with high ratings that might match the user's tastes can be recommended.

Two collaborative filtering (CF) models are individually used to infer a user's ratings of these unvisited locations. First, the personalized location recommendation is equipped with a user-based CF model, which employs user similarity introduced in Section 8.3 as a distance function between different users [46]. This model is able to capture people's mobility, such as the sequential and hierarchical properties of human movement in the physical world, while suffering from poor scalability caused by the heavy computation of user similarity. This user-based CF model is detailed in Section 9.3.2. Second, to address the problem of scalability, a location-based CF model is proposed [44]. This model uses the correlation between locations mined from many users' GPS traces [43] as a distance measure between two different locations. The location-based CF model is slightly less effective than the user-based one while being much more efficient. Refer to Section 9.3.3 for details.

9.2 Generic Travel Recommendations

This section describes the generic travel recommendation following the paradigm of "trajectories → interesting locations → popular travel sequences → itinerary planning → activities recommendation." Specifically, Section 9.2.1 introduces the detection of interest locations and travel sequences [48]. Section 9.2.2 then presents

itinerary recommendation [37, 38]. Finally, a location-activity recommender [40] is discussed in Section 9.2.3.

9.2.1 Mining Interesting Locations and Travel Sequences

9.2.1.1 Background

Traveling to an unfamiliar city or region, people usually like to know the most interesting locations and the most popular travel sequences. In fact, this kind of information evolves as time goes by and varies in quite a few factors, such as time of day, day of the week, and the seasons. For example, the Forbidden City was the most popular tourist attraction in the urban area of Beijing before 2008. However, it has recently been replaced by the Olympic Park of Beijing. Locals particularly enjoy the Olympic Park on weekend evenings during the summer. Other popular destinations include Houhai Bar Street for sightseeing in the daytime and drinking in the evening, or some newly-built movie theatres offering half-price tickets every Tuesday night. Note that the interesting places do not only include tourist attractions but also restaurants and shopping malls popular among residents. Consequently, travel agencies and travel books cannot always provide the latest and most effective recommendations that a user needs.

In order to deal with this dilemma, it is necessary to gather travel recommendations automatically and in a timely manner from social media such as the vast amount of GPS trajectories generated by the large number of users travelling in a city. GPS trajectories can be formulated in terms of users' geo-tagged photos and check-in records, or obtained from some trajectory-sharing social networking services like GeoLife [45, 42, 41]. A number of studies [23, 2] have introduced the methods for extracting trips from geo-tagged photos, and some professionals have explored the idea of mining generic travel recommendations from GPS trajectories [48, 5, 44]. Particularly, one paper [48] first proposed a learning model to infer the most interesting locations in a city as well as the popular travel sequences among these locations, followed by a few expanded studies reported in [5, 44]. The mined locations and travel sequences are used to enable a generic travel recommender illustrated in Fig. 9.1 and Fig. 9.2.

Figure 9.1 illustrates the user interface of a generic travel recommender running on desktop computers. The right column shows the top five most interesting locations and the five most experienced users in the region (specified by the present view of the map). The top five most popular travel sequences within this region are also displayed on the map. By zooming in/out and panning, a user can retrieve such results within any region. In addition, the photos taken in an interesting location will be presented on the bottom of the window after a user clicks the icon representing the location on the map.

As shown in Fig. 9.2, a user with a GPS-phone can find the top five most interesting locations as well as the five most popular sequences near her present geographic

position (denoted as the red star). Additionally, when the user reaches a location, the recommender system will provide her with a further suggestion by presenting the top three most popular sequences starting from this location.

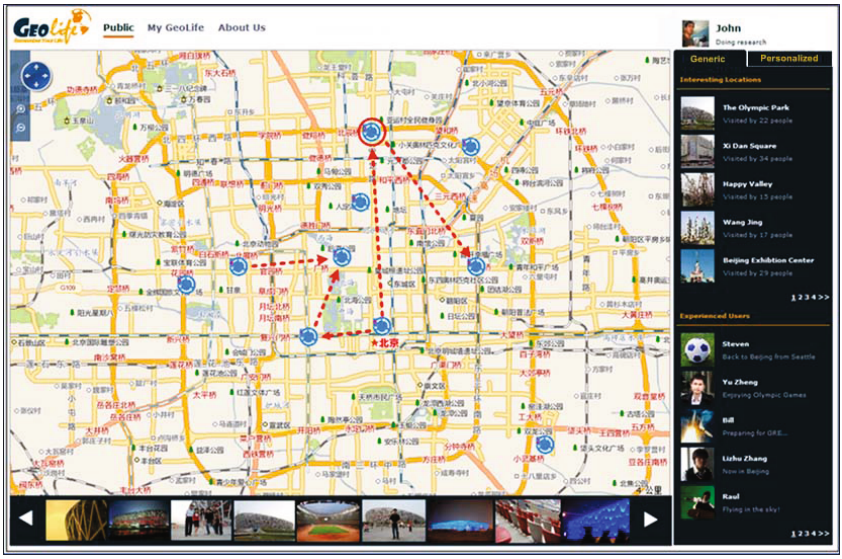


Fig. 9.1 The user interface of a generic location recommender

Fig. 9.2 Location recommendations on a GPS-phone

However, we will be faced with some challenges when conducting the generic recommendations. The first is to determine the interest level of a location. Intrinsically, the interest level of a location does not only depend on the number of users visiting this location but also on these users' travel experiences (knowledge). Intuitively, different people have different degrees of knowledge about a geospatial region. During a journey, the users with more travel experience of a region would be

more likely to visit interesting locations in that region. For example, the residents of Beijing are more capable than overseas tourists of finding high quality restaurants and shopping malls in Beijing. If we do not consider the travel knowledge of a user, the hot spots like railway stations and airports will be most recommended. Second, an individual's travel experience and interest level of a lo-cation are relative values (i.e., it is not reasonable to judge whether or not a location is interesting), and are region-related (i.e., conditioned by the given geospatial region). An individual who has visited many places in New York might have no idea about Beijing. Likewise, the most interesting restaurant in a district of a city might not be the most interesting one in the whole city (as restaurants from other districts might outperform it).

9.2.1.2 Methodology for Mining Interesting Locations

To address the above challenges, the location histories of users are first modeled with a tree-based hierarchical graph (TBHG) according to the following two steps demonstrated in Fig. 9.3.

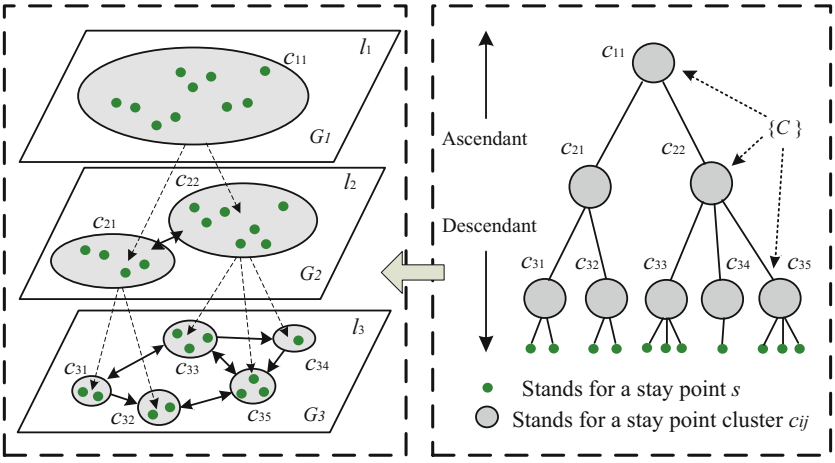


Fig. 9.3 Building a tree-based hierarchical graph

1) Formulate a shared hierarchical framework F: This step is the same as that presented in Section 8.2.2.2. That is, the stay points detected from users' GPS logs are put into a dataset, and then hierarchically clustered into geospatial regions using a density-based clustering algorithm in a divisive manner. As a consequence, the similar stay points from various users would be assigned to the same clusters on different levels. Here, a stay point stands for a location where a user stayed for a certain period of time, formally defined in Definition 8.3.

2) Build location graphs on each layer: Based on shared framework F and users' location histories, the clusters on the same level are connected with directed edges. If two consecutive stay points from one trip are individually contained in two clusters, a link is generated between the two clusters in a chronological direction according to the time serial of the two stay points. Note that different from the third step of modeling an individual's location history (introduced in Section 8.2.1), this step feeds all users' location histories (sequences of stay points) into the shared framework. Therefore, this tree-based hierarchical graph models the location history of all users in a location-based social networking service.

Then, a HITS(Hypertext Induced Topic Search)-based inference model is proposed with the TBHG. This inference model regards an individual's access to a location as a directed link from the user to that location. This model infers two values, the interest level of a location and a user's travel experience, by taking into account 1) the mutuallly reinforcing relationship between the two values and 2) the geo-regional conditions. See details in the following paragraphs.

Concept of HITS model: HITS stands for hypertext induced topic search [17], which is a search-query-dependent ranking algorithm for Web information retrieval. When the user enters a search query, HITS first expands the list of relevant pages returned by a search engine and then produces two rankings for the expanded set of pages, authority ranking and hub ranking. For every page in the expanded set, HITS assigns them an authority score and a hub score. As shown in Fig. 9.4, an authority is a Web page with many inlinks, and a hub is a page with many out-links. The key idea of HITS is that a good hub points to many good authorities, and a good authority is pointed to by many good hubs. Thus, authorities and hubs have a mutually reinforcing relationship. More specifically, a page's authority score is the sum of the hub scores of the pages it points to, and its hub score is the integration of authority scores of the pages pointed to by it. Using a power iteration method, the authority and hub scores of each page can be calculated. The main strength of HITS is ranking pages according to the query topic, which may provide more relevant authority and hub pages. However, HITS requires time consuming operations, such as online expanding page sets and calculating the hub and authority scores.

