Learning Location Correlation from GPS Trajectories

Yu Zheng, Xing Xie

Microsoft Research Asia, 4F, Sigma Building, No.49 Zhichun Road, Haidian District, Beijing 100190, China

{yuzheng, xingx}@microsoft.com

Abstract*— *People's location histories imply the location correlation that states the relations between geographical locations in the space of human behavior. With the correlation, we can enable many valuable services, such as location recommendation and sales promotion. In this paper, by taking into account a user's travel experience (knowledge) and the sequentiality that locations have been visited, we learn the location correlation from a large number of user-generated GPS trajectories. Using the location correlation, we conduct a personalized location recommendation system, which is evaluated based on a real-world GPS dataset collected by 112 users over a period of 1.5 years. As a result, our method outperforms that using the Pearson correlation.

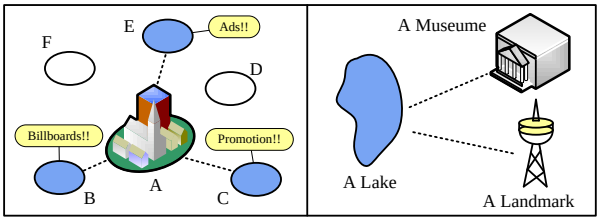
Keywords- Location Correlation, Spatial Data Mining, Location History, GPS trajectory

I. INTRODUCTION

The increasing popularity of location-acquisition technologies, such as GPS and GSM network, is leading to the collection of large spatio-temporal dataset of individuals. The dataset cannot only represent people's location histories but also imply the correlation between geographical regions. This correlation denotes the relationship between locations from the perspective of human behavior, and might indicate the probability that two locations co-occurred in people's trips.

Typically, people might visit a few locations in a trip, e.g., access some malls when shopping, travel to a branch of landmarks in a sightseeing tour, or go to a cinema from a restaurant, etc. These locations might be similar or dissimilar, nearby or far away from each other; 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 if most people have visited these places in one trip. In other cases, to buy something important like a wedding ring, an individual would access some similar shops selling jewelry in a trip. In short, these shops visited by this individual might be correlated. However, these similar shops could be far away from each other, i.e., they might not be co-located in geographical spaces. Thus, the correlation covers and is far beyond the category similarity and the geographical distance between locations.

The correlation between locations can enable many valuable services, such as location recommendation systems, mobile tour guides, sales promotion and bus routes design. For instance, as shown in Figure 1 A), a brand new shopping mall is built in location *A* recently. The mall operator is intending to set up some billboards or advertisements in other places to attract more people's attention; hence promote the sales of this mall. By mining a large number of users' location histories, we discover that, in contrast to locations *D* and *F*, locations *B*, *C* and *E* have a much higher correlation with location *A*. Hence, if putting the billboards or promotion information in locations *B*, *C* and *E*, the operator is more likely to maximize the promotion effect with minimal investment. Another example can be demonstrated using Figure 1 B). If we discover a museum and a landmark is highly correlated to a lake by analyzing many people's location histories, the museum and landmark can be recommended to tourists when they travel to the lake.



A) Put promotion information or ads. at correlated locations

B) Recommend places to tourists in terms of location correlation

Figure 1. Some application scenarios of the location correlation

In this paper, we report on an approach mining the correlation between locations from human location history. Beyond the geo-distance relationship and the business category similarity between locations, the location correlation describes the relationship between locations in the space of human behavior. The contribution of this paper lies in the follows:

1) We propose an algorithm learning the correlation between locations. This algorithm considers users' travel experiences and the sequentiality of the locations in a user's trip.
2) We conduct a personalized location recommendation system, which integrates the correlation into a collaborative filtering algorithm.
3) We evaluated the recommender by using a large-scale real-world GPS dataset collected by 112 users over a period of one year. As a result, our recommender is more effective than the baseline schemes.

II. PRELIMINARY

A. Problem Definition

Definition 1. Trajectory. A user's trajectory *Traj* is a sequence of time-stamped points, $Traj = \langle p_0, p_1, \dots, p_k \rangle$, where $p_i = (x_i, y_i, t_i)$ $(i = 0,1, \dots, k)$; t_i is a timestamp, $\forall 0 \le i < k, t_i < t_{i+1}$ and (x_i, y_i) are two-dimension coordinates of points.

Definition 2. $Dist(p_i, p_j)$ denotes the geospatial distance between two points p_i and p_j, and $Int(p_i, p_j) = |p_i.t_i - p_j.t_j|$ is the time interval between two points.

