sensors

MDPI

Article

Next Location Prediction Based on an Adaboost-Markov Model of Mobile Users †

Hongjun Wang, Zhen Yang * and Yingchun Shi

National University of Defense Technology, Hefei 230037, China; mielideman@163.com (H.W.); ycshi_eei@163.com (Y.S.)

* Correspondence: eei_yz@163.com; Tel.: +86-132-9560-9736

† This paper is an extended version of Zhen Yang, Wang Hong-jun. Important Location Identification and Personal Location Inference Based on Mobile Subscriber Location Data Preparation of Camera-Ready Contributions to SCITEPRESS Proceedings. In Proceedings of the 2018 International Conference on Smart Materials, Intelligent Manufacturing and Automation, Nanjing, China, 24–26 May 2018.

Received: 28 January 2019; Accepted: 15 March 2019; Published: 26 March 2019

Abstract: As an emerging class of spatial trajectory data, mobile user trajectory data can be used to analyze individual or group behavioral characteristics, hobbies and interests. Besides, the information extracted from original trajectory data is widely used in smart cities, transportation planning, and anti-terrorism maintenance. In order to identify the important locations of the target user from his trajectory data, a novel division method for preprocessing trajectory data is proposed, the feature points of original trajectory are extracted according to the change of trajectory structural, and then important locations are extracted by clustering the feature points, using an improved density peak clustering algorithm. Finally, in order to predict next location of mobile users, a multi-order fusion Markov model based on the Adaboost algorithm is proposed, the model order k is adaptively determined, and the weight coefficients of the 1~k-order models are given by the Adaboost algorithm according to the importance of various order models, a multi-order fusion Markov model is generated to predict next important location of the user. The experimental results on the real user trajectory dataset Geo-life show that the prediction performance of Adaboost-Markov model is better than the multi-order fusion Markov model with equal coefficient, and the universality and prediction performance of Adaboost-Markov model is better than the first to third order Markov models.

Keywords: next location prediction; trajectory division; important locations; density clustering; Markov model; Adaboost algorithm

1. Introduction

In recent years, with the rapid development of mobile communication technology and the increasingly powerful functions of smart mobile terminal networks, smart terminal devices such as mobile phones and tablet computers have gradually surpassed personal computers and become the most widely used information devices for people. At the same time, the rapid development of global navigation and positioning systems has provided accurate and real-time location information for smart mobile terminals [1]. Location-based services (LBSs) have become some of the most popular terminal information services, and a large number of related studies have also proved the hidden value of mining the resulting massive location data. Currently, location-based services mainly include location query, path planning, location sharing, etc. These functions are mostly focused on providing users with relevant services at the current location. In order to make the service more forward-looking, in recent years, a large number of domestic and foreign scholars have turned their attention to the location prediction of mobile users [2–5]. Location prediction concentrates on predicting the next location of

mobile users with maximum likelihood, on the basis of the user's past trajectory data. The current sources of trajectory data are mainly the following: (1) Data collected by the sensors, such as Global Positioning System (GPS) [6–9], wireless fidelity (Wi-Fi) sensors [10–14] and base stations [15–17]. (2) Users' check-in data on social software [18,19], in recent years, with the promotion of location-based social networks, a large number of check-in data with time and space tags have been generated. (3) Vehicle traffic data recorded by city traffic bayonets [20,21].

Mobile user location prediction has high research value and broad application prospects, as research shows that human activities have a strong regularity, and the potential predictability of user mobility is 93% [22]. The most intuitive applications include: (1) personalized recommendations and reminders [23–25]: according to the predicted next location of the user, personalized recommendations and reminders can be provided to the user. For example, the system predicts that a user's next location is a bar, the promotion of the bar and some new "friends come here" messages can be recommended to the user; (2) suspicious target tracking [26,27]: mobile user location prediction can be used to track and locate suspicious users, such as tracking users who make extreme comments on the Internet. It can effectively prevent the occurrence of events that may endanger public safety by predicting the future destination of the target users; (3) intelligent transportation [28]: a large amount of trajectory data collected by vehicle GPS devices can be used to predict traffic congestion, so that drivers can plan their driving routes and improve the scheduling efficiency of taxis.

At the same time, mobile user location prediction is also a very difficult and challenging task. First of all, due to the occlusion of high buildings in the city and the user's random switching device positioning function, it is easy to cause the loss of some important data and there is a lack of training data. Second, the activities of mobile users are uncertain, and the mobile modes exhibited by the same user may be quite different in different time periods. Furthermore, the user's trajectory data is both discrete and massive, and researchers cannot directly use the raw data for location prediction.

The existing research methods usually divide location prediction into three steps: (1) preprocessing of the original trajectory data and screening out of meaningless and abnormal location points; (2) in the remaining location points, important locations with special significance to users are constructed according to point density, visit frequency and other information, such as the user's home address, work unit and other areas inseparable from their daily life; (3) according to the training model of user's historical access information to location points, the next location of users can be predicted given the current and historical location points of users. In terms of trajectory preprocessing and important locations construction, only articles using GPS data and check-in data as experimental data contains these two steps, articles using Wi-Fi location data directly take each access point as an important location. Similarly, articles using data recorded by base stations also take each base station as an important location, while articles using data recorded by traffic bayonets take each traffic bayonet as an important location.

Ashbrook et al. directly used a K-means algorithm to cluster location points to get important locations without utilizing any preprocessing method, and a Markov model was created for each important location, with transitions to every other location. Each node in the Markov model was an important location, and a transition between two nodes represented the probability of the user traveling between those two important locations [29]. Yang et al. preprocessed the trajectory data by setting the gradient threshold to filter noise points, then used the DBSCAN algorithm to cluster the remaining location points and output the clusters as important locations [30]. Montoliu et al. used a time-based clustering algorithm to preprocess the trajectory data, defined the processed location points as the stay points, and then used the grid-based clustering algorithm to extract important locations and defined them as the stay regions [31]. Killijian et al. directly used a density clustering algorithm called DJ-Cluster to extract important locations in the original trajectory data, in view of the shortcomings of the standard Markov model, they proposed an extended mobility Markov chain called n-MMC to improve the prediction accuracy [32]. Gidófalvi et al. used a grid-based approach to preprocess the original trajectory data, then periodically extracted and managed important locations that the

user frequently visited, finally, they proposed an inhomogeneous continuous-time Markov model to predict when the user will leave his current region and where he will move next [33]. Yu et al. regarded the locations of each AP in the WLAN area where the user is located as important locations, and a step-2 Markov model was proposed to solve the state space expansion problem in high-order Markov predictor [34]. To solve the problem of the low prediction power of the standard Markov model and the problem of it being hard to determine the order of higher order Markov model in practice, Lv et al. presented a location prediction approach based on a novel adaptive multi-order Markov model, the approach first processed the raw location information of a user based on regular shape abstraction, and automatically determines the order of the model [35]. Chen et al. used the location points recorded by the traffic jam as experimental data set, they proposed the Global Markov Model and the Personal Markov Model to solve the sparse problem of personal trajectory data [21]. Rodriguez-Carrion et al. proposed a novel version of the LZ-based algorithm to perform location prediction on mobile devices [36]. Lu et al. established a Markov-based predictive model and used the trained model for location prediction of the model user [16]. Gunduz et al. first meshed the target user's activity area and then uses artificial neural networks to predict the target user's next location [4]. Reza et al. proposed a pipeline of three algorithms. First, they introduced a spatio-temporal event detection algorithm. Then, they introduced a clustering algorithm based on mobile contextual data. Their spatio-temporal clustering approach could be used as an annotation on raw sensor data [37].

Most researchers have used Markov-based models due to their good temporal trajectory data representation capability, while other researchers have attempted to deal with the location prediction problem using Hidden Markov models [38], Gaussian mixture models [39], Kalman filters [40] and so on. However, existing research methods still have some shortcomings in the three aspects of trajectory preprocessing, important locations extraction, and construction of prediction model. In particular, we have proposed an improved density clustering algorithm for extracting important locations [41]. Based on this, the paper will improve the trajectory preprocessing method and trajectory prediction model.