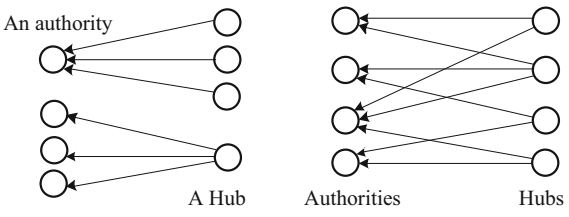


Fig. 9.4 The basic concept of HITS model

Mutually reinforcing relationship: Using the third level of the TBHG shown in Fig. 9.3 as an example, Fig. 9.5 illustrates the main idea of the HITS-based in-

ference model. Here, a location is a cluster of stay points, like c_{31} and c_{32}. This model regards an individual's visit to a location as an implicitly directed link from the individual to that location. For instance, cluster c_{31} contains two stay points respectively detected from u_1's and u_2's GPS traces, i.e., both u_1 and u_2 have visited this location. Thus, two directed links are generated respectively to point to c_{31} from u_1 and u_2. Similar to HITS, in this model, a hub is a user who has accessed many places, and an authority is a location which has been visited by many users. Intuitively, a user with rich travel experience (knowledge) in a region is able to visit many interesting places in that region, and a very interesting place in that region could be accessed by many users with rich travel experiences. Therefore, users' travel experiences (hub scores) and the interest level of locations (authority scores) have a mutually reinforcing relationship. More specifically, a user's travel experience is represented by the sum of the interest values of the locations that the user has been to, and the interest value of a location is denoted by the sum of the experiences of users who have visited this location. For simplicity's sake, in the remainder of this chapter, a user with rich travel experience (i.e., relatively high hub score) in a region is called an experienced user of that region and a location that attracts people's profound interests (relatively high authority score) is denoted as an interesting location.

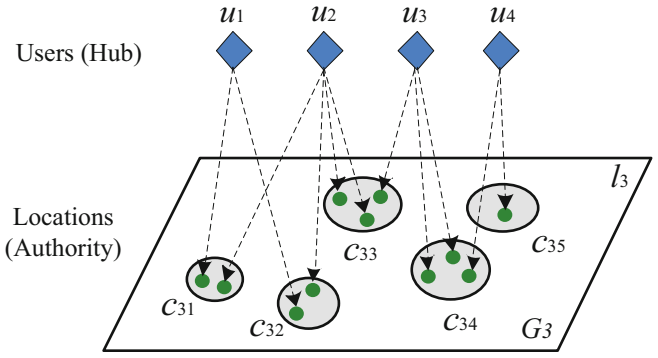


Fig. 9.5 The HITS-based inference model

Region-related: Intrinsically, a user's travel experience is region-related, i.e., a user who has a great deal of travel knowledge of a city might have no idea about another city. Also, an individual, who has visited many places in a particular part of a city might know little about another part of the city (especially if the city is very large, like New York). This concept is aligned with the query-dependent property of HITS. Thus, specifying a geospatial region (a topic query) and formulating a dataset that contains the locations in this region are needed for conducting the HITS-based inference model. However, an online data selection strategy (i.e., specifying a region based on an individual's input) will generate a great deal of resource-consuming operations, thereby diminishing the feasibility of our system. Therefore, a smart

data selection strategy should be considered to fit this region-related feature of the HITS-based inference model.

Strategy for Data Selection: Actually, on a TBHG, the shape of a graph node (cluster of stay points) provides an implicit region for its descendent nodes. These regions covered by the clusters on different levels of the hierarchy might stand for various semantic meanings, such as a city, a district, or a community. Therefore, the interest of every location can be calculated in advance using the regions specified by their ascendant clusters. In other words, a location might have multiple authority scores based on the different regions it falls in. Also, a user might have multiple hub scores conditioned by the regions of different clusters.

Definition 9.1 (Location Interest). In this system, the interest of a location (c_{ij}) is represented by a collection of authority scores $I_{ij} = \{I_{ij}^1, I_{ij}^2, \ldots, I_{ij}^l\}$. Here, I_{ij}^l denotes the authority score of cluster c_{ij} conditioned by its ascendant nodes on level l, where $1 \leq l < i$.

Definition 9.2 (User Travel Experience). In our system, a user's (e.g., u_k) travel experience is represented by a set of hub scores $e^k = \{e_{ij}^k \mid 1 \leq i < |L|, 1 \leq j \leq |C_i|\}$ (refer to Definition 8.5), where e_{ij}^k denotes u_k's hub score conditioned by region c_{ij}.

Figure 9.6 demonstrates these definitions. In the region specified by cluster c_{11}, it is possible to respectively calculate an authority score (I_{21}^1 and I_{22}^1) for clusters c_{21} and c_{22}. Meanwhile, within this region, the authority scores (I_{31}^1, I_{32}^1, I_{33}^1, I_{34}^1 and I_{35}^1) for clusters c_{31}, c_{32}, c_{33}, c_{34} and c_{35} can be inferred. Further, using the region specified by cluster c_{21}, we can also calculate authority scores (I_{31}^2 and I_{32}^2) for c_{31} and c_{32}. Likewise, the authority scores (I_{33}^2, I_{34}^2 and I_{35}^2) of c_{33}, c_{34} and c_{35} can be re-inferred within region c_{22}. Therefore, each cluster on the third level has two authority scores, which can be used on various occasions based on user inputs. For instance, as depicted in Fig. 9.6 A), when a user selects a region only covering locations c_{31} and c_{32}, the authority scores I_{31}^2 and I_{32}^2 can be used to rank these two locations. However, as illustrated in Fig. 9.6 B), if the region selected by a user covers the locations from two different parent clusters (c_{21} and c_{22}), the authority values I_{32}^1, I_{33}^1 and I_{34}^1 should be used to rank these locations.

A strategy that allows for multiple hub scores for a user and multiple authority scores for a location has two advantages. First, it is able to leverage the main strength of HITS to rank locations and users within the context of geospatial regions (query topics). Second, these hub and authority scores can be calculated offline, thereby ensuring the efficiency of a recommender system while allowing users to specify any region on a map.

Inference: Given the locations pertaining to the same ascendant cluster, we are able to build an adjacent matrix M between users and locations based on the users' visits to these locations. In this matrix, item v_{ij}^k stands for the times that u_k (a user) has visited to cluster c_{ij} (the jth cluster on the ith level). Such matrixes can be built offline for each non-leaf node. For instance, the matrix M formulated for the example shown in Fig. 9.5 can be represented as follows, where all five clusters pertain to c_{11}:

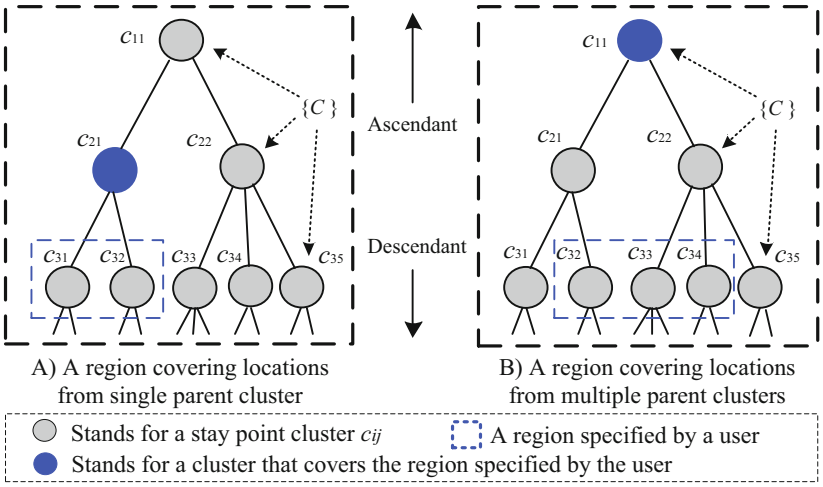


Fig. 9.6 Some cases demonstrating the data selection strategy

$$M = \begin{array}{c} \\ u_1 \\ u_2 \\ u_3 \\ u_4 \end{array} \begin{array}{c} \begin{array}{ccccc} c_{31} & c_{32} & c_{33} & c_{34} & c_{35} \end{array} \\ \begin{bmatrix} 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 2 & 0 & 0 \\ 0 & 0 & 1 & 2 & 0 \\ 0 & 0 & 0 & 1 & 1 \end{bmatrix} \end{array} \tag{9.1}$$

Then, the mutually reinforcing relationship of user travel experience e_{ij}^k and location interest I_{ij}^l is represented as follows:

$$l_{ij}^l = \sum_{u_k \in U} e_{lq}^k \times v_{ij}^k; \tag{9.2}$$

$$e_{lq}^k = \sum_{c_{ij} \in c_{lq}} v_{ij}^k \times l_{ij}^l; \tag{9.3}$$

where c_{lq} is c_{ij}'s ascendant node on the lth level, $1 \leq l < i$. For instance, as shown in Fig. 9.6, c_{31}'s ascendant node on the first level of the hierarchy is c_{11}, and its ascendant node on the second level is c_{21}. Thus, if $l = 2$, c_{lq} stands for c_{21} and $(c_{31}, c_{32}) \in c_{21}$. Also, if $l = 1$, c_{lq} denotes c_{11}, and $(c_{31}, c_{32}, \ldots, c_{35}) \in c_{11}$.

Writing them in matrix form, we use J to denote the column vector with all the authority scores, and use E to denote the column vector with all the hub scores. Conditioned by the region of cluster c_{11}, $J = (I_{31}^1, I_{32}^1, \ldots, I_{35}^1)$, and $E = (e_{11}^1, e_{11}^2, \ldots, e_{11}^4)$.

$$J = M^{\mathrm{J}} \cdot E \tag{9.4}$$

$$E = M \cdot J \tag{9.5}$$

If we use J_n and E_n to denote authority and hub scores at the nth iteration, the iterative processes for generating the final results are

$$J_n = M^{\mathrm{J}} \cdot M \cdot J_{n-1} \tag{9.6}$$

$$E_n = M \cdot M^{\mathrm{J}} \cdot E_{n-1} \tag{9.7}$$

Starting with $J_0 = E_0 = (1, 1, \ldots, 1)$, we are able to calculate the authority and hub scores using the power iteration method. Later, we can retrieve the top n most interesting locations and the top k most experienced users in a given region.