As shown in Figure 2, from each user's *Traj*, we can detect some geographic regions, called stay points, where the user stayed over a certain time interval. In contrast to a raw point p_i in a trajectory, a stay point carries a particular semantic meaning, such as the shopping mall the user accessed and the restaurant they visited. The extraction of a stay point depends on two scale parameters, a time threshold (T_r) and a distance threshold (D_r). As shown in Figure 2, $\{p_1, p_2, \ldots, p_8\}$ formulate a trajectory, and a stay point would be detected from $\{p_3, p_4, p_5, p_6\}$ if $d \le D_r$ and $Int(p_3, p_6) \ge T_r$.

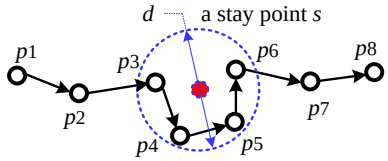


Figure 2. A trajectory and a stay point

Definition 3*: *Stay Point. A stay point s is a geographical region where a user stayed over a time threshold T_r within a distance threshold of D_r. In a user's trajectory, s is characterized by a set of consecutive points $P = \langle p_m, p_{m+1}, \ldots, p_n \rangle$, where $\forall m < i \le n$, $Dist(p_m, p_i) \le D_r, Dist(p_m, p_{n+1}) > D_r$ and $Int(p_m, p_n) \ge T_r$. Therefore, $s = (x, y, t_a, t_l)$, where

$$s.x = \sum_{i=m}^{n} p_i.x/|P|, \quad (1)$$

$$s.y = \sum_{i=m}^{n} p_i.y/|P|, \quad (2)$$

respectively stands for the average x and y coordinates of the collection P; $s.t_a = p_m.t_m$ is the user's arriving time on s and $s.t_l = p_n.t_n$ represents the user's leaving time.

Definition 4: Location History. An individual's location history h is represented as a sequence of stay points they visited with corresponding arrival and leaving times,

$$h = \langle s_0 \xrightarrow{\Delta t_1} s_1 \xrightarrow{\Delta t_2}, \ldots, \xrightarrow{\Delta t_{n-1}} s_n \rangle, \quad (3)$$

where $\forall 0 \le i < n, s_i$ is a stay point and $\Delta t_i = s_{i+1}.t_a - s_i.t_l$ is the time interval between two stay points.

However, so far, people's location histories are still inconsistent as the stay points detected from various individuals' trajectories are not identical. So, we put together the stay points detected from all users' trajectories into a dataset $\boldsymbol{S}$, and employ a clustering algorithm to partition this dataset into some clusters. Thus, the similar stay points from various users will be assigned into the same cluster.

Definition 5: Locations. $L = \{l_0, l_1, \ldots, l_n\}$ is a collection of Locations, where $\forall 0 \le i \le n, l_i = \{s | s \in \boldsymbol{S}\}$ is a cluster of stay points detected from multiple users' trajectories; $i \ne j, l_i \cap l_j = \emptyset$.

After the clustering operation, we can substitute a stay point in a user's location history with the cluster ID the stay point pertains to. In short, a user's location history can be represented as a sequence of the locations. Supposing $s_0 \in l_i, s_1 \in l_j, s_n \in l_k$, Equation (3) can be replaced with

$$h = \langle l_i \xrightarrow{\Delta t_1} l_j \xrightarrow{\Delta t_2}, \ldots, \xrightarrow{\Delta t_{n-1}} l_k \rangle. \quad (4)$$

Thus, different users' location histories become comparable and can be integrated to infer the correlation between locations. Later, we partition an individual's location history into some trips if the travel time spent between two consecutive locations exceeds a certain threshold T_p.

Definition 6: Trip: A trip is a sequence of locations consecutively visited by a user, $Trip = \langle l_0 \xrightarrow{\Delta t_1} l_1 \xrightarrow{\Delta t_2}, \ldots, \xrightarrow{\Delta t_{k-1}} l_k \rangle$, where $\forall 0 \le i \le k, \Delta t_k < T_p$ (a threshold) and $l_i \in L$ is a stay-point-cluster ID.

In short, a user's location history can be regarded as a collection of trips, $h = \{Trip\}$, and each $Trip = \langle l_i \to l_j \to \cdots \rangle$ is a sequence of locations represented by some clusters of stay points.

Definition 7: Users. $U = \{u_0, u_1, \ldots, u_m\}$ denotes the collection of users. $\forall 0 \le k \le m, u_k \in U$ is a user having a trajectory $Traj_k$, a location history h_k and certain travel experience e_k.