First, in the aspect of trajectory preprocessing, most researchers removed the trajectory outlier points by setting a time threshold or velocity threshold. Although this method is simple, it is difficult to set an appropriate threshold. Therefore, this paper first filters out the obvious noise points by setting a lower speed threshold, and then further extracts the feature points based on the structural changes of the trajectory.

Second, many researchers extracted important locations by using density clustering algorithms or grid clustering algorithms, however, the existing density clustering algorithms and grid clustering algorithms are too sensitive to the input parameters. Therefore, this paper an improved density peak clustering algorithm to extract important locations of users more accurately.

Third, existing trajectory prediction models often have the disadvantage of insufficient utilization of historical trajectory information, this paper uses a boosting algorithm to improve the predictive performance of the Markov model.

The remainder of this paper is organized as follows: the proposed approach and algorithm details are presented in the section "Proposed Framework". The data set source and experimentation results are presented in the section "Experiment". The conclusions of our work are presented in the "Conclusions" section.

2. Proposed Framework

In this section, we describe in detail how the proposed framework extracts feature points and important locations and predict the next location. Figure 1 illustrates the overall training process of the proposed framework that consists of three steps to achieve the goal of predicting the next location of mobile user.

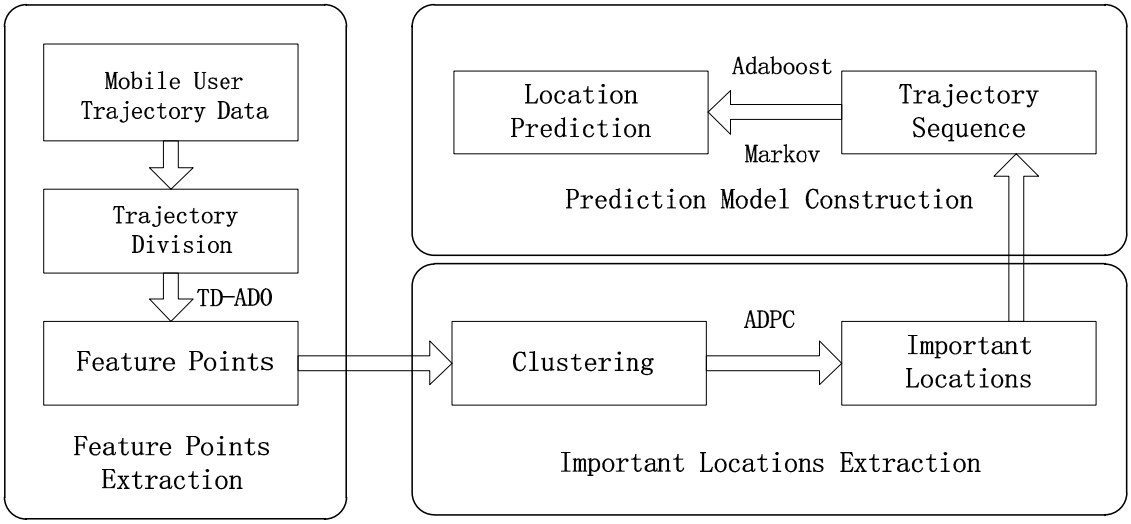

Figure 1. The framework of the proposed approach for the next location prediction of mobile user.

The training process proceeds as follows: First, a novel trajectory division method based on angle and distance offset for preprocessing trajectory data, named TD-ADO, is proposed to extract feature points and filter noise points.

Next, an improved density peak clustering algorithm named ADPC is proposed for extracting important locations from trajectory data sets filtered by TD-ADO. ADPC applies the theory of information entropy to the parameter decision of the density peak clustering algorithm, and help to determine the number of cluster centers according to the slope change trend.

Finally, the original trajectory data is converted into trajectory sequences consisting of important locations, which will be used for prediction model training. A fusion Markov with different order based on Adaboost algorithm named Adaboost-Markov model is proposed for location prediction, the weight coefficients of the training trajectory data and of each order Markov model are reasonably adjusted according to the prediction error rate, in this way, the advantages of each order model are fully leveraged and the information implicit in the prefix trajectory data is fully utilized.

2.1. Trajectory Preprocessing Algorithm

The purpose of trajectory division is to extract trajectory sampling points with large changes in user behavior, and attribute them to the data set of feature points. Each feature point serves as the end point of the previous sub-trajectory and the beginning of the next sub-trajectory, thus we can divide a complete trajectory into several continuous sub-trajectories by this method.

In order to describe our algorithm better, this paper uses the definition of the trajectory in [42]. Let TD be a trajectory dataset, and the dataset consists of n trajectories, namely $TD = \{TJ_1, TJ_2, \ldots, TJ_n\}$. The trajectory ($TJ$) is a sequence composed of a number of trajectory sampling points in chronological order, namely $TJ_i = \{P_1, P_2, \ldots, P_m\}(1 \leq i \leq n)$. Among them, $P_j(1 \leq j \leq m)$ is a trajectory sampling point composed of latitude and longitude and sampling time, namely, $<Lat_j, Lng_j, T_j>$.

Definition 1. *(sub-trajectory). The sub-trajectory of the trajectory TJ_i is expressed as $STJ_i = \{P_{s1}, P_{s2}, \ldots, P_{sk}\}$ $(1 \leq s1 < s2 < \ldots < sk \leq m)$.*

The sub-trajectories mentioned in this paper are all continuous, that is, from P_{s1} to P_{sk} are continuous trajectory sampling points.

Definition 2. *(angel offset). As shown in Figure 2, taking the trajectory TJ_i as an example, P_1 is the initial trajectory point and $\boldsymbol{P_1P_2}$ is the initial movement behavior. The angle offset refers to the angle θ_i between the movement behavior $\boldsymbol{P_iP_{i+1}}$ and $\boldsymbol{P_{i+1}P_{i+2}}$ $(1 \leq i \leq m-2)$ when they are connected at the end. For example, the*

angle θ_1 between the movement behavior $\boldsymbol{P_1P_2}$ and $\boldsymbol{P_2P_3}$ when they are connected at the end and the angle θ_2 between the movement behavior $\boldsymbol{P_2P_3}$ and $\boldsymbol{P_3P_4}$ when they are connected at the end.

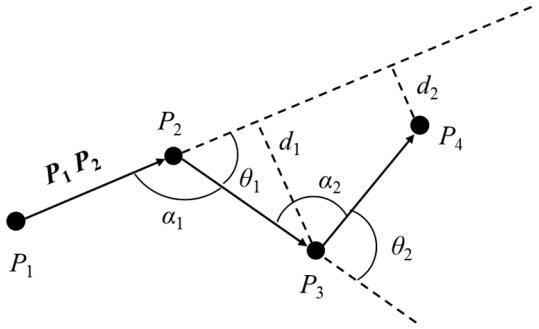

Figure 2. The angle and distance offset.

In this paper, the clockwise rotation angle is set as a positive value and the counterclockwise rotation angle is set as a negative value, i.e., θ_1 is a positive value and θ_2 is a negative value. In order to calculate the magnitude of the angle offset, the distance between two points must first be calculated from the latitude and longitude of the trajectory sampling point. The specific formula is defined in Equation (1):

$$\left.\begin{array}{c} A = \sin^2(\frac{Lat_j - Lat_i}{2}) \\ B = \cos(Lng_i)\cos(Lng_j)\sin^2(\frac{Lng_j - Lng_i}{2}) \\ d(P_i, P_j) = 2R\arcsin\sqrt{A+B} \end{array}\right\} \tag{1}$$

$d(P_i, P_j)$ is the distance between the trajectory sampling points P_i and P_j, and R is the radius of the earth. The trajectory angle offset function is defined in Equations (2) and (3):

$$\alpha_i = \arccos\frac{(|\boldsymbol{P_iP_{i+1}}|^2 + |\boldsymbol{P_{i+1}P_{i+2}}|^2 - |\boldsymbol{P_iP_{i+2}}|^2)}{2|\boldsymbol{P_iP_{i+1}}|^2|\boldsymbol{P_{i+1}P_{i+2}}|^2} \tag{2}$$

$$\theta_i = \begin{cases} \alpha - \pi, \boldsymbol{P_iP_{i+1}} \times \boldsymbol{P_{i+1}P_{i+2}} > 0 \\ \pi - \alpha, \boldsymbol{P_iP_{i+1}} \times \boldsymbol{P_{i+1}P_{i+2}} \geq 0 \end{cases} \tag{3}$$

Definition 3. *(distance offset). The distance offset is the vertical distance from the trajectory sampling points to the line where the initial movement is located.*

As shown in Figure 2, the vertical distances d_1 and d_2 from the sampling points P_3 and P_4 to the extension line of initial movement are the distance offsets. The trajectory distance offset function is defined in Equation (4):

$$d_i = |\boldsymbol{P_2P_{i+2}}| \sin \angle P_1P_2P_{i+2} \tag{4}$$

For the trajectory TJ_i, firstly, from the sampling point P_3, calculate the angle offset θ_1 and the distance offset d_1, if the absolute value of θ_1 does not exceed the angle offset threshold θ_{th}, and d_1 does not exceed the distance offset threshold d_{th}, then continue to calculate the angle and distance offset of the sampling point P_4, and so on.

