Information Sciences 181 (2011) 1264–1284

Contents lists available at ScienceDirect

Information Sciences

journal homepage: www.elsevier.com/locate/ins

A personal route prediction system based on trajectory data mining

Ling Chen [a,*], Mingqi Lv [a], Qian Ye [a], Gencai Chen [a], John Woodward [b]

[a] College of Computer Science, Zhejiang University, 38 Zheda Road, Hangzhou 310027, PR China
[b] The University of Nottingham Ningbo China, Ningbo 315100, PR China

ARTICLE INFO

Article history:
Received 31 July 2009
Received in revised form 1 October 2010
Accepted 27 November 2010
Available online 7 December 2010

Keywords:
Data mining
GPS
Route pattern
Route prediction
Privacy

ABSTRACT

This paper presents a system where the personal route of a user is predicted using a probabilistic model built from the historical trajectory data. Route patterns are extracted from personal trajectory data using a novel mining algorithm, Continuous Route Pattern Mining (CRPM), which can tolerate different kinds of disturbance in trajectory data. Furthermore, a client–server architecture is employed which has the dual purpose of guaranteeing the privacy of personal data and greatly reducing the computational load on mobile devices. An evaluation using a corpus of trajectory data from 17 people demonstrates that CRPM can extract longer route patterns than current methods. Moreover, the average correct rate of one step prediction of our system is greater than 71%, and the average Levenshtein distance of continuous route prediction of our system is about 30% shorter than that of the Markov model based method.

1. Introduction

Most people make journeys of a repetitive nature, for example to and from a place of work. This fact makes predicting the route of a person based on his previous history achievable. With the popularity of the handheld devices containing the global positioning system (GPS), e.g. some types of mobile phones and personal digital assistants (PDAs), the trajectory of a person can be easily obtained for the prediction of personal route.

Route prediction allows certain services to improve their quality. For example, if Intelligent Transportation Systems (ITSs) have access to the routes of vehicles in advance, these systems can provide route-specific traffic information [20] and better advice to drivers [27]. Researchers from Nissan showed that it was possible to improve hybrid fuel economy by up to 7.8% if the route of a vehicle was known in advance [8]. Furthermore, route prediction can also be used to provide better Location-Based Services (LBSs), as well as to improve social networks and knowledge dissemination [22]. For example, given the predicted routes of shoppers, advertisement messages can be targeted at customers who are likely to pass certain points. This approach is more efficient than traditional approaches based on location and proximity [16]. In general, as a kind of mobility service context [4], route prediction can be employed to improve context-aware applications [7].

These types of applications have motivated researchers to explore the possibilities of route prediction. Self-reporting [6] is a simple approach to obtain information about the routes people will take, however it places an additional burden on users. Currently, there are many different approaches to route prediction, which generally consist of three sequential parts: first route abstraction, followed by route pattern mining, and finally route prediction. However, most existing work has focused on predicting the routes of vehicles, rather than individual people. The route prediction of people is different than the route prediction of vehicles for the following reasons. Firstly, the routes taken by vehicles are restricted to journeys over road

* Corresponding author. Tel.: +86 13606527774.
E-mail address: lingchen@cs.zju.edu.cn (L. Chen).

0020-0255/$ - see front matter © 2010 Elsevier Inc. All rights reserved.
doi:10.1016/j.ins.2010.11.035

networks, while people may roam more freely. Secondly, the journeys taken by people include journeys made in vehicles but also journeys made without vehicles. Finally, vehicle trajectory data can be naturally segmented into individual trips as indicated by starting the engine, whereas there is no such explicit marker in personal trajectory data.

In this paper, we focus on personal route prediction rather than vehicle route prediction. Compared to vehicle route prediction, personal route prediction faces three unique challenges. First, the movement of people is more diverse than that of vehicles. Usually, the speed of people is more variable than that of vehicles, since they may take different modes of transportation. Moreover, most people spend much of their time in indoors where GPS signals are weaker. GPS signal drift is not so much of an issue for vehicles as they do not operate indoors. Second, personal routes are more varied as they are not constrained to a road network. Third, the privacy of data is a significant issue as personal route information contains the details of the exact whereabouts of a person. Solutions involve conducting prediction processes on personal mobile devices, and transmitting only identification numbers which do not contain any geographic information. An accompanying problem is the limited computational capability of mobile devices. Unlike the computational devices often found in vehicles, the computational capability of handheld mobile devices is restricted by their physical size and battery life, which both limit the computational complexity of a route prediction system.

To counter the challenges mentioned above, we propose the following three approaches. First, the system utilizes GPS, rather than cellular positioning, which allows more accurate position data to be obtained. The program can adaptively adjust the sampling rate according to the recording demands dictated by the movement of the user. Five data filters designed to remove outliers from the data make the system suitable for recording the large diverse of personal trajectory data. A novel mining algorithm, Continuous Route Pattern Mining (CRPM) is proposed to tolerate disturbances in trajectory data and allow better extraction of route patterns. Moreover, an incremental mining strategy is also available to support the long term running of the system.

Second, two decision tree based prediction algorithms, a basic one and a heuristic one, are proposed to provide offline and online route predictions. With offline prediction, both the next position and the route of a user are predicted before the trip begins. With online prediction, both the next position and the route of a user are predicted during the journey.

Third, the client–server architecture of the system allows the users to take control of their personal information. The server takes most of the computational load but cannot access the real geographic information which remains on the client side (i.e. the mobile device). Thus, it is impossible for the personal routes of users to be reconstructed on the server side of the system.

The remainder of the paper is organized as follows. Section 2 gives an overview of the related work. Section 3 demonstrates the architecture of our prediction system. Section 4 describes the details of the system in terms of data collection, route segmentation, and data cleaning. Section 5 presents the novel mining algorithm, CRPM. Section 6 demonstrates how the proposed prediction algorithms work. Section 7 provides the experimental results and discussion. Section 8 concludes the paper and gives some suggestions for potential future work.

2. Related work

According to whether a road network is exploited or not, current route abstraction approaches can be divided into two main categories, road network based approaches and geometric based approaches. Road network based approaches [4,14,15,25] assume journeys take place on a network, and usually project positions onto line segments in a road network and employ polylines to represent routes. Whereas with geometric based approaches [5,12,13,19], positions are clustered in regions and routes are represented as sequences of regions.

Several different approaches have been identified to tackle route pattern mining. Cao et al. [5] proposed a substring tree structure and improved level-wise mining algorithm to obtain route patterns efficiently. Giannotti et al. [11] presented the MiSTA algorithm, which is based on the PrefixSpan algorithm [24] and can find patterns from temporally annotated sequences. They also proposed a variant of the algorithm to perform trajectory pattern mining [12]. Gidófalvi and Pedersen [13] presented a projection based algorithm to find long, sharable route patterns. Karimi and Liu [15] proposed a predictive location model, and Simmons et al. [25] used a Hidden Markov Model (HMM) to build route models. Froehlich and Krumm [9] proposed a Hausdorff distance based clustering approach to obtain route patterns.

Currently, most research regarding route prediction focused on trajectory data obtained from vehicles. However, if the trajectory data is from people, these existing approaches would face a substantial challenge: how to deal with the relative diversity of trajectory data regarding personal movement. As discussed in Section 1, the movement of a person is likely to be more diverse than that of a vehicle, as a person can take different modes of transportation. Also, people spend the vast majority of their time indoors, and this is the reverse of the situation encountered with vehicles. Recently, Ye et al. [28] proposed a mining approach based on an individual's location history, but the approach is designed to extract significant place based life patterns, not route patterns. Froehlich and Krumm [9] proposed methods to counter the variability of vehicles' trajectory data, however, the degree of variability of vehicles' trajectory data is much lower than that of personal trajectory data.

Besides route prediction, some research has focused on the goal of destination prediction. For example, Krumm and Horvitz [17] proposed the predestination method, which uses the history of a driver's destinations and driving behaviors to predict the destination of the driver as a trip progresses. Tanaka et al. [26] proposed a destination prediction method based

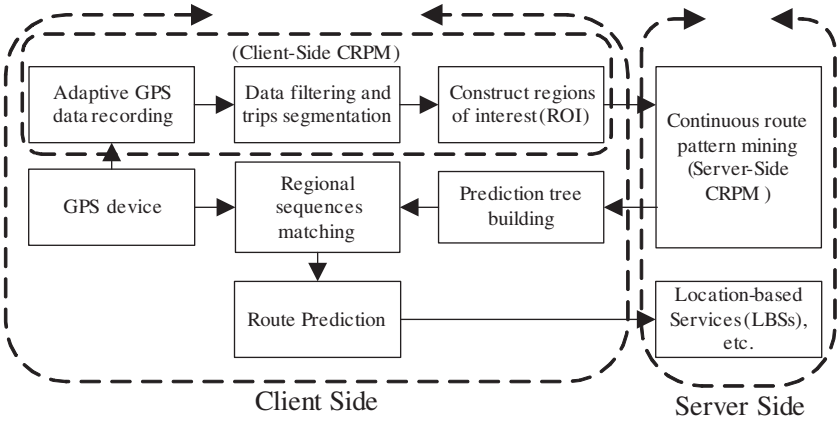

Fig. 1. The architecture of the personal route prediction system.

on driving trajectories and driving contexts. The HMM based approach proposed by Simmons et al. [25] can predict not only routes but also destinations.

Route prediction approaches have been broadly applied in ITSs and LBSs. Besides the applications mentioned in Section 1, Karbassl and Barth [14] added route prediction and estimated time-of-arrival functions to a multiple-station carsharing system. Nakata and Takeuchi [23] took the time periods into consideration and realized travel time prediction based on a probe-car system. Gidófalvi and Pedersen [13] applied route pattern mining in an intelligent ride-sharing system.

The protection of the privacy of personal trajectory data is another important issue which has received particular attention from a number of academics. For example, Gedik and Liu [10] proposed an architecture which used a flexible privacy personalization framework to support location k-anonymity for a wide range of mobile clients with context-sensitive privacy requirements. The GeoPKDD[1] project took the privacy issue as one of its major research topics. In this project, Abul et al. [1] sanitized trajectory data in a database by coarsening some sensitive trajectories which may lead to the disclosure of personal information. These researchers also devised heuristics which can preserve the usability of the coarsened database. They then conducted research on (k,δ)-anonymity for a database containing information regarding moving objects, and proposed a greedy algorithm based on clustering [2]. These approaches support the privacy of data by hiding the precise identifiable positions of users from public view. Laasonen [18] also discussed the privacy issue in his work in which a GSM network was exploited to obtain the position information of users.

Existing research has attempted to protect private information in two main different ways. Some researchers (e.g. Laasonen [18]) proposed to keep the sensitive data from being publically accessible, which completely guarantees privacy but requires a high amount of storage capacity and computational resources on the personal devices. Other researchers (e.g. Gedik and Liu [10], Abul et al. [1,2]) proposed to hide some identifiable trajectories to protect privacy. Adding or removing some artificial data can reduce the chances of personal information from being detected from the sanitized data. Based on such a mechanism, public resources can be utilized to conduct further processing.