B. *Framework*

Figure 3 describes the framework for mining location correlation. First, as shown in Lines 2~4, we detect stay points from each user's trajectories, and formulate their own location histories into a sequence of stay points. Second, as depicted in Line 5 and 6, we discover a set of locations L by clustering all users' stay points. Later, a user (u_k)'s location history (h_k) can be represented by a sequence of stay-point-clusters called locations here (refer to Lines 7 and 8). Third, we put all user's location history together, and learn each user's travel experience (e.g., e_k of u_k) using a iterative model (refer to Lines 9 and 10). Fourth, considering $\{(e_k, h_k), 0 \le k < |U|\}$, we infer the correlation between locations, $Cor(l_i, l_j)$, where $l_i \in L$ and $l_j \in L$, $\forall 0 \le i, j < |L|, i \ne j$.

MiningLocationCorrelation ($U, TRAJ, T_r, D_r, T_p$)

Input: A collection of users U and their trajectories $TRAJ = \{Traj_k\}$, a time threshold T_r and a distance threshold D_r for stay point detection, and a T_p for trip partition.
Output: A matrix *Cor* of correlation between each pair of locations.
1. S=ϕ; H = ϕ; //temporal variables
2. **Foreach** $u_k \in U$ **do**
3. ST=StayPointDetection($Traj_k$, T_r, D_r); //refer to [8] for details
4. h_k = LocHistPresent(ST); //a sequence of stay points
5. S=$S \cup ST$; // a collection of all users' stay points
6. L = Clustering(S); //detect locations by clustering the stay points
7. **Foreach** $u_k \in U$ **do**
8. h_k = LocHistRepresent(h_k, L); //a sequence of locations
9. H= $H \cup h_k$; //a collection of all users' location histories
10. E=InferUserExperience(U, L, H); //refer to Section 3
11. *Cor*=CalculateLocationCorrelation(L,E, H,T_p); //refer to Section 4
12. **Return** *Cor*.

Figure 3. The framework of our approach

III. INFERRING TRAVEL EXPERIENCE

As shown in Figure 4, we regard a user's stay on a location as an implicitly directed link from the user to that location, i.e., a user would point to many locations and a location would be pointed to by many users. Here, a green point stands for a stay

point, and a gray-circle region denotes a location, which is a cluster of stay points.

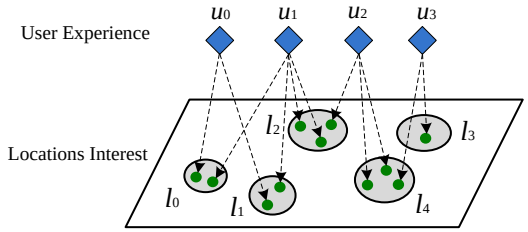


Figure 4. The model inferring user travel experience

User travel experience E and the location interest $\mathcal{T}$ have a mutual reinforcement relationship. The user with rich travel experiences in a region would visit many interesting places in that region, and a very interesting place in that region might be accessed by many users with rich travel experiences. More specifically, a user's travel experience can be represented by the sum of the interests of the locations they accessed; in turn, the interest of a location can be calculated by integrating the experiences of the users visiting it. Using a power iteration method, each user's experience and each location's interest can be calculated.

Given a collection of users U's location histories H, we can build a adjacent matrix M between users and locations. In this matrix, an item r_{ij} stands for the times that u_i has stayed in location l_j, $0 \le i < |U|, 0 \le j < |L|$. For instance, the matrix specified by Figure 4 can be represented as follows.

$$M = \begin{array}{c} \\ u_0 \\ u_1 \\ u_2 \\ u_3 \end{array} \begin{array}{c} \begin{array}{ccccc} l_0 & l_1 & l_2 & l_3 & l_4 \end{array} \\ \begin{bmatrix} 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 2 & 0 & 0 \\ 0 & 0 & 1 & 0 & 2 \\ 0 & 0 & 0 & 1 & 1 \end{bmatrix} \end{array}; \quad (5)$$

Then, the mutual reinforcement relationship of user travel experience $E = (e_0, e_1, \dots, e_m)$ and location interest $\mathcal{T} = (I_0, I_1, \dots, I_n)$ is represented as follows:

$$e_i = \sum_{l_j \in L} r_{ij} \times I_j; \quad (6)$$

$$I_j = \sum_{u_i \in U} r_{ji} \times e_i; \quad (7)$$

where e_i stands for u_i's travel experience and I_j denotes the location interest of l_j. Writing them in the matrix form,

$$\boldsymbol{E} = \boldsymbol{M} \cdot \boldsymbol{\mathcal{T}}, \quad (8)$$

$$\boldsymbol{\mathcal{T}} = \boldsymbol{M}^{\mathrm{T}} \cdot \boldsymbol{E}. \quad (9)$$