If the angle offset or distance offset of the sampling point P_i exceeds the corresponding threshold, P_i will be assigned to the data set of feature points, and define $\boldsymbol{P_iP_{i+1}}$ as the new initial movement behavior, continue to perform the above steps from the sampling point P_{i+2}. Finally, the trajectory TJ_i will be divided into several continuous sub-trajectories. The time complexity of the trajectory dividing algorithm in this paper is $O(n)$, where n is the number of trajectory sampling points.

The trajectory division algorithm using the angle and distance offset can effectively avoid the following two conditions: (1) If the trajectory is divided only by setting the angle offset threshold value, the feature points in Figure 3 cannot be extracted. Each angle offset in Figure 3 is small, but the shape of the P_1~P_6 trajectory segment is close to a semicircle due to the stacking effect of multiple clockwise rotations. Obviously, it is obviously unreasonable not to classify the six sampling points P_1~P_6 as feature points, which will lead to the omission of important locations. (2) If the trajectory is divided only by setting the distance offset threshold, missed selection of important locations may also occur. In Figure 4, since the trajectory has a regular clockwise and anti-clockwise rotation, the sampling points P_3~P_5 are all near the straight line where the initial movement behavior is located. However, the angle offsets between connected movements are already close to 90°, and it is obviously not appropriate to classify all sampling points of this trajectory segment as feature points.

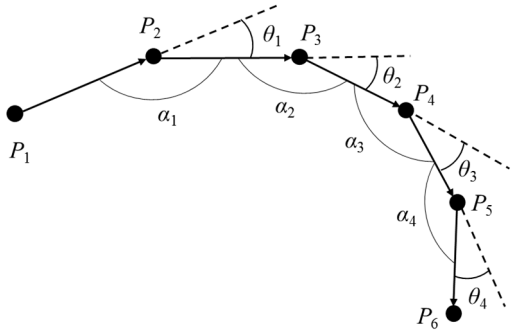

Figure 3. The continuous clockwise rotation trajectory.

The complete algorithm pseudo code is given below:

Algorithm: TD-ADO

Input: $TJ_i = \{P_1, P_2, \ldots, P_m\}$, angle offset threshold $\theta_{th,}$ distance offset threshold d_{th};
Output: Feature Points FP;
$P_1 \rightarrow$FP;
$i = 1$;
repeat
  initial movement behavior = $\boldsymbol{P_iP_{i+1}}$;
  for $j = 1$ to $m-2$ **do**
    if $\theta_j > \theta_{th}$ || $d_j > d_{th}$ **then**
      $P_{j+2} \rightarrow$FP;
      $i = j+2$;
      break;
    else
      if $j = m-2$ **then**
        $P_{j+2} \rightarrow$FP;
      end if
    end if
  end for
until $P_m \rightarrow$FP.
return FP.

The algorithm adds the initial sampling point and the final sampling point of each trajectory to the feature points, because the starting point and ending point of the trajectory usually have special meanings. For example, when a user turns on the mobile phone or turns off the airplane mode every day when he goes out, the starting point of the trajectory is often the location of the user's residence. Similarly, when the user turns off the mobile phone or turns on the flight mode before going to bed at

night, the trajectory record of the day is terminated. In this way, the trajectories in the trajectory set are divided according to the above algorithm, and each trajectory is divided into several sub-trajectories, and the extracted feature points form a data set.

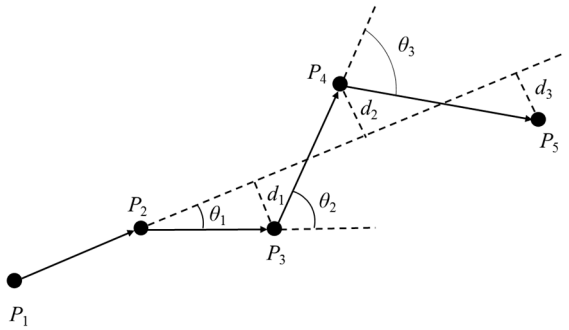

Figure 4. The regular clockwise-anticlockwise rotation trajectory.

2.2. The Important Locations Extraction Algorithm

Researchers often cluster trajectory sampling points based on their density to extract important locations. However, commonly used density clustering algorithms usually have the shortcomings that the clustering result is too sensitive to the input parameters. How to determine the appropriate input parameters has become an urgent problem to be solved. In this paper, we propose an improved density peak clustering algorithm called ADPC to cluster the feature points and extract the important locations. The density peak clustering algorithm was a new density clustering algorithm proposed by Rodriguez et al. [43]. Compared with the classical density clustering algorithms, the algorithm is simple in principle, requires only one input parameter, and does not require iteration, and it can cluster data of various shapes. However, it is necessary to select the clustering center manually through the decision graph, which not only increases the redundancy of the algorithm, but also causes subjective troubles. The improved density peak clustering algorithm ADPC incorporates the self-adaptive improvement of the cut-off distance and the clustering center, so that it can automatically determine the best input parameters and clustering center according to the distribution of sampling points.

The only input parameter of the density peak clustering algorithm is the cut-off distance d_c, this paper adopts the percentage to represent the cut-off distance d_c, arrange the distances between all two data points in ascending order, and take the distance value corresponding to a certain percentage in the distance set.

In this model, for a data set $S = \{x_i\}_{=1}^{NI}$ to be clustered, there are two quantities that need to be calculated. One is the local density of each data point ρ_i, and the other is the minimum distance between each data point and the higher density point δ_i.

The formula for the local density is given by Equation (5):

$$\rho_i = \sum_j e^{-\left(\frac{d_{ij}}{d_c}\right)^2} \tag{5}$$

δ_i is the minimum distance from data point i to data point j with a higher local density, defined in Equation (6):

$$\delta_i = \min_{\rho_i < \rho_j} (d_{ij}) \tag{6}$$

For the data point k with a global maximum density, δ_i is defined as the maximum distance between the data point and the remaining data points.

At this point, for each data point in the data set S, its local density ρ_i and the minimum distance δ_i can be calculated. The density peak clustering algorithm regards the data points that have large ρ_i and δ_i at the same time as the cluster centers, and the remaining data points are allocated to clusters that

are closest to the nearest cluster centers. As shown in Figure 5, the horizontal axis represents the local density of each point, the vertical axis represents the minimum distance of each point, and the color points are the points selected as the cluster centers.

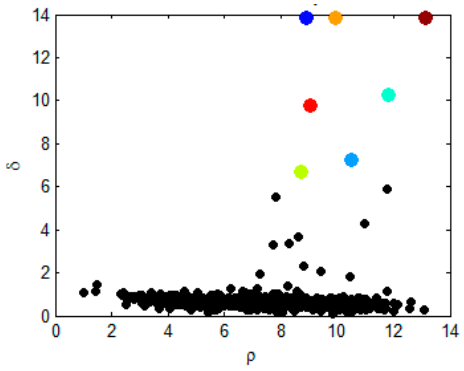

Figure 5. Decision graph.

