# Regions-of-Interest Discovering and Predicting in Smartphone Environments

Abdulrahman Al-Molegi[a], Antoni Martínez-Ballesté[a], Izzat Alsmadi[b]

[a]*Smart Health Research Group, Dept. of Computer Engineering and Mathematics, Universitat Rovira i Virgili, Av. Països Catalans, 26, 43007 Tarragona, Spain*
[b]*Department of Computing and Cyber Security, University of Texas A&M, San Antonio, TX 78224, USA*

**Abstract**

Location-Based Services (LBSs) provide users especially in smartphones with information and services based on their location and interests. Predicting people's next location could help in improving the quality of such services and, in turn, boosting people's confidence on those services. Any successful LBS algorithm or model should target three major goals: Location prediction accuracy, high throughput or fast response and efficiency in terms of utilizing smartphone resources. This paper proposes a new approach to discover and predict people's next location based on their mobility patterns, while being computationally efficient. The approach starts by discovering Regions-of-Interest (RoIs) in people's historical trajectories (which denote the locations where users have been previously, frequently). A new model based on Markov Chain (MC) is proposed to overcome the drawback of classical MC. Our proposed model considers both space and time contexts where the space represents a specific location that has been visited by a user while the time represents the location visiting time. We show how classical MC model can be extended to include movement times and how time will improve prediction accuracy. One Unique finding in our research is related to the value of integrating users' mobility location/space with time context. In particular, time context is formulated in a way to add extra information to the space context. For better abstraction during building the model, a general transformation function is used to transform the $n$-order MC into first order. Results show that the proposed approach provides significant improvements in predicting people's next location compared with the state-of-the-art models when applied on two large real-life datasets, GeoLife and Gowalla.

*Keywords:*
Location-Based Services, Regions-of-Interest Discovering, Location prediction, Markov Chain.

## 1. Introduction

Predicting human behavior is not straightforward because of its complexity and diversity. It can include a large number of attributes in which only some of those can be predicted from historical trends. Breaking human behavior into different elements such as mobility patterns, shopping habits, etc., and investigating each element separately aim to reduce the complexity of the problem into a manageable subset.

Discovering Regions-of-Interest (RoIs) (historical locations where smartphone users frequently visit) and predicting people's next location during their daily life are a key component in the success of modeling people's mobility. Consequently, a variety of services and applications can be enabled, such as ubiquitous advertising [17], healthcare applications [30], location recommendation systems [35], and so on [4, 28, 34].

Predicting user's next location is proactive function that takes user current location in addition to other information (such as previous locations or user daily, weekly or monthly movement times and trends, etc.), and produces a probability distribution over user's possible next locations.

Many companies track users' daily visited locations and record this large amount of data for many different applications. Portable devices with localization systems such as the Global Positioning System (GPS) are used to collect mobility data. Users carry smartphones and wearables most of the time. Literature shows that companies can track such data from those devices even location-based services are not enable. Additionally, users' locations can be possibly tracked even when smartphone WiFi and/or data connections are disabled.

Most of the existing studies for discovering RoIs depend on extracting contiguous GPS points that satisfy threshold conditions (e.g., stay time, distance, etc.). If contiguous GPS points satisfy the threshold conditions, it can be concluded that a RoI location has been discovered. Moreover, some other studies are based on clustering algorithms such as Density-Joinable (DJ) [39] and *k*-means. In general, few studies focused on discovering RoIs to predict people's mobility.

Smartphone devices are used to collect the traces of outdoor environments, thus, most of the recorded GPS points will be located in a regions that connect significant locations. This information helps to segment the trajectory into paths and significant locations. In this paper, an algorithm to discover RoIs is proposed which can help in building an accurate location prediction model. The proposed algorithm detects the paths and associates them with the nearest significant locations.

Different common machine learning techniques have been used to predict people's next location: *e.g.*, Markov Chain (MC) model, LZ family algorithms (LZ, LeZi Update and Active LeZi), rule-based approaches and Neural Networks (NNs). One of the significant challenges is related to how to adapt machine learning techniques in the context information of movements. Additionally, building a (one-size for all) accurate prediction model for all users is hard and sometimes impossible because predicting next location is a user-specific problem. Even if the visited locations are similar for many users, a trajectory of one user visiting different locations is most likely unique. Thus, building one prediction model for each user could be desirable. One of our future projects is to build different classes of prediction models and classify users in those different classes.

It is worth mentioning that accuracy of location prediction models depends on the accuracy as well as completeness or comprehensiveness of the collected data. Integrating space and time context is not necessarily to improve the prediction performance due to the different degrees of importance of various contexts when generating the information for location prediction. Moreover, the appropriate integration between space and time is still relatively unexplored. The time context must be formulated in a way to add extra information to the space context. To this end, including time context in a specific way could help in detecting human movements changes and as consequence, enhance the prediction model performance.

Efficiency in optimizing the usage of smartphones' resources is an important goal for location prediction models. Markov model becomes a better choice to predict people' next locations due to the high prediction accuracy and low complexity compared to other models such as NNs, LZ, Prediction by Partial Match (PPM) and Sampled Pattern Matching (SPM) [31].

### *1.1. Contributions*

This paper claims three contributions:

- Firstly, a new algorithm to discover RoIs located in people's trajectories is proposed, which helps to build an accurate users' next location prediction model. The algorithm starts by discovering Candidate RoIs (CRoIs) based on three types of thresholds: distance, time, and region density. Then, the Density-Based Spatial Clustering of Applications with Noise (DBSCAN) algorithm [8] is used to cluster the CRoIs into different RoI groups. Once the RoIs are discovered, the previously collected GPS points are converted into a sequence of RoIs that represent a series of locations visited by a user. Soundness and completeness are used as metrics to evaluate the performance of the proposed algorithm to discover RoIs.

- Secondly, a prediction model is proposed to predict people's next location; a **s**pace-**t**ime-based model (st-GMM). In st-GMM, MC model is improved to include the time context factor where different space-time integration methods are employed. To improve st-GMM accuracy, the model also applies explicit feedback during the testing phase to automatically update its transitions content. Due to the high prediction accuracy and low complexity of the models, they could be applied to an environment with limited resources such as services and applications installed in smartphones.

- Thirdly, we use two real-life datasets to evaluate the proposed approach. The experimental results demonstrate that our approach performs well in terms of the evaluation metrics, as well as time complexity.

The rest of this paper is organized as it follows. The related work is presented in Section 2. Section 3 presents the architecture of the approach used in this study. Section 4 describes the experimental data, model setting, and the experimental results. The conclusion and future work are presented in Section 5.

## 2. Related Work

This section aims to survey the literature on the two main steps of building next location prediction model: discovering RoI algorithms and location prediction models.

### *2.1. Discovering RoI Algorithms*

Several research papers and studies discussed the problem of discovering RoIs. The paper [22] represents an early work in this area. Their algorithm uses the loss and regain of GPS signals within a certain radius to infer indoor places as buildings. DJ [9] and $k$-means [5, 13] clustering algorithms were used to discover RoIs. $k$-means clustering algorithm is not tailored for geolocated data where the grouping depends on the mean of all points that belong to the same cluster. Thus, using $k$-means to discover RoIs could cause missing some RoIs. Additionally, some locations that do not carry semantic meaning might be detected as a RoIs.

The time-distance threshold method was used in [12] to extract the stay point (a sequence of GPS records in a spatial limited area.). An area could be considered as a stay point if the stay time was greater or equal to the time threshold and the distance between that area and the center point of the GPS records was smaller than the distance threshold.

In [20, 36], the authors proposed a stay points detection algorithm, called LiSPD in this paper. The stay points were detected by seeking the region where the user has spent a period more than a predetermined threshold providing that the distance between the start and end points of the region is under a specific threshold. For each detected stay point, the mean coordinate, arrival and leaving time were computed. Giving two points in the trajectory, if the thresholds are not satisfied, the whole region between these points was ignored and consequently some information was lost. In our proposed algorithm, these points are considered as a path and are associated with the nearest RoI.

### *2.2. Location Prediction Models*

During the recent years, predicting people's next locations has received a rapid and increasing attention from the research community and industry. Many studies have discussed the usage of machine learning techniques in building the prediction model. We will select a significant subset of those contributions (*i.e.* seminal papers or those appearing in relevant journals). The models proposed in the literature can be classified into two types: (i) probabilistic models, such as Markov model and (ii) supervised learning models, such as association rules, Support Vector Machine (SVM) and NNs.

#### *2.2.1. Probabilistic Models*

A probabilistic model is a statistical analysis tool that provides a distribution of possible outcomes on the basis of past data. MC model is used in [9, 3] and [5] to predict people's next location. $n$-Mobility Markov Chain ($n$-MMC) approach is introduced in [9] which depends on the sequence of $n$ previous visited locations. Three different datasets were used to test the model: Phonetic [15], GeoLife [38] in addition to a synthetic (simulated) one. The model started by discovering RoIs using DJ clustering algorithm. Then, $n$-MMC was constructed based on the probabilities of transition among the discovered RoIs. The results showed that the prediction accuracy obtained from GeoLife dataset was 69% and it was achieved when the number of previously visited locations was two ($n = 2$).

Using a Mixed Markov-Chain Model (MMM) is proposed in [3]. MMM is an improved version of MC model and it takes into account the effects of people's previous status as observable parameter and the user's own movement as an unobservable parameter. The proposed approach was evaluated using simulated and real tracing datasets, the latter was collected using indoor-GPS devices for 691 participants in a shopping mall. Pedestrians were classified into groups based on their mobility traces. For each group, different Markov models were generated. The pedestrian's next location approach works by first identifying the group a particular pedestrian belongs to and then the location most likely to be visited. However, relying solely on the last location to predict the next one affects greatly the prediction results. In order to better predict user's next location in a more precise manner, a model should typically take into consideration a *sequence of the recent locations*, and a higher-order Markov model should be modeled.

Ashbrook and Starner in [5] developed a software application to predict people's future movements based on Markov model. First and second order Markov models were used. A GPS receiver was utilized to collect location data for one user over four months. The proposed algorithm firstly discovers the regions where the time between any

two consecutive regions is equal to a time threshold. Then, $k$-means clustering algorithm was used to group the regions into clusters with a given radius. It was found that the second order model achieved a better prediction accuracy than the first order model.

The previous models focus on predicting user's next location using only locations sequences but did not consider or incorporate information about movements daily, weekly or monthly time (*i.e.* when in the day, the week or the month they often occur).

HMM is utilized in [23] to propose a hybrid method for predicting an individual next location. Real-life mobility dataset obtained from GeoLife project was used to evaluate the proposed method. Location characteristics were considered as a hidden parameter, while an individual's previous status was considered as a visible parameter. Unlike the previous models, the collected data were divided according to the associated timestamps into three clusters: day-time weekday, night-time weekday and weekend. A Hierarchical Triangular Mesh approach was used to divide earth surface into a set of triangular regions, where each region was represented by a single numeric ID. The locations in each cluster were converted into discrete codes associated to specific regions. Each cluster was used to train one specific HMM. The probabilities of all possible next locations were calculated and next location with the highest probability was returned.

Gao et al. [10] proposed a location prediction model that considers the spatio-temporal contexts of a user's trajectories. The proposed model was evaluated using a mobile dataset provided by Nokia Mobile Data Challenge which contains 80 users over one year of time. The authors divided the temporal context into hour of a day and day of the week, and estimated their distributions using Gaussian distributions. The probability of visiting a given location depends on the distributions of the hour and day in addition to the current visited location. Unlike our paper, in their paper, the authors did not consider the previous visited times and locations, which have strong relationship with the next location prediction.