9.2.1.3 Methodology for detecting travel sequences

This step detects the top k most popular sequences of locations from the graph on a layer of the TBHG (refer to Fig. 9.3 for an example). A popularity score is calculated for each location sequence within a given region based upon two factors: the travel experiences of the users taking this sequence and the interests of the locations contained in the sequence. Since there would be multiple paths starting from the same location, the interest value of this location should be distributed to these paths according to the probability that users would take a path.

Figure 9.7 demonstrates the calculation of the popularity score for a 2-length sequence (i.e., a sequence containing two locations), $A \rightarrow C$. In this figure, the graph nodes $(A, B, C, D,$ and $E)$ stand for locations, and the graph edges denote people's transition sequences among them. The number associated with an edge represents how many times that users have taken the sequence. Eq. 9.8 computes the popularity score of sequence $A \rightarrow C$, consisting of contributions from the following three parts:

- The authority score of location $A(I_A)$ weighted by the probability of people leaving by this sequence (Out_{AC}). Clearly, there are seven (5+2) links pointing to other nodes from node A, and five out of seven of these links point directly to node C. So, $Out_{AC} = 5/7$, i.e., only five sevenths of location A's authority (I_A) should be propagated to sequence $A \rightarrow C$, and the rest of I_A should be distributed to $A \rightarrow B$.
- The authority score of location $C(I_C)$ weighted by the probability of people's entering by this sequence (In_{AC}).
- The hub scores of the users (U_{AC}) who have taken this sequence.

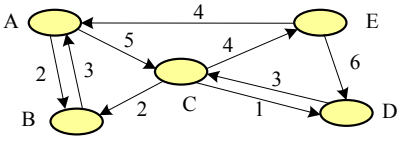


Fig. 9.7 Demonstration of mining popular travel sequences from a graph

$$
\begin{aligned}
S_{AC} &= \sum_{u_k \in U_{AC}} (I_A \cdot Out_{AC} + I_C \cdot In_{AC} + e^k) \\
&= |U_{AC}| \cdot (I_A \cdot Out_{AC} + I_C \cdot In_{AC}) + \sum_{u_k \in U_{AC}} e^k \\
&= 5 \times (\frac{5}{7} \times I_A + \frac{5}{8} I_C) + \sum_{u_k \in U_{AC}} e^k. \qquad (9.8)
\end{aligned}
$$

Following this method, the popularity score of sequence $C \to D$ is calculated as follows:

$$
S_{CD} = 1 \times \left(\frac{1}{7} \times I_C + \frac{1}{7} I_D \right) + \sum_{u_k \in U_{CD}} e^k. \qquad (9.9)
$$

Thus, the popularity score of sequence $A \to C \to D$ equals:

$$
S_{ACD} = S_{AC} + S_{CD} \qquad (9.10)
$$

The detection of popular travel sequences starts with computing the popularity score for each 2-length sequence, and then searches for 3-lenth sequences based on these 2-length sequences. Though searching for the top k n-length most popular sequences in a graph is time consuming, there are a few optimization methods using some upper bound to filter unnecessary search spaces. Moreover, the size of a location graph is usually small as the number of interesting locations in a city is limited. Meanwhile, as people do not normally travel to too many places during a trip, it is not necessary to provide a user with very long travel sequences. Sometimes, a sequence with three locations is more useful than longer ones.

9.2.2 Itinerary Recommendation

9.2.2.1 Background

The interesting locations and travel sequences mentioned above can facilitate travel to an unfamiliar place. However, people are still faced with particular challenges when planning their trips.

First, while there are many location candidates that can be considered, a traveler usually wants to maximize her travel experience, i.e., visit as many interesting locations as possible in a comfortable manner without wasting too much time traveling between locations. To achieve this task, the typical duration that people stay in a location and the average travel time between two locations should be considered. Second, an effective itinerary needs to adapt to a traveler's present location and available time length as well as her destination. Some popular travel routes can be available in a book but may not be feasible for a particular individual in the real

world because the attractions contained in these routes might be too far away, or the individual does not have enough time for the trip.

What a traveler needs is an effective itinerary, which can be adapted to the traveler's requests (consisting of a starting location, destination, and available time) and includes the information of not only a travel route passing some interesting locations but also the typical duration spent in each location and the general travel time between locations.

Itinerary recommendation has been studied in quite a few research projects. Some recommender systems [9, 3, 16] need a user's intervention when generating an itinerary for the user. For example, [9] presented an interactive travel itinerary planning system where a user defines which places to visit and avoid. Similarly, [3] reported on an interactive system where a user specifies general constraints, such as time and attractions to be included in the itinerary. The advantage of such interactive recommender systems is that the more a user knows about traveling in the area, the better the itinerary is. However, this assumption is not practical for most novice travelers who lack prior knowledge of a region.

To alleviate the human intervention and prior knowledge needed, [37, 38, 18, 8] presented relatively automated recommenders. Particularly, [37, 38] proposed a social itinerary recommendation service that generates an effective itinerary based on a user's query and social knowledge learned from user-generated GPS trajectories, with a major application scenario described as follows. Imagine that a researcher is attending a conference in Beijing. At the end of the conference, she has 8 hours to spend before catching her flight. She is a first time visitor to the city and has no idea how to plan an effective travel route, and is thereby relying on a social itinerary recommendation service. She is starting from her current location, which is automatically recognized with a GPS-enabled phone. She marks the Beijing Capital International Airport in the map as her destination, inputs 8 hours for travel duration, and sends the query. As a result, she receives an itinerary recommendation visualized on the map which shows interesting locations to be visited, a recommended amount of time to stay in each location, and an estimate of the time needed to travel between any two locations. With the information at hand, she obtains a good idea of where she might go and is able to manage her time effectively.

9.2.2.2 Methodology for Itinerary Recommendation

As illustrated in Fig. 9.8, the framework for this itinerary recommender consists of an online component and an offline component.

Offline component: This component data mines the collective social intelligence from a database of GPS trajectories in terms of the following two steps:

The first step detects stay points from each GPS trajectory and clusters these stay points into locations. A location graph can be formulated according to the method described in Section 9.2.1.2 (the bottom layer of the hierarchy shown in Fig. 9.3 is used here as a demonstration). Remember that each stay point has properties pertaining to arrival and departure times, which indicate the length of time stayed in

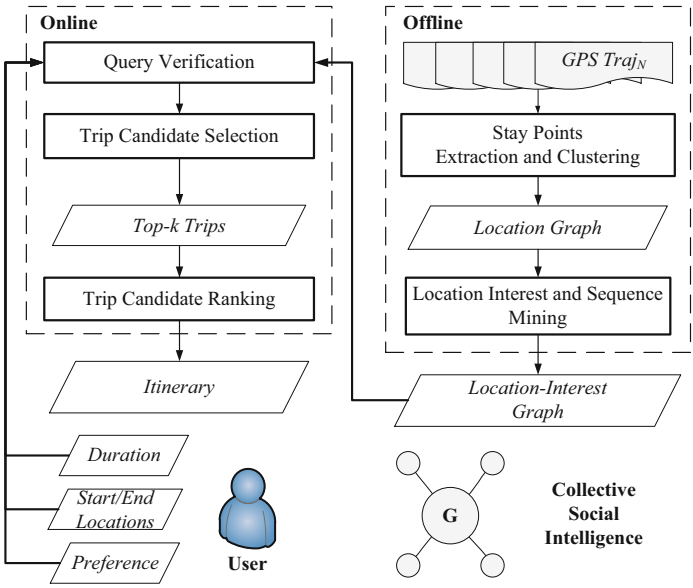


Fig. 9.8 Framework of the itinerary recommender

a location. So, the typical stay duration in each location and general travel duration between two locations (in a location graph) can be calculated, for example, using the median of all people's stay times in a location. These values associated with each location are used to estimate the duration of an itinerary. This clustering operation picks out the locations accessed by a significant number of people, ensuring the accuracy of the estimated travel and stay times. This operation also contributes to the second step by reducing the sparseness of the connections between users and locations.

The second step infers the interest value of each location in the location graph using the approach introduced in Section 9.2.1.2 (see Fig. 9.5), and calculates the popularity score of each 2-length travel sequence in terms of the method presented in Section 9.2.1.3. As a result, the output of this component is a location-interest graph, in which a node is a location associated with an interest value and a typical stay duration and an edge denotes people's transitions (between locations) and the general travel duration. The offline component will not be detailed further since they have been introduced in previous sections and in Chapter 8.

Online component: This component accepts a user-generated query (consisting of a starting location, a destination, and an available duration), and returns an effective itinerary comprised of a sequence of locations with a stay time in each location and travel times between two consecutive locations. This component can be decomposed into three steps, introduced as follows:

1) Query Verification: This step checks the feasibility of a query according to spatial and temporal constraints. In some cases, a user might set a short time length

with a far destination, making all itineraries impossible. Such queries can be filtered out by checking the distance between the start and end location with respect to the duration (of a query).

2) Trip Candidate Selection: This step searches a location-interest graph for candidate itineraries satisfying a user's query, i.e., each path has to start from the source and reach the destination within the given time length and pass some interesting locations on the way. Though there are some advanced path-finding algorithms, a straightforward method retrieving all the possible paths in a brute-force manner will work given the small size of a location-interest graph. When the start and end point of a query do not fall into any existing location in the graph, the nearest location to these points will be used.

3) Trip Candidate Ranking: This step first ranks candidate itineraries according to three factors: elapsed time ratio, stay time ratio, and interest density ratio, as introduced below. Then, these itineraries will be re-ranked according to the popularity score of the travel sequences pertaining to an itinerary.