The density peak clustering algorithm mainly has two shortcomings. On the one hand, there is no reliable reference for the selection of its input parameter d_c. The author of this algorithm, Rodriguez, gave the recommendation to select the top 1% to 2% of the cut-off distances after the distance between all data points was sorted in ascending order. However, for a data set with a disproportionately different size and distribution, this general method of parameter selection will often lead to poor results. On the other hand, the density peak clustering algorithm relies on the distribution of points on the decision graph to select the cluster center, which is too subjective.

The ADPC algorithm is an improved algorithm in this paper. Its essence is to achieve the data set clustering through the automatic selection of the cut-off distance and the clustering centers.

Rodriguez gave an idea for reference, that is, consider the product of ρ_i and δ_i:

$$\gamma_i = \rho_i \delta_i \tag{7}$$

First, ρ_i and δ_i are normalized, then γ_i is obtained for each data point, and arrange them in descending order. The larger the γ_i, the more likely it is to be a cluster center.

To visually show the relationship between d_c and γ_i, Equation (7) can also be expressed as:

$$\gamma_i = \delta_i \sum_j e^{-\left(\frac{d_{ij}}{d_c}\right)^2} \tag{8}$$

Finding the minimum value of the information entropy based on γ_i to achieve the adaptive cut-off distance and obtaining the maximum value of the slope change trend to achieve automatic selection of cluster centers is the focus of this paper.

This paper proposes a cut-off distance adaptive method based on the minimization of information entropy. In information theory, entropy is used as a measure of system uncertainty. The greater the entropy, the stronger the uncertainty of the system. The formula for calculating information entropy is as follows:

$$H = -\sum_{i=1}^{n} p_i \log p_i \tag{9}$$

Similarly, apply the definition of entropy to this algorithm, replace the probability value in the information entropy formula with γ, that is, the larger the γ, the more likely it is that the corresponding data point is the cluster center. If the points in the dataset have the same γ, the uncertainty of the system is the greatest, and the difficulty of determining the cluster center is also highest at this time.

On the other hand, if the entropy value is the minimum, the distribution difference of γ is the most obvious, and it is the easiest to determine the cluster center.

Combining Equations (8) and (9), this paper gives the relationship between cut-off distance d_c and entropy H with Equation (10):

$$H = -\sum_{i=1}^{n}\left(\delta_i \sum_j e^{-\left(\frac{d_{ij}}{d_c}\right)^2}\right)\log\left(\delta_i \sum_j e^{-\left(\frac{d_{ij}}{d_c}\right)^2}\right) \tag{10}$$

Let the cut-off distance d_c gradually increase from zero, find d_c that makes the entropy value H the minimum, and use it as the optimal cut-off distance for the next clustering, so that the cut-off distance is self-adaptive.

Take the data set of Ref. [43] as an example and get the result shown in Figure 6. The horizontal axis indicates the cut-off distance d_c, and the vertical axis indicates the entropy value H. It can be seen from the figure that when the entropy value H reaches the minimum, the cut-off distance d_c is 1.5%.

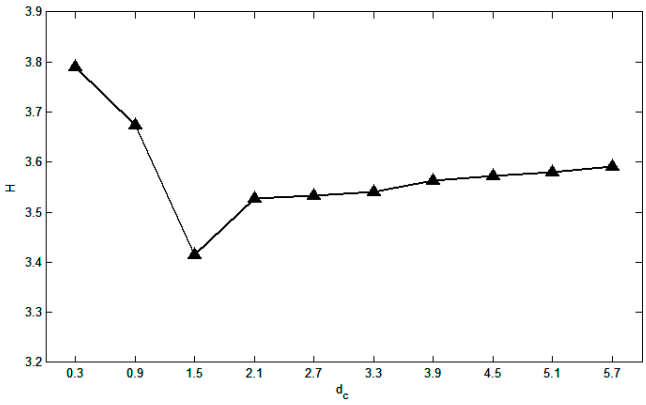

Figure 6. The trend of information entropy under different cut-off distances.

Taking the 2000 random sample points given in Ref. [43] as an example, we select the first 1.5% d_c, calculate the first 40 γ_i, and then use MATLAB to generate a decision map based on the results. The results are shown in Figure 7. The first five points are the actual cluster centers, and there is a steep downward trend from the cluster center to the first non-cluster center. Therefore, this paper uses the trend of slope change to automatically select the cluster center and define this trend as $tend_i$:

$$tend_i = (i-1)\frac{\gamma_{i-1} - \gamma_i}{\gamma_i - \gamma_{i+1}} \tag{11}$$

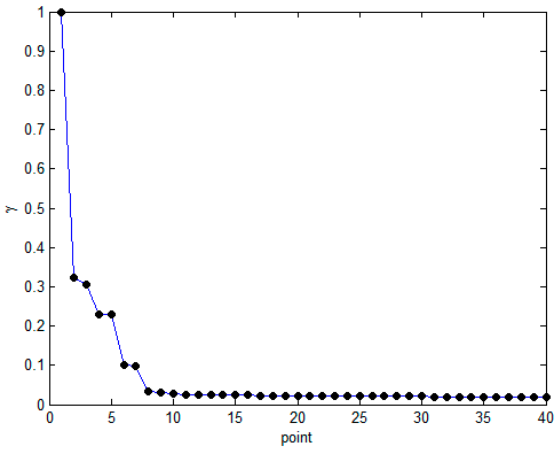

Figure 7. γ_i sort graph.

Considering that there may be a tendency similar to the slope change from point 1 to point 2 in Figure 7, the weight $i-1$ is introduced for $tend_i$. However, as the number of data points increases, the weight $i-1$ may cause $tend_i$ to increase rapidly, resulting in erroneous judgments.

In this regard, the paper uses the average value of all γ_i in the ranking graph as the threshold θ to solve the problem that the weight will cause the rapid increase of $tend_i$. First, the data points whose γ_i is larger than the threshold θ are selected from 40 data points, and *tend* of the selected data point is calculated. As shown in Figure 8, after the screening, there are still seven data points remaining. It can be seen that the *tend* at the sixth point is the largest, so the first five points are selected as potential cluster centers. The density peak clustering algorithm can well classify the data points with very small ρ_i and large δ_i into outliers. However, relying solely on the product of ρ_i and δ_i to select cluster centers may result in false selection of cluster centers: For the point with large ρ_i and small δ_i, that is, two high density points within the same cluster are likely to be mistakenly selected as clustering centers and the cluster is divided. Therefore, the optimal d_c obtained by Equation (10) is set as the distance threshold of the potential cluster center, and the algorithm is prevented from erroneously judging the point with large ρ_i and small δ_i as the cluster center.

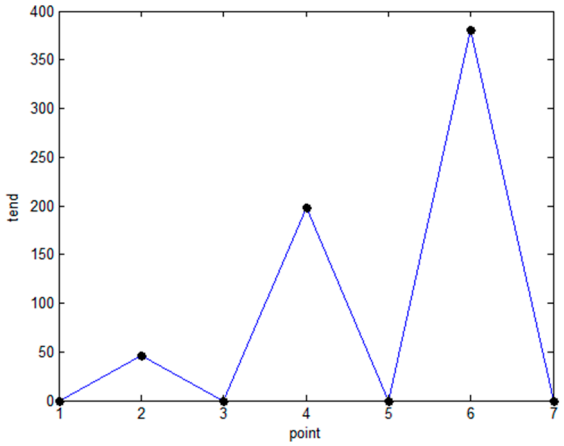

Figure 8. Slope change trend graph.

The concrete steps of ADPC are as follows:

Step 1. Calculate the product γ_i of normalized ρ_i and δ_i for each data point.

Step 2. Find d_c that minimizes the entropy.

Step 3. Sort γ_i of each data point in descending order, and select the appropriate number of points according to the total number of points in the data set and generate sorting charts in descending order.

Step 4. Calculate the average value θ of γ in the γ_i sort diagram and use θ as the threshold to filter out data points with γ_i greater than θ.

Step 5. According to the slope change trend formula, the *tend* of the filtered data points is obtained.

Step 6. The $i-1$ data points before the data point i with the largest tend are regarded as potential cluster centers.

Step 7. First, use the first data point as the actual cluster center to determine whether the distance between the second data point and its distance is less than the distance threshold d_c. If it is less than, then the second data point is treated as a non-clustering center; otherwise, use the second data point as the actual clustering center.