Finally, both location and activity transitions of users were taken into account in [12]. A dataset of 100 users, using GPS devices for two weeks, was used to test the proposed model. The stay points were extracted from the collected data using time-distance threshold method. The activities of the users were considered as invisible states, while the locations were considered as visible observations. The destination the user will visit is predicted by estimating the next activity first.

#### *2.2.2. Supervised Learning Models*

Supervised learning algorithms such as association rules, SVM and NNs have been applied to predict people's location.

Morzy [25] used association rules to predict the next location of a moving object. The Network-based Generator of Moving Objects was used to generate different instances of synthetic moving objects datasets. The moving objects' trajectories were obtained by dividing the movement area into a set of rectangular regions (cells) of fixed sizes. Each cell was assigned a discrete coordinate to identify the position of the cell in the movement area. Traj-PrefixSpan algorithm (a modified version of PrefixSpan algorithm [27]) was used to discover frequent trajectories that were used to generate the movement rules. An FP-Tree index structure was used to speed up the process of looking for the trajectories discovered by Traj-PrefixSpan algorithm. To determine the next location, different matching strategies were used between the trajectories of the moving object and the generated movement rules . The results showed that the accuracy was close to 80%.

In [24], the authors used the frequent sequential pattern algorithm [2] to extract the frequent movement patterns, called Trajectory-patterns (T-patterns). The T-patterns were ordered in a prefix tree. Each node in the tree contains the dimensions of the visited region ID, support and the pointers to another regions appear sequentially in the trajectories. The best T-pattern path that matches the given moving object trajectory was obtained. The score for each node in the relative path was calculated based on support value. Then the children of the best node are selected as next possible locations. A dataset obtained from GeoPKDD project was used. The dataset includes trajectories of 1700 cars which were equipped with GPS receiver for over a week.

In [7], the authors proposed a technique to predict movement in mobile network based on association rules mining. The histories of mobile user's movement that were recorded in the core base station network were used for evaluation purposes. These histories contain the dimensions of mobile user id, source cell, destination cell and travel date. For each mobile user, the movement history was arranged in a descending order from the newest to the oldest movements. A list of elements was generated where the first cell was the current location of the mobile user and the last cell was

the first location recorded. Apriori algorithm [1] was applied to extract the association rules that satisfy the specified minimum support and confidence. The best accuracy results were obtained for the association rules of order three. Thus, the next cells of the mobile user could be predicted if the two last cells crossed were known. As a result of the prediction, the network resources were effectively managed. Instead of reversing the resources in the cells that the mobile user might not cross, the network could reserve the resources only in the appropriate time when mobile user visits the predicted cells.

Kedia [14] proposed an object tracking approach for wireless sensor network using association rules technique. A simulation model was designed to generate an object and a set of randomly distributed sensors. The sensor nodes were used to record and transmit the object movement information in the monitoring environment to the sink. Apriori algorithm was applied to discover association rules from the transaction database recorded in the sink. The discovered association rules were used to predict the next location of the object. Kedia proved that the approach could save significant amount of energy where only the sensor nodes in the predicted location must be active and the other nodes can stay in sleep mode.

Ryan and Brown [29] applied the association mining rules to the problem of occupant location prediction. The proposed method was evaluated using three types of datasets: simulated, UCC and Augsburg dataset. An occupancy simulator was developed to generate a movement data for 9 persons lasted for 2 months and 3 weeks. UCC dataset was collected for 5 occupants for a period of 1-2 months, while Augsburg dataset was collected for 4 occupants for 1-2 weeks in summer and 1-2 months in fall. In the association rules combination phase, Apriori algorithm was initialized with all itemset of attributes size + 1 instead of standard 1-itemsets. In their paper, the authors assumed that a visited locations sequences always occurs at the same time. Thus, the existence of shifts in the transitions time were ambiguous for their model.

A major shortcoming of the previous methods was that if no appropriate rule or route was matched, they failed to predict the next locations.

Location/contact traces (location, stay duration and social contacts) were utilized in [33] to build a prediction framework for people's future contextual information movement. A scanning system based on Wi-Fi and Bluetooth scanners was built to collect WiFi access point MAC addresses and Bluetooth MAC addresses, respectively. The sensing traces dataset was collected for 123 participants for 6 months. UIM clustering technique [32] was used to cluster Wi-Fi traces data into locations. A regular location was considered if the user visits the location at the same time for at least a specific number of days. The dataset rows that belong to the same period of sensing time were assigned to the same locations a user visits. These records were used to train a supervised model such as SVM, Naive Bayesian, and k-NN. The experiments showed an increase in performance.

NNs have shown to be useful in location prediction model [26]. The authors used NNs technique to predict mobile user's movement in cellular communication network. The evaluation of the prediction scheme was conducted on a dataset obtained from the MH's history of movement pattern. The dataset contains two mobiles MH1 and MH2 with regular and random movement patterns, respectively. The movement data was represented as a pair of values: the crossed distance of the mobile user in cells and the movement direction. A three-layer artificial NNs trained with Backpropagation technique was used to build movement prediction model. The hidden layer contained 8 neurons, while both input and output layers contained two neurons representing the distance and the direction of a mobile user. Sigmoid function was used for activation. This prediction was used to automatically update the location information of the mobile user which reduces the traffic load in cellular communication network.

Some previous studies are based on Recurrent Neural Networks (RNNs) which are useful with sequential data due to the neurons' internal memories that are used to maintain information about the previous input. Kaaniche and Kamoun [11] introduced a mobility prediction model in wireless Ad Hoc networks. Random Waypoint Mobility (RWM) model was used to generate location time series dataset for evaluating the efficiency of the mobility predictor. The dataset contained 30 location time series with 400 patterns each one. The first 200 patterns were used for the training and the rest is used for testing. The proposed RNNs architecture had three layers: input, hidden and output layers. The three layers were arranged in a feedforward connection and trained with Backpropagation algorithm. The input layer received the external signals coming from the time series of previous visited location and the feedbacks from previous state of network output. The characteristics of the learned patterns were stored in the hidden layer. The location prediction could be used to estimate the expiration time of the links connecting the nodes enabling them to select the most stable paths which improves routing performances. Unfortunately, the author did not present details on the results.

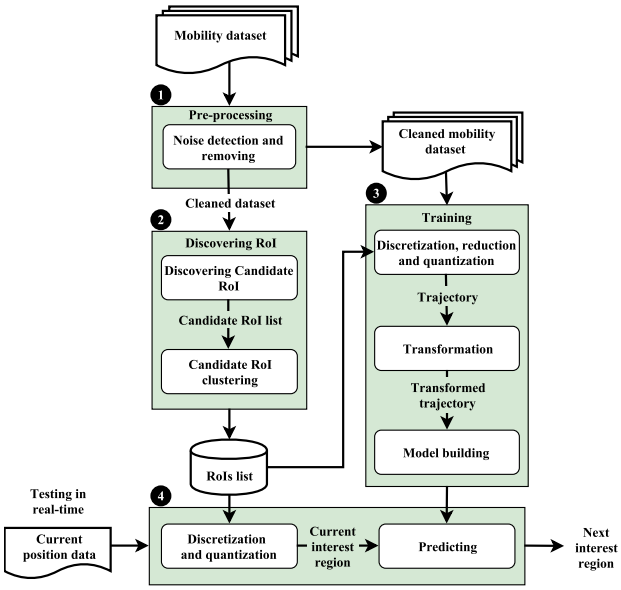



Figure 1: Our proposed next interest region discovering and prediction approach

The work presented in [21] introduced a RNN model called Spatial Temporal Recurrent Neural Network (ST-RNN) for predicting users' next location. Two typical datasets called Global Terrorism Database (GTD) and Gowalla dataset were used to evaluate the effectiveness of ST-RNN model. The recurrent structure was utilized to capture not only the local temporal contexts but also the periodical ones. The spatial and temporal values were divided into discrete bins in order to produce the time-specific and distance-specific transition matrices. The corresponding transition matrix was calculated for each specific temporal value in one time bin and similarly for each specific spatial value. Apart from prediction accuracy, the computational complexity of NNs and RNNs is extremely high.

The main drawback in most of these models is that only the space context is considered, missing other information of trajectories which is important to build accurate prediction models. Additionally, discovering RoIs in people's mobility data has not been considered in most of previous mobility prediction studies. Because of the intensive computations of some techniques to train, in addition to mobile devices restrictions on resource consumption, some algorithms may sacrifice prediction accuracy for a higher prediction speed.

## 3. The Proposed Approach

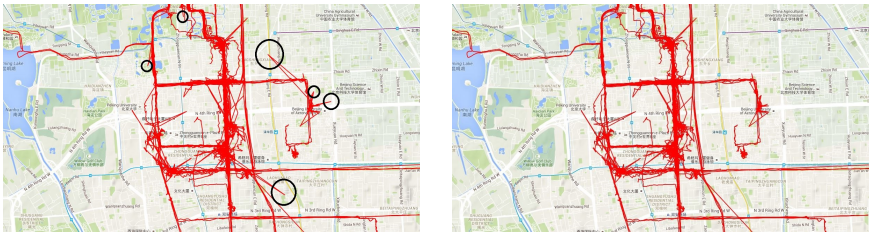
Figure 1 shows our proposed next interest region discovering and prediction approach which includes four main steps:

- First, the dataset is preprocessed to detect and remove possible noise.
- In the second step, RoIs located in a user movement region are discovered.
- The third step involves several preprocessing steps: Discretization, reduction, quantization and transformation of the training dataset, and building a prediction model.
- The last step involves evaluating the prediction model using a testing dataset.

Next sections cover with details our approach different steps.

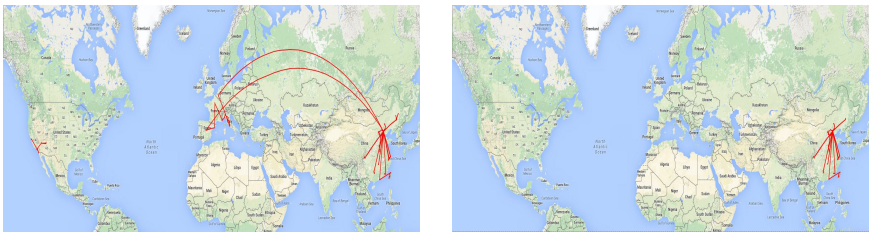
### *3.1. Mobility Data Preprocessing*

**Definition 1. Mobility data.** A mobility data is a sequence of GPS points in the form of $((x_i, y_i), t_i)$, $i = 0 \ldots n$, where $(x_i, y_i)$ denotes a GPS coordinates, $x_i$ is latitude, $y_i$ is longitude, $t_i$ is the timestamp when GPS coordinate was recorded represented by standard time format 'YYYY-MM-DD hh:mm:ss am/pm' and $n$ is the length of the sequence.



(a) Before preprocessing. (b) After preprocessing.

Figure 2: Noise detection and removal



(a) Before preprocessing. (b) After preprocessing.

Figure 3: Removing the points outside our target boundary.

In mobility data preprocessing stage, dataset is converted to a suitable form for further processing. Noises that may come from different sources are removed. Such noise can be due to mispositioning software or hardware system, or losing GPS signals due to indoor usage or mobile device battery drain or problems. Once the dataset is cleaned, RoIs can be discovered.

In general, mobility data are not accurate and consistent. For example, in a short period of time, the distance between two adjacent points could be large. The inconsistencies negatively affect the process of discovering RoIs and consequently the prediction process as a whole.