- *Elapsed Time Ratio (ETR): ETR* is a ratio between the time length of a recommended itinerary and that given by a user. The bigger value this factor has, the more substantially the time given by a user is leveraged by an itinerary. If the total time needed for an itinerary is much shorter than the available time, the remaining time is wasted.
- *Stay Time Ratio (STR):* This factor considers how the available time is spent by calculating a ratio between the time that a user could stay in a location and that for traveling between locations. Intuitively, travelers prefer to spend more time in interesting locations rather than traveling to them. Therefore, an itinerary with a bigger *STR* is considered a better choice, i.e., a user can spend a longer time visiting actual places.
- *Interest Density Ratio (IDR): IDR* is the sum of the interest values of the locations contained in an itinerary. The general assumption is that visitors like to visit as many highly interesting locations as possible on a trip. Therefore, the bigger the IDR value, the better the itinerary. In the implementation, the IDR of an itinerary should be normalized to [0, 1] and divided by the maximum IDR in the candidate itineraries.
  As shown in Fig. 9.9, a good itinerary candidate is a point located in the upper-right corner of this cube, i.e., simultaneously having larger *ETR*, *STR*, and *IDR* values.
  To rank candidate itineraries, a Euclidian distance in these three dimensions is calculated as Eq. 9.11:

$$ED = \sqrt{\alpha_1 \cdot (ETR)^2 + \alpha_2 \cdot (STR)^2 + \alpha_3 \cdot (IDR)^2} \tag{9.11}$$

- *Popular Travel Sequence:* This factor represents how popular a recommended itinerary is (according to people's travel history), by summing up the popularity scores of the travel sequences contained in the itinerary (the popularity score of a travel sequence is computed in Section 9.2.1.3). This factor uses collective social knowledge to further differentiate the itineraries returned by the first

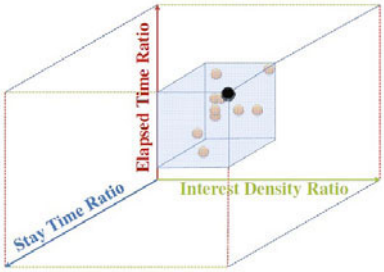


Fig. 9.9 Idea itinerary candidates

ranking step and guarantees the feasibility of an itinerary. As a result, the top k itineraries will be recommended to a user.

9.2.3 Location-Activity Recommendation

Besides the need for an itinerary, people usually have two types of questions in mind when traveling. They wonder where to go for sightseeing and food, and they wonder what there is to do at a particular location. The first question corresponds to location recommendation given a particular activity query, which might include restaurants, shopping, movies/shows, sports/exercise, and sightseeing. The second question corresponds to activity recommendation given a particular location query.

This section introduces a location-activity recommender system [40] which answers the above questions by mining a myriad of social media, such as tips-tagged trajectories or check-in sequences. Regarding the first question, this system provides a user with a list of interesting locations, e.g., the Forbidden City and the Great Wall, which are the top k candidate locations for conducting a given activity. With respect to the second question, if a user is visiting the Olympic Park of Beijing, the recommender suggests that the user can also try some exercise activities and nice restaurants nearby. This recommender integrates location recommendation and activity recommendation into one knowledge-mining process, since locations and activities are closely related in nature.

9.2.3.1 Data Modeling

Location-activity matrix: As mentioned before, to better share experiences, an individual can add comments or tips to a point location in a trajectory. For example, in Foursquare a user can leave some tips or a to-do-list in a venue so that her friends are able to view these tips when they arrive at the venue. Sometimes, these tips and to-do-lists clearly specify a user's activity in a location, enabling us to study the correlation between user activities and a location, for instance, what kinds of activities

can be performed in a location, and how often a particular activity is conducted in the location. Consequently, a location-activity matrix can be built, in which rows stand for locations and columns represent activities, as shown in the middle part of Fig. 9.10. An entry in the matrix denotes the frequency of an activity performed in a location. For example, if 5 users had dinner and 7 people watched a movie in this location in a week, the frequency of activity "dining" and "watching movies" is 5 and 7 respectively. This frequency denotes the popularity of an activity in a location and indicates the correlation between an activity, and a location.

If this location-activity matrix is completely filled, the above-mentioned recommendations can be easily achieved. Specifically, when conducting the location recommendation given an activity, we can rank and retrieve the top k locations with a relatively high frequency from the column that corresponds to that activity. Likewise, when performing activity recommendation for a location, the top k activities can be retrieved from the row corresponding to the location.

However, the location-activity matrix is incomplete and very sparse. Intuitively, people will not leave tips and to-do-lists in every restaurant and shopping mall. In short, many venues will not have labels of user activities. To address this issue, the information from another two matrices, respectively shown in the left and right part of Fig. 9.10, can be leveraged. One is a location-feature matrix; the other is an activity-activity matrix.

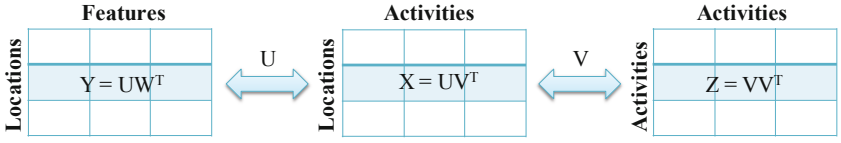


Fig. 9.10 The collaborative location-activity learning model

Location-feature matrix: In this matrix, a row stands for a location, and a column denotes a category (referred to as feature in this section), such as restaurants, cafes, and bars, illustrated in the left part of Fig. 9.10. Usually, a location might include multiple points of interest (POI) pertaining to different categories. For example, a mall would include different types of shops, movie theaters, and cafes. Further, a movie theater could have a few bars and restaurants inside. At the same time, a single venue could belong to multiple categories. For instance, some bars can also be regarded as a restaurant or a cafe. The motivation for building this location-feature matrix lies in the insight that people could carry out similar activities in similar locations.

Specifically, this matrix is built based on a POI database. Each POI in this database is associated with a set of properties, including name, address, GPS coordinates, and category. Given a location, the number of POIs pertaining to each category (and falling in this location) can be counted. Note that a location can be represented by a point or a small region [48, 46] like a cluster of stay points mentioned in Section 8.2.1.1, depending on the data source from different applications.

Suppose there are 4 restaurants, 2 bars and 5 shops in a location, a feature vector $v = < \ldots, 4, 2, 5, \ldots >$ is formulated for the location. To further differentiate the representativeness of each category in a location, a TF-IDF (term frequency-inverse document frequency [26, 27]) value is calculated for each category according to Eq. 8.7. Intuitively, if POIs of a category occur in a region many times, this POI category is important in representing this region. Furthermore, if a POI category (e.g., "museum" or "natural parks") occurs rarely in other regions, the category is more representative for the region in which it is located than a common POI category (e.g., "restaurant") that appears in many places. As a result, each item in a location-feature matrix is a TF-IDF value of a category in a location.

Activity-activity matrix: The activity-activity matrix, demonstrated in the right part of Fig. 9.10, models the correlation between two different activities, which contributes to the inferences of user activities that can be performed at a location. In other words, if a user performs some activity at a location, how likely would she perform another activity? One possible way to calculate this correlation is based upon user-generated tips and to-do-lists. In case the user-generated data is not large enough, the results returned by a search engine (like Google and Bing) can be used to compute the correlation as the correlation between two activities should be generally reflected by the World Wide Web. Specifically, we can send a pair of activities like "*shopping*" and "*food*" as a query to a search engine and count the returned results. The bigger this count is, the more these two activities are correlated. For example, the count of results returned for "*shopping*" and "*food*" is much larger than that of "*sports*" and "*food*," indicating that the former pair of activities is more related than the latter. Later, these counts are normalized into $[0, 1]$, representing the correlation between different pairs of activities.

9.2.3.2 Collaborative Inference

The data modeling has allowed for the compilation of location-activity, location-feature, and activity-activity matrices. The objective is to fill the missing entries in the location-activity matrix with the information learned from the other two matrices. A collaborative filtering (CF) approach based on collective matrix factorization [31] can be employed to train a location-activity recommender, using these matrices as inputs. Specifically, to infer the value of each missing entry an objective function is defined according to Eq. 9.12, which is iteratively minimized using a gradient descent method. Based on the filled location-activity matrix, it is possible to rank and retrieve the top k locations/activities as recommendations to users.

As shown in Fig. 9.10, a location-activity matrix $X_{m \times n}$ can be decomposed into a product of two matrices $U_{m \times k}$ and $V_{n \times k}$ (the superscript "T" for $V^{\mathrm{T}}_{n \times k}$ denotes the matrix transpose), where m is the number of locations and n stands for the number of activities. k is the number of latent factors (topics), usually $k < n$. In the implementation, k was set to 3 as there are three topics: location, activity, and feature. Likewise, location-feature matrix $Y_{m \times l}$ is decomposed as a product of matrices $U_{m \times k}$ and $W_{l \times k}$, and activity-activity matrix $Z_{n \times n}$ is decomposed as a self-product of $V_{n \times k}$.

So, this location-activity matrix shares the location information with the location-feature matrix via $U_{m\times k}$, and shares the activity knowledge with the activity-activity matrix via $V_{n\times k}$. In short, the inference model propagates the information among $X_{m\times n}$, $Y_{m\times l}$ and $Z_{n\times n}$ by the low-rank matrices $U_{m\times k}$ and $V_{n\times k}$. Finally, an objective function is formulated as Eq. 9.13:

$$\begin{aligned} L(U,V,W) = \frac{1}{2}\|I\circ(X-UV^{\mathrm{T}})\|_F^2 + \frac{\lambda_1}{2}\|Y-UW^{\mathrm{T}}\|_F^2 + \\ \frac{\lambda_2}{2}\|Z-VV^{\mathrm{T}}\|_F^2 + \frac{\lambda_3}{2}\left(\|U\|_F^2+\|V\|_F^2+\|W\|_F^2\right), \end{aligned} \tag{9.12}$$

where $\|\cdot|_F$ denotes the Frobenius norm, and I is an indicator matrix with its entry $I_{ij}=0$ if X_{ij} is missing, I_{ij}=1 otherwise. The operator "$\circ$" denotes the entry-wise product. The first three terms in the objective function control the loss in matrix factorization, and the last term controls the regularization over the factorized matrices so as to prevent over-fitting. λ_1,λ_2 and λ_3 are three parameters respectively weighting the contributions of location features, activity correlations, and the regularization term. These parameters can be learned using a training dataset.