3. Architecture

Inspired by the structure of route prediction systems for vehicles, our personal route prediction system consists of three major modules: (1) the data preparation module collects the personal trajectory data from a GPS device and does data filtering; (2) the mining module extracts complete route patterns of users, which is achieved with the CRPM algorithm; (3) the personal route prediction module chooses most suitable route patterns to build a probability tree for prediction.

The work of Laasonen [18] inspired the client–server architecture of our system, which can protect personal privacy and reduce the computational load on the client (i.e. handheld devices). Consider that the volume of personal trajectory data is limited by the sampling rate of GPS and the storage capacity of handheld devices has increased significantly, sensitive geographic information is all stored in personal mobile devices. Under this architecture, most of the computation is performed on the server, but it does not have sufficient information to reconstruct the routes of users. The users can keep all the prediction results in their handheld devices as private data, and they can decide on whether to send these results to LBSs providers for better services or not.

Fig. 1 presents the client–server architecture of the personal route prediction system. The client side records data from a GPS device, then filters outliers, and finally segments data into trips. Then, the client constructs regions of interest (ROIs) from trajectory data, and sends Regional-Temporal Sequences (RTSs, see Section 5) to the server. The server employs the

[1] http://www.geopkdd.eu/.

CRPM algorithm to extract route patterns, and transmits them to the client for route prediction. The client builds a prediction tree based on the received route patterns, and predicts the route of the user by regional sequence matching. Users can choose to send their future routes to the server to get specific services. Since RTSs do not contain real geographic information, the communication between the client and the server will not disclose the privacy of users.

Based on the architecture of the system, it is possible to apply sophisticated restrictions to the utilization of prediction results. For example, users can send their personal future routes during daytime (i.e. in office hours under their professional activities) and choose not to send information in the evenings (i.e. outside office hours under their private activities), as their activities later in the day might be more private. Thus the user has control of what data is sent from his private mobile device to a public server.

4. Data preparation

The data preparation module is designed to record personal trajectory data from a GPS device. An adaptive recording program has been developed to counter the diversity of personal movement, and five data filters are utilized to clean the trajectory data.

4.1. Data collection

The high variability of personal trajectories can be recorded more efficiently and effectively if the positions are recorded at different rates, with the recording rate being high when there is a lot of movement, and low when there is little movement. A mobile phone program written in Python connects to a GPS device via Bluetooth and records the positions of the user along with time stamps which make up the trajectory data. Users are required to turn on their mobile phones and GPS devices in the morning, carry these devices with them during the day, and turn them off at night. Fig. 2 shows the state chart of the recording program. The program can automatically recover from a Bluetooth failure, and the sampling interval is adaptively adjusted according to the speed of the user. A minimum sampling rate is defined to ensure the recording of a sufficient number of positions, and the maximum sampling rate is determined by the GPS device.

4.2. Data filtering

Due to the uncertainty of the data obtained from GPS devices, outliers need to be removed before route pattern extraction is begun. Besides, route pattern mining also requires that recorded data is segmented into trips, which is not a problem for the data of vehicles which is naturally segmented when the engine is switched off. However, asking users to manually turn on and off their GPS devices several times a day for the purpose of trip segmentation would drastically decrease the usability of the system and the reliability of the data.

Five filters have been developed to remove outliers and segment trajectory data into trips according to different criteria. A single parameter is associated with each filter and the units of each parameter are described in Section 7.

4.2.1. Duplication filter

If the distance between two consecutive positions is smaller than a threshold λ_{dup}, the duplication filter removes the second position.

4.2.2. Speed filter

It is assumed that individuals move at a constant speed between two consecutive positions, and there is a reasonable speed range for individuals. The speed filter removes the second position if the speed between two consecutive positions is unreasonable (i.e. the speed exceeds λ_{speed}).

4.2.3. Acceleration filter

The acceleration filter removes the positions that contribute to an acceleration of over a threshold λ_{accel}.

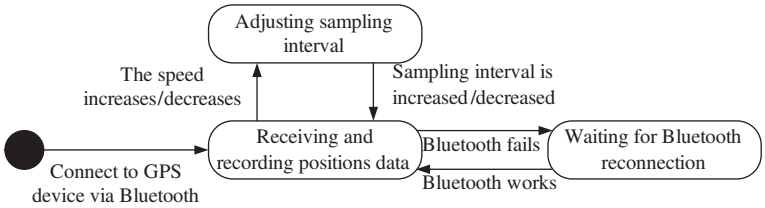

Fig. 2. The state chart of the recording program.

4.2.4. Total distance filter

The total distance filter is designed to remove the redundant position data which is recorded when users are inside buildings. Given a window size δ, we first calculate the centroid for each δ-sequential-position sequence in a trip. Then the maximum distance between these centroids is calculated to estimate whether the trip contributes to a reasonable movement distance. A trip will be dropped, if its maximum centroid distance is shorter than a threshold λ_{dis}.

4.2.5. Angle filter

The angle filter is used to smooth the trajectory data. Reading in three sequential positions A, B, C, it calculates the angle of $\angle$ABC. Since the sampling interval between two contiguous positions is relatively small, if an angle of $\angle$ABC is smaller than a threshold λ_{ang}, the position B is probably an outlier, because people are unlikely to take sharp turns during a few seconds. The angle filter is scheduled to run repeatedly for each trip until there are no outliers left to be removed.

The criterion for splitting GPS data into trips is the time gap between two consecutive positions, as a long period of no movement indicates the end of a trip. Fig. 3 shows the algorithm of data filtering and segmentation. In the algorithm, T is the array containing all recorded GPS data of a user, λ_{time_gap} is the time threshold used to segment GPS data into separate trips, λ_{trj_cnt} is the threshold used to remove short trips. *Funct* is one of the data filtering functions described above, which returns *true* if the positions comply with the restrictions of the data filters. Otherwise, the *Funct* returns *false*, and the corresponding positions will be removed.

The data filtering process can remove most outliers and greatly reduce the amount of GPS data which needs to be stored. The output trips are organized as Spatial-Temporal Sequences (STSs), which are in the form $\langle(x_0,y_0,t_0),\ldots,(x_k,y_k,t_k)\rangle$, where (x_i,y_i) is the longitude–latitude coordinate at time stamp $t_i(i=0,\ldots,k)$.

5. The mining of route patterns

In this section, we describe how to extract route patterns from the segmented trajectory data. The CRPM algorithm is performed in a distributed manner. As shown in Fig. 4, the pseudocode from line 1 to line 9 are executed on the client, and the pseudocode from line 10 to line 14 are executed on the server.

Our mining approach starts by using sequences of cells to abstract personal trajectory data (line 1 in Fig. 4). Given the STSs of a user, the algorithm equally divides the area visited by the user into cells, and trips can be represented by sequences of cells. Linear interpolation is used to ensure that all the cells that users have passed can be extracted (line 2 in Fig. 4). In Fig. 4, Cell-Temporal Sequence (CTS) is in the form $\langle(C_0,t_0),\ldots,(C_k,t_k)\rangle$, where C_i is the cell at time stamp $t_i(i=0,\ldots,k)$.

Regions-Of-Interest (ROIs) are frequently visited regions, and are constructed based on visiting density (from line 3 to line 8 in Fig. 4), which is similar to the work of Giannotti et al. [12]. A threshold $\lambda_{density}$ is employed to judge whether a cell is frequently accessed. The neighboring cells with similar density are merged to regions. Line 9 in Fig. 4 is to transform STSs to RTSs based on ROIs.

Definition 1. A *Regional-Temporal Sequence (RTS)* is a sequence in the form $\overline{S}=\langle S_0,\ldots,S_k\rangle$, each of the item in the sequence is a couple $S_i=(R_i,T_i)$ where region $R_i(i=0,\ldots,k)$ is a set of merged neighboring cells, and $T_i(i=0,\ldots,k)=\left(T_{in}^{(i)},T_{out}^{(i)}\right)$, $\forall_{0\leqslant i<k}T_{in}^{(i)}<T_{out}^{(i)},T_{out}^{(i)}\leqslant T_{in}^{(i+1)}$.

R_i is the ith ROI constructed at the previous step. $T_{in}^{(i)}$ and $T_{out}^{(i)}$ are the times the user enters and leaves R_i respectively. Given two couples S_n and S_m (m not equal to n) in a RTS, although R_n may equal R_m (as the user may revisit the same ROI in a single trip), T_n never equals T_m. In other words, the spatial component of a RTS may repeat, but the temporal component always

Algorithm 1 datafilter (T, λ_{time_gap}, λ_{trj_cnt}, *Funct*)

```
1.  T_tmp = Ø;
2.  for each route r_i in T do
3.      r_tmp = Ø;
4.      for each position p_j in r_i do
5.          if (Funct(p_j, r_i) returns true) && (Time(p_j)-Time(p_j-1) <= λ_time_gap)
6.              r_tmp = Append (r_tmp, p_j);
7.          else if (Time(p_j)-Time(p_j-1) > λ_time_gap)
8.              split_trip(r_i, r_tmp)
9.      if (Size(r_tmp) > λ_trj_cnt)
10.         T_tmp = Append(T_tmp, r_tmp);
11. return T_tmp;
```

Fig. 3. The algorithm for data filtering and segmentation.

Algorithm 2 CRPM (STS, λ_{time})

```
1.  grid = build_grid(STS);
2.  CTS = interpolation(grid, STS);
3.  for each item GTelem_i in CTS do
4.      compute_density(grid, GTelem_i);
5.  if (incremental mining) do
6.      load_past_trajctories(STS, dbFile);
7.      update_density(grid, gridFile);
8.  Regions = bound_regions_of_interest(grid);
9.  RTS = translation(CTS, Regions);
10. PS = {RTS};
11. while (PS is not empty) do
12.     P = pop(PS);
13.     PS' = extend_projection(P, λ_time);
14.     PS = Append(PS, PS');
```

Fig. 4. The algorithm for route pattern mining.

increases. Moreover, $T_{out}^{(m)} = T_{in}^{(n)}$ means item n occurs immediately after item m (i.e. $n = m + 1$). The entering time $T_{in}^{(i)}$ of region R_i is set as the time stamp of the first position in the region, while the leaving time $T_{out}^{(i)}$ is set as the last time stamp in that region. Fig. 5 shows examples of trips that are represented in CTS (top) and RTS (bottom).