If we use $\boldsymbol{\mathcal{T}}_n$ and $\boldsymbol{E}_n$ to denote location interests and travel experiences at the nth iteration, the iterative processes for generating the final results are

$$\boldsymbol{\mathcal{T}}_n = \boldsymbol{M}^{\mathrm{T}} \cdot \boldsymbol{M} \cdot \boldsymbol{\mathcal{T}}_{n-1} \quad (10)$$

$$\boldsymbol{E}_n = \boldsymbol{M} \cdot \boldsymbol{M}^{\mathrm{T}} \cdot \boldsymbol{E}_{n-1} \quad (11)$$

Starting with $\boldsymbol{\mathcal{T}}_0 = \boldsymbol{E}_0 = (1,1,\dots,1)$, we are able to calculate the final results using the power iteration method.

IV. LOCATION CORRELATION

First, we claim that the correlation between two locations does not only depend on the number of users visiting the locations in a trip but also lie in these users' travel experiences. Second, the two locations continuously accessed by a user would be more correlated than those being visited discontinuously. In short, the correlation between two locations can be calculated by integrating the travel experiences of the users U' who have visited them in a trip in a weighted manner. Formally, the correlation between location A and B can be calculated as

$$Cor(A,B) = \sum_{u_k \in U'} \alpha \cdot e_k, \quad (12)$$

where U' is the collection of users who have visited A and B in a trip, $e_k$is u_k's travel experience, $u_k \in U'$, and $0 < \alpha \le 1$ is a dumping factor, which will decrease as the interval between these two locations' index in a trip increases. For example, in our experiment we set $\alpha = 2^{-(|j-i|-1)}$, where i and j are indices of A and B and in the trip they pertain to; i.e., the more discontinuously two locations being accessed by a user ($|i\text{-}j|$ would be big, thus α will become small), the less contribution the user can offer to the correlation between these two location.

As depicted in Figure 5, 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 a node is the index of this node in the sequence. According to Equation (12), from $Trip_1$ we can calculate $Cor(A,B) = e_1$ and $Cor(B,C) = e_1$, since these locations have been consecutively accessed by u_1(i.e., $\alpha = 1$). However, $Cor(A,C) = \frac{1}{2} \cdot e_1$ (i.e., $\alpha = 2^{-(|2-0|-1)} = \frac{1}{2}$) as u_1 traveled to B before visiting C. In other words, the correlation between location A and C that we can sense from $Trip_1$ might not that strong as if they are consecutively visited by u_1. Likewise, we can learn $Cor(A,C) = e_2$, $Cor(C,B) = e_2$, $Cor(A,B) = \frac{1}{2} \cdot e_2$ from $Trip_2$, and infer $Cor(B,A) = e_3, Cor(A,C) = e_3, Cor(B,C) = \frac{1}{2} \cdot e_3$ from $Trip_3$. Later, we can integrate these correlation inferred from each user's trips and obtain the following results.

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

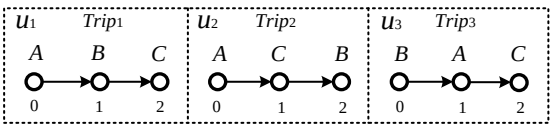


Figure 5. A case calculating the correlation between locations

Figure 6 formally describes the algorithm for inferring correlation between locations. Here, b is a constant, which is set to 2 in our experiment. $|Trip|$ stands for the number of locations contained in the $Trip$and $Trip[i]$ represents the ith location in $Trip$. For example, regarding $Trip_1$, $|Trip| = 3$, $Trip[0] = A$ (the first location), $Trip[1] = B$, $Cor(Trip[0], Trip[1]) = Cor(A,B)$.

Supposing we have n trips in a dataset and the average length of a trip is m, this mining algorithm takes $O(2|L|^2 + \frac{m(m-1)}{2} \cdot n)$ time. So, the overall computing complexity F of our approach is the combination of inferring user travel experience

and calculating location correlation, i.e., $Q = O(2w|L||U| + 2|L|^2 + \frac{m(m-1)}{2} \cdot n)$.