In the objective function, the first term $(X-UV^{\mathrm{T}})$ measures the prediction loss of the location-activity matrix. The second term $(Y-UW^{\mathrm{T}})$ measures the prediction loss of the location-feature matrix. Minimizing it enforces the location latent factor U to be good as well in representing the location features. In other words, it helps to propagate the information of location features Y to the prediction of X. The third term $(Z-VV^{\mathrm{T}})$ measures the prediction loss of the activity-activity correlations. Minimizing it enforces the activity latent factor V to be good as well in representing the activity correlations. In other words, it helps to propagate the information of activity correlations Z to the prediction of X.

In general, this objective function is not jointly convex to all the variables, $U_{m\times k}$, $V_{n\times k}$, and $W_{l\times k}$. Also, there is no closed-form solution for minimizing the objective function. As a result, a numerical method such as the gradient descent is employed to determine the local optimal solutions. Specifically, the gradient (denoted as ∇) for each variable is represented as follows. Using the gradient descent, a converged $X_{m\times n}$ is returned with all the entries filled:

$$\nabla_U L = \left[I\circ(UV^{\mathrm{T}}-X)\right]V + \lambda_1(UW^{\mathrm{T}}-Y)W + \lambda_3 U, \tag{9.13}$$

$$\nabla_V L = \left[I\circ(UV^{\mathrm{T}}-X)\right]^{\mathrm{T}}U + 2\lambda_2(VV^{\mathrm{T}}-Z)V + \lambda_3 V, \tag{9.14}$$

$$\nabla_W L = \lambda_1(UW^{\mathrm{T}}-Y)^{\mathrm{T}}U + \lambda_3 W. \tag{9.15}$$

9.3 Personalized Travel Recommendations

While a generic travel recommender system can provide users with a variety of locations regardless of their personal interests, a personalized recommender offers locations matching an individual's preferences, which are learned from the individual's location history [44, 46]. Specifically, a personalized recommender uses a particular individual's number of visits to a location as their implicit rating of that location, and predicts the user's interest in an unvisited location in terms of their location history and those of other users. A matrix between users and locations, like the M shown in Eq. 9.16, is formulated, where rows stand for users and columns denote users' ratings of locations (represented by the times that a user has been to a location). One approach for building this matrix with user-generated GPS trajectory has been introduced in Section 8.2.1.

$$M = \begin{array}{c} \\ u_1 \\ u_2 \\ u_3 \\ u_4 \end{array} \begin{array}{c} \begin{array}{ccccc} c_{31} & c_{32} & c_{33} & c_{34} & c_{35} \end{array} \\ \begin{bmatrix} 1 & 1 & 2 & 0 & 4 \\ 1 & 1 & 3 & 0 & 2 \\ 0 & 0 & 1 & 2 & 0 \\ 0 & 0 & 0 & 1 & 1 \end{bmatrix} \end{array} \tag{9.16}$$

Based on user-location matrix M, a collaborative filtering model can be employed to infer a user's ratings of some unvisited location. Later, by ranking and retrieving the top k unvisited locations in terms of the inferred values from the row (in M) corresponding to a particular user, we can provide the user with a personalized recommendation.

In the following sections, the general idea of a CF model is first introduced and then two types of CF-based models that have been used in previous literature to create a personalized location recommender system. One is a user-based location recommender [46]; the other is an item-based one [44].

9.3.1 Collaborative Filtering

Collaborative filtering is a well-known model widely used in recommender systems. The general idea behind collaborative filtering [11, 24] is that similar users make ratings in a similar manner for similar items. Thus, if similarity is determined between users and items, a potential prediction can be made as to the rating of a user with regards to future items. According to [4], algorithms for collaborative recommendations can be grouped into two general classes: memory-based (or heuristic-based) and model-based.

Memory-based: Memory-based algorithms are essentially heuristics that make ratings predictions based on the entire collection of previously rated items by users [1]. That is, the value of the unknown rating for a user and an item is usually computed as an aggregate of the ratings of some other (usually, the N most similar) users

for the same item. There are two classes of memory-based collaborative filtering: user-based [29, 25] and item-based techniques [19, 28].

1) User-based techniques are derived from similarity measures between users. The similarity between two users (A and B) is essentially a distance measure and is used as a weight. In other words, when predicting user A's rating of an item, the more similar user A and B are, the more weight user B's rating of the item will carry. In most approaches, the similarity between two users is based on their ratings of items that both users have rated, using the Pearson correlation or the Cosine similarity measures. Spertus et al. [32] present an extensive empirical comparison of six distinct measures of similarity for recommending online communities to members of the Orkut social network. As a result, under the circumstances of the above-mentioned approach, they found that the Cosine similarity measure showed the best empirical results ahead of other measures, such as log odds and point-wise mutual information.

2) Item-based techniques predict the ratings of one item based on the ratings of another item. Examples of binary item-based collaborative filtering include Amazon's item-to-item algorithm [22], which computes the Cosine similarity between binary vectors representing the purchases in a user-item matrix. Slope One [19] is the simplest form of non-trivial item-based collaborative filtering. Its simplicity makes it especially easy to implement it efficiently while its accuracy is often on par with more complicated and computationally expensive algorithms. This algorithm is detailed in Section 9.3.2.2.

Model-based: In contrast to memory-based methods, model-based algorithms [10, 12] use the collection of ratings to form a model, which is then used to predict ratings. For example, Breese et al. [4] proposed a probabilistic approach to collaborative filtering. It is assumed that rating values are integers between 0 and n, and the probability expression is the probability that a user will give a particular rating to an item given that user's ratings of previously rated items. Hofmann et al. [12] proposed a collaborative filtering method in a machine learning framework where various machine learning techniques (such as artificial neural networks) coupled with feature extraction techniques can be used.

9.3.2 Location Recommenders Using User-Based CF

This section presents a personalized location recommender system [46] using a user-based CF model. Given a user-location matrix like M shown in Eq. 9.16, this location recommender operates according to the following three steps:

1) Infer the similarity between users: This personalized location recommender estimates the user similarity between two users in terms of their location histories (detailed in Section 8.3), instead of using traditional similarity measures, such as the Cosine similarity or the Pearson correlation. Typically, in a user-based CF model, the similarity between two users is based upon the ratings provided by both users. For example, the similarity between u_1 and u_2 (shown in Eq. 9.16) can be represent-

ed by the Pearson correlation between the ratings $< 1,1,2,4 >$ and $< 1,1,3,2 >$. Eq. 9.17 details the computation of the Pearson correlation.

$$sim(u_p, u_p) = \frac{\sum_{i \in S(R_p) \cap S(R_q)} (r_{pi} - \overline{R_p}) \cdot (R_{qi} - \overline{R_q})}{\sqrt{\sum_{i \in S(R_p) \cap S(R_q)} (r_{pi} - \overline{R_p})^2 \cdot \sum_{i \in S(R_p) \cap S(R_q)} (R_{qi} - \overline{R_q})^2}} \tag{9.17}$$

Notations: The ratings from a user u_p are represented as an array $R_p =< r_{p0}, r_{p1}, \ldots, r_{pn} >$, where r_{pi} is u_p's implicit ratings (the occurrences) in a location i. $S(R_p)$ is the subset of R_p, $\forall r_{pi} \in S(R_p)$, $r_{pi} \neq 0$, i.e., the set of items (locations) that has been rated (visited) by u_p. The average rating in R_p is denoted as $\overline{R_p}$. In this example, $R_1 =< 1,1,2,0,4 >$, $R_2 =< 1,1,3,0,2 >$, $S(R_1) =< 1,1,2,4 >$, and $S(R_2) =< 1,1,3,2 >$.

This rating-based similarity measure well represents the similarity between two users when the items rated by users are relatively independent, such as books, videos, or music. However, dealing with locations (especially, when these locations are derived from user-generated trajectories), this approach loses the information of people's mobility (e.g., the sequential property) and the hierarchical property of locations. As we mentioned in Section 8.3, users accessing the same locations (A, B, C) could be similar to each other. However, they would be more similar if they visited these locations in the same sequence like $A \rightarrow B \rightarrow C$. Also, people who visited the same building could be more similar to one another than those who traveled to the same city.

Two studies based on a real GPS trajectory dataset generated by 109 users over a period of 2 years, one using the geographical model [21], the other using the semantic model [35], have shown that the user similarity based on location history outperforms the Cosine similarity and the Pearson correlation.

2) Location selection: For a user, this step selects some locations that have not been visited by the user but have been accessed by other users. Note that the inferred rating of a location would not be very accurate if the location has only been accessed by a few users. At the same time, using the personalized recommender, a user needs to have some location data accumulated in the system.

3) Rating inference: Given the user-location matrix, user p's interest (r_{pi}) in a location i can be predicted according to the following three Equations, which is a common implementation of user-based collaborative filtering. All the notations used here have the same meanings with that of Eq. 9.17:

$$r_{pi} = \overline{R_p} + d \sum_{u_q \in U'} sim(u_p, u_q) \times (r_{qi} - \overline{R_q}); \tag{9.18}$$

$$d = \frac{1}{|U'|} \sum_{u_q \in U'} sim(u_p, u_q); \tag{9.19}$$

$$\overline{R_p} = \frac{1}{|S(R_P)|} \sum_{i \in S(R_p)} r_{pi}. \tag{9.20}$$

As shown in Eq. 9.18, the similarity between users u_p and u_q, $sim(u_p, u_q)$, is essentially a distance measure and is used as a weight. That is, the more similar u_p and u_q are, the more weight r_{qi} will carry in the prediction of r_{pi}. Here, $sim(u_p, u_q)$ is calculated according to the method introduced in Section 8.3.3. However, different people may visit places a varying number of times (e.g., one user might visit a park twice while another person may access the same park four times, although both of them are equally interested in the park), i.e., they use the rating scale differently. Therefore, an adjusted weighted sum is used here.

