Journal of Cloud Computing: Advances, Systems and Applications

RESEARCH Open Access

Self-adaptive trajectory prediction for improving traffic safety in cloud-edge based transportation systems

Bin Xie[1,2], Kun Zhang[1*], Yi Zhao[1], Yunchun Zhang[1], Ying Cai[1] and Tian Wang[1]

Abstract

Intelligent transportation brings huge benefits to humans' life and Industrial production in terms of vehicle control and traffic management. Now, the development of edge-cloud computing has once again promoted intelligent transportation into a new era. However, the development of intelligent transportation inevitably produces a large amount of data, which brings new challenges to data privacy protection and security. In this paper, we propose to develop an improved trajectory prediction framework based on the self-adaptive trajectory prediction model (SATP), which could significantly enhance traffic safety in transportation systems. The proposed framework is capable of guaranteeing the accurate trajectory prediction of moving target under different application scenarios. In particular, to reduce the size of original trajectory point data collected by sensors, the angle change and minimum description length (MDL) principle are first combined to remove the redundant points in raw trajectories. The obtained points can then be reduced for model using the two-step clustering method. To further enhance the prediction performance, we add the "self-transfer" to the original model to solve the problems that the state of original SATP model may be discontinuous. Furthermore, we propose to develop a trajectory complementation method based on Bezier curve to improve the prediction accuracy. Finally, by comparing the two-step clustering method with the commonly-used SinglePass and density-based clustering method (DBCM) algorithms, the proposed two-step clustering policy greatly reduce the time cost of clustering. At the same time, by comparing the improved SATP model with the original model, the results show that the improved SATP method can greatly improve the speed of prediction model.

Keywords: Trajectory prediction, Trajectory clustering, Hidden Markov model, Bezier curve, Big data

Section I: introduction

In the past few decades, intelligent transportation has become an effective way to manage vehicles, improve traffic system performance, enhance travel safety, and provide travelers with more choices. Cloud computing [1–3] provides services to users in a shared resource pool, and users do not need to care about the operation and maintenance of equipment. The edge-cloud-based intelligent transportation system can further improve road safety, traffic productivity, and travel reliability. However, while enjoying these benefits, we inevitably face data privacy and security issues [4–6] arising from intelligent transportation.

With the rapid development of wireless communication and Global Navigation Satellite Systems (GNSS) [7] techniques, it possible for us to systematically track object movements while collecting a large amount of trajectory data, such as vessel positioning data and animal movement data. Moving target trajectory prediction refers to real-time prediction of the moving target's current trajectory by using a large amount of historical behavior trajectory information of the moving target.

* Correspondence: zhangkun@njust.edu.cn
[1]School of Computer Science and Engineering Nanjing University of Science and Technology, Nanjing, China
Full list of author information is available at the end of the article

Thus, we should make corresponding operations [8, 9] before predicting the moving target's behavior trajectory. Moving target trajectory prediction has been widely used in various fields and has recently attracted interest from researchers, such as urban planning, location services, national defense, military, traffic management and vehicle routing, security applications such as barrier monitoring, multiple tasks [10–14].

Over the past decades, many trajectory prediction methods for moving targets have been proposed. Monreale et al. [15] proposed a trajectory prediction method named WhereNext, which could find out the locations where moving targets often visited according to the pattern. It could then use the T-pattern tree to extract the history trajectory which had the highest matching degree with the current trajectory as the predicted trajectory. Ying et al. [16] predicted the location of the moving target at the next moment based on the semantic features of geography and trajectory. This method predicted the location of the next moment by mining the frequent behavior features of the same kind of moving targets. Song et al. [17] proposed a state space model based on user mobility model, and used Markov transfer probability to mine the transformation between moving targets in different states. Ishikawa et al. [18] divided the map into a grid of different sizes and found the grid of the moving target by using the R-tree and Markov chain to describe the probability of the moving target's transfer between the grids. Ma et al. [19] applied the hidden Markov theory to the urban taxi movement trajectory model, which can provide users with decision-making support for the ride route. Asahara [20] used mixed Markov chain model to predict pedestrian trajectories. They took into account the moving targets' individual characteristics and historical status. Killijian [21] extended the mobile Markov chain (Mobility Markov Chain, MMC) model to predict the location of the moving target. The essence of the model was the high order Markov model, the prediction accuracy can reach 70% ~ 95%, but the computational cost was large. Qiao et al. [22] modelled the complex motion model of moving objects by using the Gauss hybrid model, then analyzed the probability distribution of different motion patterns. The self-adaptive trajectory prediction (SATP) model based on hidden Markov model (HMM) model proposed by Qiao et al. [23, 24] reduced the number of hidden states by using the clustering algorithm based on density, and then used HMM to predict the trajectory. However, the execution speed of the method was still slow. Moreover, the solution was poor when the state appears stayed and discontinuous. Furthermore, deep learning technology [25–28] widely used in image processing can also provide solutions for trajectory prediction.