Step 8. By analogy, determine whether the distance between the kth potential cluster center and all actual cluster centers are greater than the distance threshold, and then treat it as the actual cluster center or non-cluster center according to the judgment result.

Step 9. Finally, according to all the selected actual cluster centers, cluster the remaining data points.

Figure 9 illustrates the process of clustering important points from points of feature, there are five simplified trajectories consists of feature points, after clustering the points of feature extracted by the TD-ADO algorithm, four clusters C_1 to C_4 are formed, and the feature points that have not been red circled are judged as noise points. In this way, the important locations consist of all sampling points in the red circle.

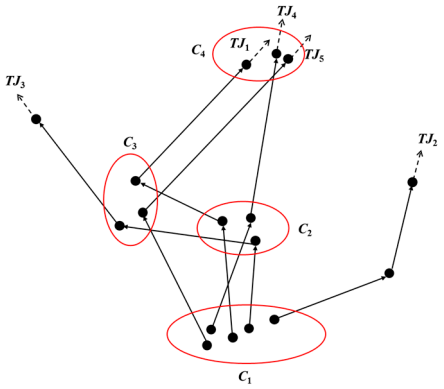

Figure 9. The diagram of important location clusters.

2.3. Locations Prediction Model

In order to facilitate the understanding of the following, this paper gives the following definitions.

Definition 4. *(Markov chain). Suppose there is a random process* $\{X_n, n \in T\}$*, and there are finite states* $i_0, i_1, \ldots, i_n \in I$*, if* $P\{X_{n+1} = i_{n+1} \mid X_0 = i_0, X_1 = i_1, \ldots, X_n = i_n\} = P\{X_{n+1} = i_{n+1} \mid X_n = i_n\}$*, then* $\{X_n, n \in T\}$ *is a Markov chain, that is, the next state of the object is only related to the state at this time, also called the first-order Markov chain. If the next state of the object is not only related to the state at this time, but also related to the first k-1 states, it is called a k-order Markov chain.*

Definition 5. *(Trajectory sequence): The user's trajectory sequence is a simplified representation of its original trajectory after trajectory division and density clustering, and is composed of extracted important locations. For example, the original trajectory of the user is expressed as* $TJ_i = \{P_1, P_2, \ldots, P_m\}(1 \leq i \leq n)$*, and the processed simplified trajectory is expressed as* $TS_i = \{C_{k1}, C_{k2}, \ldots, C_{kj}\}(1 \leq i \leq n)$*.*

Definition 6. *(Prefix trajectory sequence): Suppose that given a trajectory sequence* $TS_i = \{C_{k1}, C_{k2}, \ldots, C_{kj}\}(1 \leq i \leq n)$*, and there is* $1 < l < j$*, to predict which important location* C_{kl} *is, then* $\{C_{k1}, C_{k2}, \ldots, C_{k(l-1)}\}$ *is its corresponding prefix trajectory sequence, whose elements are called prefix trajectory elements.*

Definition 7. *(Trajectory Markov chain): The state space of the trajectory Markov chain is composed of the user's important locations* $C_1, C_2, \ldots, C_n$*.*

If the user's next important location depends on the current location and the past *k*-1 locations, the Markov chain is called a *k*-order trajectory Markov chain, namely:

$$C_p = \arg\max_{C_{n+1}} \{p(C_{n+1} | C_{n-k+1}, C_{n-k+2}, \ldots, C_n)\} \tag{12}$$

where C_p represents the prediction result of the user's next important location, and C_n represents the current important location of the user.

The low-order Markov model has the advantages of small time cost and fast calculation speed, but its prediction accuracy is not high enough. Although the higher-order Markov model has higher prediction accuracy, the spatial state expansion problem makes it difficult to apply to the large-scale trajectory data set, and some researchers [35] have pointed out that there is a limit to the prediction

performance growth caused by the increase of the order. If the user's historical trajectory data is difficult to match the Markov model with high-order, the prediction performance of the model will be reduced due to the sparse rate. As shown in Figure 10, taking a historical trajectory pattern tree of a user as an example, it is assumed that a third-order Markov model is adopted, and the prefix trajectory sequence is $C_2 \rightarrow C_4 \rightarrow C_5$, and the next important location of the user is predicted based on the mobile pattern tree. However, there is no historical trajectory with $C_2 \rightarrow C_4 \rightarrow C_5$ as the prefix sequence in this pattern tree, and the third-order Markov model cannot make prediction. Moreover, when the order of the Markov model is high, even if a prefix trajectory sequence as $C_2 \rightarrow C_4 \rightarrow C_5$ is found in the user historical trajectory sequence pattern tree, there is a high possibility that the number of matches is too small and the sparse rate is still high, which may lead to poor prediction result.

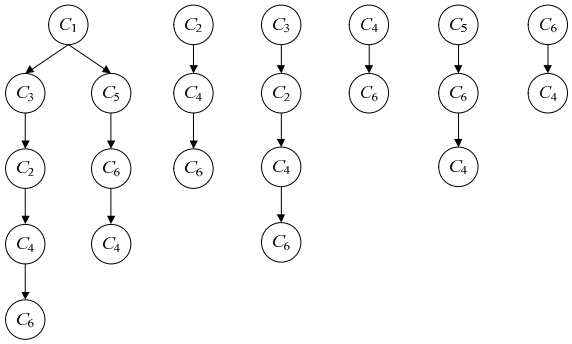

Figure 10. User historical trajectory sequence pattern tree.

In view of the above problems, the first-order Markov model only utilizes the current interest region of the user, and the utilization rate of the historical trajectory data is not high; while the high-order Markov model has problems of difficulty in matching and high sparse rate, the historical trajectory data cannot be fully utilized as the same. Therefore, combining low-order models with high-order models is a feasible method. Then, reasonably determining the influence coefficient of each model with different order is an important prerequisite for ensuring good prediction performance. From a macro perspective, high-order Markov model usually has higher prediction accuracy, and should have greater "discourse power" in the fusion model. In order to determine the specific influence coefficient of each model with different order, a mobile user location prediction method based on Adaboost-Markov model is proposed in this paper. As a representative lifting method, Adaboost algorithm can combine a plurality of weak classifiers by changing the probability distribution of training data and the weight coefficient of weak classifier to generate a strong classifier [44]. In this paper, the fusion model order k is adaptively determined, and the first-order to kth-order Markov model are used as k weak predictors. The probability distribution of user trajectory data and the weight coefficients of each Markov model are changed by Adaboost algorithm, and finally a multi-order fusion Markov is generated, which is used to predict user important location.

The order k is determined by the maximum matching step length of the user prefix trajectory sequence and the historical trajectory sequence. Taking Figure 11 as an example, the user's original trajectory data is preprocessed to establish a historical trajectory sequence database. The user determines the number of elements of the prefix trajectory sequence, and inputs the prefix trajectory sequence to be predicted. If the same sequence as the prefix trajectory sequence can be found in the historical trajectory sequence, the matching is successful, and the number of prefix trajectory elements is set to the model order k; otherwise, if the matching fails, the order is automatically reduced, remove the element from the prefix trajectory sequence that is furthest from the current state, and continue to seek a match in the historical trajectory sequence database, repeat the above steps until the match is successful.

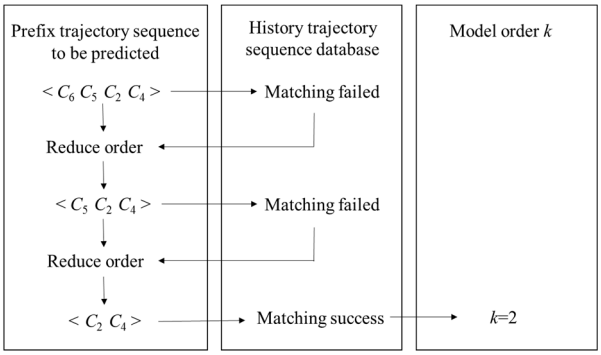

Figure 11. Example of adaptive determination of order *k*.