First, instead of using the absolute values of ratings, the deviations from the average rating of the corresponding user are used, i.e., $r_{qi} - \overline{R_q}$, where $\overline{R_q}$ denotes the average rating of u_q. Second, a normalizing factor d is involved, calculated in terms of Eq. 9.19 where U' is the collection of users who are the most similar to u_q. Third, u_p's rating scale is considered by calculating the average rating ($\overline{R_p}$) of u_p as Eq. 9.20, where $S(R_p)$ represents the collection of locations accessed by u_p.

Actually, these equations illustrate a well-known method [1, 24], which has been used widely in many recommendation systems. Therefore, it is not necessary to explain them in more detail.

9.3.3 Location Recommenders Using Item-Based CF

The user-similarity-based CF model accurately reflects the sequential and hierarchical properties of locations, providing an individual with effective location recommendations. However, this model has a relatively poor scalability as it needs to compute the similarity between each pair of users. Though the approximated method introduced in 8.4.1 can be used to alleviate this problem to some extent, the constantly increasing number of users in a real system leads to a huge computational burden. To address this issue, a location recommender using item-based collaborative filtering was proposed in [44], which is comprised of the following two steps: 1) Mining the correlation between locations, and 2) rating inference.

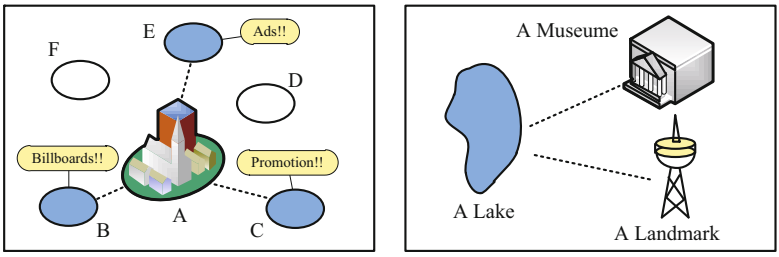
9.3.3.1 Mining the Correlation between Locations

There are a variety of approaches to determine the correlations between locations, for example, according to the distance between them (i.e., in geographical spaces)

[14], or in terms of the category of POIs located in a location (i.e., in category spaces) [30]. However, this section focuses on introducing the correlation between locations in the spaces of user behavior, specifically, to what extent two locations are correlated in people's minds [43, 47].

Typically, people might visit a few locations during a trip, such as going to a few shopping malls, traveling to a bunch of landmarks on a sightseeing tour, or going to a cinema after a restaurant. These locations might be similar or dissimilar, or nearby or far away from one another; but they are correlated from the perspective of human behavior. For example, a cinema and a restaurant are not similar in terms of the business categories they pertain to. However, in a user's mind, these places would be correlated as many people visit both these places during a trip. As another example, when shopping for something important like a wedding ring, an individual will visit similar shops selling jewelry sequentially. In short, these shops visited by this individual might be correlated. However, these similar shops might be far away from each other, i.e., they might not be co-located in geographical spaces. Therefore, this kind of correlation between locations can only be inferred from a large number of users' location history in a collective way.

The correlation between locations mentioned above can enable many valuable services, such as location recommender systems, mobile tour guides, sales promotions and bus route design. For instance, as shown in Fig. 9.11 A), a new shopping mall was built in location A recently. The mall operator is intending to set up some billboards or advertisements in other places to attract more attention, thereby promoting sales at this mall. Knowing that locations B, C and E have a much higher correlation with location A in contrast to locations D and F (according to a large number of users' location histories), the operator is more likely to maximize the promotion effect with minimal investment by putting the billboards or promotion information in locations B, C and E. Another example can be demonstrated using Fig. 9.11 B). If a museum and a landmark are highly correlated to a lake in terms of people's location histories, the museum and landmark can be recommended to tourists when they travel to the lake. Otherwise, people would miss some fantastic places even if they are only two hundred meters away from these locations.



A) Put promotion information or ads. at correlated locations

B) Recommend places to tourists in terms of location correlation

Fig. 9.11 Some application scenarios of this location correlation

However, mining the correlation from people's location histories faces the following two challenges. First, the correlation between two locations does not only depend on the number of users visiting the two locations but also lies in these users' travel experiences. The locations sequentially accessed by the people with more travel knowledge would be more correlated than the locations visited by those having little idea about the region. For instance, some overseas tourists might randomly visit some places in Beijing because they are not familiar with the city. However, the local people of Beijing are more capable of determining the best itineraries for a visit there.

Second, the correlation between two locations, A and B, also depends on the sequences in which both locations have been visited. 1) This correlation between A and B, $Cor(A,B)$, is asymmetric; i.e., $Cor(A,B) \neq Cor(B,A)$. The semantic meaning of a travel sequence $A \rightarrow B$ might be quite different from $B \rightarrow A$. For example, on a one-way road, people would only go to B from A while never traveling to A from B. 2) The two locations continuously accessed by a user would be more correlated than those being visited discontinuously. Some users would reach B directly from A ($A \rightarrow B$) while others would access another location C before arriving at B ($A \rightarrow C \rightarrow B$). Intuitively, the $Cor(A,B)$ indicated by the two sequences might be different. Likewise, in a sequence $A \rightarrow C \rightarrow B$, $Cor(A,C)$ would be greater than $Cor(A,B)$, as the user consecutively accessed $A \rightarrow C$, but traveled to B after visiting C.

In short, the correlation between two locations can be calculated by integrating the travel experiences of the users visiting them on a trip in a weighted manner. Formally, the correlation between location A and B can be calculated as Eq. 9.21.

$$Cor(A,B) = \sum_{u_k \in U'} \alpha \cdot e_k, \tag{9.21}$$

where U' is the collection of users who have visited A and B on a trip; e_k is u_k's travel experience, $u_k \in U'$, (Section 9.2.1.2 details the method for calculating a user's travel experience) . $0 < \alpha \leq 1$ is a dumping factor, decreasing as the interval between these two locations' index on a trip increases. For example, in the experiment of [43, 44], $\alpha = 2^{-(|j-i|-1)}$, where i and j are indices of A and B in the trip they pertain to. That is, the more discontinuously two locations being accessed by a user ($|i-j|$ would be big, thus α will become small), the less contribution the user can offer to the correlation between these two locations.

Figure 9.12 illustrates Eq. 9.21 using an example, in which three users (u_1, u_2, u_3) respectively access locations (A, B, C) in different manners and create three trips ($Trip_1$, $Trip_2$, $Trip_3$). The number shown below each node denotes the index of this node in the sequence. According to Eq. 9.21, for $Trip_1$, $Cor(A,B) = e_1$ and $Cor(B,C) = e_1$, since these locations have been consecutively accessed by u_1(i.e., $\alpha = 1$). However, $Cor(A,C) = 1/2 \cdot e_1$ (i.e., $\alpha = 2^{-(|2-0|-1)} = 1/2$) as u_1 traveled to B before visiting C. In other words, the correlation (between location A and C) that can be sensed from $Trip_1$ might not be that strong if they are not consecutively visited by u_1. Likewise, $Cor(A,C) = e_2$, $Cor(C,B) = e_2$, $Cor(A,B) = 1/2 \cdot e_2$ in

terms of $Trip_2$, and $Cor(B,A) = e_3$, $Cor(A,C) = e_3$, $Cor(B,C) = 1/2 \cdot e_3$ by $Trip_3$. Later, the correlation inferred from each user's trips is integrated as follows.

$$Cor(A,B) = e_1 + \frac{1}{2} \cdot e_2; \quad Cor(A,C) = \frac{1}{2} \cdot e_1 + e_2 + e_3;$$
$$Cor(B,C) = e_1 + \frac{1}{2} \cdot e_3; \quad Cor(C,B) = e_2; \quad Cor(B,A) = e_3.$$

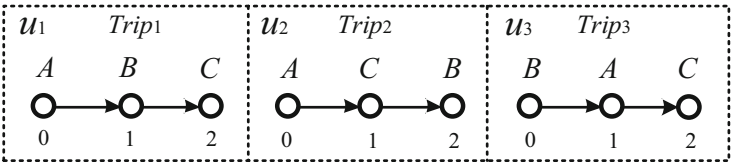


Fig. 9.12 An example calculating the correlation between locations

9.3.3.2 Rating Inference

The typical way to estimate the similarity between two items is calculating the Cosine similarity between two rating vectors that correspond to the two items. For instance, the similarity between location l_3 and l_5 can be represented by the Cosine similarity between the two rating vectors, $< 3,2,1,0 >^{\mathrm{T}}$ and $< 4,2,0,1 >^{\mathrm{T}}$. This rating-based similarity measure is fast, however it neglects the information of user mobility patterns among locations. As a result, the rating-based method is not a very effective similarity measure for an item-based location recommender, in which people's mobility patterns is a key factor determining the quality of recommendations.

Instead of using the rating-based similarity measure, the recommender presented in [44] integrates the location correlation (introduced in Section 9.3.3.1) into an item-based collaborative filtering (specifically, the Slope One algorithm) [19], thereby inferring the ratings of a particular user to some unvisited locations.

1)*The Slope One algorithms*

Notations: The ratings from user u_p, called an evaluation, are represented as array $R_p =< r_{p0}, r_{p1}, \ldots, r_{pn} >$, where r_{pj} is u_p's implicit ratings in location j. $S(R_p)$ is the subset of R_p, $\forall r_{pj} \in S(R_p)$, $r_{pj} \neq 0$. The collection of all evaluations in the training set is χ. $S_j(\chi)$ means the set of evaluations containing item j, $\forall R_p \in S_j(\chi)$, $j \in S(R_p)$. Likewise, $S_{i,j}(\chi)$ is the set of evaluations simultaneously containing item i and j.

The Slope One algorithms [19] are famous and representative item-based CF algorithms, which are easy to implement, efficient to query, and reasonably accurate. Given any two items i and j with ratings r_{pj} and r_{pi} respectively in some user

evaluation $R_p \in S_{j,i}(\chi)$, the average deviation of item i with regard to item j is calculated as Eq. 9.22.

$$dev_{j,i} = \sum_{R_p \in S_{j,i}(\chi)} \frac{r_{pj} - r_{pi}}{S_{j,i}(\chi)}, \tag{9.22}$$

Given that $dev_{j,i} + r_{pi}$ is a prediction for r_{pj} based on r_{pi}, a reasonable predictor might be the average of all the predictions, as shown in Eq. 9.23.

$$P(r_{pj}) = \frac{1}{|w_j|} \sum_{i \in w_j} (dev_{j,i} + r_{pi}), \tag{9.23}$$

where $w_j = \{i | i \in S(R_p), i \neq j, |S_{j,i}(\chi)| > 0\}$ is the set of all relevant items.