CalculateLocationCorrelation (L, E, H, T_p)

Input: A collection of users' travel experiences E and their location histories H, location collection L, and a threshold T_p for trip partition.
Output: A matrix *Cor* describing the correlation between locations.

1. **Foreach** location $l_p \in L$ **Do**
2. **Foreach** location $l_q \in L, p \neq q$ **Do**
3. $Cor(l_p, l_q) = 0$; //initialize the location correlation
4. **Foreach** $h_k \in H$ **Do** //each user's location history
5. *TP*=TripPartition(h_k, T_p); //partition u_k's location history into trips
6. **Foreach** *Trip* **in** *TP* **Do**
7. **For** $i = 0; i < |Trip|; i++$ // ith location contained in *Trip*
8. **For** $j = i + 1; j < |Trip|; j++$
9. $\alpha = b^{-(j-i-1)}$; // dumping factor, b is a constant
10. $Cor(Trip[i], Trip[j]) += \alpha \cdot e_k$;
11. **Foreach** $l_p \in L$ **Do**
12. **Foreach** $l_q \in L, p \neq q$ **Do** //normalization
13. $Cor(l_p, l_q) = Cor(l_p, l_q) / \|Cor(l_p, l_0), \dots, Cor(l_p, l_{|L|-1})\|_1$
11. **Return** Cor;

Figure 6. Algorithm learning the correlation between locations

V. CASE STUDY

Notation: As shown in Equation (5), we have a matrix *M* describing the relationship between each user and each location. Here, we can regard the times an individual has stayed at a location as their implicit ratings on the location. The ratings from a user u_p, called an *evaluation*, is represented as an array $R_p = \langle r_{p0}, r_{p1}, \dots, r_{pn} \rangle$, where r_{pj} is u_p's implicit ratings (the occurrences) in location $l_j, 0 \leq j < |L|$. $S(R_p)$ is the subset of the R_p, $\forall r_{pj} \in S(R_p)$, $r_{pj} \neq 0$, i.e., the set of items (locations) which has been rated (visited) by u_p. The average of ratings in R_p is denoted as $\overline{R_p}$, and the number of elements in a set S is $|S|$. The collection of all *evaluations* in the training set is $\mathcal{X}$. $S_j(\mathcal{X})$ means the set of evaluations containing item j, $\forall R_p \in S_j(\mathcal{X})$, $j \in S(R_p)$. Likewise, $S_{i,j}(\mathcal{X})$ is the set of evaluations simultaneously containing item i and j.

A. Baseline Schemes: Traditional CF Models

Collaborative filtering is a well-known model widely used in recommendation systems. CF model can be partition into two categories; the user-based and item-based inference methods.

1) The Pearson correlation-based CF. The Pearson correlation reference scheme [6] is the most popular and accurate user-based CF model using the similarity between users, $sim(u_p, u_q)$, to weight the ratings from different individuals. Equation (15) and (16) give a formal description on calculating $P(r_{pj})$, the predicted u_p's ratings on location l_j. Refer to [7] for details.

$$sim(u_p, u_q) = \frac{\sum_{i \in S(R_p) \cap S(R_q)} (r_{pi} - \overline{R_p}) \cdot (r_{qi} - \overline{R_q})}{\sqrt{\sum_{j \in S(R_p) \cap S(R_q)} (r_{pj} - \overline{R_p})^2 \cdot \sum_{j \in S(R_p) \cap S(R_q)} (r_{qj} - \overline{R_q})^2}} \quad (15)$$

$$P(r_{pj}) = \overline{R_p} + \frac{\sum_{R_q \in S_j(\mathcal{X})} sim(u_p, u_q) \times (r_{qj} - \overline{R_q})}{\sum_{R_q \in S_j(\mathcal{X})} sim(u_p, u_q)}; \quad (16)$$

As the number of users in a system is much larger and increases much faster than the number of items, the user-based CF models are not that efficient than the item-based methods.

2) The Slope One algorithms [7] 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 $R_p \in S_{j,i}(\mathcal{X})$, we consider the average deviation of item i with regard to item j as Equation (17).

$$dev_{j,i} = \sum_{R_p \in S_{j,i}(\mathcal{X})} \frac{r_{pj} - r_{pi}}{|S_{j,i}(\mathcal{X})|}, \quad (17)$$

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.

$$P(r_{pj}) = \frac{1}{|\mathcal{W}_j|} \sum_{i \in \mathcal{W}_j} (dev_{j,i} + r_{pi}), \quad (18)$$

where $\mathcal{W}_j = \{i | i \in S(R_p), i \neq j, |S_{j,i}(\mathcal{X})| > 0\}$ is the set of all relevant items. Further, the number of evaluations simultaneously contain two items has been used to weight the prediction regarding different items. 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 better predictor for item *A* than u_p's ratings of item *C* is.