After the model order k is determined, the first-order to kth-order Markov model is merged, that is, the Adaboost-Markov model is composed of k m-order ($m = 1, 2, \ldots, k$) Markov models, The weight coefficient α_m of each model is negatively correlated with the prediction error of itself, the larger the prediction error, the smaller the weight coefficient of the corresponding model, and vice versa. For training trajectory sequences with N elements, the prediction error e_m of the m-order ($m = 1, 2, \ldots, k$) Markov model is calculated as follows:

$$e_m = \sum_{i=1}^{N} w_{mi} I_m^i \tag{13}$$

where w_{mi} is the weight of the training samples corresponding to the m-order ($m = 1, 2, \ldots, k$) Markov model, which is initially set to $1/N$, and $\sum_{i=1}^{N} w_{mi} = 1$. I_m^i is the predictor of the m-order Markov model for the training sample i, and its formula is as shown in Equation (14):

$$I_m^i = \begin{cases} 1, error\ prediction \\ 0, correct\ prediction \end{cases} \tag{14}$$

After calculating the prediction error e_m, the model weight coefficient αm can be calculated as follows:

$$\alpha_m = \frac{1}{2} \log \frac{1 - e_m}{e_m} \tag{15}$$

Then, the weight coefficient of the m-th order model obtained by the above method is used to update the weight of the training sample, and the weight $w_{m+1,i}$ of the training sample corresponding to the m+1 order ($m = 1, 2, \ldots, k-1$) Markov model is calculated, improve the sample weight of mispredicted in the m-order Markov model and reduce the weight of sample with the correct prediction result, so as to make the mispredicted samples get more attention in the next round of learning. The calculation method is as shown in Equations (16) and (17):

$$w_{m+1,i} = \frac{w_{mi}}{Z_m} \exp(\alpha_m I_m^i) \tag{16}$$

$$Z_m = \sum_{i=1}^{N} w_{mi} \exp(\alpha_m I_m^i) \tag{17}$$

where Z_m is the normalization factor, which makes $w_{m+1,i}$ a probability distribution. Then, the prediction error e_{m+1} and the model weight coefficient α_{m+1} are calculated according to the new training sample weight and the m+1 order Markov model, and so on, until the prediction error

e_k of the k-order Markov model and the model weight coefficient α_k are calculated. Finally, normalize the weight coefficients of each order model, its formula is as follows:

$$\beta_m = \frac{\alpha_m}{\sum_{m=1}^{k} \alpha_m} \tag{18}$$

The Adaboost-Markov model can be expressed as follows:

$$G(x) = \sum_{m=1}^{k} \beta_m G_m(x) \tag{19}$$

Among them, $G_m(x)$ represents the m-order Markov model.

3. Experiments

In this section, in order to verify the clustering performance of the ADPC algorithm, we first compare the accuracy, normalized mutual information (NMI) and F-measure of DBSCAN algorithm, density peak clustering algorithm and ADPC algorithm on the UCI dataset [45]. The data set information is shown in Table 1. Then the real mobile user trajectory dataset Geolife [9] is used to detect the prediction performance of our method, this data set information is shown in Table 2. The experimental environment is the Windows 10 64-bit operating system, Intel Core i7-6700HQ @ 2.60 GHz CPU, 8 G memory, using MATLAB2014a for simulation experiments. On the parameter selection of the trajectory preprocessing algorithm, $\theta_{th} = 15^{\circ}$, $d_{th} = 3$ m.

Table 1. UCI dataset used in the experiment.

Data Set	Category	Sample	Dimension
Iris	3	150	4
Aggregation	7	788	2
Waveform	3	500	21
Wine	3	178	13

Table 2. Geolife dataset used in the experiment.

Data Collection Time	Number of Users	Number of Trajectories
2007.4~2012.8	182	17624

3.1. Clustering Performance

From Figures 12–14, it can be seen that the trend of ADPC>DPC>>DBSCAN on the accuracy rate, F-measure, and normalized mutual information of the algorithm as a whole indicates to some extent the ADPC algorithm's superior performance on clustering.

However, the performance of the ADPC algorithm on the high-dimensional datasets such as Waveform and Wine is still not ideal. For the two-dimensional Aggregation dataset, the accuracy, F-measure value, and normalized mutual information of the ADPC algorithm have reached more than 90%, and the performance is good. Therefore, the ADPC algorithm can be used for two-dimensional location points.

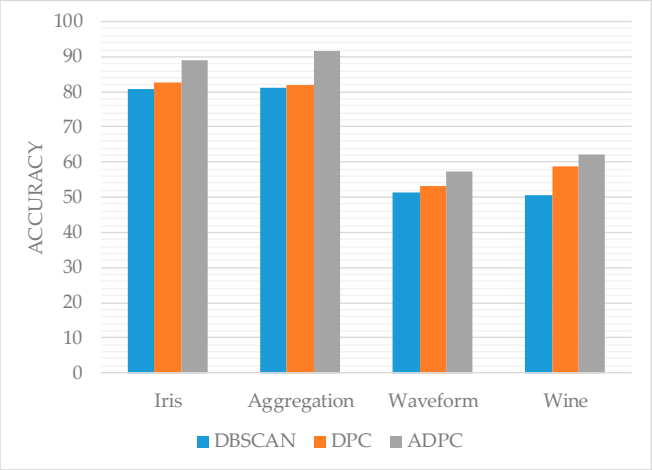

Figure 12. The accuracy of each algorithm.

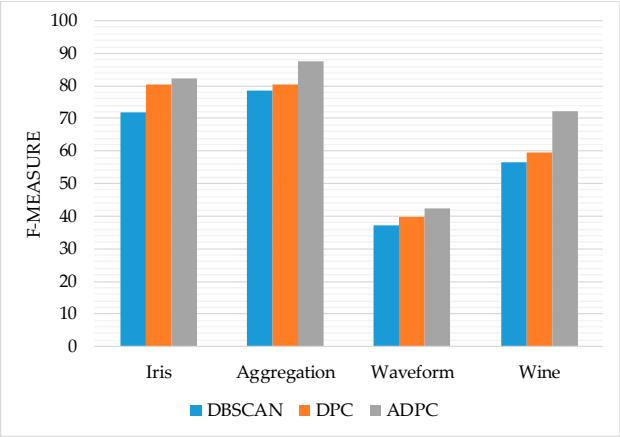

Figure 13. The F-measure of each algorithm.

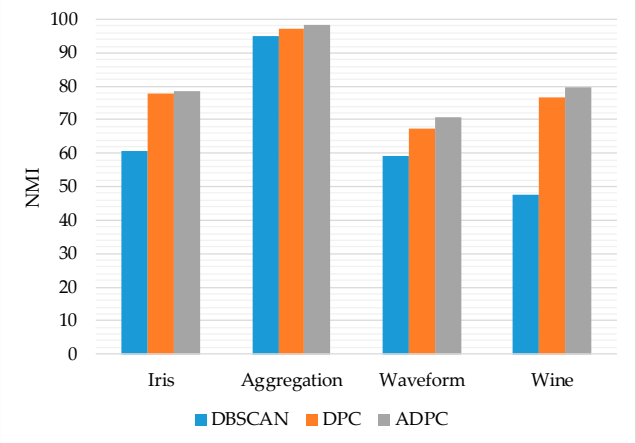

Figure 14. The normalized mutual information of each algorithm.

3.2. Prediction Performance Analysis of Various Model

In this section, 90% of the Geolife trajectory data set are randomly selected for training models, and the remaining 10% of the trajectory data is used to detect prediction performance. Firstly, in order to verify the rationality of weight coefficient allocation using Adaboost algorithm, the prediction accuracy of the Adaboost-Markov model, the multi-order fusion Markov model with equal weight coefficients and the common Markov model under different prefix elements are compared.

As shown in Figure 15, the horizontal axis represents the number of prefix elements, and the vertical axis represents the prediction accuracy. For the common Markov model, the number of prefix elements is equivalent to the model order, the Adaboost-Markov model and the multi-order fusion Markov model with equal weight coefficients adopt the adaptive method to determine the model order k. Therefore, with the increase of the number of prefix elements, the prediction accuracy of the common Markov model firstly shows an increasing trend, reaching a maximum at the second-order, and then gradually decreasing due to the increase of the matching sparse rate. The Adaboost-Markov model effectively overcomes this problem: he prediction accuracy first increases with the number of prefix elements, when the number of prefix elements is 3, the maximum is basically reached, then the prediction accuracy is stable near the maximum.