Further, the number of evaluations that simultaneously contain two items has been used to weight the prediction regarding different items, as presented in Eq. 9.25. Intuitively, to predict u_p's rating of item A given u_p's ratings of item B and C, if 2000 users rated the pair of A and B whereas only 20 users rated pair of A and C, then u_p's ratings of item B is likely to be a far better predictor for item A than u_p's ratings of item C is.

$$P(r_{pj}) = \frac{\sum_{w_j} (dev_{j,i} + r_{pi}) \cdot |S_{j,i}(X)|}{\sum_{w_j} |S_{j,i}(X)|} \tag{9.24}$$

2) *The Slope One algorithm using the location correlation:* Intuitively, to predict u_p's rating of location A given u_p's ratings of location B and C, if location B is more related to A beyond C, then u_p's ratings of location B is likely to be a far better predictor for location A than u_p's ratings of location C is. Therefore, as shown in Eq. 9.25, the $|S_{j,i}(\chi)|$ in Eq. 9.24 is replaced by the correlation cor_{ji} (inferred in Section 9.3.3.1):

$$P(r_{pj}) = \frac{\sum_{i \in S(R_p) \wedge i \neq j} (dev_{j,i} + r_{pi}) \cdot cor_{ji}}{\sum_{i \in S(R_p) \wedge i \neq j} cor_{ji}}, \tag{9.25}$$

where cor_{ji} denotes the correlation between location i and j, and $dev_{j,i}$ is still calculated as Eq. 9.22.

In contrast to the number of observed ratings (i.e., $|S_{j,i}(\chi)|$) used by the weighted Slope One algorithm, the mined location correlation considers more human travel behavior, such as the travel sequence, user experience, and transition probability between locations. Using Eq. 9.25, an individual's ratings of locations they have not accessed can be inferred. Later, the top n locations with relatively high ratings can be recommended to the user.

9.3.4 Open Challenges

Although they are able to provide useful recommendations to an individual, personalized location recommender systems are faced with some open challenges during real implementation. These challenges include cold start problems, data sparseness, and scalability, which will be discussed individually.

9.3.4.1 Cold Start

A cold start is a prevalent problem in recommender systems, as a system cannot draw inferences for users or items when it has not yet gathered sufficient information. Generally, the problem is caused by new users or new items entering a recommender system. This sub-section discusses the possible solutions dealing with new locations and new users with respect to personalized location recommender systems.

New Location problem: When a new location is added to a location recommendation system, it usually has few ratings with which to determine the correlation (or similarity) between this new location and other places. As a consequence, the new location is hardly recommended to users even if it is a good place to visit. One possible solution is assigning the new location with a small number of users' ratings of existing places that are similar to the new location. Specifically, this method estimates the similarity between a new location and some existing locations (with an adequate number of ratings) according to their category information (e.g., the feature vector shown Fig. 9.10) [46]. For example, if both the new location and an existing place belong to the category of <restaurant, café, and bar>, the ratings that a user gave to a previous existing place might be similar to the new location. The locations in the system having a closer distance to the new location will be given a higher weight. As a result, the k most similar places can be selected for new locations. Then, the ratings of a few users to these similar places can be utilized to estimate their ratings to the new location, for instance, using the average mean of existing ratings. With these virtually generated ratings, the new location can be recommended to real users, thereby getting real ratings in return. The virtual ratings can be removed from the system once the new location has obtained enough ratings. Note that we can only select a few similar locations and users to generate the virtual ratings for a new location. Otherwise, these virtual ratings will dominate future inferences.

New User problem: When signing up in a recommender system, a new user has no location data accumulated in the system. Therefore, the recommender cannot offer her personalized recommendations effectively. One possible solution is providing an individual with some of the most popular recommendations at first regardless of her interests. If we have an individual's profile (e.g., likes movies), the popular locations with a category matching the individual's preferences can be recommended. Regarding the users who have only visited a very limited number of locations, some similarity-based mapping methods can be used to propagate a user's rating to a

visited location to a few similar places that have not been accessed by the individual. Specifically, a similarity between two locations can be determined using either the location correlation (if both locations have sufficient ratings), or categories of these two locations as mentioned in the new location problem.

9.3.4.2 Data Sparseness

Intuitively, a user-location matrix is very sparse as a user can only visit a certain number of locations. The prediction of an individual's interest in a location is unable to be very accurate based on such a sparse matrix. Some approaches using additional information can improve the inferences. For example, the method introduced in Section 9.2.3 uses the category information of a location. Also, propagation and similarity-based mapping methods (mentioned in the above section) can be employed to reduce the empty entries in a matrix. Alternative methods can transfer this user-location matrix into a user-category matrix, which has a smaller number of columns and fewer empty entries than the former. As shown in Fig. 9.13, the similarity between users can be determined in terms of this user-category matrix with sufficient ratings, and then used in turn in the user-location matrix for location recommendation. However, this is still a very challenging problem that remains open to research and real implementation.

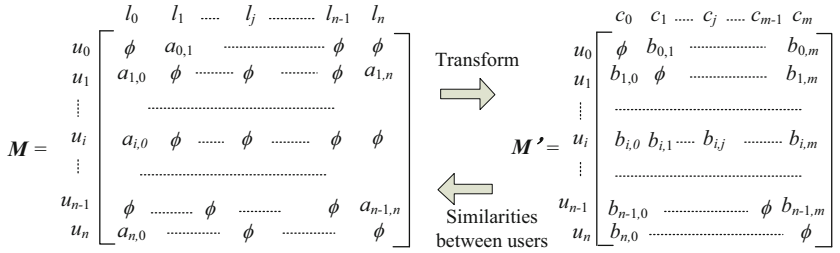


Fig. 9.13 Transforming a user-location matrix into a user-category matrix

9.3.4.3 Scalability

While the number of locations is limited in the real world and usually much smaller than that of users, the problem concerning the scalability of a location recommender arises in large part from the increasing number of users. Some approximation approaches can be used to reduce this problem to some extent.

For example, in the user-based CF model (introduced in Section 9.3.2), the mechanisms demonstrated in Fig. 8.10 can be employed to diminish the computations when a new user joins the system. First, existing users in a recommendation system can be clustered into groups according to the similarity between one another.

The users who are similar to a new user can then be found solely in the groups that the new user pertains to. Second, a location history can be quickly built for a new user by directly inserting her stay points into existing framework F. Later, we can compute the similarity between the new user and the representative user of a cluster based on this location history, thereby determining which group the new user belongs to. Third, the shared framework and user clusters can also be updated at a relatively low frequency, e.g., 1 update per month, as the arrival of a few users will not significantly change them. Fourth, once the similarity between the new users and other users in the cluster is computed, the similarity can be used for a time even if the new user has new data uploaded to the system. Adding a few trajectories will not change the similarity between two users significantly.

An alternative way to enhance the scalability of a location recommender system uses the item-based CF model presented in Section 9.3.3. The correlation (or similarity) between two locations can be updated at a very low frequency, e.g., once a month, as the arrival of a few new users does not change them significantly.

9.4 Summary

This chapter explores research topics in a location-based social network from the perspective of understanding locations with user-generated GPS trajectories. Using travel as a main application scenario, both generic and personalized travel recommendations are studied.

The generic travel recommender starts by finding interesting locations and travel sequences from a large amount of raw trajectories, and then offers itinerary and location-activity recommendations. By tapping into the collective social knowledge, these recommendations help people to travel to an unfamiliar place and plan their journey with minimal effort.

The personalized location recommender provides a particular user with locations matching her preferences, based on the location history of this user and that of others. Regarding a user's visits to a location as an implicit rating to that location, two kinds of collaborative filtering-based models are proposed to predict the user's interests in unvisited places. One is a user-based CF model, which incorporates the similarity between two different users (derived from their location histories) as a distance function between them. The other is a location-based CF model using the correlation between two different locations (inferred from many users' GPS trajectories) as a distance measure between them. The user-based CF model is able to accurately model an individual's behavior while it suffers from the increasing scale of users. The location-based CF model is efficient and reasonably accurate.

Some challenges are also discussed in the end of this chapter, aiming to encourage more research effort into this field.