The purpose of this paper is to improve the convergence speed of the model and the efficiency of prediction. The main contents of this paper are as follows: (1) Streamline the trajectory points according to the MDL (Minimum Description Length) principle, which can reduce the amount of data to be processed and speed up the model training and model prediction; (2) Using the trajectory clustering algorithm to reduce the number of hidden states in the HMM model. At the same time, combine the SinglePass algorithm and DBCM (density-based clustering method) algorithm into a two-step clustering algorithm, which reduces the time complexity of the original density-based clustering algorithm and accelerates the speed of trajectory point clustering; (3) Integrate the initial state transition with the implicit state transition probability matrix in the SATP model, and add self-transition in the implicit state transition probability matrix to solve the problem of state stay and discontinuity; (4) Completing the predicted trajectory by using the Bezier curve [29] to improve the accuracy of trajectory prediction.

Section II: analysis of algorithms

The flow chart of the moving target trajectory prediction method in this paper is shown in the Fig. 1. Based on this flow chart, it can be seen that the process of moving target trajectory prediction can be divided into the following two parts: action mode training and action trajectory prediction. The action mode training is mainly divided into: simplifying the historical trajectory points, aggregating the historical trajectory points, and the training and storage of historical trajectory action mode models. The process of trajectory prediction can be divided into the following steps: simplifying the current trajectory points, calculating the possible hidden state chains corresponding to trajectory points, calculating the hidden state chains with maximum probability, calculating the transition probabilities of subsequent states, reducing the hidden state of trajectory points, and completing the trajectory points. The loop is repeated many times to get a number of subsequent states to predict a relatively long trajectory.

Trajectory point simplification method based on MDL principle

In order to satisfy the rapid completion of trajectory point clustering and predictive model training based on big data, this paper uses the minimum description length (MDL) [30] to simplify the trajectory points. Only those points in the trajectory that best describe the trajectory will be retained, for reaching a balance between accuracy and simplicity.

The calculation complexity of the MDL principle is relatively high. Therefore, before the calculation, we first

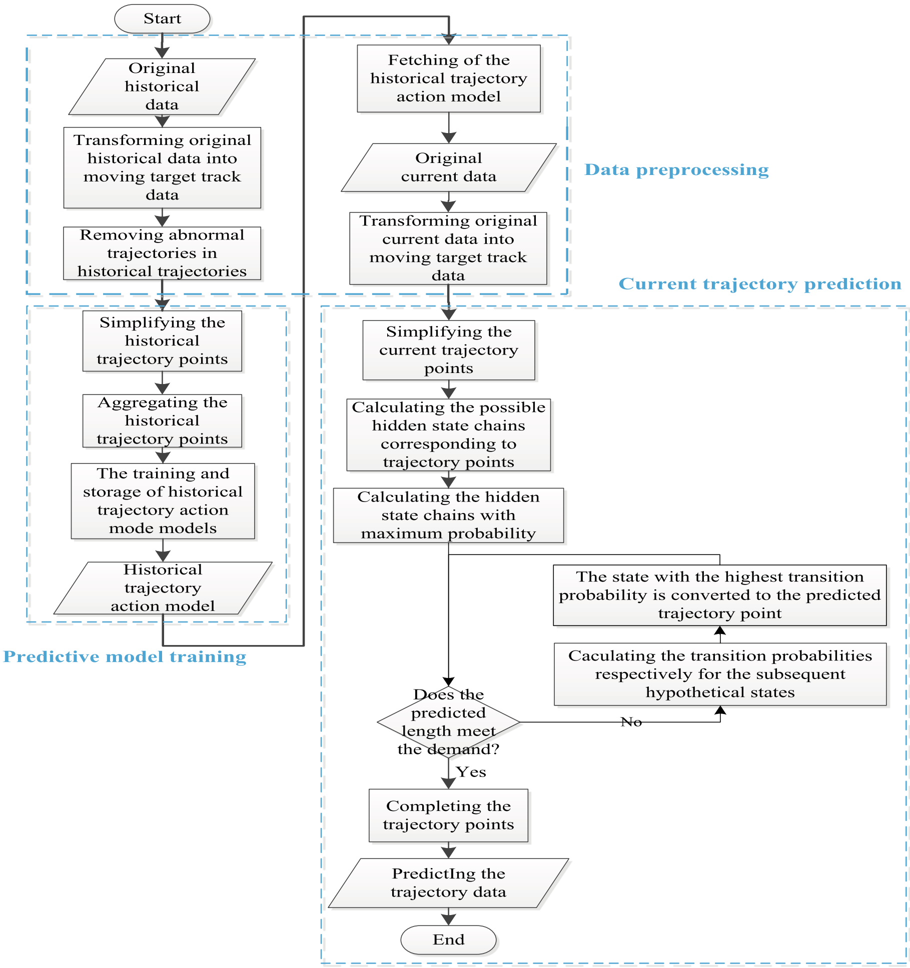
Start
Original historical data
Transforming original historical data into moving target track data
Removing abnormal trajectories in historical trajectories
Fetching of the historical trajectory action model
Original current data
Transforming original current data into moving target track data
Data preprocessing
Current trajectory prediction
Simplifying the historical trajectory points
Aggregating the historical trajectory points
The training and storage of historical trajectory action mode models
Historical trajectory action model
Predictive model training
Simplifying the current trajectory points
Calculating the possible hidden state chains corresponding to trajectory points
Calculating the hidden state chains with maximum probability
The state with the highest transition probability is converted to the predicted trajectory point
Caculating the transition probabilities respectively for the subsequent hypothetical states
Does the predicted length meet the demand?
No
Yes
Completing the trajectory points
PredictIng the trajectory data
End