The mining preprocessing described in Section 4 is performed on mobile devices. The RTSs are then transmitted to the server. Since the system only sends the IDs of regions in RTSs to the server, the information received by the server is insufficient to recover the actual trips, and thus the privacy of users' trajectory data is protected. We first define continuous route as follows:

Definition 2. A RTS represents a *continuous route* if and only if $\forall_{0 \leqslant i < k} T_{in}^{(i+1)} - T_{out}^{(i)} \leqslant \lambda_{time}$ where λ_{time} is the time interval threshold.

In the definition, λ_{time} is the maximum tolerated interval between two consecutive ROIs in a continuous route. There are two extremes concerning this parameter. If $\lambda_{time} = 0$, the mining results equal the results of a substring mining algorithm, e.g. substring tree mining [5]. If $\lambda_{time} \to +\infty$, the mining results equal the results of a general sequential pattern mining algorithm, e.g. PrefixSpan [24].

One of the advantages of the proposed algorithm is that mined routes are robust to slight disturbances in trajectory data. Fig. 6 shows an example, in which there are three trips starting from the top-left corner and terminating in the bottom-right corner. Assume the minimum support threshold $\lambda_{min\_sup} = 3$, the longest sequential pattern that substring mining algorithm can find is $\langle B, C, D, E, F\rangle$. However, the three trips actually start from the same area (i.e. the building in cell A), and they take different detours because there is an obstacle in front of the building (e.g. a parterre). If the time spent on passing the obstacle is less than an acceptable interval, a longer route pattern, $\langle A, B, C, D, E, F\rangle$, will emerge from the data. The blank between cells A and B will not affect the continuity of this route pattern, as the distance is short in the real world. The disturbance mentioned here could also be caused by a number of other factors, for example, obstacles on the road, GPS signal drift, and short term device failure, etc. The design of the CRPM algorithm can improve the reliability of route pattern mining as it is more robust against the sources of such outliers.

The CRPM algorithm differs from the PrefixSpan algorithm in that it includes a threshold λ_{time}. The critical function (line 13 in Fig. 4) is described in Fig. 7. In this function, P is a projection which contains a continuous prefix and the RTSs that contain the prefix. P will be extended with a new regional element if the time gap between the last element of the prefix and the new element is less than the threshold λ_{time}. The continuous route patterns are extracted from RTSs and returned to the client for further use.

Besides the client–server architecture, incremental mining strategy is also implemented to reduce the computational load on mobile devices. On the client side, the most time consuming process is the computation of visiting density (lines 3 and 4 in Fig. 4), the computational complexity of which is $O(N^2)$, where N is the number of cells that form the side of a region. The incremental mining strategy attempts to reuse previous mining results, including the past CTSs and the densities of cells, which can greatly reduce the time to process new trajectory data. Lines 6 and 7 in Fig. 4 show the incremental mining strategy.

A corresponding restriction of the incremental mining strategy is that the grid cannot be resized. As the existing CTSs are built based on the previous definition of the active grid, these cells remain the same as the cells used in the previous mining (i.e. the cells are effectively fixed by previous mining activity). As a result of the static nature of cells, it is inevitable that

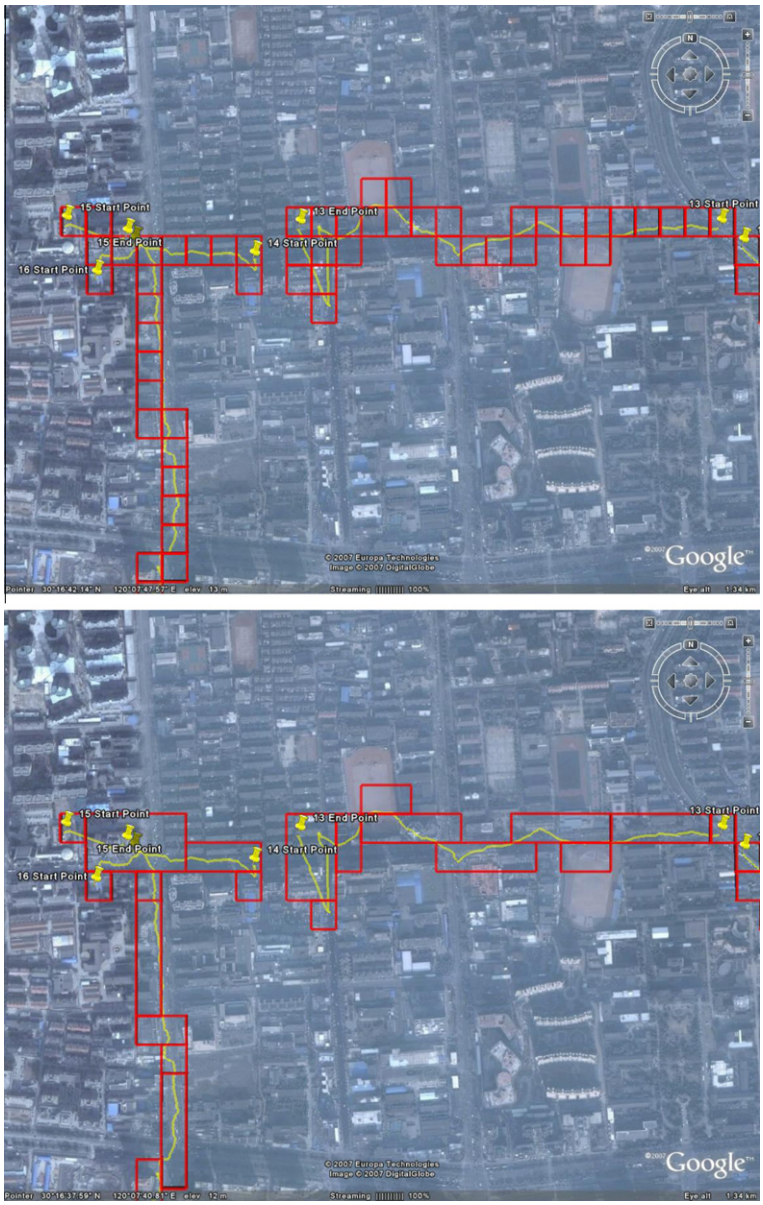

Fig. 5. The trips displayed in CTSs (top) and RTSs (bottom).

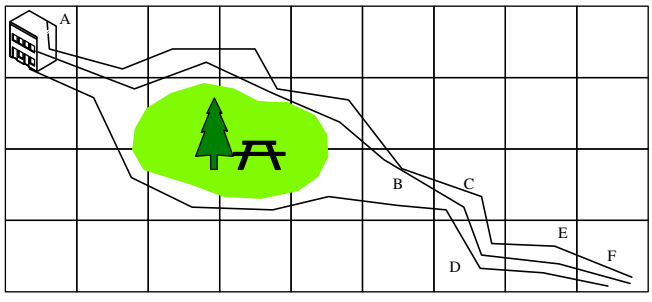

Fig. 6. An example of trips which contain a disturbance.

Algorithm 3 extend_projection (P, λ_{time})

```
1.  PS={Ø}
2.  for each Region-Temporal Sequence RTS_i in P do
3.      lastRTelem = get_lastProj(RTS_i);
4.      for each item RTelem_j in RTS_i do
5.          if (RTelem_j.in - lastRTelem.out < λ_time)
6.              P' = generate_proj(RTS_i, RTelem_j);
7.          else
8.              break;
9.          update_support(Append(P.prefix, RTelem_j.R), P');
10.         PS = Append(PS, P');
11. return P';
```

Fig. 7. Extending projection procedure in CRPM.

using the same trajectory data, the results of an incremental mining process might be different from the results of a standard mining process where the cells are not restricted by previous mining activities.

6. Route prediction

We first analyze the probabilistic model used in route prediction (Section 6.1), and then propose two prediction algorithms: the basic algorithm (Section 6.2) and the heuristic algorithm (Section 6.3).

6.1. Probabilistic model

We build the route prediction model based on a probabilistic analysis. The ROIs which are constructed in the mining procedure are used to describe personal route patterns. Given the current ROI r of a user or a series of ROIs $(r_1, r_2, r_3, \ldots, r_k)$, the prediction of the next ROI r' that the user will visit is determined by the joint probability:

$$P(r', r) = P(r'|r) \bullet P(r) \tag{1}$$

or

$$P(r', r_1, r_2, r_3 \ldots r_k) = P(r'|r_1, r_2, r_3 \ldots r_k) \bullet P(r_1, r_2, r_3 \ldots r_k), \tag{2}$$

where $P(r', r)$ is the probability that r' and r occur. Here, it means the probability that a person visits r and r' in turn in the same trip. Because the person is currently at ROI r, i.e. $P(r) = 1$, then $P(r', r) = P(r'|r)$. Route prediction is performed by recursively predicting the next ROI as the ROI with the highest probability. The difference between our prediction strategy and the methods based on the basic Markov model (i.e. taking the tuple $s = \langle r, r' \rangle$ as the state, where r is the current ROI, and r' is the next ROI) is that, the probability matrix is updated not only with the current ROI, but also with the route patterns which contain the given ROIs. In other words, the probability matrix of next ROI is $M(r, \overline{P})$, where r is the current ROI and $\overline{P}$ is the set of candidate route patterns which can be used in route prediction. $\overline{P}$ is always a subset of the complete route patterns set.

$$M(r, \overline{P}) = \begin{bmatrix} P(r_1|r, P_1) & P(r_2|r, P_1) & \cdots & P(r_k|r, P_1) \\ P(r_1|r, P_2) & P(r_2|r, P_2) & \cdots & P(r_k|r, P_2) \\ \cdots & \cdots & \cdots & \cdots \\ P(r_1|r, P_j) & P(r_2|r, P_j) & \cdots & P(r_k|r, P_j) \end{bmatrix}, \tag{3}$$

where the conditional probability $P(r_k|r, P_j)$ is the probability of going from ROI r to ROI r_k along the route pattern P_j. As the number of personal trips which support a specific route pattern can be given by the route pattern mining procedure, the conditional probability in the matrix can be calculated. Since

$$P(r', r) = \sum_{i=1}^{j} P(r'|r, P_i). \tag{4}$$

The $M(r,\overline{P})$ can be expressed as:

$$M(r,\overline{P}) = \left[\sum_{i=1}^{j} P(r_1|r,p_i)\cdots\sum_{i=1}^{j} P(r_k|r,p_i)\right] = \left[\frac{N_r,r_1}{N_r}\cdots\frac{N_r,r_k}{N_r}\right], \tag{5}$$

where N_{r,r_k} is the total number of supports to the candidate route patterns that contain a move from ROI r to ROI r_k.

The critical part of the probability matrix calculation is to decide which route patterns should be selected as candidates for route prediction. This decision is made on the basis of the current trip obtained via online observation. Suppose that nine ROIs are extracted, as shown in Fig. 8, and there are three candidate route patterns: $P_1 = \langle r_2,r_3,r_4\rangle$, $P_2 = \langle r_1,r_3,r_6,r_7\rangle$, $P_3 = \langle r_1,r_3,r_5,r_8,r_9\rangle$, whose total numbers of supports are 4, 6, 5, respectively, where the total number of supports is the number of times a pattern is present in data.