One of the mobility dataset we have used in this paper was collected from various cities in China, USA, and Europe, but the majority of the data was from China (its cities are assumed to be our target cities for prediction model). The preprocessing step on this particular dataset is conducted as it follows:

- Measuring the user travel speed between each two adjacent points based on time and distance. As shown in Figure 2, if the speed is high (*e.g.*, 300km/h), the point is removed [37].
- Calculating the distance between every point in the mobility data and the center point of China (Figure 3), then the points that are located outside the target boundary are removed.

### *3.2. Discovering RoIs*

Typically, mobility data should be processed first before it can be used in prediction models and, thus, each GPS coordinate in a user mobility data should first be converted into discrete values associated to specific RoI. The RoIs that are used to describe the movements of a certain user must be identified and eventually can be used to build a user's trajectories. RoIs discovery algorithm has to satisfy the soundness and completeness properties which are used as metrics to evaluate the effectiveness of the algorithm. More details are provided in Section 4.3.1.

**Definition 2. Soundness property.** The algorithm to discover RoIs is sound only if it can find or discover interest regions.

**Definition 3. Completeness property.** The algorithm to discover RoIs is complete if can find all interest regions.

**Algorithm 1:** Discovering RoI.

**Input :** $\mathbb{G}$: a sequence of GPS-Point and time, $\delta$: distance threshold, $\tau$: time threshold, *rd*: region density threshold.

**Output:** A set of RoI.

*CRoI*.add($\mathbb{G}$[1].GPS-Point)
*CRoIindex* ← 1
*RegionDensity*.add(1)
**for** *i* ← *2 to* $\mathbb{G}$.*length* **do**
    *isBelongToCRoI* ← **False**
    *mini* ← ∞
    **for** *j* ← *1 to CRoI.length* **do**
        *dist* ← HaversineDistance ($\mathbb{G}$[*i*].GPS-Point,*CRoI*[*j*]) // computing the distance
        **if** *dist* < $\delta$ **then**
            *isBelongToCRoI* ← **True**
            *RegionDensity*[*j*] ← *RegionDensity*[*j*] + 1
            **break**
        **else if** *dist* < *mini* **then**
            *mini* ← *dist*
            *x* ← j //*x* is the index of the nearest CRoI
        **end**
    **end**
    **if** *not isBelongToCRoI* **then**
        ΔT ← $\mathbb{G}$[*i*].*time* - $\mathbb{G}$[*CRoIindex*].*time* // computing the overall spent time
        **if** Δ*T* > $\tau$ **then**
            **if** *RegionDensity*[*x*] > *rd* **then**
                // region density
                *CRoI*.add($\mathbb{G}$[*i*].GPS-Point) // new CRoI
                *RegionDensity*.add(1)
                *CRoIindex* ← i
            **else**
                *RegionDensity*[*x*] ← *RegionDensity*[*x*] + 1
            **end**
        **else**
            *RegionDensity*[*x*] ← *RegionDensity*[*x*] + 1
        **end**
    **end**
**end**
RoI ← DBSCAN(*CRoI*)
**return** *RoI*

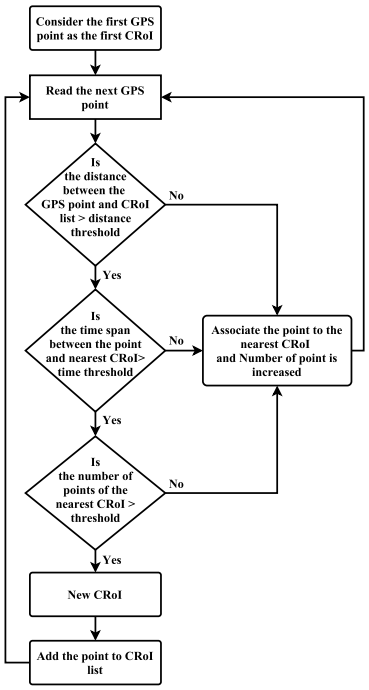



Figure 4: Discovering Candidate RoIs (DCRoIs).

We propose Discovering RoI (DRoI) (see algorithm 1); a new algorithm that takes a sequence of GPS points from user's mobility data in order to produce a set of RoIs $\mathcal{R}^u = \{r_1, r_2, \ldots, r_m\}$, where $m$ is the number of the obtained RoIs.

DRoI is accomplished in two levels: The first level is called Discovering Candidate RoI (DCRoI) (lines 1 to 33) and the second one is called Candidate RoI Clustering (CRoIC) (line 34).

- DCRoI: In the first level, mobility data is grouped based on three types of thresholds: distance ($\delta$), stay time ($\tau$) and region density ($rd$), where $\delta$, $\tau$ and $rd$ are three tuning parameters.
- CRoIC: Based on DBSCAN algorithm, the second level performs clustering on the CRoIs to obtain the RoIs.

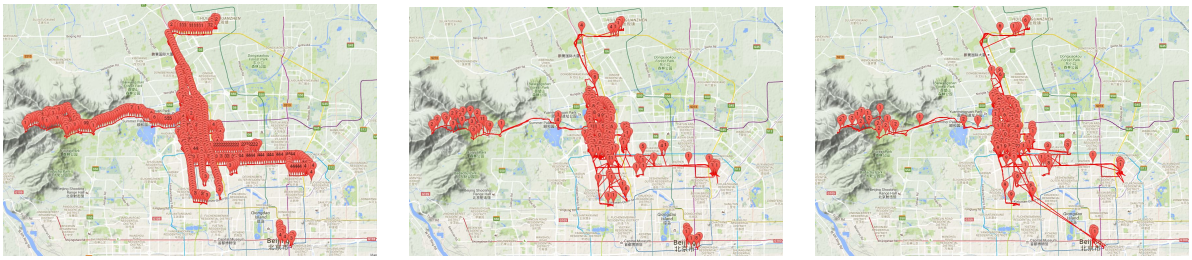
### *3.2.1. DCRoI*

Figure 4 shows a flowchart of DCRoI algorithm. The first GPS point is added to the CRoI list (line 1). It is assumed that participants have started collecting GPS points from RoIs. The GPS points of each discovered RoI are counted. Then, all GPS points are compared with the CRoI list, which starts with only one at the beginning. We begin by examining the first threshold, which is the distance $\delta$ (lines 7-17) computed using Haversine Distance, Equation 1:

$$distance = 2r \arcsin\left(\sqrt{\sin^2(\tfrac{\phi_2-\phi_1}{2}) + \cos(\phi_1)\cos(\phi_2)\sin^2(\tfrac{\lambda_2-\lambda_1}{2})}\right) \quad (1)$$

where $\phi$ is latitude, $\lambda$ is longitude and $r$ is the radius of the sphere. If the distance is less than a given threshold, the current GPS point is associated with the nearest CRoI and its density is increased by one. Otherwise, the second threshold (time spent in the current CRoI $\tau$) is computed (line 19). If the spent time is less than a given threshold, the current GPS point is also associated with the nearest CRoI and one point is counted. Otherwise, the last threshold (the density of the nearest CRoI to the current point $rd$) is compared to a given threshold (line 21). The current GPS point is also associated with the nearest CRoI in case the region density is less than a given threshold. In other words, if the GPS points satisfy the three threshold conditions, they are considered as new CRoIs. Otherwise, they will be associated with the nearest CRoI. The threshold conditions are as the followings:

- The distance between any two CRoIs must be larger than a given threshold, Figure 5a.

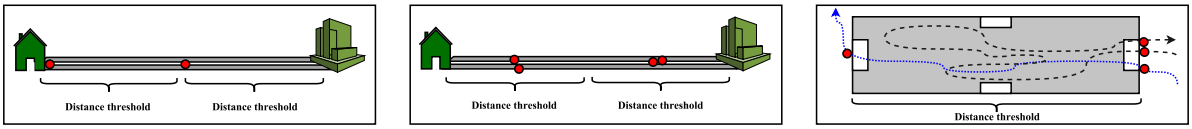


(a) After applying distance threshold.

(b) After applying distance and time thresholds.

(c) After applying distance, time and region density thresholds.

Figure 5: The first level: Discovering CRoIs.




(a) DCRoI algorithm.

(b) LiSPD algorithm.

(c) Indoor senario.

Figure 6: Discovering CRoI in mobility data.

- The time spent in a CRoI must be longer than a given threshold, Figure 5b.
- The density of the nearest CRoI to the current point must be greater than a given threshold, Figure 5c.

According to [16], it is assumed that people spend 87% of their time indoors, 8% outdoors and 5% in a vehicle. During outdoors, GPS points are recorded when users move from one RoI to another. Thus, the paths that connect the RoIs must be given a higher attention than the RoIs themselves. The GPS points that don't satisfy the threshold conditions, will be then associated with the nearest CRoI, instead of ignoring them, as in LiSPD. Therefore, the RoI will contain the interest region itself and the relevant paths that lead to a another RoI.

Figure 6 illustrates the behavior of the algorithm compared to LiSPD according to some scenarios. To move from one RoI to another, the state of the user changes from indoor to outdoor and vice versa. Let us consider the user's commuting movement. Suppose that the distance between home and workplace locations is two times greater than the threshold distance. In our algorithm, only two CRoIs are considered, Figure 6a. The first one is the first recorded GPS point when the user leaves home which represents the ID of the first region. The second CRoI is considered at the beginning of the second region after the threshold conditions are satisfied. In LiSPD algorithm, four CRoIs are considered (Figure 6b). After time and distance thresholds, the first and second points are discovered in the first and second regions, respectively. For each region, the mean values of all recorded points are calculated. The same scheme is repeated when the user returns home. It can be noticed that the two CRoIs that were discovered by our proposed algorithm are more accurate than those obtained by the LiSPD algorithm. The four points discovered by LiSPD are located in insignificant locations such as road. In addition, discovering many CRoIs in the first level negatively affects the clustering process of the second level and RoI discovering process.

For more clarification, consider the user moves to a new indoor RoI such as the workplace (Figure 6c). Only one RoI should be discovered for that building. The last recorded GPS point before the user enters the workplace building is considered as CRoI, assuming that the threshold conditions are satisfied at this point. Inside the building, a few GPS points might be recorded. When the user leaves the building, new GPS points are recorded. Two cases are presented in this situation based on region density threshold. In the first case where the region density threshold is not considered, the new recorded GPS point is considered a new CRoI if the distance between the last CRoI and the new recorded GPS point is larger than $\delta$ and the time span is greater than $\tau$. When region density threshold is considered in the second case, the region density is not greater than $rd$. Thus, the new recorded GPS point is not considered as a new CRoI but referred to the nearest CRoI.

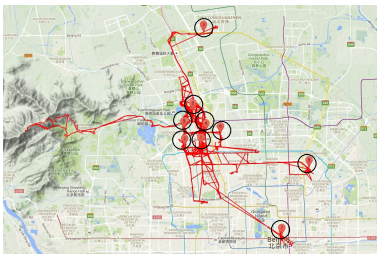


Figure 7: The second level: CRoI clustering.

CRoIs list is used to avoid considering more than one CRoI in the same region. Region density threshold condition is used to address some of the noise cases that are more likely to occur during GPS points recording process, which could lead to considering a wrong RoI. If GPS points are recorded every $T$ times and the threshold time that a user must stay in any region is $\tau$, then any RoI region must contain at least $rd = \tau/T$ points. If GPS signal is lost for any reason, few GPS points may be recorded in such regions. Thus, these points will be assigned to the nearest CRoI even if the time and distance thresholds are satisfied. In a normal case, during time threshold and within a distance threshold, $rd$ points are recorded and CRoIs are detected.