Fig. 1 Trajectory prediction method flow chart

make a filter to trajectory point according to the change of the direction of the trajectory point. For the trajectory consisting timestamped points $\{P_1, P_2, \cdots, P_n\}$, start from P_3, calculating the slopes k_1, k_2 of the two segments and $P_{i-1}P_i$, respectively. If $|k_1 - k_2| > angle$, it indicates that the change of direction at point P_{i-1} is large enough, so this point needs to be preserved; otherwise, it shows that there is almost no change of direction at point P_{i-1}. Then this point can be removed at this time. The angle is set to 5^o in this paper. We also study how to choose the angle in our future work.

After the first filter, the trajectory points have only reached a certain level of simplification but have not reached the optimal simplification, i.e., The filtered data can't completely represent trajectory points. Therefore, this paper makes a second simplification based on the MDL principle. The MDL principle was originally proposed to compress spatial data. Its formula is composed of L(H) and L(D|H), where L(H) represents the cost of the compression model and L(D|H) represents the overhead of data D after compression by model H. When L(H) + L(D|H) takes the minimum, the compression of the data is optimal because it is used to store the model and store the compression and the length of the data is minimal. Since there is no data compression model in this paper (that is, no data restoration is needed), this paper designs an MDL formula that is applicable to this project. It should meet the requirements: the more the number of trajectory points that are ultimately selected, then the more assumption condition L(H) is, that is the corresponding data overhead L(D|H) is smaller. Conversely, when the number of final selected trajectory points is smaller, then the smaller the condition L(H) is, the corresponding data overhead L(D|H) is bigger. In order to meet this demand, we designed the MDL formula in this paper which is shown below.

$$\begin{cases} LH = \left|trace'\right| = \sum_{i=1}^{k-1} PS_i PS_{i+1}, \\ L(D|H) = miss\left(trace, trace'\right) = \sum_{i=1}^{k-1} \sum_{j=J}^{K} dis\left(PS_i, P_j, PS_{i+1}\right), \\ dis(A, B, C) = 2 \times (1 + \ cos\angle ABC) \times |B \sim AC| \end{cases} \quad (1)$$

where $trace = \{P_1, P_2, \cdots, P_n\}$ is the original trajectory point $trace = \{PS_1, PS_2, \cdots, PS_k\}$ is a streamlined trajectory, the MDL formula is used to solve the description overhead and description ability of the simplistic trajectory. $|trace'|$ indicates the length of the trajectory *trace* Same as it, $|PS_iPS_{i+1}|$ indicates the line segment $|PS_iPS_{i+1}|$. $miss(trace', trace)$ represents the error between the trajectory and the trajectory, and $index(PS_i, trace)$ represents the subscript of the point PS_i in the original trajectory point sequence trace. $|B\sim AC|$ indicates the height in ΔABC where the bottom edge is AC and the apex angle is $\angle ABC$, as shown in Fig. 2. K, J are given as by:

$$\begin{aligned} K &= index(PS_{i+1}, trace) + 1, \\ &= index(PS_{i+1}, trace) + 1. \end{aligned}$$

The goal of applying MDL principle is to when the value of the formula $L(H) + L(D|H)$ reaches the smallest. The selection of the reduced trajectory point can best describe the original trajectory. This formula simplifies the calculation of L(D|H) with respect to the original formula. The calculation of the vertical and angular distances between the line segment and the line segment is modified to calculate the high and vertical angle cosines of the triangle, which can be satisfied under the same requirements. It can thus reduce the amount of calculations accordingly. The height of the triangle can be calculated using Helen's formula, and the cosine of the top corner can be calculated using the cosine theorem.