If the user is currently at ROI r_3, and no pervious online observation is available, the probability matrix is:

$$M(r_3,P) = [P(r_4|r_3,P_1) \quad P(r_6|r_3,P_2) \quad P(r_5|r_3,P_3)] = \left[\frac{4}{15} \quad \frac{2}{5} \quad \frac{1}{3}\right]. \tag{6}$$

This means that there are three possible next ROIs, r_4, r_6, r_5, of which r_6 has the highest probability of 0.4. If the online observation shows that the person moves to r_3 from r_1, then the probability matrix becomes:

$$M(r_3,P) = [P(r_6|r_3,P_2) \quad P(r_5|r_3,P_3)] = \left[\frac{6}{11} \quad \frac{5}{11}\right], \tag{7}$$

which means pattern P_1 is no longer a candidate pattern for route prediction and can be discarded. Our approach has more predictive capability compared to the basic Markov model based approach which only uses the current state for prediction, whereas our method has several historical states available to it. By using higher-order Markov models, in which the probability of the next state is dependent on the current state and the previous n-1 states (taking n as the order of the model), the prediction quality can also be improved significantly. However, selecting an appropriate order for a higher-order Markov model remains an open research question, and in practice, the quantity of data available for analysis may greatly influence the choice of the order of a Markov model [3]. Our prediction approach is also different from the higher-order Markov model based methods as it can adaptively exploit historical information (i.e. the online observations and route patterns of users) to gain maximum predictive power, without being given the order n of a Markov model.

6.2. The basic route mining algorithm

The proposed prediction procedures are conducted in three stages: prediction tree building, route pattern matching, and route prediction. As the computational capability of mobile devices is limited, standard search strategies such as Breadth First Search are unsuitable in this case, so a tree data structure is constructed for the purpose of pattern matching. Given the route patterns produced by CRPM, the prediction tree organizes all the patterns and possible sub-patterns by their prefixes. Let us consider the route patterns in Fig. 8 as an example, and the prediction tree for the three route patterns *P1*, *P2* and *P3* is presented in Fig. 9. Every node in the prediction tree, with the exception of the root, represents a ROI in route patterns. Each node contains two numbers. The numbers not in parentheses are the serial numbers of the ROIs. The numbers inside the parentheses are the total numbers of supports for the route patterns. For example, the numbers in the bottom-left node "9(5)" mean the pattern $\langle 1,3,5,8,9\rangle$ is supported by 5 recorded trips. The advantage of a prediction tree is that route pattern matching can be conducted without searching for an entry. The children of the root include all possible ROIs of the given route patterns, in this case ROIs 1 to 9. The route pattern matching procedure utilizes these entries to start matching and to pick up the initial candidate patterns. For example, the child of root, ROI r_3 has three children, which indicates three possible continuing trips. These three route patterns are the candidates for route prediction in the next stage.

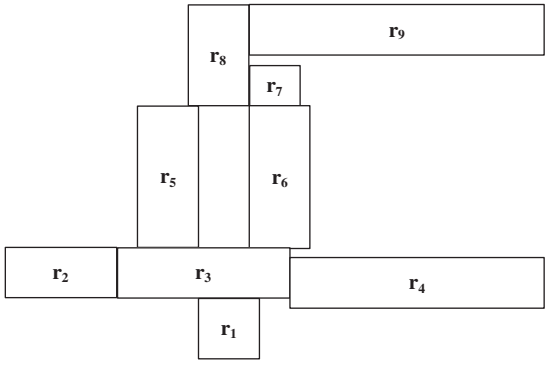

Fig. 8. An example of the route patterns of a user.

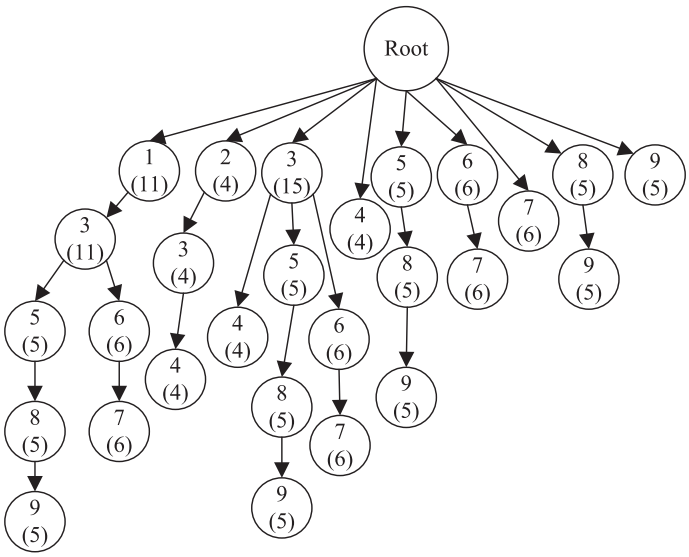

Fig. 9. The prediction tree for route patterns: $P_1 = \langle 2,3,4\rangle$, $P_2 = \langle 1,3,6,7\rangle$, $P_3 = \langle 1,3,5,8,9\rangle$.

The aim of the route pattern matching procedure is to select the candidate patterns to calculate a probability matrix. Given a ROI sequence as input, the algorithm attempts to find the patterns whose prefixes match the ROI sequence. To reduce the computational load on mobile devices, the input sequence only includes the latest ROIs (indicated by the threshold λ_{recent_regs}) visited by a user, which is reasonable as the recent movement is more important for route prediction than the earlier movement. The matching procedure extracts the subset of candidate patterns by looking for a sub-tree according to the ROI sequence. However, there is not always a sub-tree which matches the input sequence, and reasons for this are given in the next subsection. In this case, the matching algorithm repeatedly removes the oldest ROI in the sequence until a match is found. The procedure is guaranteed to halt when the ROI sequence is reduced to a single ROI as there is a node for each ROI from the root node (see Fig. 9). As a result of this, our algorithm adaptively tunes itself to the most appropriate length of the sequence of ROIs (with user defined upper bound on the length λ_{recent_regs}). With this approach, the threshold λ_{recent_regs} can be set within a range (e.g. 1–5) rather than being fixed at a particular value. This has the advantage of avoiding the problem of selecting the order parameter of a higher-order Markov model.

Given candidate patterns, the prediction procedure generates a set of routes based on the probabilistic model. The ROI with the highest probability will be output as the most likely ROI to be visited next.

Let us consider the route patterns in Fig. 8 to demonstrate how the prediction algorithm works. The online observations show a user has just passed three ROIs: r_4, r_3, r_5 in sequence. The algorithm starts from the child of the root, which represents ROI r_4, to find a prefix matching. However, there is no sub-tree indicating the sequence $\langle r_4, r_3, r_5\rangle$. The algorithm then deletes $r_4$from the sequence, and attempts to find a match with the shorter sequence. Since there is a sub-tree which matches the sequence $\langle r_3, r_5\rangle$ (namely, $\langle r_3, r_5, r_8, r_9\rangle$), its children are used as the prediction of the remainder of this route. As there is only one candidate pattern which is a continuation of the sequence $\langle r_3, r_5\rangle$, the prediction is performed based on this pattern, and generates the predicted route as $\langle r_8, r_9\rangle$.

6.3. The heuristic route prediction algorithm

In this subsection we introduce a heuristic algorithm which is an extension of the basic matching algorithm described in the previous subsection. The basic algorithm, which shortens the input sequence when a matching failure occurs, discards partial information obtained during online observation. There are several reasons why a match cannot be found. First, the input sequence may relate to multiple short route patterns, whereas none of these patterns can be fully matched. This can be overcome by using more trajectory data to extract longer route patterns. Second, the online observation includes noise resulting from inaccurate position information which can only be improved by using more accurate positioning techniques. Third, the matching failure can be caused by random short trips. For example, in Fig. 8, a person may take the route $\langle r_1, r_3, r_4, r_6\rangle$, because his bicycle is parked in ROI r_4. The heuristic prediction algorithm can counter the mis-matching caused by random short trips. With the prediction tree building procedure and route prediction procedure being similar to the basic algorithm, the heuristic prediction algorithm applies a different pattern matching strategy. When a matching failure occurs, the heuristic algorithm skips the mismatched ROI, and continues the matching procedure with the next several ROIs in the input sequence. The number of ROIs, which will be used in re-matching, is controlled by the threshold λ_{skip_regs}. If matching failure still happens, the algorithm then shortens the input sequence. This skipping mechanism can also filter some inaccurate positions which occur during online observation, which might be caused by GPS signal drift. An important criterion of

```
getPrediction_heur (Pred_tree, Online_data, Grid)
1.  reg_input = getRegSeq(Online_data, Grid)
2.  start_idx = 0
3.  if reg_input.length > λ_recent_regs
4.     start_idx = reg_input.length - λ_recent_regs
5.  while (has no complete matching)
6.     pCurNode = Pred_tree.root
7.     idx = start_idx
8.     for (; idx < reg_input.length; idx++)
9.        if pCurNode has a child ROI r
10.          pCurNode = pCurNode.children[r]
11.       else if idx is not the last index of input region sequence
12.          for i in range(0,   λ_skip_regs)
13.             idx++
14.             if pCurNode has a child ROI r
15.                pCurNode = pCurNode.children[r]
16.                break
17.          if cannot find a matching ROI then break
18.       else break
19.    if idx is the last index of input region sequence
20.       pCurNode indicates the candidate patterns
21.    else start_idx++
22. pred_Res = {Ø}
23. pNextNode = Pred_Tree.stepdown(pCurNode)
24. while pNextNode is not NULL
25.    pred_Res.append(pNextNode)
26.    pNextNode = Pred_Tree.stepdown(pNextNode)
27. return pred_Res;
```

Fig. 10. The heuristic algorithm for route prediction.

the algorithm is that the skipped ROI cannot be the last ROI in the input sequence, as the last (current) ROI plays a critical role in prediction.

Fig. 10 shows the heuristic algorithm for route prediction, where *Pred_tree* is the prediction tree built from the route patterns, *Online_data* is the real-time input data obtained via online observation, and *Grid* is a map which can project real positions to corresponding ROIs.

7. Performance evaluation and discussion

We evaluated the personal route prediction system on a dataset which consisted of more than 900 real trips recorded over a period of one month from 17 participants, each being either student of or member of staff at Zhejiang University in the People's Republic of China. The NOKIA N70 mobile phones and the HOLUX 1000 GPS devices were used as client devices, which participants carried with them during this period. A participant might take different means of transportation during a single trip. For example, on daily commute to work a participant may walk from home to a bus stop, and then take a bus to work. Obtaining this information could give insights into the effects of transportation means on the parameter setting and the overall performance of the system. For safety reasons, participants were not asked to input their means of transportation and therefore this information was not recorded in our experiment. We conducted pilot experiments to determine suitable values for sensitive and important parameters, leaving other parameters set at an arbitrary level. We do not claim optimality for these values. Table 1 shows the total numbers of recorded positions of participants.