*3.2.2. CRoI Clustering*

It can be observed that, in general, the density of GPS points in RoI is higher than in other regions because people tend either to move slowly or not to move at all. To obtain a set of RoIs in the movement area, DBSCAN (one of the most common clustering algorithms) is utilized to cluster CRoIs (Figure 7). Different clustering algorithms can be used in this step. We use DBSCAN because it does not require specifying the number of clusters a priori as in the case of $k$-means. It also groups points together with many nearby neighbors and marks outliers that are too far from the nearest neighbors.

***Time complexity***. For a dataset with $n$ GPS points and $m$ CRoIs, the DRoI algorithm has the following time complexity features:

- For every iteration, distances, elapsed time and region density between GPS points and the list of all CRoIs are calculated $(m \cdot n)$, where $m < n$.
- Time complexity of DBSCAN algorithm is $\log m$ [8].
- Therefore, the time complexity is $m \cdot n + \log m$ instead of $n^2 + \log m$ in LiSPD.

*3.3. Next Location Prediction Model Construction*

This section presents the process of building the prediction model which considers both locations and time intervals as inputs and predicts user's next location. Various processes are involved before building the prediction model: Discretization, reduction, quantization and transformation.

*3.3.1. Discretization, Reduction and Quantization*

The collected mobility data is categorized into two types:

- Continuous, such as GPS coordinates.
- Discrete, such as cellular station IDs and Wi-Fi access points MAC addresses.

Continuous data is converted into discrete positions for the sake of manageability. In the discretization process, the GPS points of training dataset are converted to the RoI IDs that belong to. In other words, using the RoI list, the continuous values of the GPS training data are converted from a series of spatial points into discrete values that represent a sequence of RoIs that the user visits. With the discrete values, many types of calculations are more easily performed than the continuous one.

After discretization of the GPS points, the RoI IDs may possibly be repeated. Thus, the reduction process is used to remove the similar successive RoIs. Regarding the timestamps associated with GPS points, all recorded timestamps are ignored except the time before the movement to the next RoI (*i.e.* the leaving time from the current RoI). The obtained timestamps are then quantized into a specific time interval such as splitting the day into different timeslots or the week into weekdays and weekend etc. In our models, we choose to represent the time interval by the hour part of the leaving time from the RoI. In this case, the time intervals are the total number of hours per day (*i.e.* 24). Different time intervals such as weekday type, month of the year and timeslots are used in Section 4.5.4.

**Definition 4. Trajectory.** We denote the set of users by $\mathcal{U}$, the set of RoIs by $\mathcal{R}$, and the set of time intervals by $\mathcal{T}$. A user's trajectory is represented as a sequence of points $\mathcal{P}^u = \{p_1 \rightarrow p_2 \rightarrow \ldots \rightarrow p_n\}$, where $u \in \mathcal{U}$ and $n$ is the length of the trajectory. Each point $p_i$ is represented as a tuple $(r_i, t_i)$ where $r_i \in \mathcal{R}$ is the interest region part defined by its GPS coordinates $(x_{r_i}, y_{r_i})$ and $t_i \in \mathcal{T}$ is the hour part of the leaving time from the $r_i$ (and also the entering time of $r_{i+1}$)

The model predicts user's next RoI $r_{i+1}$ on the basis of previously collected mobility records by modeling:

$$P(r_{i+1} = \hat{r}|p_i, \ldots, p_1) \tag{2}$$

where $\hat{r}$ is a RoI in the user sequence trajectory $\mathcal{R}$.

*3.3.2. Generalized Markov Chain Model: GMM*

Markov Chain Model, MC is one of the popular models that are used in people's movement prediction. A state in MC (state model) corresponds to the RoI while state transition corresponds to the movement from one RoI to the next. The procedures of building a first order MC model are different from second, third, and n-order MC models. In this study, a general transformation function is used to transform $n$-order MC model into first order. Thus, all $n$-order MC models will be treated as a first order which provides more abstraction for $n$-order MC. Here, order indicates the number of visited RoIs taken as input to the prediction model.

Given a set of observable objects, $\mathbb{O} = \{o_1, \ldots o_m\}$ and a sequence $\mathbb{S} = \left\{o_i, o_j, \ldots o_r, \ldots o_k, o_p, \ldots, o_l\right\}$ where $o_i, o_j, o_r, o_k, o_p, o_l \in \mathbb{O}$, then the n-transformed sequence, $\mathbb{S}^{(n)}$, is defined as the following:

$$\mathbb{S}^{(n)} = \left\{\left(\underbrace{o_i, o_j, \ldots}_{n}; o_r\right), \ldots, \left(\underbrace{o_k, o_p, \ldots}_{n}; o_l\right)\right\} \tag{3}$$

Based on that, the $n$-order MC model can be defined as the following:

$$P(o_{n+1}|o_n, \ldots, o_1) = \frac{N(o_1, \ldots, o_n \rightarrow o_{n+1})}{\sum_{\acute{o} \in \mathbb{O}} N(o_1, \ldots, o_n \rightarrow \acute{o})} \tag{4}$$

where $N(o_1, \ldots, o_n \rightarrow o_{n+1})$ is the number of times $o_1, \ldots, o_n \rightarrow o_{n+1}$ occurs in $\mathbb{S}^{(n)}$. In the case of next location prediction, the interest regions represent the set of observable objects, while trajectories represent the sequence. The probability is represented using the five most different variables as it follows: $P(Nxt_r = r_j|Cur_r = r_i, Cur_t = t_i, Prv_r = r_{i-1} : r_{i-order}, Prv_t = t_{i-1} : t_{i-order})$ which means: What is the probability that the next RoI is $Nxt_r = r_j$ given that the current RoI is $Cur_r = r_i$, the current time is $Cur_t = t_i$, the previous RoIs are: $Prv_r = r_{i-1} : r_{i-order}$ and the previous times are: $Prv_t = t_{i-1} : t_{i-order}$. When order equals to 1, the previous RoIs and times are removed.

**Definition 5. Trajectory Transformation.** The interest region of the trajectory is transformed into a pair of values $((r_i, r_{i+1}, ..., r_{i+order-1}), r_{i+order})$, $i = 1 \ldots n$, where the first value represents the current interest region location preceded by $order - 1$ visited locations and the second one is the next location.

**Example 1.** Given $\mathbb{O} = \{1, 2, 3, 4, 5\}$ and $\mathbb{S} = \{1, 4, 2, 3, 2, 3, 2, 1, 5, 1\}$.

The transformation of the trajectory to first, second and third orders based on equation (3) will be as the following:

$\mathbb{S}^{(1)}$ = {(1→ 4), (4→ 2), (2→ 3), (3→ 2), (2→ 3), (3→ 2), (2→ 1), (1→ 5), (5→ 1) }.

$\mathbb{S}^{(2)}$ = {(1, 4→ 2), (4, 2→ 3), (2, 3→ 2), (3, 2→ 3), (2, 3→ 2), (3, 2→ 1), (2, 1→ 5), (1, 5→ 1) }.

$\mathbb{S}^{(3)}$ = {(1, 4, 2→ 3), (4, 2, 3→ 2), (2, 3, 2→ 3), (3, 2, 3→ 2), (2, 3, 2→ 1), (3, 2, 1→ 5), (2, 1, 5→ 1) }.

Table 1: s-GMM probability matrix.

| | $r_1$ | $r_2$ | $r_3$ | $r_4$ | $r_5$ |
|---|---|---|---|---|---|
| $r_1$ | - | 0.14 | - | 0.77 | 0.09 |
| $r_2$ | 0.1 | - | 0.27 | 0.63 | - |
| $r_3$ | - | 0.73 | - | - | 0.27 |
| $r_4$ | 0.36 | 0.48 | - | - | 0.15 |
| $r_5$ | 0.72 | - | 0.11 | 0.17 | - |

Table 2: t-GMM probability matrix.

| | $r_1$ | $r_2$ | $r_3$ | $r_4$ | $r_5$ |
|---|---|---|---|---|---|
| $t_0$ | - | - | - | 1 | - |
| $t_2$ | - | - | 0.81 | 0.19 | - |
| $t_3$ | - | - | - | 0.33 | 0.66 |
| $t_8$ | - | 0.37 | 0.44 | - | 0.19 |
| $t_{13}$ | 0.71 | - | - | - | 0.29 |
| ⋮ | | | ⋮ | | |

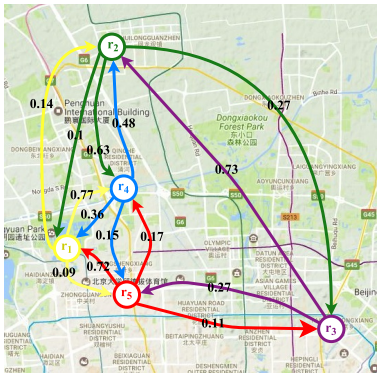



Figure 8: s-GMM transitions probability graph.

Based on the different types of trajectory information, different prediction model can be built. For the trajectory that involves only the space, a space-based model (s-GMM) is built, whereas a space-time based model (st-GMM) is built for the trajectory that has both space and time. Additionally, only the time information can be used without considering any space information, thus, a time-based prediction model (t-GMM) is built. Those were our different experimental models that we developed and set as the benchmark.

**s-GMM**. This model is similar to classical MC and it only considers RoIs (*i.e.* sequence of locations visited by a user). Thus, the observable objects-set of this model is $\mathbb{O} = \{r_1, r_2, \ldots, r_m\}$, where $m$ is the number of discovered RoIs. To build $n$-order s-GMM, the RoI transitions matrix is calculated. First, we construct different prefixes $n$ RoIs based on the trajectories. Then, we compute the frequency of each distinct RoI that appears after every prefix. The frequency value is set to 0 if there is no movement from any prefix to any RoI. The frequencies of the RoI transitions matrix are then normalized to get the probability matrix using equation 5 (see Table 1). The rows of the matrix represent the from-location(s) (the last $n$-visited interest region), while the columns represent the to-location (*i.e.* the next interest region). Each row of the matrix represents the predicted scores for RoIs of a certain user. s-GMM probability matrix can be represented as a graph where nodes and arrows represent the RoIs and the transitions between them, respectively (see Figure 8).

$$P(Nxt_r = r_j | Cur_r = r_i) = \frac{N(r_i \rightarrow r_j)}{\sum_{r_k \in \mathcal{R}} N(r_i \rightarrow r_k)} \tag{5}$$

The main issue in this model is that, the next location of the user in a specific region will be always the same region that has the high probability value. For example, based on the probability matrix in Table 1, the next location for $r_1$ is always $r_4$.