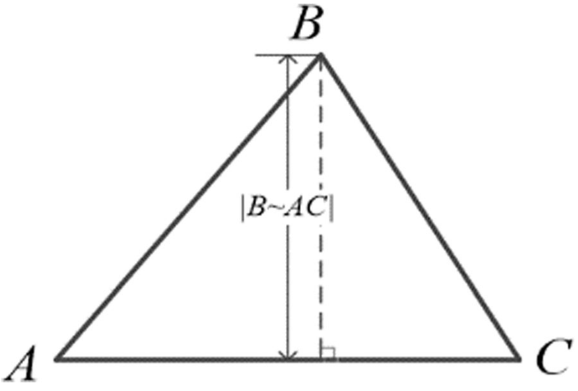

Fig. 2 Schematic diagram of a triangle ΔABC with a base AC and an apex $\angle ABC$

Trajectory point clustering method based on two-step clustering

This section focuses on specific methods based on two-step clustering. The purpose of the two-step clustering is to reduce the computational complexity of the trajectory point clustering, and to reduce the matrix size of the hidden state matrix in the hidden Markov model that will be mentioned later. In this paper, the trajectory points are clustered once by the SinglePass algorithm. The reason for we use SinglePass algorithm is that this algorithm is very suitable for clustering flow text. After the first step clustering, the cluster centers are obtained. Each cluster is composed of several trajectory points and cluster centers. For the cluster centers obtained by the first step cluster, a clustering algorithm based on density-based clustering method (DBCM) [31–35] is used for the secondary clustering. Compared with the existing clustering algorithm (e.g., DBSCAN), DBCM does not require embedding the data in a vector space and maximizing explicitly the density field for each data point.

The first step of SinglePass clustering algorithm is sensitive to parameter of cluster radius, but since the trajectory point data itself has a distance and there is a secondary clustering, the parameters of the first step cluster can be set to a relatively small value according to the specific requirements. In extreme terms, if the radius parameter is set to 0, it can be understood that each trajectory point itself is a cluster, which is equivalent to directly performing the secondary clustering. For example, the distance radius parameter $d_1 = 0.1$ is set in this paper. Note that we also can select the other value of d_1.

The basic steps of the DBCM algorithm are shown as follows:

1) Calculate the density of each cluster center point i obtained after one-step clustering. The local density of the point $i : \rho_i = \sum \tau(d_{ij} - d_2)$, $\tau(\nu) = \begin{cases} 1, \nu < 0 \\ 0, \nu \geq 0 \end{cases}$, where the parameter d_2 is the boundary threshold, The smaller the value of d_2, the smaller possible range will cover cluster.
2) Calculate the minimum distance from the point i to all other points above its density $\kappa_i = \min_{j:\rho_j>\rho_i} d_{ij}$.
3) Cluster centers are recognized as points for which the values of ρ and κ are anomalously large. Here, the algorithm comprehensively measures the influence of two factors on the cluster center through the product factor ψ. The product factor ψ_i for point i is defined as shown in eq. (2).

$$\psi_i = \ norm\ \rho_i \times \ norm\ \kappa_i \quad (2)$$

where $norm\ \rho_i$ and $norm\ \kappa_i$ are normalized values, the normalization method uses the normalization of the dispersion and maps the values to the interval [0, 1]. Specifically, *norm* ρ_i is defined as follows:

$$norm\ \rho_i = \frac{\rho_i - \ min\{\rho_1, \rho_2, \cdots, \rho_n\}}{max\{\rho_1, \rho_2, \cdots, \rho_n\} - \ min\{\rho_1, \rho_2, \cdots, \rho_n\}} \tag{3}$$

$$B = \begin{bmatrix} 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 & 0 & 2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 2 & 0 & 0 & 0 & 2 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 3 & 2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

The calculation method of *norm* κ_i is similar to this and will not be described again. The larger ψ, the larger the center density of the clusters and the further the distance between the centers of the different clusters. Sort the ψ values from large to small, and select the point with the larger ψ value as the cluster center point. Since the transition from the non-cluster center point to the cluster center point, the ψ value will increase greatly, so the number of clusters will be determined according to the power law.