A participant's total number of recorded positions varied greatly for the following reasons. First, different participants habitually traveled along different routes. For example, Participant 1 drove from the west of Hangzhou to Zhejiang University everyday. While in contrast, Participant 14 lived on campus, and he often stayed on the campus during weekdays resulting in different total distances travelled. Second, different participants used different transportation methods, resulting in different sampling rates. For example, Participant 17 often took a bus, while Participant 15 often rode a bicycle. Third, different par-

Table 1
The total positions recorded by the data collection system and the total positions after the data filtering.

ID	Total positions	Total positions after data filtering	Time periods
1	308,338	91,315	11.12.2007–12.20.2007
2	192,666	68,302	12.23.2007–01.25.2008
3	132,448	48,480	01.25.2008–02.20.2008
4	90,024	35,080	02.20.2008–03.21.2008
5	79,149	22,456	03.21.2008–03.31.2008
6	145,678	30,504	11.08.2007–12.24.2007
7	150,813	47,805	12.26.2007–01.25.2008
8	40,692	12,854	01.30.2008–02.23.2008
9	141,464	39,045	02.25.2008–03.20.2008
10	41,254	12,451	11.08.2007–12.22.2007
11	222,853	65,035	11.08.2007–12.21.2007
12	33,291	8355	12.24.2007–01.25.2008
13	17,616	5624	12.28.2007–01.27.2008
14	93,927	29,179	12.24.2007–01.25.2008
15	173,445	40,527	01.26.2008–03.20.2008
16	174,553	7406	10.10.2007–02.21.2008
17	121,545	43,516	11.08.2007–12.24.2007

ticipants spent different amounts of time outside buildings. For example, Participant 9 often went out during his spare time, while Participant 16 typically stayed at the dormitory on campus.

In this section, we evaluate the performance of our system from three different aspects: data filtering and trip segmentation (Section 7.1), route pattern mining (Section 7.2), and route prediction (Section 7.3). Then we discuss how to set the parameters of the system (Section 7.4).

7.1. Data filtering and trip segmentation

The five data filters were applied to the position data obtained via the GPS devices in the following order: (1) Duplication Filter; (2) Speed Filter; (3) Acceleration Filter; (4) Total Distance Filter; and (5) Angle Filter. These data filters are not mutually independent of each other. For example, the outliers that contribute to high speed may also be correlated with high acceleration. It is therefore difficult to evaluate each data filter independently. Another challenge is that the parameters of these filters are hard to tune. In our work, the parameters were set by experience obtained from data analysis.

We set the time threshold λ_{time_gap} (see Section 4.2) to 120 s to accommodate short interruptions to a trip. For instance, a traffic signal or the coincidental meeting of an acquaintance will result in a temporary stop on an actually continuous journey. The speed threshold and acceleration threshold were set to 27 m/s and 10 m/s^2 respectively. The λ_{ang} (see Section 4.2) was set to $\pi/6$ radians. Fig. 11 shows the proportion of valid positions in the original trajectory data. It can be seen that about 75% of positions were filtered as redundant positions or outliers according to the filters. Most of the positions which were filtered out were indoors, as one would expect that indoor readings are susceptible to signal drift. Fig. 12 shows the total number of trips made by each participant. As mentioned in Section 4.2, the trips which contained less than λ_{trj_cnt} positions (set to 20 in the experiment), were deleted from the trajectory data by the filters. Fig. 13 shows the average number of positions in a trip for each participant.

In practice, the parameters for data filtering and trip segmentation should be set adaptively. For example, while processing the trajectory data recorded when the user takes a high-speed train, the speed threshold and acceleration threshold should be adjusted to higher values.

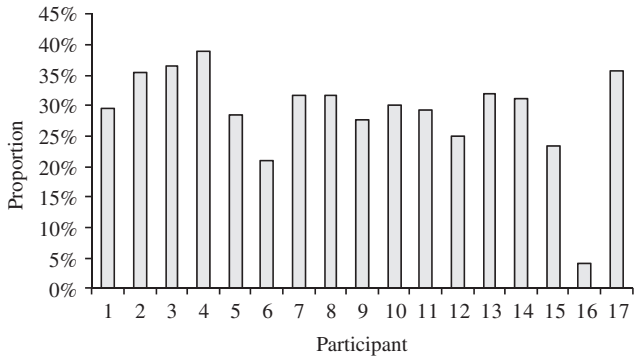

Fig. 11. The proportion of valid positions in original trajectory data.

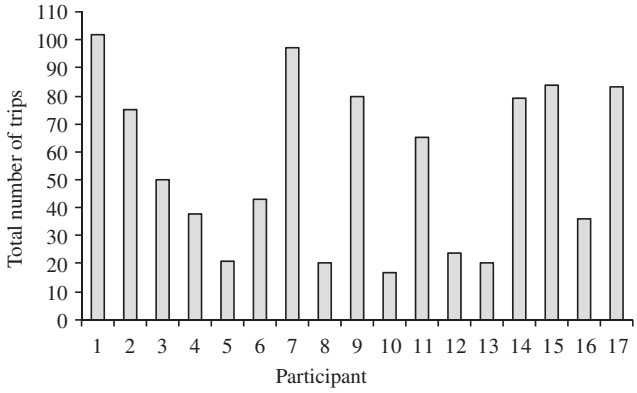

Fig. 12. The total number of trips.

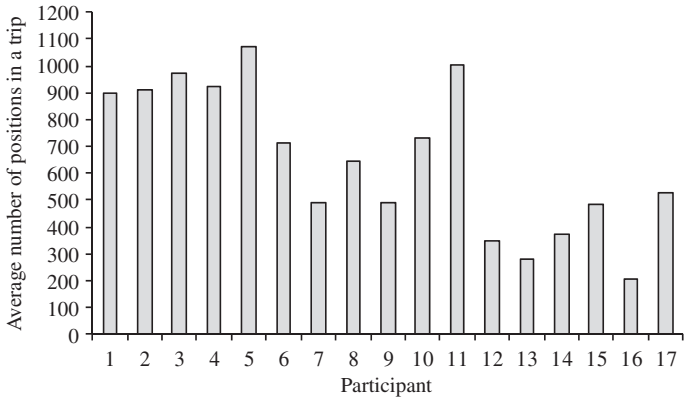

Fig. 13. The average number of positions in a trip.

7.2. Routes pattern mining

From Table 1, it can be seen that Participants 8, 10, 12, and 13 have the least total number of positions (less than 50,000). From Fig. 12, it can be seen that Participants 5, 8, 10, and 13 have the least total number of trips (less than 22). In order to obtain sufficient data for reliable route pattern mining and route prediction, the trajectory data of Participants 8, 10, and 13, (i.e. the intersection of the above two sets), were not used. In the route pattern mining experiment, we set λ_{min_sup} to 5 (see Section 5).

The performance of route pattern mining can be adjusted through different parameter settings, which provides more flexibility in route mining (Section 5). The parameter *max_reg_size* sets the maximum total number of cells on the side of a ROI. In our experiment, the length of a cell is set to 50 m. That means the parameter *max_reg_size* restricts the maximum side length of a ROI to *max_reg_size* $\times$ 50 m. This parameter represents a tradeoff between the accuracy and the computational load. A small *max_reg_size* provides more accurate mining results but incurs a higher computational cost, as the number of ROIs would increase. In practice, street block level accuracy is usually considered as the bottom line to differentiate between two neighboring parallel roads. In Hangzhou city, where the experiments were conducted, the distance between two adjacent parallel roads is generally less than 250 m. Therefore, *max_reg_size* was set within the range of 1–5 cells, corresponding to a cell with sides of length 50 m. Fig. 14 discloses the relationship between *max_reg_size* and the total number of ROIs for each participant. It can be seen that when *max_reg_size* increases from 1 to 2, the total number of ROIs decreases dramatically, and the decrease trend becomes less significant with further increase of *max_reg_size*.

The key parameter for CRPM is λ_{time} which defines the maximum time interval that can be accepted in a continuous route pattern (see Section 5). As discussed in Section 4.3, when $\lambda_{time} = 0$, CRPM corresponds to substring mining, e.g. substring tree mining [5]. According to the definition of CRPM, with the increase of λ_{time}, the total number of route patterns, including the route patterns of all possible lengths, will increase. In the experiment, we first set λ_{time} to 0 s, and then increased it in increments of 1 s to 5 s. Since the minimum side length of a region was 50 m, if λ_{time} was set to 5 s, a region would not be regarded as a disturbance under the condition that the movement speed of a participant is less than 10 m per second. Since the experiment was conducted in the urban districts of Hangzhou city, this condition was satisfied by most of the participants. The

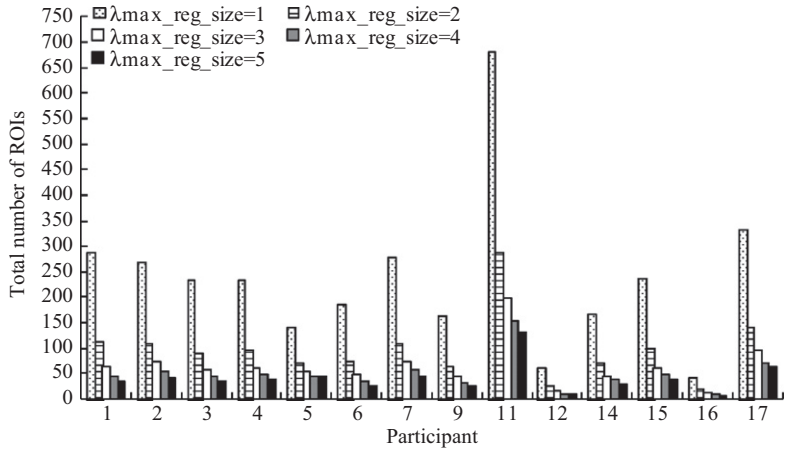

Fig. 14. The relationship between *max_reg_size* and the total number of region-of-interest.