$$P(r_{pj}) = \frac{\sum_{i \in S(R_p) \wedge i \neq j} (dev_{j,i} + r_{pi}) \cdot |S_{j,i}(\mathcal{X})|}{\sum_{i \in S(R_p) \wedge i \neq j} |S_{j,i}(\mathcal{X})|}. \quad (19)$$

B. Our Location Correlation-Based CF Model

In this case study, we integrated the location correlation into the Slop One algorithm to achieve a more effective and accurate item-based CF model, which can predict a user's interest in a location they have not been.

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. In contrast to the number of observed ratings (i.e., the number of people who have visited two locations) used by the weighted Slope One algorithm, the location correlation mined from multiple users' location histories carries more semantic meanings. Formally, our approach can be represented as

$$P(r_{pj}) = \frac{\sum_{i \in S(R_p) \wedge i \neq j} (dev_{j,i} + r_{pi}) \cdot c_{ji}}{\sum_{i \in S(R_p) \wedge i \neq j} c_{ji}}, \quad (20)$$

where c_{ji} denotes the correlation between location l_i and l_j, and $dev_{j,i}$ is still calculated as Equation (17).

Using Equation (20), we can predict an individual's ratings on the locations they have not accessed, and then rank these locations in terms of the predicted ratings. Later, the top *n* locations with relatively high ratings can be recommended to the individual.

VI. EXPERIEMENT

A. Settings

Dataset: Carrying a GPS-enabled device, 112 users (49 females and 63 males) recorded their outdoor movements with GPS logs from May 2007 to Dec. 2008. As a result, the total distance of the GPS logs exceeded 254,030,449 kilometers,

and the total number of GPS points reached 9,432,747. Most parts of this dataset were created in Beijing, China, and other parts covered 36 cities in China as well as a few cities in the USA, South Korea, and Japan. Considering privacy issues, we use these datasets anonymously.

Stay point detection: In this experiment, we set T_r to 20 minutes and D_r to 250 meters for stay point detection. Refer to paper [11] for more justifications.

Clustering: We use a density-based clustering algorithm, OPTICS (Ordering Points To Identify the Clustering Structure), to cluster the extracted stay points into some geospatial regions. In the evaluation step, we set the core-distance to 100 meters and configure the minimum number of points to 8.

Trip partition: In the experiment, we investigate the performance of our methods changing over T_p.

Using these locations and users' trips, a location graph can be constructed as illustrated in Figure 10 B). Here, a node stands for a location and an edge between two locations denotes that there is at least one trip passing the two locations. Table II shows the detailed information of this region.

Subjects: We invited 23 subjects from the 112 users to participate in a user study, which evaluates the effectiveness of the two applications powered by the location correlation mined from the given GPS dataset. These subjects have logged their location histories over a year and have been in the selected region for more than 6 years, i.e., they know this region well.

B. Evaluation Approach

Strategy: Respectively using our location correlation-based CF model and two baseline methods, we infer each subject's interest level (ratings) in each location that the subject has not visited. Then, the top 10 locations with relatively high ratings are retrieved as the recommendation for the subject. Later, the subject can view the recommendation on a Web map, and offer a rating on each recommended location with a level described in Table III.

Table III. Users' interest levels in a location

Ratings	Explanations
4	I'd like to plan a trip to that location.
3	I'd like to visit that location if passing by.
2	I have no feeling, but don't oppose others to visit it.
1	This location does not deserve to visit.

Measurements: First, we evaluate the ranking performance of the top 10 locations recommended to each subject using *nDCG* and *MAP*. Later, we calculate an average *nDCG* and *MAP* by aggregating the results from multiple subjects.

Baseline Scheme: The Pearson-based CF model and the weighted Slope One. Refer to Section 5.2.1 for details.