4) For the remaining non-clustered center data points, the points are assigned to the clusters of the neighbor nodes that are closest to them and have a higher density than them.

DBCM has one parameter: the boundary threshold d_2. Since the result of first step clustering is theoretically a circular cluster, the distance between adjacent cluster centers is at least $2 \times d_1$. Therefore, d_1 should be set to at least $2 \times d_1$ in the secondary clustering. This paper sets d_2 to $2 \times d_1$ (if d_1 is set smaller in the application, d_2 should be larger. If $d_2 < 2 \times d_1$, the secondary clustering algorithm cannot be executed; if d_2 is set smaller in the application, then the speed of the secondary clustering speed will be slower; if d_2 is set larger in the application, there will be too much excessive loss of hidden state quantity).

The two-step clustering proposed in this paper can speed up the clustering speed of the trajectory points because the event complexity of the DBCM clustering algorithm is applied to the trajectory points is $O(n^2)$, and n is the number of trajectory points. For the massive trajectory point data, so the first step is is to use the SinglePass clustering method to initially "concentrate" a large number of trajectory points into a smaller number of clusters, and then use DBCM to concentrate the clusters. Conducting secondary clustering can greatly reduce the input of secondary clustering. Based on the aforementioned analysis, it can be concluded that the two-step clustering method contains the SinglePass clustering and DBCM clustering. Suppose the number of trajectory points is n, m represents the number of clusters. Thus, when the SinglePass is used to cluster the data, the computational complexity of SinglePass is $O(nm)$. Now, the large number of trajectory points will be reduced into a smaller number of clusters, i.e., m. The computational complexity DBCM is $O(m^2)$ when the DBCM is used to cluster the data that have been clustered by SinglePass. Thus, the computational complexity of the proposed strategy is $O(nm + m^2$, which is also less than n^2, i.e., the complexity of two-step clustering is less than DBCM. Thus, two-step clustering effectively speeding up the trajectory point clustering speed.

Improved trajectory prediction method

In this paper, based on the hidden Markov model, the dataset is used to train the model firstly to generate the implicit state attribution probability and the implicit state transition probability in the model. Then, for the trajectory to be predicted, we enumerate all possible subsequent hidden states, use the forward algorithm to calculate the probability of each state and take the most probable state as the follow-up state predicted, and we use the hidden state center (cluster center) as the prediction trajectory point .The result of the model training is to obtain the state transition probability matrix A and the explicit state probability matrix B. We explain the model training and model prediction steps of this method in detail with the example as shown in Fig. 3.

In Fig. 3, there are five historical trajectories (The five-pointed star represents the trajectory point. The order of the five trajectories is shown by the arrow. The dotted circle in the figure represents the clustering effect in the previous step, in the present example, clusters c1-c5 are obtained after clustering 17 trajectory points. To adapt to the model, clusters are called "states" in the following steps to represent the hidden states in the model.) First, they are used for model training. The steps are as follows:

1) The mesh size is firstly determined based on the historical trajectory point coordinate range and the cluster diameter. Assume that in this example, the mesh is divided as shown in the figure, resulting in sixteen grids b1-b16, making all historical trajectory points in a grid.
2) The state transition probability matrix A and the explicit state probability matrix B are established, where the number of rows and columns of A is respectively the number of states plus one and the number of states (that is, in this example, A is a matrix of 6 rows and 5 columns, in the matrix each value represents the probability of shifting from the number of rows minus one represented state to the