**t-GMM**. Typically, human mobility behavior is strongly influenced by the time interval information such as movement time, time of the day, day of the week etc. In other words, humans have trends in what they do in the different times of the day, week, etc. This model considers the time intervals sequence to predict the next RoI. This means that the time intervals are used to produce the candidate RoIs based on their relationships with the current RoI. To build $n$-order t-GMM, the time transitions matrix is calculated. We First construct different prefixes $n$ time intervals based on the trajectories. Then, we compute the frequency of each distinct RoI that appears after every prefix. The frequencies of the time transitions matrix are then normalized to get the time probability matrix using equation 6 (see Table 2). The rows of the matrix represent the from-time(s) (*i.e.* the current time when $n$ equals to 1), while the

Table 3: The RoI transition matrix.

| | $r_1$ | $r_2$ | $r_3$ | $r_4$ | $r_5$ |
|---|---|---|---|---|---|
| $r_1$ | - | 6 | - | 33 | 4 |
| $r_2$ | 5 | - | 13 | 30 | - |
| $r_3$ | - | 11 | - | - | 4 |
| $r_4$ | 24 | 32 | - | - | 10 |
| $r_5$ | 13 | - | 2 | 3 | - |

Table 4: The time matrix.

| | $r_1$ | $r_2$ | $r_3$ | $r_4$ | $r_5$ |
|---|---|---|---|---|---|
| $r_1$ | - | 8,7 | - | 0,23,2 | 3,4,2 |
| $r_2$ | 17,18 | - | 8,10 | 17,15,18 | - |
| $r_3$ | - | 7,8 | - | - | 9,11,13 |
| $r_4$ | 18,19,21 | 8,9 | - | - | 8,14 |
| $r_5$ | 13,12,11 | - | 4 | 1,3,11 | - |

Table 5: Tran-Time matrix.

| | $r_1$ | $r_2$ | $r_3$ | $r_4$ | $r_5$ |
|---|---|---|---|---|---|
| $r_1$,7 | - | 2 | - | - | - |
| $r_1$,8 | - | 4 | - | - | - |
| $r_1$,0 | - | - | - | 19 | - |
| $r_1$,23 | - | - | - | 13 | - |
| $r_1$,2 | - | - | - | 1 | 2 |
| $r_1$,3 | - | - | - | - | 1 |
| $r_1$,4 | - | - | - | - | 1 |
| $r_2$,17 | 3 | - | - | 8 | - |
| $r_2$,18 | 2 | - | - | 13 | - |
| $r_2$,8 | - | - | - | 5 | - |
| $r_2$,10 | - | - | - | 8 | - |
| $r_2$,15 | - | - | - | 9 | - |
| $r_3$,8 | - | 6 | - | - | - |
| $r_3$,7 | - | 5 | - | - | - |
| $r_3$,9 | - | 1 | - | - | - |
| $r_3$,11 | - | 2 | - | - | - |
| $r_3$,13 | - | 1 | - | - | - |
| $r_4$,18 | 8 | - | - | - | - |
| $r_4$,19 | 5 | - | - | - | - |
| $r_4$,21 | 11 | - | - | - | - |
| $r_4$,8 | - | 21 | - | - | 8 |
| $r_4$,9 | - | 11 | - | - | - |
| $r_4$,14 | - | - | - | - | 2 |
| $r_5$,12 | 4 | - | - | - | - |
| $r_5$,13 | 3 | - | - | - | - |
| $r_5$,11 | 6 | - | - | 1 | - |
| $r_5$,4 | - | - | 2 | - | - |
| $r_5$,1 | - | - | - | 1 | - |
| $r_5$,3 | - | - | - | 1 | - |

Table 6: dep-st-GMM probability matrix.

| | $r_1$ | $r_2$ | $r_3$ | $r_4$ | $r_5$ |
|---|---|---|---|---|---|
| $r_1$,7 | - | 1 | - | - | - |
| $r_1$,8 | - | 1 | - | - | - |
| $r_1$,0 | - | - | - | 1 | - |
| $r_1$,23 | - | - | - | 1 | - |
| $r_1$,2 | - | - | - | 0.33 | 0.67 |
| $r_1$,3 | - | - | - | - | 1 |
| $r_1$,4 | - | - | - | - | 1 |
| $r_2$,17 | 0.27 | - | - | 0.73 | - |
| $r_2$,18 | 0.13 | - | - | 0.87 | - |
| $r_2$,8 | - | - | - | 1 | - |
| $r_2$,10 | - | - | - | 1 | - |
| $r_2$,15 | - | - | - | 1 | - |
| $r_3$,8 | - | 1 | - | - | - |
| $r_3$,7 | - | 1 | - | - | - |
| $r_3$,9 | - | 1 | - | - | - |
| $r_3$,11 | - | 1 | - | - | - |
| $r_3$,13 | - | 1 | - | - | - |
| $r_4$,18 | 1 | - | - | - | - |
| $r_4$,19 | 1 | - | - | - | - |
| $r_4$,21 | 1 | - | - | - | - |
| $r_4$,8 | - | 0.72 | - | - | 0.28 |
| $r_4$,9 | - | 1 | - | - | - |
| $r_4$,14 | - | - | - | - | 1 |
| $r_5$,12 | 1 | - | - | - | - |
| $r_5$,13 | 1 | - | - | - | - |
| $r_5$,11 | 0.86 | - | - | 0.14 | - |
| $r_5$,4 | - | - | 1 | - | - |
| $r_5$,1 | - | - | - | 1 | - |
| $r_5$,3 | - | - | - | 1 | - |

columns represent the to-location (*i.e.* the next interest region).

$$P(Nxt_r = r_j | Cur_t = t_i) = \frac{N(t_i, r_j)}{\sum_{r_k \in \mathcal{R}} N(t_i, r_k)} \tag{6}$$

**st-GMM**. This model is similar to previous s-GMM except that it considers time intervals obtained from the GPS timestamp. The transformation of the trajectory will be modified to include the time interval $t$ as it follows: $((r_i, r_{i+1}, \ldots, (r_{i+order-1}, t_{i+order-1})), r_{i+order})$, where the first part contains the current location and time preceded by: $order-1$, visited RoIs and the second one is the next RoI.

To model space-time information, we consider three cases. The first case where the contribution of time information is considered as independent from the space. The model is referred to as: ind-st-GMM. The second case where the contribution of time and space information are dependent on each other, and the model is referred to as: dep-st-GMM. The final case where the contribution of space and time are adjusted using weight values. This model is refereed to as: weighted-st-GMM.

In ind-st-GMM model, the space and time matrices are built separately as explained in the previous models (*i.e.*, s-GMM and t-GMM). Both space and time information contribute equally in making the decision of next location prediction. In this case, we first find the probability of each type occurring separately, and then multiply the probabilities. The probability matrix of this model is calculated based on the equation 7.

$$P(Nxt_r = r_j | Cur_r = r_i, Cur_t = t_i) = P(r_j|r_i) \cdot P(r_j|t_i) \tag{7}$$

Building weighted-st-GMM model is similar to ind-st-GMM except that the probability of each type is multiplied by weight value to make more balanced decisions on next location prediction. Then, the final weighted probability values are summed as shown in equation 8.

$$P(Nxt_r = r_j | Cur_r = r_i, Cur_t = t_i) = W_s \cdot P(r_j|r_i) + W_t \cdot P(r_j|t_i) \tag{8}$$

where $W_s$ and $W_t$ are the weight variables in which their values belong to the interval [0,1] and $W_s + W_t$ is equal to

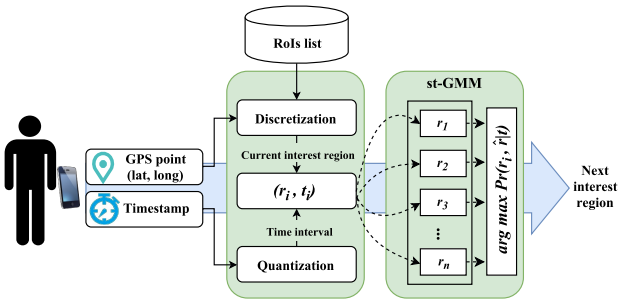



Figure 9: Next location prediction architecture.

1. When one weight value is equal to zero, it means that the contribution of specific information is excluded from the prediction decision.

In dep-st-GMM, the space-time information is used by integrating both into one matrix. dep-st-GMM can be tailored as it follows. Everyday trajectory is first divided into 24 time intervals, each of which lasts one hour long. Then, for each RoI, a list of time intervals are extracted. The transitions and time matrices among RoIs are calculated. Time matrix represents different time intervals among regions (see Table 4 for a portion of user mobility data). Finally, transitions and time matrices are integrated into one matrix called Tran-Time matrix (see Table 5). For example, the number of movements from $r_1$ to $r_2$, $r_4$ and $r_5$ are 6, 33 and 4, respectively. From user's trajectories, we observed that the six times of movements from $r_1$ to $r_2$ were two main times at 7 o'clock and four times at 8 o'clock. The probability matrix of this model is calculated based on equation 9 as shown in Table 6.

$$P(Nxt_r = r_j|Cur_r = r_i, Cur_t = t_i) = \frac{N(r_i \rightarrow r_j|t_i)}{\sum_{r_k \in \mathcal{R}} N(r_i \rightarrow r_k|t_i)} + [\![N(r_i \rightarrow r_j|t_i) = 0]\!] * P(r_j|r_i) \tag{9}$$

where

$$[\![ \text{ x } ]\!] \rightarrow \begin{cases} 1, & \text{if x is ture.} \\ 0, & \text{otherwise.} \end{cases}$$

If the $N(r_i, r_j|t)$ is 0, the model will work according to only the transitions probability matrix $P(r_j|r_i)$ computed in equation 5. In this case, dep-st-GMM applies explicit feedback to automatically update the Tran-Time matrix by adding the new time to the region's time movements list.

### *3.4. Next Location Prediction*

Before predicting the next location, user's current position (in real-time) or trajectories previously collected (in testing) are preprocessed. The GPS coordinates are converted into their RoI IDs, while the timestamps are quantized into specified time intervals. The trajectory will be represented as a sequence of regions in case of s-GMM, and a sequence of time intervals in case of t-GMM. In case of st-GMM, the trajectory will be represented as a sequence of tuples $(r_i, t_i)$ where $r_i$ is the RoI and $t_i$ is the time interval.

s-GMM and t-GMM are utilized to predict next location based on s-GMM and t-GMM probability matrix, respectively. Determining the next location depends only on the number of transitions among the RoIs. Specifically, the most frequently visited RoI is assigned the highest probability.

To make prediction using s-GMM, a set of probability values is obtained based on the current RoI. The RoI with the highest probability value is predicted to be the most likely next RoI of a certain user as shown in equation 10. Similarly, in t-GMM probability matrix, the location of the column with the highest value for the row that represents the $n$ previous time interval(s) is retrieved (see equation 11).

$$s - GMM(r_i) = \arg\max_{\hat{r} \in \mathcal{R}} P(\hat{r}|r_i) \tag{10}$$

$$t - GMM(r_i) = \arg\max_{\hat{r} \in \mathcal{R}} P(\hat{r}|t_i) \tag{11}$$

Table 7: GeoLife dataset trajectory values.

| | Latitude | Longitude | Altitude | Number of days | Timestamp |
|---|---|---|---|---|---|
| $GPS_1$ | 39.984094 | 116.319236 | 492 | 39744.2451 | 2008-10-23 05:53:05 |
| $GPS_2$ | 39.984198 | 116.319322 | 492 | 39744.2452 | 2008-10-23 05:53:06 |
| $GPS_3$ | 39.984224 | 116.319402 | 492 | 39744.2452 | 2008-10-23 05:53:11 |
| $GPS_4$ | 39.984211 | 116.319389 | 492 | 39744.2453 | 2008-10-23 05:53:16 |
| ⋮ | | | ⋮ | | |

Table 8: Gowalla dataset trajectory values.

| | Check-in time | Latitude | Longitude | Location ID |
|---|---|---|---|---|
| $GPS_1$ | 2010-07-24 13:45:06 | 53.3648119 | -2.2723465833 | 145064 |
| $GPS_2$ | 2010-07-24 3:44:58 | 53.360511233 | -2.276369017 | 1275991 |
| $GPS_3$ | 2010-07-24 13:44:46 | 53.3653895945 | -2.2754087046 | 376497 |
| $GPS_4$ | 2010-07-24 13:44:38 | 53.3663709833 | -2.2700764333 | 98503 |
| ⋮ | | ⋮ | | |

where $\hat{r}$ is the predicted location and $\mathcal{R}$ is the set of discovered RoIs in a user movement region.