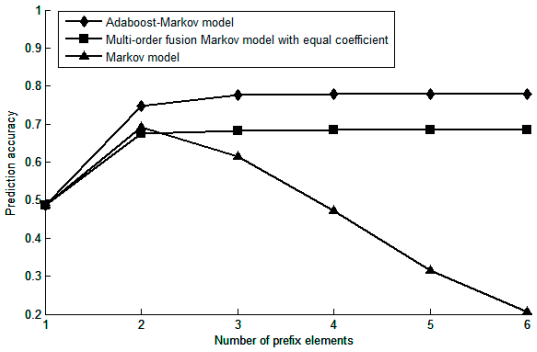

Figure 15. The relationship between the number of prefix elements and the prediction accuracy.

Moreover, it can be seen from the figure that the weight coefficient of the model determined by the Adaboost algorithm is reasonable and effective, the prediction accuracy of the Adaboost-Markov model is higher than the multi-order fusion Markov model with equal weight coefficients regardless of the number of prefix elements in the given sequence to be predicted.

In order to verify the universality of the model, this paper randomly selects the trajectory data of eight users from the Geolife data set, and compares the first to third-order Markov model and the Adaboost-Markov model under three prefix elements on different user trajectory data sets. As shown in Figure 16, the prediction accuracy of the Adaboost-Markov model is better than that of the first to third-order Markov model on different trajectory data sets. It can be seen that the proposed model does not depend on a specific data set and has good universality.

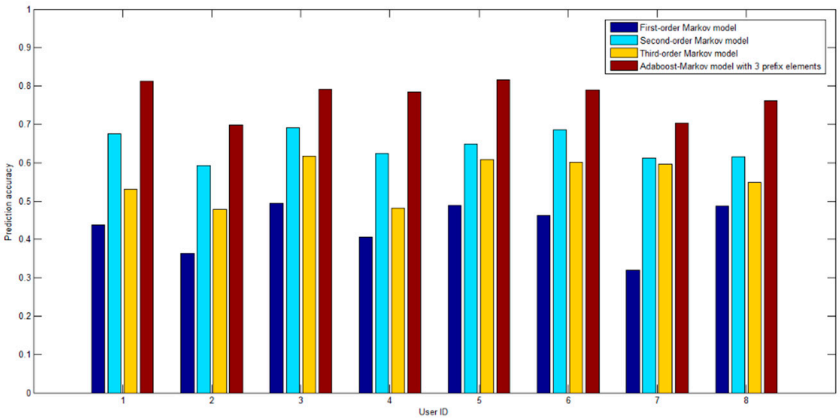

Figure 16. Prediction accuracy comparison under different trajectory data set.

In the location prediction of mobile users, multi-step prediction ability is also one of the keys to reflect the prediction performance of the model, that is, predicting the region of interest where the user is after n steps. In order to verify the multi-step prediction ability of the model, the prediction accuracy

of the Adaboost-Markov model with 3 prefix elements and the first to third -order Markov model are compared under different prediction steps, the experimental results are shown in Figure 17.

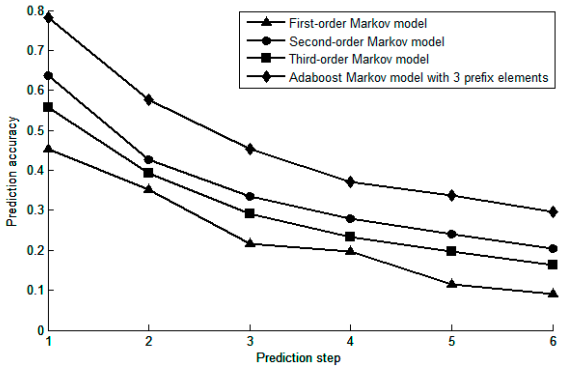

Figure 17. Relationship between prediction steps and prediction accuracy.

It can be seen from the figure that with the increase of the prediction step size, the prediction accuracy of each model decreases to some extent, but the prediction accuracy of the Adaboost-Markov model is better than the other three models. This is because the common Markov model directly uses a fixed-length step transition matrix when performing multi-step prediction, and has poor adaptability to the trajectory data. In this experiment, the Adaboost-Markov model selects three prefix elements, and its order k is flexibly changed in the first to third-order according to the specific trajectory data, which not only ensures that the time cost is not too large, but also makes the prediction accuracy is close to the maximum, so the prediction accuracy of the Adaboost-Markov model is relatively high.

4. Conclusions

This paper studies a location prediction method based on the Markov model for mobile user trajectory data. The method of combining trajectory division and density clustering is used to process trajectory data, the original trajectory data is transformed into a trajectory sequence composed of important locations, and the weight coefficient is determined for the multi-order fusion Markov model using the Adaboost algorithm. Experiments on the real user trajectory data set Geolife prove that the proposed method overcomes the low prediction accuracy of low-order Markov model and the high sparse rate of high-order Markov model to some extent, and makes full use of the user's prefix trajectory information. The next step will take into account factors such as weather, time (working days and rest days) and user-related social data, in order to further improve the prediction accuracy of the model.

Author Contributions: H.W. conceived of the idea, developed the proposed methods and wrote the paper. H.W. and Z.Y. performed the experiments. Y.S. advised the research and revised the paper.

Funding: This work was supported by the National Natural Science Foundation of China (Project No. 61273302).

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Guo, C.; Liu, J.N.; Fang, Y.; Luo, M.; Cui, J.S. Value Extraction and Collaborative Mining Methods for Location Big Data. *J. Softw.* **2014**, *25*, 713–730.
2. Widhalm, P.; Nitsche, P.; Brandie, N. Transport mode detection with realistic Smartphone sensor data. In Proceedings of the International Conference on Pattern Recognition, Tsukuba, Japan, 11–15 November 2015; IEEE: Piscataway, NJ, USA, 2012; pp. 573–576.
3. Shih, D.H.; Shih, M.H.; Yen, D.C.; Hsu, J.H. Personal mobility pattern mining and anomaly detection in the GPS era. *Am. Cartogr.* **2016**, *43*, 55–67. [CrossRef]