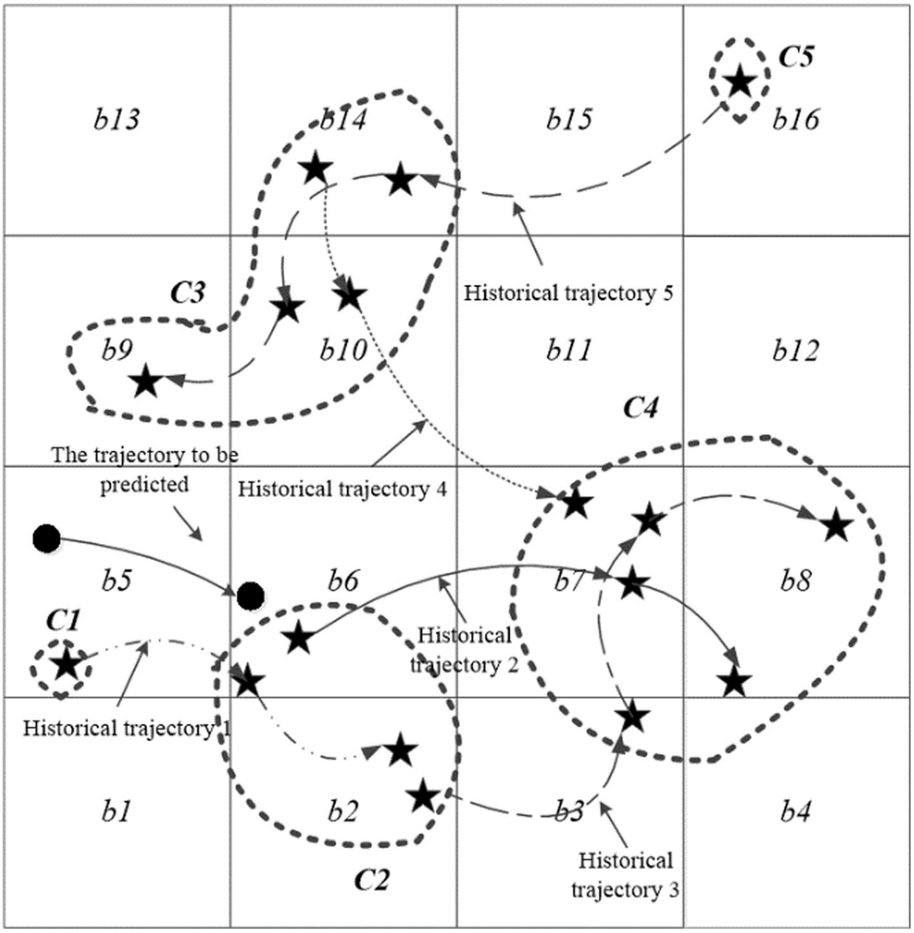

Fig. 3 Example for trajectory prediction method introduction

state represented by the number of columns, where the extra first behavior is the probability that the initial state transitions to the first state); At the same time, the number of rows and columns of B represents the number of states and the number of grids (in this example, B is a matrix of 5 rows and 16 columns. Each value in the matrix indicates that the number of columns indicates the probability that a point in the grid may belong to the state indicated by the row number). All the values in both matrices default to zero.

3) The historical trajectories used for training are traversed, and the processing method of historical trajectory 2 is described in detail in the following steps.
4) First of all, the first point of the historical trajectory 2 is b6, its belonging state is c2, and the previous state is the initial state, so we add 1 to $A[1][2]$ and add 1 to $B[2][6]$. The second point of the historical trajectory 2 is at the grid b7, and the affiliation state is c4. At this time, the previous state is c2, so the $A[3][4]$ is increased by 1 and the $B[4][7]$ is increased by 1 at the same time .The third point of the historical trajectory 2 is at the grid b8, the belonging state is c4, and the previous state is c4, so the $A[5][4]$ is increased by 1 and the $B[4][8]$ is increased by 1 at the same time.
5) Do a similar operation on the other trajectories to get the following matrix:

$$(\mathrm{A}=)\begin{bmatrix} 1 & 2 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 2 & 0 \\ 0 & 0 & 3 & 1 & 0 \\ 0 & 0 & 0 & 3 & 0 \\ 0 & 0 & 1 & 0 & 0 \end{bmatrix}$$

$$B=\begin{bmatrix} 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 & 0 & 2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 2 & 0 & 0 & 0 & 2 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 3 & 2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

6) For each value in the matrix A and matrix B, it is divided by the sum of all the values in its row to get the final probability matrix (the training part is completed so far).

$$A = \begin{bmatrix} 0.2 & 0.4 & 0.2 & 0 & 0.2 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0.33 & 0 & 0.67 & 0 \\ 0 & 0 & 0.75 & 0.25 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 \end{bmatrix}$$

In the example above, the detailed steps of the model training are explained. The result of the model training is to obtain the state transition probability matrix A and the explicit state probability matrix B. The two matrices are related to the prediction. The probability calculation method used in the prediction of this paper is the forward algorithm, whose essence is to calculate the probability of the next possible state, regardless of the moving target's previous state, and selects the largest probability as the predicted state. In the following content, the specific method of prediction will be described in detail around this example (the trajectory to be predicted has been shown in the figure, and it currently has two trajectory points):

1) For the trajectory points currently existing in the trajectory to be predicted, the probabilities of all the states that proceed from the initial state to this point are calculated sequentially from the initial state using the forward algorithm according to the matrix A and B.
2) First, the first point of the trajectory to be predicted, where the grid is b5, and the previous state is the initial state, so the calculation should use the first row of A and the fifth column of B. The specific calculation method is the probability that the initial state transferres to each other state multiplied by the probability that the point belongs to the state(i.e., the value of the first row in A is multiplied by the value of the first column in B to get a probability vector). The calculation of this step is shown in Table 1.