In all st-GMM models, time context is included to improve the prediction model. Figure 9 shows the overview of location prediction part. Determining the next location depends not only on the number of transition but also on the movement time. The RoI with the highest value is retrieved as the location the user will visit next as shown in equation 12.

$$st - GMM(r_i, t_i) = \arg\max_{\hat{r} \in \mathcal{R}} P(\hat{r}|r_i, t_i) \tag{12}$$

## 4. Experiments and Results

Various experiments are conducted to evaluate the performance of the proposed approach. The settings of the experiment, the datasets used and experimental results are explained in this section.

### *4.1. Geolife and Gowalla Datasets*

The proposed models are evaluated using two publicly available real-world datasets, *i.e.*, GeoLife and Gowalla.

- **GeoLife** [38] is a large GPS trajectory dataset which belongs to GeoLife project (Microsoft Research Asia). The dataset was collected from 182 users in a period of over five years (from April 2007 to August 2012). It contains 24,876,978 recorded points and 18,670 trajectories with a total distance of 1,292,951 kilometers and a total duration of 50,176 hours. The dataset covers different cities located in China, USA and Europe, but the majority of the data was from Chinese cities. GPS trajectories were recorded using different GPS loggers and GPS phones every $1 \sim 5$ seconds or every $5 \sim 10$ meters per point. The trajectories are represented by a sequence of timestamped points and each point contains latitude, longitude, altitude and other information. Table 7 shows a partial trajectory of one user. After pre-processing phase, the number of trajectories and recorded points are: 17,467 and 23,232,752, respectively.

- **Gowalla** [6] provides check-in records that contain the location ID, the corresponding check-in timestamp and other information, Table 8. This dataset contains 6,442,890 check-ins over the period from February 2009 to October 2010. The dataset was preprocessed by removing the users who had the same number of check-ins and locations. Moreover, users with less than 20 check-ins or less than 5 locations were removed from the dataset. Finally, the check-in records are organized as locations sequences.

The two datasets: GeoLife and Gowalla have several general differences such as: the size or the number of records in each dataset, input variables in each dataset, the distribution of the points in the movement area, in addition to meta-data related to the collection process. The most important difference between them is that GeoLife should be preprocessed in order to discover the RoIs. Different RoIs' discovery algorithms are used in the experiments. The RoIs in Gowalla dataset are given different IDs during collecting the data.

### 4.2. Approach Specification

#### 4.2.1. DRoI Algorithm Specifications

To select the parameters specifications, a grid search method is implemented on one user trajectories. For DCRoI, three parameters are considered: the distance ($\delta$), stay time ($\tau$) and region density ($rd$). In this study those are: $\delta$ = 200 meters, $\tau$ = 20 minutes and $rd$ = 400 points. For CRoI clustering using DBSCAN, two parameters are required: maximum distance $\varepsilon$ between any two points and the minimum number of points $minPts$ required to form a dense region. In this study, those are: $\varepsilon$ = 300 meters and $minPts$ = 3. $k$-means algorithm is used with $k$ = 8.

#### 4.2.2. Prediction Models Specifications

In this paper, first, second, third, fourth and fifth order GMM models are implemented. The models are based on s-GMM, t-GMM and st-GMM. The trajectories' datasets of each user are split into 80% training and 20% testing set. The partition is repeated 10 times with different randomly selected trajectory files. The accuracy score of each case from each user is then calculated and the final results of all users are averaged. The weight values $W_s$ and $W_t$ are set to 0.8 and 0.2, respectively.

### 4.3. Metrics

#### 4.3.1. DRoI Algorithm Evaluation Metrics

In order to evaluate the performance of various algorithms used to discover RoIs, trajectory dataset must be annotated with meaningful locations. The GeoLife dataset used in this study is not originally annotated. We asked 10 participants to manually annotate the RoIs into a set of semantically meaningful labels based on the density of trajectories and the real locations on the maps. Each participant annotated the RoIs for 20 randomly selected users' trajectory. These manually annotated outlying locations serve as ground truth in the experiments. Soundness and completeness properties are used as metrics to evaluate the effectiveness of the algorithms used to discover RoIs. Soundness and completeness are defined in the Equations: 13 and 14:

$$completeness = \frac{|correct\ RoI\ discovered|}{|all\ real\ RoIs|} \tag{13}$$

$$soundness = 1 - \frac{|incorrect\ RoI\ discovered|}{|all\ RoIs\ discovered|} \tag{14}$$

#### 4.3.2. Prediction Evaluation Metrics

The prediction evaluation procedure starts with reading the previously visited RoIs in each trajectory sequentially. Then, the next RoIs to be visited are predicated based on these readings. Finally, the predicted RoIs are mapped into the real data. Accuracy score is used as a metric in all the experiments to assess the efficiency of the prediction model. The **Accuracy@N** is defined as the ratio of the number of correct predictions to the total number of predictions.

To compute the Accuracy@N score, first, a ranked list is populated with all potential next RoIs arranged in a descending order according to their probabilities. Then, the percentage of RoIs that are accurately predicted in the top $N$ within the ranked list are calculated. In our study, we only report Accuracy@N with $N$ = 1 and 2.

Suppose that $U$ is the set of users, $L_u$ denotes a set of real visited RoIs by a user $u$ in the test data and $P_{N,u}$ the set of top $N$ predicted RoIs, the Accuracy@N can be defined as it follows:

$$Accuracy@N = \frac{1}{|U|} \sum_{u \in U} \frac{|L_u \cap P_{N,u}|}{|L_u|} \tag{15}$$

#### 4.3.3. Comparison to Other Prediction Models

The proposed approach is compared with baseline models in the task of predicting user's next location [1]. We selected those models in particular as based on using the same datasets we have used or based on their usage of MC prediction models.

[1] We have implemented all baseline models based on our understanding of the original papers.

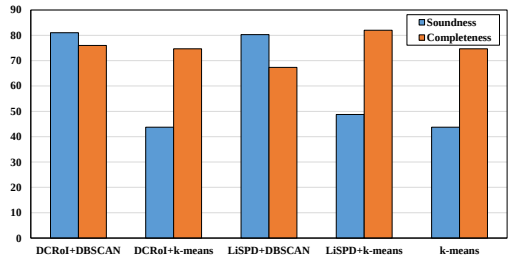



Figure 10: Soundness and completeness of DRoI algorithms.

- **MC**: It is the basic baseline method, that uses $k$-means for extracting RoIs in a user movement region and MC for location prediction [5].
- ***n*-MMC**: MC model is used to predict user's next location [9]. $n$-MMC depends on the sequence of $n$ previous visited locations and DJ Cluster for discovering RoI.
- **HPHD**: It is proposed in [10] where spatial, hourly and daily information are used for location prediction.
- **AR**: It is proposed in [7, 14] to predict next location of a moving object.
- **NNs**: This model is introduced in [19, 18, 26] for predicting people next movements.

Additionally, we compare our algorithm of discovering RoIs with **LiSPD** [20, 36] and ***k*-means** to show the different RoIs discovered by these algorithms. The same trajectories and parameters specifications are considered for these algorithms. For both HPHD, AR and NNs, $k$-means is used as a basic algorithm for extracting RoIs.

### *4.4. Comparison Results: DRoI Algorithms*

Figure 10 illustrates a comparison of RoIs discovery algorithms in terms of soundness and completeness. High completeness value means that most of the significant locations are discovered, while the high soundness value means that most of the insignificant locations are ignored. The best soundness and completeness values are achieved by DCRoI+DBSCAN and LiSPD+$k$-means by 81% and 82%, respectively. Furthermore, it can be noted that the variance of completeness values is much smaller than variance of soundness values. Thus, soundness is more appropriate metric to evaluate the proposed algorithms. The highest soundness values are 81% and 80% for DCRoI+DBSCAN and LiSPD+DBSCAN, respectively. DCRoI is better than LiSPD when DBSCAN is used as a second level algorithm. Discovering many CRoIs in the first level using LiSPD negatively affects the clustering process of the second level. Additionally, it can be observed that using $k$-means results in high completeness value because of the large defined number of clusters ($k = 8$) which might be greater than the number of the real RoIs of some users. DBSCAN results in better soundness than $k$-means. Grouping in DBSCAN depends on the density of points with many nearby neighbors rather than the mean of all points that belong to the same cluster.

### *4.5. Comparison Results: Location Prediction*

Table 9 illustrates the performance comparison on the two datasets in terms of Accuracy@N with N = 1 and 2. We can find that in the GeoLife dataset, the models where $k$-means is used to find the RoIs (*i.e.* MC, HPHD, AR and NNs) do not perform well. The reason is that $k$-means clustering algorithm is not tailored for geolocated data where the grouping process depends on the mean of all points that belong to the same cluster. MC, HPHD and AR models have achieved approximately similar results under Accuracy@1. NNs slightly improves the results comparing with MC, HPHD and AR, but still performs poor in terms of predicting the future locations. $n$-MMC improves the performance greatly comparing with the other baseline models. Extracting the RoIs using DJ cluster is better than $k$-means, where the number of discovered RoIs depends on the people's mobility traces [39].

The worst results are obtained by t-GMM when only time sequences are used in building the prediction model. In contrast, a great improvement that is brought by s-GMM which is considered as the best model among the compared ones. Using the space information results in higher prediction accuracy than time information. Additionally, using

Table 9: Performance comparison on the datasets evaluated by Accuracy@N. Best scores are in bold.

| Models | GeoLife | | Gowalla | |
|---|---|---|---|---|
| | Accuracy@1 | Accuracy@2 | Accuracy@1 | Accuracy@2 |
| MC | 58.46 | 66.91 | - | - |
| *n*-MMC | 68.6 | - | - | - |
| HPHD | 58.07 | 73.23 | 22.37 | 36.99 |
| AR | 59.32 | 67.43 | 21.23 | 36.29 |
| NNs | 63.01 | 82.05 | **25.07** | **42.3** |
| s-GMM | 68.92 | 84.21 | 21.38 | 36.84 |
| t-GMM | 47.32 | 64.53 | 19.11 | 32.76 |
| ind-st-GMM | 66.93 | 78 | 20.42 | 37.48 |
| dep-st-GMM | **70.52** | **88.4** | 22.23 | 39.49 |
| weighted-st-GMM | 69.73 | 88.25 | 22.3 | 41.58 |

Table 10: Performance comparison with different order evaluated by Accuracy@1. Best scores are in bold.