Measurements: *MAP* is the most frequently used summary measure of a ranked retrieval run. In our experiment, it stands for the mean of the precision score after each good recommendation is retrieved. Regarding the mobile tour guide, a retrieved location is a good recommendation if its integrated ground truth equals to 3. With regard to the personalized recommendation, a recommended location is deemed as a good recommendation if its interest level rated by a subject is greater than 2. For instance, $G =< 4,1,3,1,2,1,1,3,1,1 >$is a rating vector for the top 10 locations recommended to a subject; the *MAP* of the *G* is computed as

$$MAP = (1 + 2/3 + 3/8)/3 = 0.681$$

nDCG is used to compute the relative-to-the-ideal performance of information retrieval techniques. The discounted cumulative gain of a rating vector *G* is computed as follows: (we set b = 2 here)

$$DCG[i] = \begin{cases} G[1], & if\ i = 1 \\ DCG[i-1] + G[i], & if\ i < b \\ DCG[i-1] + \frac{G[i]}{log_b i}, & if\ i \geq b \end{cases} \quad (21)$$

Given the ideal discounted cumulative gain *DCG'*, then *nDCG* at *i*-th position can be computed as $NDCG[i] = DCG[i]/DCG'[i]$.

C. Results

1) Effectiveness

Using the average *NDCG* and *MAP*, Table V compares the effectiveness of different methods in conducting the personalized location recommendation. Clearly, our approach (*Experience + Sequentiality*) outperforms the weighted Slope One algorithm (T-Test of *NDCG@5*, p=0.0053<0.01; T-Test of *MAP*, p=0.0049<0.01). Although our method is slightly weaker than the Pearson correlation-based CF model in terms of the average *NDCG* and *MAP*, the T-test result (*NDCG@5*, p=0.678>>0.01; *MAP*, p=0.741>>0.01) shows that the advantage of the Pearson correlation is not significant and not clear. In other words, some users thought the recommendation generated by our method is even better than that of the Pearson correlation-based scheme. Thus, we can claim that at least our method is as effective as the Pearson correlation-based one.

Table I. Effectiveness of different methods in performing the personalized location recommendation

	Ours	The Pearson Correlation-Based	The Weighted Slope One
NDCG@5	**0.840**	0.862	0.762
NDCG@10	**0.922**	0.938	0.891
MAP	**0.798**	0.804	0.665

2) Efficiency

Suppose we have a GPS dataset generated by *T* users. From this dataset, we discover *k* locations and *n* trips; the average length (number of locations) of a trip is *m*. Thus, to predict a user's interest level in a location, the upper bounds of computing complexity of different methods are as follows:

The Pearson correlation-based CF model: $O(k \times (T-1)^2)$;

The Weighted Slope One algorithm: $O(T \times k(k-1))$;

Our method (*Exp + Seq*): $O(T \times k(k-1) + Q)$,

where $Q = 2wkT + 2k^2 + m(m-1)n/2$ is the total computing complexity of inferring the correlation, and *w* is the iteration times.

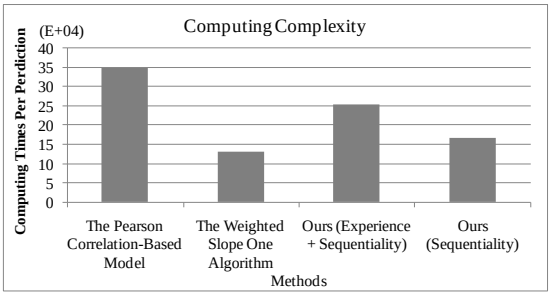


Figure 14. Average computing complexity in computing a prediction

Given the GPS dataset, Figure 14 depicts the upper bound of computing complexity of different methods in calculating a prediction. Clearly, our method is much more efficient than the Pearson correlation-based CF model, while being slightly slower than the weighted Slope One algorithm.

VII. RELATED WORK

Mining human location history has attracted intensive attention in past years [5][20]. Previously, most work focuses on detecting significant locations of a user [1][7], predicting the user's movement among these locations and recognizing user-specific activities at a location [7] [10], etc. Recently, Gonotti et al. [4] mined similar sequences from users' moving trajectories; Mamoulis et al. [12] proposed a framework for retrieving maximum periodic patterns in spatio-temporal data. MSMLS [8] predicts where a driver may be going as a trip progresses. Eagle et al [3] aimed to recognize the social pattern in daily user activity from the dataset collected by Bluetooth-enabled mobile phones. Zheng et al. [17][18] classified people's GPS trajectories into different categories of transportation modes, such as driving and taking a bus. Instead of understanding user behavior, we aim to mine people's location histories to learn the correlation between locations.

Zheng et al. [19][20] performed a generic travel recommender that provides a user with the top interesting locations and travel sequences mined from GPS trajectories. In contrast to this work, we conduct a personalized location recommendation, which predicts an individual's interests in an unvisited location based on her location history and that of others. Li and Zheng et al. [11] mined the similarity between individuals from their GPS trajectories, and incorporate this user similarity into a personalized location recommender [21]. Differing from this work, we do not estimate the similarity between each pair of users, which causes heavy computation.