4. Gunduz, S.; Yavanoglu, U.; Sagiroglu, S. Predicting Next Location of Twitter Users for Surveillance. In Proceedings of the 2013 13th International Conference on Machine Learning and Applications, Miami, FL, USA, 4–7 December 2013; IEEE Computer Society: Piscataway, NJ, USA, 2013; pp. 267–273.
5. Bogomolov, A.; Lepri, B.; Staiano, J.; Oliver, N.; Pianesi, F.; Pentland, A. Once Upon a Crime: Towards Crime Prediction from Demographics and Mobile Data. In Proceedings of the 16th International Conference on Multimodal Interaction, Istanbul, Turkey, 12–16 November 2014; ACM: New York, NY, USA, 2014; pp. 427–434.
6. Yuan, J.; Zheng, Y.; Zhang, L.; Xie, X.; Sun, G. Where to find my next passenger. In Proceedings of the 13th International Conference on Ubiquitous Computing, Beijing, China, 17–21 September 2011.
7. Pang, L.X.; Chawla, S.; Liu, W.; Zheng, Y. On mining anomalous patterns in road traffic streams. In Proceedings of the 7th International Conference on Advanced Data Mining and Applications, Beijing, China, 17–19 December 2011.
8. Zheng, Y.; Liu, Y.; Yuan, J.; Xie, X. Urban Computing with Taxicabs. In Proceedings of the 13th International Conference on Ubiquitous Computing, Beijing, China, 17–21 September 2011; pp. 89–98.
9. Zheng, Y.; Xie, X.; Ma, W.Y. GeoLife: A collaborative social networking service among user, location and trajectory. *IEEE Data Eng. Bull.* **2010**, *33*, 32–39.
10. Kotz, D.; Henderson, T. CRAWDAD: A Community Resource for Archiving Wireless Data at Dartmouth. *IEEE Pervasive Comput.* **2005**, *4*, 12–14. [CrossRef]
11. Scellato, S.; Musolesi, M.; Mascolo, C.; Latora, V.; Campbell, A.T. NextPlace: A Spatio-temporal Prediction Framework for Pervasive Systems. In Proceedings of the International Conference on Pervasive Computing, Seattle, WA, USA, 21–25 March 2011; Springer: Berlin/Heidelberg, Germany, 2011; pp. 152–159.
12. Li, H.; Shou, G.; Hu, Y.; Guo, Z. Mobile Edge Computing: Progress and Challenges. In Proceedings of the 2016 4th IEEE International Conference on Mobile Cloud Computing, Services, and Engineering (Mobile Cloud), Oxford, UK, 29 March–1 April 2016.
13. Wang, B.; Hu, Y.; Shou, G.; Guo, Z. Trajectory Prediction in Campus Based on Markov Chains. In Proceedings of the International Conference on Big Data Computing & Communications, Shenyang, China, 29–31 July 2016; Springer: Cham, Switzerland, 2016.
14. Lee, S.; Lim, J.; Park, J.; Kim, K. Next Place Prediction Based on Spatiotemporal Pattern Mining of Mobile Device Logs. *Sensors* **2016**, *16*, 145. [CrossRef]
15. Eagle, N.; Pentland, A. Reality mining: Sensing complex social systems. *Pers. Ubiquitous Comput.* **2006**, *10*, 255–268. [CrossRef]
16. Lu, X.; Wetter, E.; Bharti, N.; Tatem, A.J.; Bengtsson, L. Approaching the Limit of Predictability in Human Mobility. *Sci. Rep.* **2013**, *3*, 2923. [CrossRef] [PubMed]
17. Gonzalez, M.C.; Hidalgo, C.A.; Barabasi, A.L. Understanding individual human mobility patterns. *Nature* **2008**, *453*, 779–782. [CrossRef] [PubMed]
18. Cho, E.; Myers, S.A.; Leskovec, J. Friendship and mobility: User Movement in Location-Based Social Networks. In Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, San Diego, CA, USA, 21–24 August 2011.
19. Wei, L.Y.; Zheng, Y.; Peng, W.C. Constructing popular routes from uncertain trajectories. In Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, Beijing, China, 12–16 August 2012.
20. Chen, M.; Liu, Y.; Yu, X. NLPMM: A Next Location Predictor with Markov Modeling. In Proceedings of the Pacific-Asia Conference on Knowledge Discovery and Data Mining, Tainan, Taiwan, 13–16 May 2014; Springer: Cham, Switzerland, 2014.
21. Chen, M.; Yu, X.; Liu, Y. Mining moving patterns for predicting next location. *Inf. Syst.* **2015**, *54*, 156–168. [CrossRef]
22. Song, C.; Qu, Z.; Blumm, N.; Barabási, A.L. Limits of Predictability in Human Mobility. *Science* **2010**, *327*, 1018–1021. [CrossRef] [PubMed]
23. Feitosa, R.M.; Labidi, S.; dos Santos, A.L.S.; Santos, N. Social Recommendation in Location-Based Social Network Using Text Mining. In Proceedings of the 2013 4th International Conference on Intelligent Systems, Bangkok, Thailand, 29–31 January 2013; IEEE Computer Society: Piscataway, NJ, USA, 2013.

24. Bao, J.; Zheng, Y.; Mokbel, M.F. Location-based and preference-aware recommendation using sparse geo-social networking data. In Proceedings of the 20th International Conference on Advances in Geographic Information Systems, Beach, CA, USA, 7–9 November 2012.
25. Symeonidis, P.; Ntempos, D.; Manolopoulos, Y. Location-Based Social Networks. In *Recommender Systems for Location-Based Social Networks*; Springer: New York, NY, USA, 2014.
26. Wernke, M.; Skvortsov, P.; Dürr, F.; Rothermel, K. A classification of location privacy attacks and approaches. *Pers. Ubiquitous Comput.* **2014**, *18*, 163–175. [CrossRef]
27. Michael, K.; Clarke, R. Location privacy under dire threat as uberveillance stalks the streets. *Precedent (Sydney NSW)* **2012**, *108*, 24–29.
28. Yuan, J.; Zheng, Y.; Xie, X.; Sun, G. T-Drive: Enhancing Driving Directions with Taxi Drivers. *IEEE Trans. Knowl. Data Eng.* **2012**, *25*, 220–232. [CrossRef]
29. Ashbrook, D.; Starner, T. Using GPS to learn significant locations and predict movement across multiple users. *Pers. Ubiquitous Comput.* **2003**, *7*, 275–286. [CrossRef]
30. Yang, P.; Zhu, T.; Wan, X.; Wang, X. Identifying significant places using multi-day call detail records. In Proceedings of the 2014 IEEE 26th International Conference on Tools with Artificial Intelligence (ICTAI), Limassol, Cyprus, 10–12 November 2014.
31. Montoliu, R.; Blom, J.; Gatica-Perez, D. Discovering places of interest in everyday life from smartphone data. *Multimed. Tools Appl.* **2013**, *62*, 179–207. [CrossRef]
32. Killijian, M.O. Next place prediction using mobility Markov chains. In Proceedings of the First Workshop on Measurement, Privacy, and Mobility, Bern, Switzerland, 10 April 2012.
33. Gidófalvi, G.; Dong, F. When and where next: Individual mobility prediction. In Proceedings of the 1st ACM SIGSPATIAL International Workshop on Mobile Geographic Information Systems, Redondo Beach, CA, USA, 6–9 November 2012; pp. 57–64.
34. Yu, X.-G.; Liu, Y.-H.; Wei, D.; Tian, M. Hybrid Markov model for mobile path prediction. *J. Commun.* **2006**, *27*, 61–69.
35. Lv, M.; Chen, L.; Chen, G. Position prediction based on adaptive multi-order Markov model. *J. Comput. Res. Dev.* **2010**, *47*, 1764–1770.
36. Rodriguez-Carrion, A.; Garcia-Rubio, C.; Campo, C. Performance evaluation of LZ-based location prediction algorithms in cellular networks. *IEEE Commun. Lett.* **2010**, *14*, 707–709. [CrossRef]
37. Rawassizadeh, R.; Dobbins, C.; Akbari, M.; Pazzani, M. Indexing Multivariate Mobile Data through Spatio-Temporal Event Detection and Clustering. *Sensors* **2019**, *19*, 448. [CrossRef] [PubMed]
38. Qiao, S.; Shen, D.; Wang, X.; Han, N.; Zhu, W. A Self-Adaptive Parameter Selection Trajectory Prediction Approach via Hidden Markov Models. *IEEE Trans. Intell. Transp. Syst.* **2015**, *16*, 284–296. [CrossRef]
39. Qiao, S.-J.; Jin, K.; Han, N.; Tang, C.J.; Gesangduoji, G.L. Trajectory prediction algorithm based on Gaussian mixture model. *J. Softw.* **2015**, *26*, 1048–1063.
40. Pathirana, P.N.; Savkin, A.V.; Jha, S. Robust extended Kalman filter applied to location tracking and trajectory prediction for PCS networks. In Proceedings of the 2004 IEEE International Conference on Control Applications, Taipei, Taiwan, 2–4 September 2004; IEEE: Piscataway, NJ, USA, 2004; Volume 63–68.
41. Yang, Z.; Wang, H.-J. Important Location Identification and Personal Location Inference Based on Mobile Subscriber Location Data. *MATEC Web Conf.* **2018**, *173*, 03086. [CrossRef]
42. Lee, J.G.; Han, J.; Whang, K.Y. Trajectory clustering: A partition-and-group framework. In Proceedings of the ACM SIGMOD International Conference on Management of Data, Beijing, China, 11–14 June 2007; pp. 593–604.
43. Rodriguez, A.; Latio, A. Clustering by fast search and find of density peaks. *Science* **2014**, *344*, 1492–1496. [CrossRef] [PubMed]
44. Li, H. *Statistical Learning Method*; Tsinghua University Press: Beijing, China, 2012; pp. 137–139.
45. Lichman, M. *UCI Machine Learning Repository*; University of California: Irvine, CA, USA, 2013; Available online: http://archive.ics.uci.edu/ml (accessed on 19 March 2019).