| Dataset | Models | Order | | | | |
|---|---|---|---|---|---|---|
| | | First | Second | Third | Fourth | Fifth |
| GeoLife | MC | 58.46 | 64.65 | 63.89 | 63.72 | 63.67 |
| | *n*-MMC | 68.6 | 69.1 | 63.2 | - | - |
| | HPHD | 58.07 | 64.15 | 65.19 | 65.01 | 64.32 |
| | s-GMM | 68.92 | 72.53 | 70.96 | 69.41 | 66.45 |
| | t-GMM | 47.32 | 50.39 | 52.01 | 52.35 | 52.22 |
| | ind-st-GMM | 66.93 | 70.99 | 70.69 | 68.5 | 66.25 |
| | dep-st-GMM | **70.52** | 73.44 | 71.16 | 69.56 | 66.58 |
| | weighted-st-GMM | 69.73 | **73.86** | **71.19** | **69.92** | **67.23** |
| Gowalla | MC | - | - | - | - | - |
| | *n*-MMC | - | - | - | - | - |
| | HPHD | **22.37** | 22.4 | 20.36 | 17.7 | 14.89 |
| | s-GMM | 21.38 | 22.92 | 20.48 | 17.80 | 14.95 |
| | t-GMM | 19.11 | 19.66 | 14.47 | 11.21 | 9.73 |
| | ind-st-GMM | 20.42 | 22.47 | 20.19 | 18.13 | 17.53 |
| | dep-st-GMM | 22.23 | 24.06 | 22.41 | 20.66 | 17.64 |
| | weighted-st-GMM | 22.3 | **24.17** | **22.64** | **20.74** | **17.65** |

only the RoIs sequences (or only time sequences) are not enough inputs in location prediction as the models can not well capture the users' mobility behaviors. Including the time factor in ind-st-GMM did not cause any performance improvement. Using the probability of time transition matrix can change the final prediction decision when combining with the probability of space transition matrix. In other words, using time context as a basic component either individually as, in t-GMM or with space context, as in both ind-st-GMM and HPHD is not recommended. This is due to the different degrees of importance of space and time context when generating the information for location prediction (*i.e.*, space information is the most important one). Thus, the time context must be formulated in a way to add more information to the space context as in weighted-st-GMM. ind-st-GMM and HPHD models are similar in terms of using time information as a basic components in building the prediction model. However, they are different in terms of probability calculation where HPHD model depends on the Gaussian distribution and therefore there is no probability that equals to zero as in ind-st-GMM model. weighted-st-GMM performs better than ind-st-GMM and HPHD. This is due to the importance of space information where it is given higher weight than time information. Moreover, we can observe that, dep-st-GMM outperforms the compared models on the GeoLife dataset measured by all the metrics. The superiority of dep-st-GMM can be attributed to using the algorithm of RoIs discovery and the addition of time context information.

On Gowalla dataset, it is clearly shown that NNs results in a better prediction accuracy than other models. While NNs have the ability to learn non-linear and complex relationships, probabilistic models in general depend on distributional assumptions. On the other hand, NNs are computationally intensive due to the large number of parameters that should to be updated numerous times. As a result, using smartphone devices with limited resources to train NNs and make a prediction can be computationally expensive and time-consuming.

MC and *n*-MMC models are excluded from the comparison. They are based on the classical MC but with different clustering algorithms. Gowalla dataset is used without discovering the interest regions.

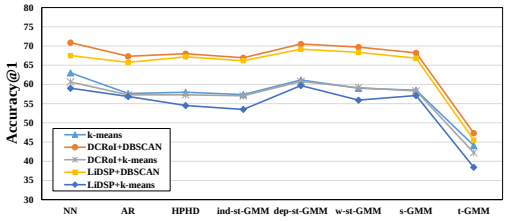
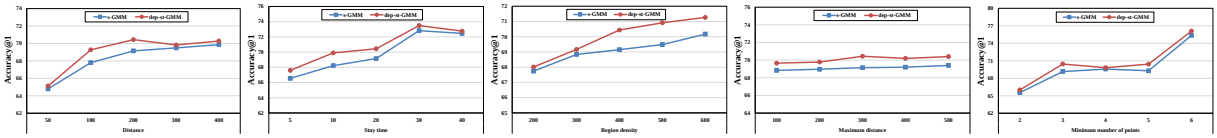



Figure 11: Prediction models with different RoIs discovery algorithms.



(a) Distance. (b) Stay time. (c) Region density. (d) Maximum distance. (e) Minimum number of points.

Figure 12: The effects of DRoIs parameters.

#### *4.5.1. Impact of Different Orders*

In order to explore the influence of different orders, we perform experiments with first, second, third, fourth and fifth order. Results are shown in Table 10. From the table, we can see that the prediction accuracies of dep-st-GMM and weighted-st-GMM are higher than others. On Gowalla dataset, HPHD achieved the best results when order equals to 1. However, the prediction accuracy often decreases when the order is greater than 2 where more constraints must be applied. The best result on both datasets is achieved by weighted-st-GMM. It shows that the impact of RoIs sequences and time interval information are more significant on modeling user's behavior than using only one of them. When $n$-order equals to 2, weighted-st-GMM achieved better accuracy with 73.86% and 24.17% on GeoLife and Gowalla datasets, respectively. In all experiments, second order results in a higher prediction accuracy than other $n$-order.

#### *4.5.2. Impact of RoI Discovering Algorithms*

Discovering all locations that represent a real RoI to a certain user enables to build a real trajectory. Then, it helps the model and extract the user's mobility pattern which means more accurate prediction models. Therefore, the effect of RoIs discovering algorithms is explored.

As stated in section 3.2, discovering RoIs takes place in two levels of clustering. For the first level, DCRoI and LiSPD algorithms are used, whereas DBSCAN and *k-means* are used for the second level. For testing, different algorithms of discovering RoIs are considered: DCRoI+DBSCAN, LiSPD+DBSCAN, DCRoI+$k$-means and LiSPD+$k$-means. $k$-means is used also as one clustering level. The results are reported in Figure 11.

It is shown that when using the algorithm we proposed for discovering RoIs (DCRoI+DBSCAN), the models perform better. This indicates that DRoI is effective in discovering the significant RoIs, which enables the models to extract a user' mobility behavior and as a consequence, it can be used to build more accurate models. DCRoI+DBSCAN algorithm produces highest accuracy rate of the prediction model and then LiSPD+DBSCAN. The three other algorithms (*i.e.* DCRoI+$k$-means, LiSPD+$k$-means and $k$-means) have almost the same prediction accuracy rate. Using DCRoI in the first level improves the performance of the prediction models, better than LiSPD. Regarding second level, DBSCAN results in better performance than $k$-means.

Adding time factor to a real discovered RoI positively affects the prediction accuracy. When $k$-means is used, some significant RoIs could be missed. Also, some regions that do not carry semantic meaning (like 'roads') might be detected as a RoI. Thus, including the time to RoIs that represent insignificant locations will not provide a good prediction accuracy.

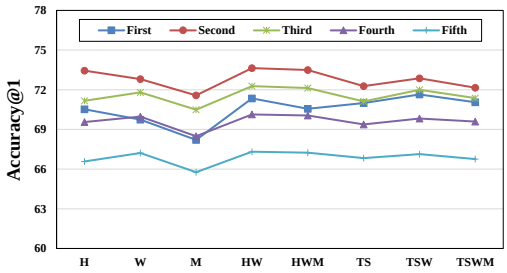



Figure 13: The effect of different time intervals information on dep-st-GMM performance.

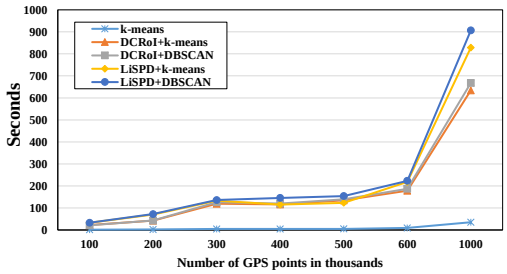



Figure 14: Running time of different RoI discovery algorithms.

### *4.5.3. Impact of DRoI Parameters*

We also study the impact of the five DRoIs parameters on the performance of s-GMM and dep-st-GMM models: the distance ($\delta$), stay time ($\tau$), region density ($rd$), maximum distance ($\varepsilon$) and the minimum number of points ($minPts$). We conduct several experiments with various values as shown in Figure 12. We start by varying the value of one parameter while fixing the others and then studying how the prediction accuracy is affected. The same procedure is then repeated for the rest of the parameters.

In general, we observe that the models performance gives the best results as long as we increase the parameters values. Small values mean that many RoIs will be discovered which limits its efficiency in discovering significant regions and, hence, impairing the prediction performance. On the contrary, large values mean that few RoIs with large areas will be discovered. Thus, it will be easy for the models to extract the movement pattern and then predict the next location.

### *4.5.4. Impact of Different Time Intervals*

In addition to the hour part of the timestamp (H), we consider dep-st-GMM with different time intervals types: weekday (W), month (W), hour weekday (HW), hour weekday month (HWM). Moreover, we experiment with timeslots day hours (TS), timeslots weekday (TSW), timeslots weekday month (TSWM). Timeslots imply that the day hours are divided into: 'Morning', 'Noon', 'Afternoon', 'Evening' and 'Night'. Weekday types are either weekend days or weekday days.

As shown in Figure 13, the prediction model with HW showed the best performance. Using weekday types is shown to be very useful and such information is a significant predictor of users next location. It shows that users follow similar patterns in the weekdays in comparison to the weekends. Results showed also that with time related features, using small time periods (e.g. hours of the day) is more predictive features than larger time features such as the month of the year. Users may follow similar behavioral patterns across several months (when excluding vacation months).

### *4.5.5. Running Time*

Finally, we measure the exact running time of the RoIs discovery algorithms and the prediction models. For the RoI discovery algorithms, we simply run the algorithms on trajectories of different users with different number of GPS points, and then measure the amount of time. For the prediction models, we run the models on trajectories of a

Table 11: Running time in seconds.

| Models | Model building | Predicting |
|---|---|---|
| AR | 0.59±0.11 | 0.37±0.07 |
| NNs | 1464.3±23.7 | 3.07±0.66 |
| HPHD | 24.74 ± 2.11 | 1.63 ± 0.24 |
| s-GMM | 0.19±0.03 | 0.36±0.03 |
| t-GMM | 0.17±0.02 | 0.43±0.12 |
| ind-st-GMM | 0.2±0.02 | 0.42±0.06 |
| dep-st-GMM | 0.24±0.01 | 0.42±0.1 |
| weighted-st-GMM | 24.80±1.90 | 0.48±0.01 |

randomly selected user and measure the amount of time for building the model and the predicting. For each model, the experiment is repeated 10 times, and the average and standard deviation are then calculated. All experiments were conducted on iMac, equipped with a 3.06 GHz Intel Core 2 Due CPU and 4GB RAM.

Figure 14 shows the algorithms' running time in seconds. As expected, running times grow gradually with the dataset size. We observe that our algorithm is faster that LiSPD due to the less number of iterations. Of course, k-means is still significantly faster, but it is not tailored for geolocated data.

Table 11 shows the prediction models' exact running time in seconds. It can be noted that, during model building, our proposed models are less than others. During the prediction, all st-GMM models' running times are more than AR's due to including an extra factor (the time context). t-GMM is faster that s-GMM during model building but slower during the prediction phase due to the large number of time intervals compared to the number of RoIs for some users. However, NNs model has the highest running time. This is not surprising as the NNs model has a large number of parameters that need to be updated numerous times. It is worth mentioning that our models are faster than others for any dataset size.

*4.5.6. Summary*

Overall results show that weighted-st-GMM outperforms most of the state-of-the-art models despite the fact that the model is simple. As the training set size increases (for example, 90% training and 10% testing), the accuracy is improved since the model can learn more (weighted-st-GMM achieved 74.7%).

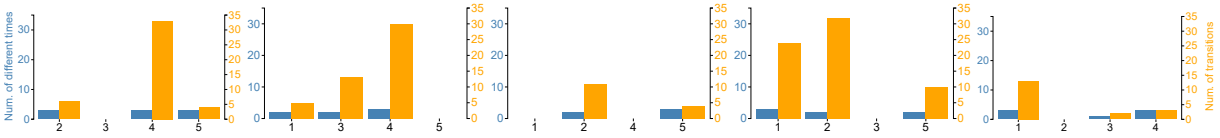
Regarding the different evaluated orders, the prediction accuracy improves as the $n$ order increases but then decreases gradually when $n > 2$ due to having more constraints that must be applied to get the next location. For example, the inputs of a fifth order model are the last five RoIs the user visited which are rarely visited or occurred in the same RoIs order. In first order Model, only the current RoI is considered. Thus, more constraints means less accuracy.