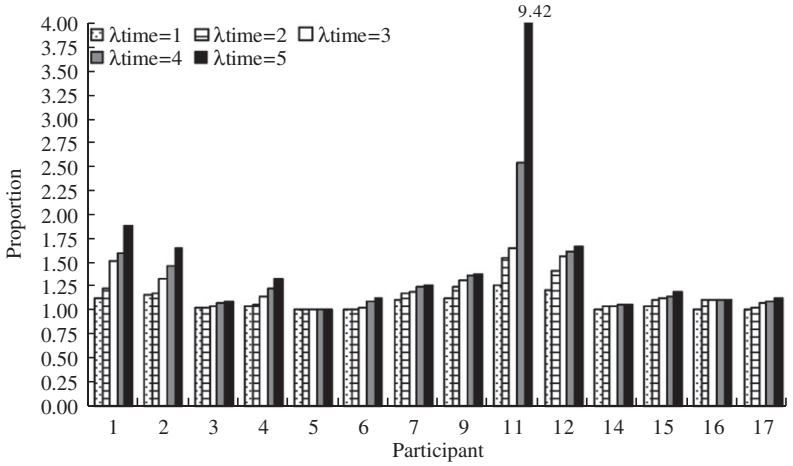

Fig. 15. The relationship between the total number of route patterns and λ_{time}.

max_reg_size was set to 2 during the experiment. Fig. 15 shows the relationship between the total number of route patterns and λ_{time}. Because the absolute value of the total number of route patterns is different from one participant to another, in the figure, the proportional relationships between the total numbers of patterns generated with different λ_{time} and the number of patterns generated when $\lambda_{time} = 0$ are exploited to show the effects of λ_{time}. The results in the figure can be regarded as the comparison between the traditional substring pattern mining and CRPM. It can be found in Fig. 15 that the proportion increases with λ_{time}. When $\lambda_{time} = 5$, the total number of patterns of some participants (e.g. Participant 11) increases much more than that of other participants. It might be because there are many short-term stops in the trajectory data of Participant 11. By using a large λ_{time}, more random interruptions are tolerated, and more route patterns and sub-patterns can be extracted.

The most important benefit of CRPM is that it can extract longer continuous route patterns than ordinary substring mining techniques. To demonstrate this we first provide the following two definitions:

Definition 3. Regional Sequence Containment ($\supseteq$). A regional sequence $\overline{S}_1\left(\overline{S}_1 = \langle R_0, \ldots, R_j\rangle\right)$ *contains* another regional sequence $\overline{S}_2\left(\overline{S}_2 = \langle R_0, \ldots, R_k\rangle\right)$, i.e. $\overline{S}_1 \supseteq \overline{S}_2$, if and only if $\overline{S}_2$ is a subsequence of $\overline{S}_1$.

Definition 4. Maximum Distinct Subset (MDS). A regional sequences set $\overline{P}_1$ is a *Maximum Distinct Subset* of another regional sequences set $\overline{P}_2$ if and only if $\overline{P}_2 \forall \overline{S}_2 \in \overline{P}_2, \exists \overline{S}_1 \in \overline{P}_1 \wedge \overline{S}_1 \supseteq \overline{S}_2$, and $\forall \overline{S}_1 \in \overline{P}_1, \quad \nexists \overline{S}_3 \in \overline{P}_1 \wedge \overline{S}_1 \supseteq \overline{S}_3 \wedge \overline{S}_1 \neq \overline{S}_3$.

It can be seen that MDS is a set without redundant route patterns, which can be used to evaluate CRPM. Table 2 shows the longest and average lengths of route patterns in MDS with different λ_{time} values for each participant. Table 2 clearly shows that increasing λ_{time} helps to find longer route patterns.

Table 2
The longest and average length of route patterns in MDS for each participant.

ID	Longest pattern (regions)				Average length (regions)			
	λ_{time} = 0	λ_{time} = 3	λ_{time} = 5	λ_{time} = 10	λ_{time} = 0	λ_{time} = 3	λ_{time} = 5	λ_{time} = 10
1	41	45	55	58	12.75092937	17.02949853	21.15970516	31.01136913
2	31	40	40	42	12.955	15.852	17.48070175	22.73561644
3	35	35	35	41	12.06	12.20886076	12.48235294	16.93181818
4	23	28	31	34	7.165562914	7.891719745	8.975	19.02729258
5	10	10	10	10	2.507692308	2.507692308	2.507692308	2.507692308
6	14	14	17	17	5.352941176	5.288888889	5.574468085	6.777777778
7	7	9	10	12	2.113821138	2.410852713	2.523076923	2.808823529
9	10	14	14	17	2.891566265	3.654761905	3.744186047	4.0
11	34	47	80	80	10.51875	12.56155508	23.12989324	50.95671726
12	11	17	17	17	5.290322581	7.875	7.589743590	8.5
14	11	11	11	11	2.743902439	2.841463415	2.879518072	2.988095238
15	13	13	13	13	3.710144928	4.063829787	4.220689655	4.506329114
16	4	5	5	7	1.857142857	2.0	2.0	2.478260870
17	40	40	40	40	10.58928571	11.13953488	11.13043478	12.61818182

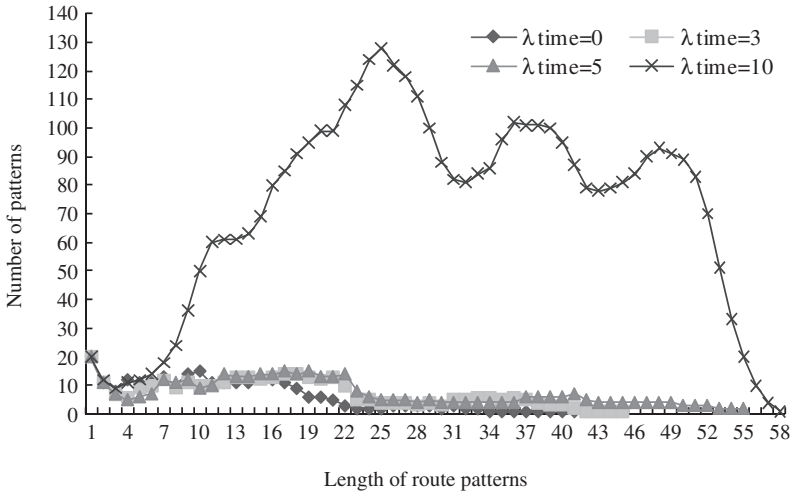

Fig. 16. The distribution of the length of Participant 1's route patterns with different λ_{time}.

Using the trajectory data of Participants 1 and 6, Figs. 16 and 17 present how λ_{time} affects the length of route patterns in MDS. These two figures show that with the increase of λ_{time}, the total number of long route patterns increases rapidly. When λ_{time} = 10, the total number of route patterns increases especially rapidly, as shown in Fig. 16. This might be because Participant 1 makes longer repetitive daily trips. Since small interruptions are within the tolerance indicated by λ_{time}, the number of long route patterns increases quickly.

We also used feedback from the participants to evaluate the effectiveness of the mining system. After the trajectory data was collected from the participants, each one whose data we used was asked to complete a short questionnaire. There were five randomly selected route patterns from their own 50% longest route patterns. Of the 17 participants 14 were selected to complete the questionnaire, for reasons stated at the start of this Section. We presented all five of the selected route patterns to each particular participant using Google Maps. All the participants were asked to answer the three following questions about each of the five route patterns mined from their data.

Question 1. Is this a typical route that you might take?
Question 2. If you do take the route, how often do you follow it?
Question 3. What proportion of the complete journey does the route pattern represent?

The responses to the first question indicate that all the participants considered the extracted route patterns as representations of the trips they realistically made over the month of recording. The answers to the second question are presented in Fig. 18, which shows that 84.3% of the extracted route patterns corresponded to trips taken more than once a week. This is partly because the experiment was only conducted for a period of one month, and the recorded trajectory data was insuf-

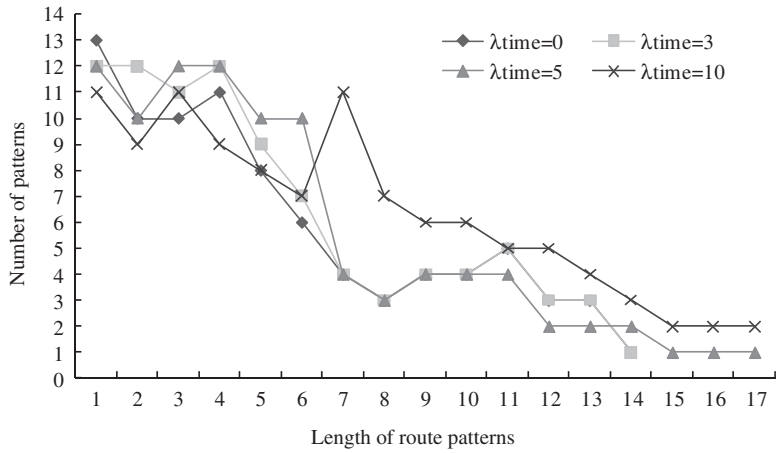

Fig. 17. The distribution of the length of Participant 6's route patterns with different λ_{time}.

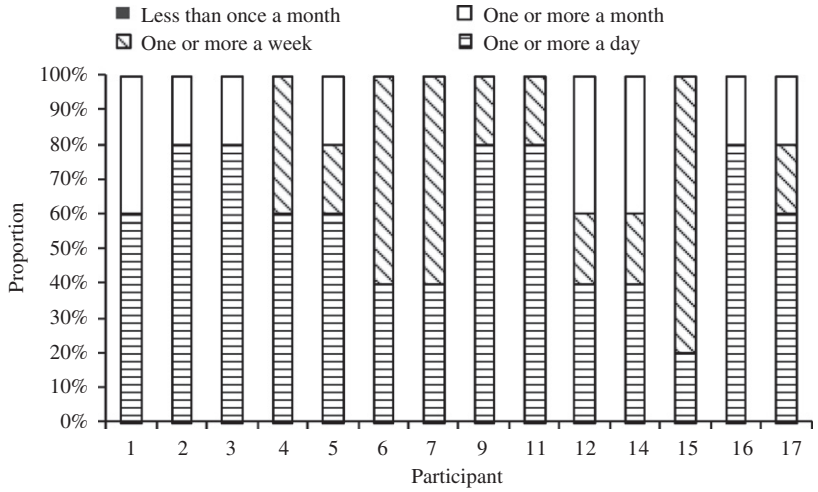

Fig. 18. The statistical results of the second question in human judgment.

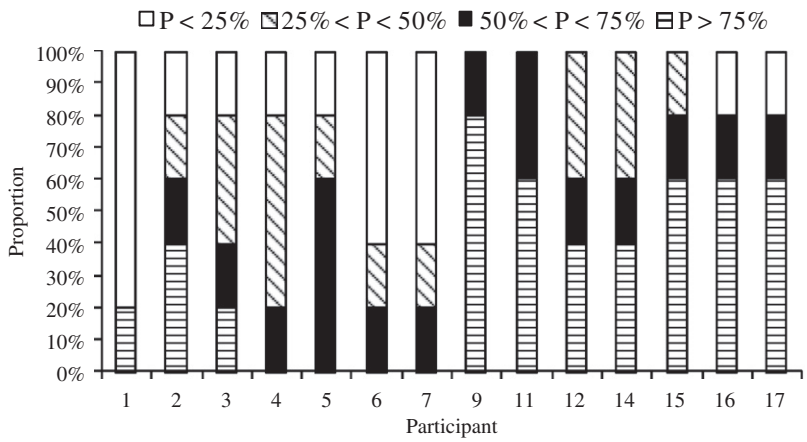

Fig. 19. The statistical results of the third question in human judgment.

ficient to mine route patterns with a lower frequency. Answers to the third question (see Fig. 19) show that most of the route patterns were considered to cover more than 50% of the length of real trips which were taken on a repetitive basis. However,

there were some short patterns due to practical reasons, e.g. GPS signal drift, long-term stops, and battery running out of power etc., which could be overcome by employing more trajectory data.