Co-location pattern mining [13] [6] [12] [14] [16] aims to find classes of spatial objects that are frequently located together. The major differences between these work and ours lie in two aspects: 1) We infer the correlation between each pair of locations rather than the co-located patterns of location categories. 2) We use human behaviors to estimate the correlation between two locations rather than the geospatial distance between them.

VIII. CONCLUSION

In this paper, by considering the user travel experience and the visited sequence between locations, we mine the correlation between locations from people's location histories. Beyond the geo-distance and the category relationship between locations, the correlation describes the relationship between locations in the space of human behavior. Using the correlation, we conduct a personalized location recommender, which is evaluated by a real-world GPS dataset collected by 112 users over 1.5 years. As a result, our recommender is more effective than the weighted Slope one algorithm with a slightly additional computation. In addition, in contrast to the Pearson correlation-based CF model, our method is much more efficient while keeping the similar effectiveness.

REFERENCES

[1] Ashbrook, D., and Starner, T. Using GPS to learn significant locations and predict movement across multiple users. Personal and Ubiquitous Computing 7(5), 275-286.

[2] Adomavicius, G. and Tuzhhilin, A. Toward the Next Generation of Recommender Systems: A Survey of the State-of-the-Art and Possible Extensions. IEEE Transaction on Knowledge and Data Engineering. 17, 6, 734-749.

[3] Eagle, N. et al. Reality mining: sensing complex social systems. Personal and Ubiquitous Computing 10(4), 255-268.

[4] Gonotti, F., et al. Trajectory pattern mining. In Proc. of KDD'07, pp. 330-339.

[5] Gonz áez, M. C., Hidalgo, C. A., Understanding individual human mobility patterns. Nature 453 (June 2008), 779-782.

[6] Huang, Y., Shekhar, S., and Xiong, H.. Discovering co-location patterns from spatial datasets: A general approach. TKDE, 16(12):1472-1485.

[7] Hariharan, R. et al. Project Lachesis: Parsing and Modeling Location Histories, In Proc. of GIScience 2004, pp. 106-124.

[8] Krumm, J.et al. Predestination: Inferring Destinations from Partial Trajectories. In Proc. of the Ubicomp'03, pp. 243-260.

[9] Lemire D. and Maclachlan A. Slope One Predictors for Online Rating-Based Collaborative Filtering. In Proc. of SDM 2005.

[10] Liao, L., et al. Building Personal Maps from GPS Data. In proc. of IJCAI MOO05, pp. 249-265.

[11] Li, Q. and Zheng, Y. et al. Mining user similarity based on location history. In Proc. of GIS'08. pp.1-10

[12] Mamoulis, N. et al. Mining, Indexing and Querying Historical Spatiotemporal Data. In Proc. of KDD'04, pp. 236-245.

[13] Morimoto, Y.. Mining frequent neighboring class sets in spatial databases. In Proc. of SIGKDD, 2001, pp. 353-358.

[14] Vladimir, E. and Lee, I.. Data mining techniques for autonomous exploration of large volumes of geo-referenced crime data. In Proceedings of Geocomputation, 24-26, 2001.

[15] Xiao, X., Xie, X., Luo, Q. Density-based co-location pattern discovery. In Proc. of GIS'08. pp.11-20.

[16] Zhang, X., N. Mamoulis, D. W. Cheung, and Y. Shou. Fast mining of spatial collocations. In Proc. of SIGKDD, pp. 384-393, 2004.

[17] Zheng, Y., Li Q., Xie X., Ma, W. Y.. Understanding mobility based on GPS data. In Proc. of Ubicomp'08, pp. 312-321.

[18] Zheng, Y., Liu, L., Wang, L. Xie, X. Learning transportation modes from raw GPS data for geographic applications on the Web. In Proc. of WWW 2008, pp. 247-256.

[19] Zheng, Y., Zhang, L., Xie X., Ma, W. Y. Mining interesting locations and travel sequences from GPS trajectories for mobile users. In Proc. of WWW2009, 2009, pp. 791-800.

[20] Zheng, Y. Chen, Y. Xie, X., Ma, W., Y. GeoLife2.0: A Location-Based Social Networking Service, In Proc. of MDM 2009, pp.357-358.

[21] Zheng, Y. Zhang, L., Ma, Z. Xie, X., Ma, W., Y. 2010b. Recommending friends and locations based on individual location history. To appear in ACM Transaction on the Web