In addition to that, the impact of the time factor becomes little important when the order is high. For example, with first order, there is an obvious difference between the prediction accuracy of space-based and space-time-based models but the gap between them is narrowing with the increase of the order.

In general, it is possible for a user to change his/her mobility patterns over time. All the prediction models are unable to deal with new context of movements. In other words, the models fail to make a prediction for new locations in unknown territories (*i.e.* the unseen locations) that did not appear in the user's history. Applying explicit feedback to automatically update the transition matrix can be used as a simple solution to improve the prediction accuracy.

As indicated in the description of the prediction phase, the model retrieves the RoI with the highest probability value given the current location. In this regard, it is possible to have the same probability value for more than one location. In our model, we choose to retrieve the first one added to transition matrix.

***Movement Regularity***. As shown in Figure 15, the movement regularity of a user can be identified based on the transition matrix (Table 3) and time matrix (Tables 4). For example, the number of transitions from $r_5$ to $r_4$ is equal to the number of movement times between them. This means that a user moves to $r_4$ in different times. Thus, there is no regularity in the movement between these two locations. Additionally, the RoIs that are visited by the user at a specific time can be extracted. For example, $r_2$ can be considered as a workplace because the number of transitions from $r_4$ to $r_2$ is much greater than the number of the different values of movement times.




(a) Movements from $r_1$. (b) Movements from $r_2$. (c) Movements from $r_3$. (d) Movements from $r_4$. (e) Movements from $r_5$.

Figure 15: User movement regularity.

## 5. Conclusion and Future Work

This paper presents an approach to discover Regions-of-Interest (RoIs) in the users movement area and then predicts their future locations, which play a key role in the success of advanced location-based services. The RoIs discovery algorithm (DRoI) presented in this paper includes two steps. The first step is to group GPS points based on three threshold conditions: distance, time and region density, which identifies a set of Candidate RoIs (CRoIs). The second step is to perform clustering using DBSCAN algorithm on the CRoIs to obtain the RoIs. Soundness and completeness are used as metrics to evaluate the proposed RoIs discovery algorithm. We found that our algorithm is an effective one, compared to other relevant algorithms in the literature. While the algorithm is able to find most of the RoIs, the algorithm's overall execution time is less than other algorithms. Based on Markov Chain (MC), a model for predicting a user's next location is proposed. Moreover, a general transformation function for the trajectory is used to include the space and time context. Any order MC will be processed as first order, which helps to make more abstraction on $n$-order. The latter performs better than space-based and time-based models, but the gap between the models' prediction accuracy is narrowing with the increase of the order. We evaluated the proposed approach with real-world datasets: GeoLife and Gowalla. Overall, the approach where a DRoI and second order MC were used achieved better accuracy.

Future work will include investigating the approach performance on more practical system through real-time data. We also plan to evaluate and classify different users' behaviors/patterns based on their daily, weekly and monthly habits. Additionally, more contextual information will be incorporated, such as RoI arrival time, the time spent in a RoI and the distances between RoIs. Last, since we only focused on the user next location prediction, we will consider, in future, our approach to other problems of user behaviors' prediction.

## References

[1] Agrawal, R., Srikant, R., 1994. Fast algorithms for mining association rules in large databases. In: Proceedings of the 20th International Conference on Very Large Data Bases. VLDB '94. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, pp. 487–499.

[2] Agrawal, R., Srikant, R., 1995. Mining sequential patterns. In: Data Engineering, 1995. Proceedings of the Eleventh International Conference on. IEEE, pp. 3–14.

[3] Asahara, A., Maruyama, K., Sato, A., Seto, K., 2011. Pedestrian-movement prediction based on mixed Markov-chain model. GIS '11 Proceedings of the 19th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems, 25–33.

[4] Asgari, F., Gauthier, V., Becker, M., 2013. A survey on human mobility and its applications. arXiv preprint arXiv:1307.0814.

[5] Ashbrook, D., Starner, T., 2003. Using GPS to learn significant locations and predict movement across multiple users. Personal and Ubiquitous Computing 7, 275–286.

[6] Cho, E., Myers, S. A., Leskovec, J., 2011. Friendship and mobility: User movement in location-based social networks. In: Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. KDD '11. ACM, New York, NY, USA, pp. 1082–1090.

[7] Daoui, M., Belkadi, M., Chamek, L., Lalam, M., Hamrioui, S., Berqia, A., 2013. Mobility prediction and location management based on data mining (December), 1–4.

[8] Ester, M., Kriegel, H., Sander, J., Xu, X., 1996. A density-based algorithm for discovering clusters in large spatial databases with noise. In Proceedings of 2nd International Conference on KDD, 226–231.

[9] Gambs, S., Killijian, M.-O., Del Prado Cortez, M. N. n., 2012. Next place prediction using mobility Markov chains. Proceedings of the First Workshop on Measurement Privacy and Mobility MPM 2012, 1–6.

[10] Gao, H., Tang, J., Liu, H., 2012. Mobile location prediction in spatio-temporal context. Nokia Mobile Data Challenge Workshop.

[11] H. Kaaniche, F. K., 2010. Mobility Prediction in Wireless Ad Hoc Networks using Neural Networks. Journal of Telecommunications 2 (1), 95–101.

[12] Huang, W., Lit, M., Hut, W., Song, G., Xie, K., 2013. Hierarchical Destination Prediction Based on GPS History, 972–977.

[13] Kang, J. H., Welbourne, W., Stewart, B., Borriello, G., 2004. Extracting places from traces of locations. In: Proceedings of the 2nd ACM international workshop on Wireless mobile applications and services on WLAN hotspots. ACM, pp. 110–118.

[14] Kedia, S. P., 2012. An object tracking scheme for wireless sensor networks using data mining mechanism. 2012 IEEE Network Operations and Management Symposium, 526–529.
[15] Killijian, M.-O., Roy, M., Trédan, G., 2010. Beyond san francisco cabs: Building a*-lity mining dataset. pp. 75–78.
[16] Klepeis, N. E., Nelson, W. C., Ott, W. R., Robinson, J. P., Tsang, A. M., Switzer, P., Behar, J. V., Hern, S. C., Engelmann, W. H., 2001. The national human activity pattern survey (nhaps): a resource for assessing exposure to environmental pollutants. Journal of Exposure Science and Environmental Epidemiology 11 (3), 231–252.
[17] Krumm, J., 2011. Ubiquitous advertising: The killer application for the 21st century. IEEE Pervasive Computing 10 (1), 66–73.
[18] L. Vintan, A. Gellert, J. P., Ungerer, T., 2004. Person movement prediction using neural networks. In First Workshop on Modeling and Retrieval of Context, Ulm, Germany 114 (4), 1–12.
[19] Leca, C.-L., Nicolaescu, I., Rîncu, C.-I., 2015. Significant location detection & prediction in cellular networks using artificial neural networks. Computer Science and Information Technology 3 (3), 81–89.
[20] Li, Q., Zheng, Y., Xie, X., Chen, Y., Liu, W., Ma, W.-Y., 2008. Mining user similarity based on location history. In: Proceedings of the 16th ACM SIGSPATIAL international conference on Advances in geographic information systems. ACM, p. 34.
[21] Liu, Q., Wu, S., Wang, L., Tan, T., 2016. Predicting the Next Location : A Recurrent Model with Spatial and Temporal Contexts, 194–200.
[22] Marmasse, N., Schmandt, C., 2000. Location-aware information delivery withcommotion. In: International Symposium on Handheld and Ubiquitous Computing. Springer, pp. 157–171.
[23] Mathew, W., Raposo, R., Martins, B., 2012. Predicting future locations with hidden markov models. Proceedings of the 2012 ACM Conference on Ubiquitous Computing, 911–918.
[24] Monreale, A., Pinelli, F., Trasarti, R., 2009. WhereNext : a Location Predictor on Trajectory Pattern Mining. Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining - KDD '09, 637–645.
[25] Morzy, M., 2007. Mining frequent trajectories of moving objects for location prediction. Proceedings of the 5th International Conference on Machine Learning and Data Mining in Pattern Recognition, 667–680.
[26] Parija, S., Ranjan, R. K., Sahu, P. K., 2013. Location prediction of mobility management using neural network techniques in cellular network. In: Emerging Trends in VLSI, Embedded System, Nano Electronics and Telecommunication System (ICEVENT), 2013 International Conference on. IEEE, pp. 1–4.
[27] Pei, J., Han, J., Mortazavi-Asl, B., Pinto, H., Chen, Q., Dayal, U., Hsu, M., 2001. Prefixspan: Mining sequential patterns by prefix-projected growth. In: Proceedings of the 17th International Conference on Data Engineering. IEEE Computer Society, pp. 215–224.
[28] Rao, B., Minakakis, L., 2003. Evolution of mobile location-based services. Communications of the ACM 46 (12), 61–65.
[29] Ryan, C., Brown, K. N., 2012. Occupant Location Prediction Using Association Rule Mining, 23–28.
[30] Solanas, A., Patsakis, C., Conti, M., Vlachos, I. S., Ramos, V., Falcone, F., Postolache, O., Pérez-Martınez, P. A., Di Pietro, R., Perrea, D. N., et al., 2014. Smart health: A context-aware health paradigm within smart cities. IEEE Communications Magazine 52 (8), 74–81.
[31] Song, L., Kotz, D., Jain, R., He, X., 2006. Evaluating next-cell predictors with extensive wi-fi mobility data. IEEE Transactions on Mobile Computing 5 (12), 1633–1649.
[32] Vu, L., Do, Q., Nahrstedt, K., 2011. Jyotish: Constructive approach for context predictions of people movement from joint wifi/bluetooth trace. Pervasive and Mobile Computing 7 (6), 690–704.
[33] Vu, L., Nguyen, P., Nahrstedt, K., Richerzhagen, B., 2014. Characterizing and modeling people movement from mobile phone sensing traces. Pervasive and Mobile Computing 17, 220–235.
[34] Xia, H., Qiao, Y., Jian, J., Chang, Y., 2014. Using smart phone sensors to detect transportation modes. Sensors 14 (11), 20843–20865.
[35] Yu, Y., Chen, X., 2015. A survey of point-of-interest recommendation in location-based social networks. In: Workshops at the Twenty-Ninth AAAI Conference on Artificial Intelligence.
[36] Yuan, J., Zheng, Y., Zhang, L., Xie, X., 2013. T-finder: A recommender system for finding passengers and vacant taxis. IEEE Transactions on Knowledge and Data Engineering, 2390–2403.
[37] Zheng, Y., 2015. Trajectory data mining: an overview. ACM Transactions on Intelligent Systems and Technology (TIST) 6 (3), 1–29.
[38] Zheng, Y., Xie, X., Ma, W., 2010. Geolife: A collaborative social networking service among user, location and trajectory. IEEE Data Engineering Bulletin 2 (33), 32–40.
[39] Zhou, C., Frankowski, D., Ludford, P., Shekhar, S., Terveen, L., 2004. Discovering personal gazetteers: an interactive clustering approach. In: Proceedings of the 12th annual ACM international workshop on Geographic information systems. ACM, pp. 266–273.