7.3. Route prediction

We evaluated the performance of the proposed prediction algorithms from two aspects, including the prediction of the next ROI and continuous route prediction. Since the experiment only lasted for one month, the extracted route patterns only covered a fraction of the length of the real repetitive trips, as shown in Fig. 19. In order to make a reasonable route prediction evaluation, we used the trajectory data of the participants whose longest route patterns were longer than 10 ROIs, with $\lambda_{time} = 0$. According to the data in Table 2, the trajectory data of Participants 1–4, 6, 11, 12, 14, 15, and 17 had route patterns longer than 10 ROIs. The trajectory data from the 10 participants was split into two parts, which were used for training and testing. For each participant, we randomly selected 10% of the recorded trips as a test set and the remaining 90% as a training set to build his personal route patterns. Basic Markov model and 2nd order Markov model were used as the baselines for the purposes of a comparison.

Fig. 20 shows the correct rates of the prediction for the next ROI by using the basic algorithm under five different λ_{recent_regs} settings. We set λ_{time} to 5 when mining route patterns. In the evaluation, each ROI sequence in the testing set was split into two disjoint sequences: the input sequence and the result sequence, for example $\langle r_1, r_2, r_3, r_4\rangle$ could be split into $\langle r_1, r_2\rangle$ and $\langle r_3, r_4\rangle$. The first ROI in the result sequence was the ground truth for the next ROI prediction. For example, a ROI sequence $\langle r_1, r_2, r_3\rangle$ produces two test cases: the first test case with $\langle r_1\rangle$ as the input sequence and r_2 as the ground truth for the next ROI prediction; and the second test case with $\langle r_1, r_2\rangle$ as the input sequence and r_3 as the corresponding ground truth for the next ROI prediction. It can be seen from Fig. 20 that when the length of input ROI sequence is longer than 1, i.e. $\lambda_{recent\_regs} > 1$, the basic algorithm provides much better performance than that of the basic Markov model based method. Additionally, when λ_{recent_regs} increases from 1 to 2, the correct prediction rate increases significantly, but this improvement is limited when λ_{recent_regs} increases from 2 to 5. We speculate that this might be because the limited amount of available trajectory data. From Fig. 20, it can also be seen that when setting λ_{recent_regs} to 2, the correct rate of our algorithm is also better than that of the 2nd order Markov model based method, which indicates a suitable value for λ_{recent_regs}.

In Fig. 21 we compare the four different algorithms in terms of how well they predict the next ROI. We set λ_{recent_regs} to 3 in this experiment. It shows that the basic algorithm and the heuristic algorithm achieve a higher percentage of correct predictions than the Markov model based methods. Additionally, the difference between the performance of the basic algorithm and the heuristic algorithm is insignificant, as most of the participants are students who live on campus, and their trajectory data is very repetitive.

As the length of input sequence increases, the percentage of correctly predicted next ROI increases as one would intuitively expect, see Fig. 22. When the length of input sequence exceeds 30, the basic and the heuristic algorithms can achieve correct rates of about 90%. There is very little difference between the basic and heuristic algorithms in this experiment. The number of correct predictions made by the basic Markov model based method also increases as the length of input sequence increases. This might be because, for most people, there are only few long repetitive trips in their daily lives, and the next ROI can be predicted with more certainty the longer the input sequence.

We also evaluated the performance of continuous route prediction. The test cases were built by splitting testing sequences into input sequences and ground truths for route prediction. For example, the test sequence: $\langle r_1, r_2, r_3\rangle$ produces

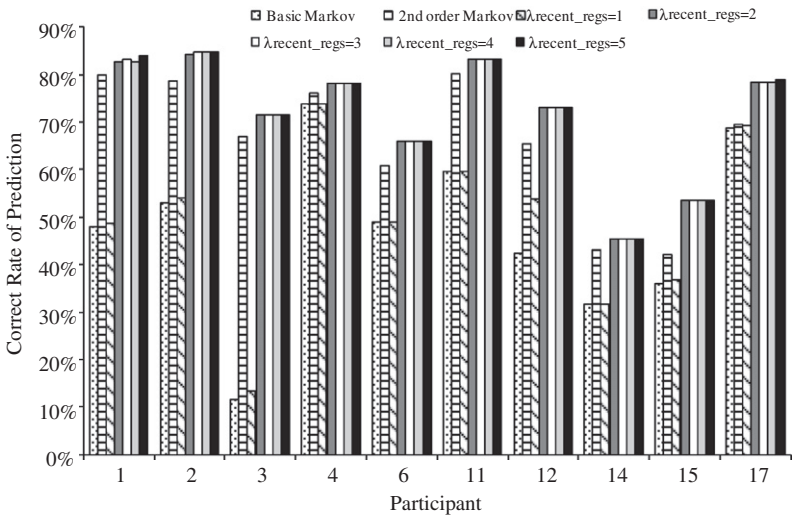

Fig. 20. The correct rate of next ROI prediction using basic algorithm with different λ_{recent_regs}.

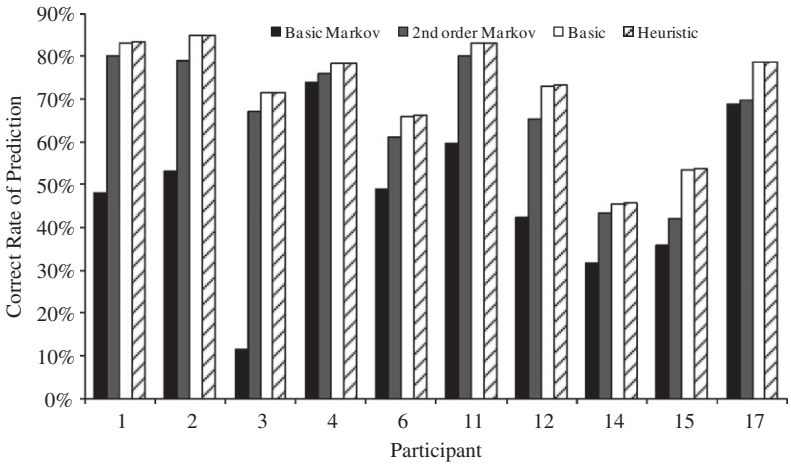

Fig. 21. The correct rate of next ROI prediction using different algorithms.

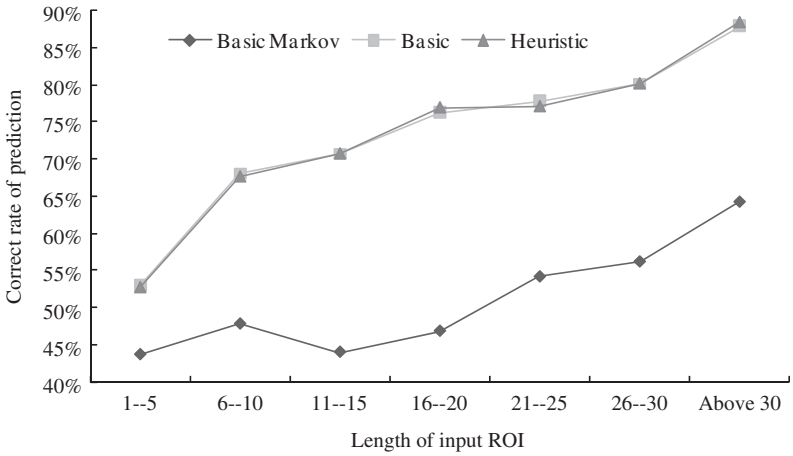

Fig. 22. The correct rate of predicting next ROI with different lengths of input sequence.

two test cases: the first test case with $\langle r_1 \rangle$ as the input sequence, $\langle r_2, r_3 \rangle$ as the ground truth for route prediction, and the second test case with $\langle r_1, r_2 \rangle$ as the input sequence, $\langle r_3 \rangle$ as the corresponding ground truth. The baselines for the evaluation were the basic Markov model based method and the 2nd order Markov model based method. To evaluate the performance of our route prediction system we used Levenshtein distance [21], which was a traditional metric used to measure the difference between two sequences.

Fig. 23 shows the average length of predicted routes under different mining settings and prediction algorithms, with λ_{time} set to 0 and 5 s. T0 and T5 stand for the patterns mined by setting λ_{time} to 0 and 5 respectively. It can be seen that the effects of prediction algorithm are insignificant, but the increase of λ_{time} can lead to longer lengths of route prediction.

Fig. 24 shows the average Levenshtein distance results for different prediction algorithms. In the figure, Basic and Heuristic stand for the basic algorithm and the heuristic algorithm, and T0 and T5 stand for setting λ_{time} to 0 and 5. Note that the results of the basic and the 2nd order Markov model based methods are obtained based on the patterns mined by setting λ_{time} to 5. It can be seen that the average Levenshtein distance of the basic Markov model based method is much larger than that of the basic and the heuristic methods, i.e. using the basic Markov model based method leads to larger difference between predicted results and the ground truth. Additionally, the average Levenshtein distances of the basic and the heuristic algorithms are slightly smaller than that of the 2nd order Markov model based method. The results also indicate that using the patterns mined by CRPM with an appropriate λ_{time} can greatly improve the accuracy of route prediction when compared to Markov model based methods.

Table 3 summarizes the performance of our system for one step prediction and continuous route prediction. These results are obtained based on the patterns mined with $\lambda_{time} = 5$. It can be seen that, for one step prediction, the average correct rates of our algorithms are much higher than that of the basic Markov model based method, and are also about 5% higher than that of the 2nd order Markov model based method. For route prediction, the average Levenshtein distance of our algorithms is

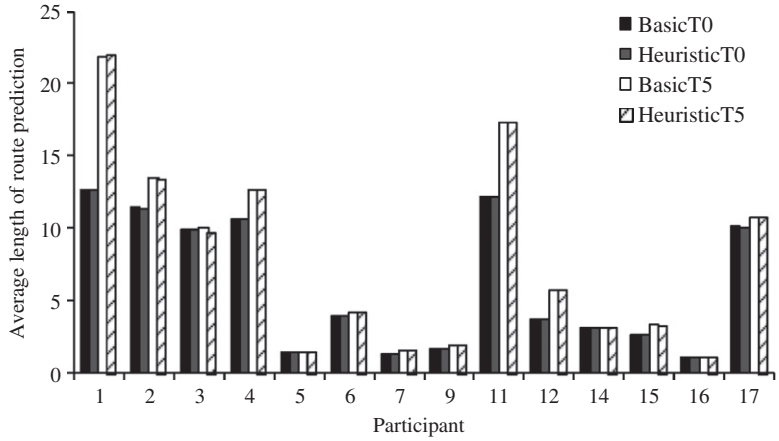

Fig. 23. The average length of prediction results with different mining settings.