References

1. Adomavicius, G., Tuzhilin, A.: 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**, 734–749 (2005)
2. Arase, Y., Xie, X., Hara, T., Nishio, S.: Mining people's trips from large scale geo-tagged photos. In: Proceedings of the international conference on Multimedia, MM '10, pp. 133–142. ACM, New York, NY, USA (2010)
3. Ardissono, L., Goy, A., Petrone, G., Segnan, M.: A multi-agent infrastructure for developing personalized web-based systems. ACM Trans. Internet Technol. **5**, 47–69 (2005)
4. Breese, J.S., Heckerman, D., Kadie, C.M.: Empirical analysis of predictive algorithms for collaborative filtering. In: Proceedings of the International 14th Conference on Uncertainty in Artificial Intelligence, pp. 43–52 (1998)
5. Cao, X., Cong, G., Jensen, C.S.: Mining significant semantic locations from gps data. Proc. VLDB Endow. **3**, 1009–1020 (2010)
6. Counts, S., Smith, M.: Where were we: communities for sharing space-time trails. In: Proceedings of the 15th annual ACM international symposium on Advances in geographic information systems, GIS '07, pp. 10:1–10:8. ACM, New York, NY, USA (2007)
7. Cranshaw, J., Toch, E., Hong, J., Kittur, A., Sadeh, N.: Bridging the gap between physical location and online social networks. In: Proceedings of the 12th ACM international conference on Ubiquitous computing, Ubicomp '10, pp. 119–128. ACM, New York, NY, USA (2010)
8. De Choudhury, M., Feldman, M., Amer-Yahia, S., Golbandi, N., Lempel, R., Yu, C.: Automatic construction of travel itineraries using social breadcrumbs. In: Proceedings of the 21st ACM conference on Hypertext and hypermedia, HT '10, pp. 35–44. ACM, New York, NY, USA (2010)
9. Dunstall, S., Horn, M.E.T., Kilby, P., Krishnamoorthy, M., Owens, B., Sier, D., Thiébaux, S.: An automated itinerary planning system for holiday travel. J. of IT & Tourism **6**(3), 195–210 (2003)
10. Getoor, L., Sahami, M.: Using probabilistic relational models for collaborative filtering. In: Working Notes of the KDD Workshop on Web Usage Analysis and User Profiling (1999)
11. Goldberg, D., Nichols, D., Oki, B.M., Terry, D.: Using collaborative filtering to weave an information tapestry. Commun. ACM **35**, 61–70 (1992)
12. Hofmann, T.: Collaborative filtering via gaussian probabilistic latent semantic analysis. In: Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval, SIGIR '03, pp. 259–266. ACM, New York, NY, USA (2003)
13. Horozov, T., Narasimhan, N., Vasudevan, V.: Using location for personalized poi recommendations in mobile environments. In: Proceedings of the International Symposium on Applications on Internet, pp. 124–129. IEEE Computer Society, Washington, DC, USA (2006)
14. Huang, Y., Shekhar, S., Xiong, H.: Discovering colocation patterns from spatial data sets: A general approach. IEEE Trans. on Knowl. and Data Eng. **16**, 1472–1485 (2004)
15. Hung, C.C., Chang, C.W., Peng, W.C.: Mining trajectory profiles for discovering user communities. In: Proceedings of the 2009 International Workshop on Location Based Social Networks, LBSN '09, pp. 1–8. ACM, New York, NY, USA (2009)
16. Kim, J., Kim, H., Ryu, J.h.: Triptip: a trip planning service with tag-based recommendation. In: Proceedings of the 27th international conference extended abstracts on Human factors in computing systems, CHI EA '09, pp. 3467–3472. ACM, New York, NY, USA (2009)
17. Kleinberg, J.M.: Authoritative sources in a hyperlinked environment. In: Proceedings of the ninth annual ACM-SIAM symposium on Discrete algorithms, SODA '98, pp. 668–677. Society for Industrial and Applied Mathematics, Philadelphia, PA, USA (1998)
18. Kumar, P., Singh, V., Reddy, D.: Advanced traveler information system for hyderabad city. Intelligent Transportation Systems, IEEE Transactions on **6**(1), 26–37 (2005)
19. Lemire, D., Maclachlan, A.: Slope one predictors for online rating-based collaborative filtering. In: Proceedings of SIAM Data Mining. SIAM press (2005)

20. Li, Q., Zheng, Y., Xie, X., Chen, Y., Liu, W., Ma, W.Y.: Mining user similarity based on location history. In: Proceedings of the 16th ACM SIGSPATIAL international conference on Advances in geographic information systems, GIS '08, pp. 34:1–34:10. ACM, New York, NY, USA (2008)
21. Li, Q., Zheng, Y., Xie, X., Chen, Y., Liu, W., Ma, W.Y.: Mining user similarity based on location history. In: Proceedings of the 16th ACM SIGSPATIAL international conference on Advances in geographic information systems, GIS '08, pp. 34:1–34:10. ACM, New York, NY, USA (2008)
22. Linden, G., Smith, B., York, J.: Amazon.com recommendations: Item-to-item collaborative filtering. IEEE Internet Computing **7**, 76–80 (2003)
23. Lu, X., Wang, C., Yang, J.M., Pang, Y., Zhang, L.: Photo2trip: generating travel routes from geo-tagged photos for trip planning. In: Proceedings of the international conference on Multimedia, MM '10, pp. 143–152. ACM, New York, NY, USA (2010)
24. Nakamura, A., Abe, N.: Collaborative filtering using weighted majority prediction algorithms. In: Proceedings of the Fifteenth International Conference on Machine Learning, ICML '98, pp. 395–403. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA (1998)
25. Resnick, P., Iacovou, N., Suchak, M., Bergstrom, P., Riedl, J.: Grouplens: an open architecture for collaborative filtering of netnews. In: Proceedings of the 1994 ACM conference on Computer supported cooperative work, CSCW '94, pp. 175–186. ACM, New York, NY, USA (1994)
26. Salton, G., Buckley, C.: Term-weighting approaches in automatic text retrieval. Inf. Process. Manage. **24**, 513–523 (1988)
27. Salton, G., Fox, E.A., Wu, H.: Extended boolean information retrieval. Commun. ACM **26**, 1022–1036 (1983)
28. Sarwar, B., Karypis, G., Konstan, J., Reidl, J.: Item-based collaborative filtering recommendation algorithms. In: Proceedings of the 10th international conference on World Wide Web, WWW '01, pp. 285–295. ACM, New York, NY, USA (2001)
29. Shardanand, U., Maes, P.: Social information filtering: algorithms for automating "Word of Mouth". In: Proceedings of the SIGCHI conference on Human factors in computing systems, CHI '95, pp. 210–217. ACM Press/Addison-Wesley Publishing Co., New York, NY, USA (1995)
30. Sheng, C., Zheng, Y., Hsu, W., Lee, M.L., Xie, X.: Answering top- similar region queries. In: Proceedings of Database Systems For Advanced Applications, vol. 5981, pp. 186–201. Springer (2010)
31. Singh, A.P., Gordon, G.J.: Relational learning via collective matrix factorization. In: Proceeding of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining, KDD '08, pp. 650–658. ACM, New York, NY, USA (2008)
32. Spertus, E., Sahami, M., Buyukkokten, O.: Evaluating similarity measures: a large-scale study in the orkut social network. In: Proceedings of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining, KDD '05, pp. 678–684. ACM, New York, NY, USA (2005)
33. Takeuchi, Y., Sugimoto, M.: Cityvoyager: An outdoor recommendation system based on user location history. In: Proceedings of the 3rd International Conference Ubiquitous Intelligence and Computing, pp. 625–636. Springer press (2006)
34. Xiao, X., Zheng, Y., Luo, Q., Xie, X.: Finding similar users using category-based location history. In: Proceedings of the 18th SIGSPATIAL International Conference on Advances in Geographic Information Systems, GIS '10, pp. 442–445. ACM, New York, NY, USA (2010)
35. Xiao, X., Zheng, Y., Luo, Q., Xie, X.: Finding similar users using category-based location history. In: Proceedings of the 18th SIGSPATIAL International Conference on Advances in Geographic Information Systems, GIS '10, pp. 442–445. ACM, New York, NY, USA (2010)
36. Ying, J.J.C., Lu, E.H.C., Lee, W.C., Weng, T.C., Tseng, V.S.: Mining user similarity from semantic trajectories. In: Proceedings of the 2nd ACM SIGSPATIAL International Workshop on Location Based Social Networks, LBSN '10, pp. 19–26. ACM, New York, NY, USA (2010)

37. Yoon, H., Zheng, Y., Xie, X., Woo, W.: Smart itinerary recommendation based on user-generated gps trajectories. In: Proceedings of the 7th international conference on Ubiquitous intelligence and computing, UIC'10, pp. 19–34. Springer-Verlag, Berlin, Heidelberg (2010)
38. Yoon, H., Zheng, Y., Xie, X., Woo, W.: Social itinerary recommendation from user-generated digital trails. Personal and Ubiquitous Computing (2011)
39. Zheng, V.W., Cao, B., Zheng, Y., Xie, X., Yang, Q.: Collaborative filtering meets mobile recommendation: A user-centered approach. In: Proceedings of AAAI conference on Artificial Intelligence (AAAI 2010), pp. 236–241. ACM, New York, NY, USA (2010)
40. Zheng, V.W., Zheng, Y., Xie, X., Yang, Q.: Collaborative location and activity recommendations with gps history data. In: Proceedings of the 19th international conference on World wide web, WWW '10, pp. 1029–1038. ACM, New York, NY, USA (2010)
41. Zheng, Y., Chen, Y., Xie, X., Ma, W.Y.: Geolife2.0: A location-based social networking service. In: Proceedings of the 2009 Tenth International Conference on Mobile Data Management: Systems, Services and Middleware, MDM '09, pp. 357–358. IEEE Computer Society (2009)
42. Zheng, Y., Wang, L., Zhang, R., Xie, X., Ma, W.Y.: Geolife: Managing and understanding your past life over maps. In: Proceedings of the The Ninth International Conference on Mobile Data Management, pp. 211–212. IEEE Computer Society, Washington, DC, USA (2008)
43. Zheng, Y., Xie, X.: Learning location correlation from gps trajectories. In: Proceedings of the 2010 Eleventh International Conference on Mobile Data Management, MDM '10, pp. 27–32. IEEE Computer Society, Washington, DC, USA (2010)
44. Zheng, Y., Xie, X.: Learning travel recommendations from user-generated gps traces. ACM Trans. Intell. Syst. Technol. **2**, 2:1–2:29 (2011)
45. Zheng, Y., Xie, X., Ma, W.Y.: Geolife: A collaborative social networking service among user, location and trajectory. IEEE Data Eng. Bull. **33**(2), 32–39 (2010)
46. Zheng, Y., Zhang, L., Ma, Z., Xie, X., Ma, W.Y.: Recommending friends and locations based on individual location history. ACM Trans. Web **5**, 5:1–5:44 (2011)
47. Zheng, Y., Zhang, L., Xie, X., Ma, W.Y.: Mining correlation between locations using human location history. In: Proceedings of the 17th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems, GIS '09, pp. 472–475. ACM, New York, NY, USA (2009)
48. Zheng, Y., Zhang, L., Xie, X., Ma, W.Y.: Mining interesting locations and travel sequences from gps trajectories. In: Proceedings of the 18th international conference on World wide web, WWW '09, pp. 791–800. ACM, New York, NY, USA (2009)