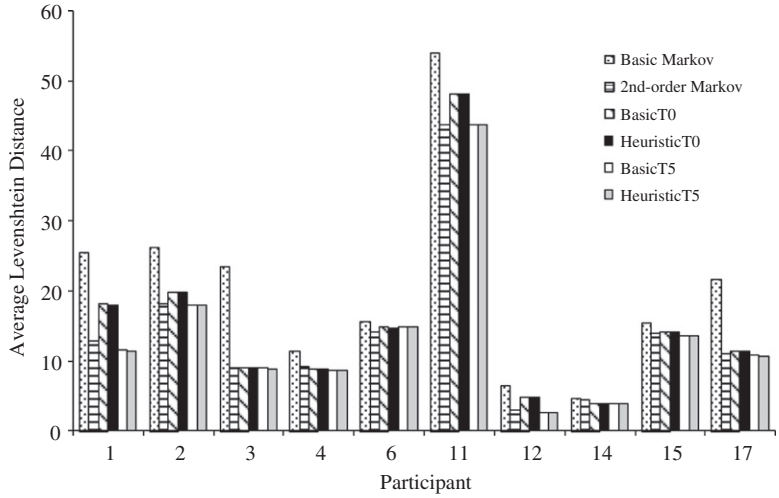

Fig. 24. The average Levenshtein distance of route prediction using different prediction algorithms.

Table 3
The prediction performances of different algorithms.

	Basic Markov	2nd-order Markov	Basic	Heuristic
Average correct rate of one step prediction	47.42%	66.35%	71.75%	71.86%
Average Levenshtein distance of route prediction	20.35	13.88	13.65	13.61

about 30% smaller than that of the basic Markov model based method, and is slightly smaller than that of the 2nd order Markov model based method.

7.4. Guideline to parameter settings

The experimental results indicate that the parameter settings can greatly affect the performance of the proposed system. Therefore, to achieve more reliable prediction results, the parameters for route pattern mining and route prediction should be fine tuned to suit the the typical commuting habits of users.

During route pattern mining, the *max_reg_size* should be set according to the size of the area covered by the movement of a user. If a user repetitively makes long trips, then the *max_reg_size* should be set to a large value, otherwise, *max_reg_size* should be set to a small value to achieve a higher accuracy of route abstraction. Additionally, the λ_{time} needs to be set in terms of the frequency of movement. For example, a small λ_{time} might be suitable for an office clerk who has a very predictable daily

pattern in terms of the trips he makes, whereas a large λ_{time} is more suitable for a roaming journalist who is likely to make a large number of erratic trips. The λ_{recent_regs} in route prediction should be set according to the computational capability of the client device. The accuracy of prediction can be improved by increasing λ_{recent_regs}, but this would be at the cost of more computational time.

Inappropriate setting of these parameters might decrease the performance of our system under some operating conditions, and determining suitable setting for these parameters is problematic for novel users. We consider this to be the main drawback of the system in its current state of development. The results of the experiment suggest guidelines on how to set these parameters, however the results are constrained by the limited amount of trajectory data used in the experiment. Additionally, our system does not take other important factors into consideration (e.g. temporal attributes of trips and transportation means). The accuracy of route prediction could be further improved by exploiting these related factors.

8. Conclusions and future work

In this paper, we present a practical personal route prediction system, which runs on mobile phones with GPS capabilities. In the proposed system, an adaptive recording strategy is employed to counter the different archiving demands associated with different modes of transportation. Five specially designed data filters are employed to remove outliers from the raw personal trajectory data. The client–server distributed architecture guarantees the privacy of personal data and greatly reduces the computational load on the client device which has limited computational capacity. We also proposed a novel algorithm, CRPM, for continuous personal route pattern mining, and two decision tree based algorithms for personal route prediction.

The experimental results show that CRPM can extract more routes and longer route patterns than traditional substring mining methods. Moreover, the proposed basic and heuristic prediction algorithms outperform the basic and 2nd order Markov model based methods in terms of average correctly predicted next ROI and average Levenshtein distance of continuous route prediction.

The proposed system has many potential applications. For example, appropriate notifications can be delivered to users in terms of their predicted routes, and advertisers can send targeted sales messages to shoppers who will be passing a particular shopping mall. The proposed system can also be used in ITSs to send relevant traffic information to users.

This work will be extended in the following aspects. First, we will exploit machine learning techniques to automatically configure the parameters of the system. Second, we will utilize other attributes of trips, e.g. temporal attributes and transportation means, to further improve the performance of the system. Third, we will try to estimate the mode of transportation based on personal trajectory data. Moreover, we will attempt to improve the performance of route prediction by using simple interactive tools, e.g. a tool for users to report their intended destinations on a map via their mobile phones.

Acknowledgement

This work was funded by the Natural Science Foundation of China (No. 60703040), the Zhejiang Provincial Natural Science Foundation of China (No. Y107178), Science and Technology Department of Zhejiang Province (No. 2007C13019).

References

[1] O. Abul, M. Atzori, F. Bonchi, F. Giannotti, Hiding sensitive trajectory patterns, in: Proceedings of IEEE International Conference on Data Mining Workshops, 2007, pp. 693–698.
[2] O. Abul, F. Bonchi, M. Nanni, Never walk alone: uncertainty for anonymity in moving objects databases, in: Proceedings of International Conference on Data Engineering, 2008, pp. 376–385.
[3] D. Ashbrook, T. Starner, Using GPS to learn significant locations and predict movement across multiple users, Personal and Ubiquitous Computing 7 (5) (2003) 275–286.
[4] A. Brilingaitė, C.S. Jensen, Enabling routes of road network constrained movements as mobile service context, Geoinformatica 11 (1) (2007) 55–102.
[5] H. Cao, N. Mamoulis, W. Cheung, Mining frequent spatio-temporal sequential patterns, in: Proceedings of IEEE International Conference on Data Mining, 2005, pp. 82–89.
[6] L. Chen, S. Benford, Your way your missions: from location-based to route-based pervasive gaming, in: Proceedings of International Conference on Advances in Computer Entertainment Technology, 2007, pp. 232–233.
[7] L. Chen, S. Benford, Designing planned route based interactions for context-aware applications, in: Proceedings of International Conference on Mobile Technology, Applications, and Systems, 2007, pp. 628–631.
[8] Y. Deguchi, HEV charge/discharge control system based on car navigation information, in: Proceedings of SAE Convergence International Congress & Exposition on Transportation Electronics, 2004, pp. 1–4.
[9] J. Froehlich, J. Krumm, Route prediction from trip observations, in: Proceedings of Intelligent Vehicle Initiative Technology Advanced Controls and Navigation System, SAE World Congress & Exhibition, 2008.
[10] B. Gedik, L. Liu, Protecting location privacy with personalized *k*-anonymity: architecture and algorithms, IEEE Transactions on Mobile Computing 7 (1) (2008) 1–18.
[11] F. Giannotti, M. Nanni, D. Pedreschi, Efficient mining of temporally annotated sequences, in: Proceedings of SIAM Conference on Data Mining, 2006, pp. 346–357.
[12] F. Giannotti, M. Nanni, D. Pedreschi, F. Pinelli, Trajectory pattern mining, in: Proceedings of ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2007, pp. 330–339.
[13] G. Gidófalvi, T.B. Pedersen, Mining long, sharable patterns in trajectories of moving objects, Geoinformatica 13 (1) (2009) 27–55.
[14] A. Karbassl, M. Barth, Vehicle route prediction and time of arrival estimation techniques for improved transportation system management, in: Proceedings of IEEE Intelligent Vehicles Symposium, 2003, pp. 511–516.

[15] H. Karimi, X. Liu, A predictive location model for location-based services, in: Proceedings of International Symposium on Advances in Geographic Information Systems, 2003, pp. 126–133.
[16] S. Keegan, G.M. O'Hare, M.J. O'Grady, Easishop: Ambient intelligence assists everyday shopping, Information Sciences 178 (3) (2008) 588–611.
[17] J. Krumm, E. Horvitz, Predestination: Inferring destinations from partial trajectories, in: Proceedings of International Conference on Ubiquitous Computing, 2006, pp. 243–260.
[18] K. Laasonen, Clustering and prediction of mobile user routes from cellular data, in: Proceedings of European Conference on Principles of Data Mining and Knowledge Discovery, 2005, pp. 569–576.
[19] A.J.T. Lee, Y.A. Chen, W.C. Ip, Mining frequent trajectory patterns in spatial–temporal databases, Information Sciences 179 (13) (2009) 2218–2231.
[20] W.H. Lee, S.S. Tseng, W.Y. Shieh, Collaborative real-time traffic information generation and sharing framework for the intelligent transportation system, Information Sciences 180 (1) (2010) 62–70.
[21] V.I. Levenshtein, Binary codes capable of correcting deletions, insertions, and reversals, Soviet Physics Doklady 10 (8) (1966) 707–710.
[22] R. Monclar, A. Tecla, J. Oliveira, J.M. Souza, MEK: Using spatial–temporal information to improve social networks and knowledge dissemination, Information Sciences 179 (15) (2009) 2524–2537.
[23] T. Nakata, J. Takeuchi, Mining traffic data from probe-car system for travel time prediction, in: Proceedings of ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2004, pp. 817–822.
[24] J. Pei, J. Han, B. Mortazavi-Asl, H. Pinto, Q. Chen, U. Dayal, M. Hsu, PrefixSpan: Mining sequential patterns by prefix-projected growth, in: Proceedings of International Conference on Data Engineering, 2001, pp. 215–224.
[25] R. Simmons, B. Browning, Y. Zhang, V. Sadekar, Learning to predict driver route and destination intent, in: Proceedings of Intelligent Transportation Systems Conference, 2006, pp. 127–132.
[26] K. Tanaka, Y. Kishino, T. Terada, S. Nishio, A destination prediction method using driving contexts and trajectory for car navigation systems, in: Proceedings of ACM Symposium on Applied Computing, 2009, pp. 190–195.
[27] K. Torkkola, K. Zhang, H. Li, H. Zhang, C. Schreiner, M. Gardner, Traffic advisories based on route prediction, in: Proceedings of Workshop on Mobile Interaction with the Real World, 2007, pp. 33–36.
[28] Y. Ye, Y. Zheng, Y. Chen, J. Feng, X. Xie, Mining individual life pattern based on location history, in: Proceedings of International Conference on Mobile Data Management: Systems, Services and Middleware, 2009, pp. 1–10.