Table 1 The calculation of first point probability

	Calculation equation	Probability
P(c1)	0.2 × 1	0.2
P(c2)	0.4 × 0	0
P(c3)	0.2 × 0	0
P(c4)	0 × 0	0
P(c5)	0.2 × 0	0

3) For the second point and follow-up point (in this case, the trajectory has only two points. In practical applications, the method for calculating the actual existence of subsequent points is similar). The probability calculation method is slightly different from the previous step, that is, it needs to be calculated. The prior probability of the previous step is added and the calculated probabilities are summed. That is, if we are looking for the probability that the second point belongs to c2, because we are not sure about the state of the first point, we should find that “the first point belongs to c1 and the second point belongs to c2” and “the first point belongs to c2 and the second point belongs to c2”, “the first point belongs to c3 and the second point belongs to c2”, “the first point belongs to c4 and the second point belongs to c2” and “the first point belongs to c5 and the second point belongs to the probability of c2, and then sums the probabilities to get the probability that the second point belongs to c2. In the previous step, the probability that “the first point belongs to c1“ has been calculated, while “the first point belongs to c1 and the second point belongs to c2“ needs to be added to the former by the limitation that “the state from the first point to the second point is transferred from c1 to c2 and the second point belongs to c2”, so the solution of this probability is: P(c1) ∗ P (state transition from c1 to c2) ∗P (second point belongs to c2) (that is, the solution results of the first step multiplied by second row and second column in A, and second row and first column in B). After all the above probabilities are calculated in a similar way, they are summed to obtain the probability that the second point belongs to c2. Similarly, the same problem can be solved for the probability that the second point belongs to c1. The solution method is shown in Table 2.
4) After that, it is needed to start solving the probabilities of predicting the state. The solution to this probability is similar to the previous step, but since there is no specific trajectory point, there is no need to add the explicit state transition probability in the solution equation, in other words, no B matrix is needed. The solution method for the

Table 2 The calculation of second point probability

	Probability
P(c1)	0.2
P(c2)	0
P(c3)	0
P(c4)	0
P(c5)	0

next prediction state is shown in Table 3. It can be seen that the probability that the next state in c4 is the largest, so we should take the center point of the c4 cluster as the next predicted trajectory point.

5) After predicting the position of the next trajectory point, if the predicted length does not meet the demand, the prediction needs to be continued. On the basis of step (4), similar calculations are performed again, and the results as shown in Table 4 are obtained. That is, the state with the greatest probability of the next step is c4, and the center point of the c4 cluster is taken as the predicted trajectory point of the next step.

When the predicted length reaches the demand, the calculation is stopped, and the predicted trajectory point is complemented (the following section will describe the completion method in detail). At this point, the trajectory prediction step is completed.

Trajectory complement method based on Bezier curve interpolation

After using the SATP model to predict the trajectory points, we get some distant trajectories (hidden states), and the demand in this paper can predict relatively continuous motion trajectories. Therefore, this section introduces the trajectory complement method based on the Bezier curve in detail. In the previous research, the two element functions are used to fit the trajectory point, but the trajectory point may appear the same horizontal coordinates and different ordinates. Therefore, this method can't meet the requirements of this paper. In addition, the author finds [21] that the Bezier curve is better to complement the trajectory with less trajectory points, and does not need to be trained in advance but can achieve a relatively small error, so this method will be used to complement the trajectory point.

The steps for a Bezier-based trajectory completion method will be described in detail with Fig. 4. There exist five points (a blue, five-pointed star) in a trace, where the distance between point B and point C is too large. This can be judged from B to C needing to make up points operation. In this example, the effect after the complement is shown in the figure. Among them, three red five-pointed stars are the points obtained by applying the complement method. The procedure of the point-of-replenishment operation in this example is described as follows:

Table 3 The calculation of first predicted point probability

	Probability
P(c1)	0
P(c2)	0.033
P(c3)	0
P(c4)	0.067
P(c5)	0

Table 4 The calculation of second predicted point probability

	Probability
P(c1)	0
P(c2)	0.01089
P(c3)	0
P(c4)	0.08911
P(c5)	0

1) Calculating the distance *dis* from B to C. Dividing the *dis* by a shaping parameter *PDIS* to obtain 3, determining that BC needs to fill 3 points between two points.
2) Finding the vector $\overrightarrow{AB}$, then calculating the control point $A^{'}$ of the Bezier curve according to the coordinates of the vector $\overrightarrow{AB}$ and the point B, and calculating the equation of the Bezier curve according to the points B, $A^{'}$, C.
3) Substituting $t = 0.33$, $t = 0.67$, $t = 1$ into the Bezier equation respectively (because step (1) requires 3 points, 3 values are 1/3, 2/3 and 3/3 respectively), finding the coordinates of the three points that need to be complemented, and thus the completion of this point is completed.

Section III: experimental results

This paper uses the improved SATP model to predict the moving target's trajectory points, in order to adapt to the mass of trajectory point data. Furthermore, in order to reduce the amount of the data and speed up the model training and prediction. This paper adopts the MDL principle to simplify the trajectory points and two-step clustering algorithm for clustering the trajectories in order to reduce the number of implicit states in the model training. After the trajectory prediction, Bezier interpolation is also used to complete the trajectory point.

Realization and verification of trajectory point clustering algorithm

In order to improve the computational efficiency for the prediction of moving target's trajectory, this paper introduces a two-step clustering based on SinglePass and DBCM on the trajectory points before training on the improved SATP model. This section implements SinglePass clustering, DBCM clustering and two-step clustering algorithm, respectively. The proposed method will be evaluated in terms of clustering effect and clustering speed. The clustering results of the three clustering algorithms on the same data are shown in (a), (b), and (c) in

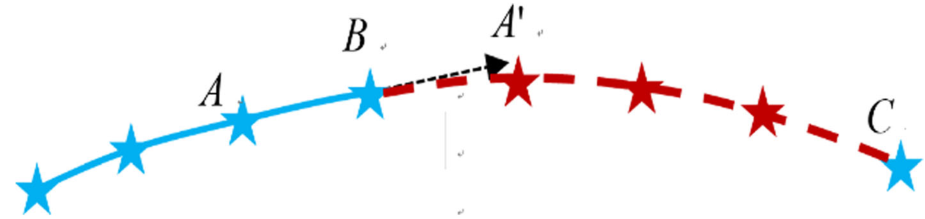

Fig. 4 Example for Bezier trajectory completion method

Fig. 5, respectively. It can be seen from the Fig. 5(a) that if the SinglePass clustering algorithm is used alone, the clustering effect is poor, it can't recognize irregularly shaped clusters. Thus, the clustering results obtained by SinglePass does not meet the needs of this paper. By observing the Fig. 5(b) and Fig. 5(c), we find that the clustering results obtained by DBCM and two-step clustering outperform the SinglePass, i.e., some samples categories are correctly distinguished. Therefore, the clustering results obtained by DBCM and two-step clustering can meet the requirements of this paper. On the other hand, we also find that the effect of using DBCM algorithm is similar to that of using the two-step clustering algorithm proposed in this paper (Fig. 6).

Realization and verification of trajectory prediction methods

The trajectory prediction method proposed in this paper improves the prediction speed, but at the same time it may reduce the prediction accuracy. Therefore, after implement the algorithm, this paper also uses the same project experimental data to test the improved model and algorithm, and compare it with the original model from two aspects of time consumption and prediction accuracy.

This paper selects the first 1 billion to 2 billion pieces of raw data (about 3 months to 6 months) as the input of the training part, and selects 100,000 pieces of raw data (about 2.5 h) as the input of the prediction part. Then trains the original SATP model and the improved SATP model proposed and perform trajectory prediction separately. Finally, the time of model training (including trajectory point reduction and trajectory point clustering steps), model predictive time, predictive deviation degree, and predictive accuracy of the two models are respectively counted.

From the two graphs in Fig. 7 (a) and (b), the original SATP.

model spends more time on model training than the improved SATP model. When the amount of data reaches 1.6 billion, the training time of the original SATP model has exceeded 30 min, and the improved SATP model exceeds 30 min when the data volume reaches 2 billion. Therefore, the improved SATP model is significantly faster in time than the original SATP model. At the same time, in the model prediction, the improved SATP model reduces the prediction time by 12 s on average compared with the original SATP model, and can control the prediction time of each trajectory within 100 milliseconds. It can be concluded that the improved SATP model is significantly faster than the original SATP model.

As can be seen from the two graphs in Fig. 8 (a) and (b), as the amount of training data increases, the

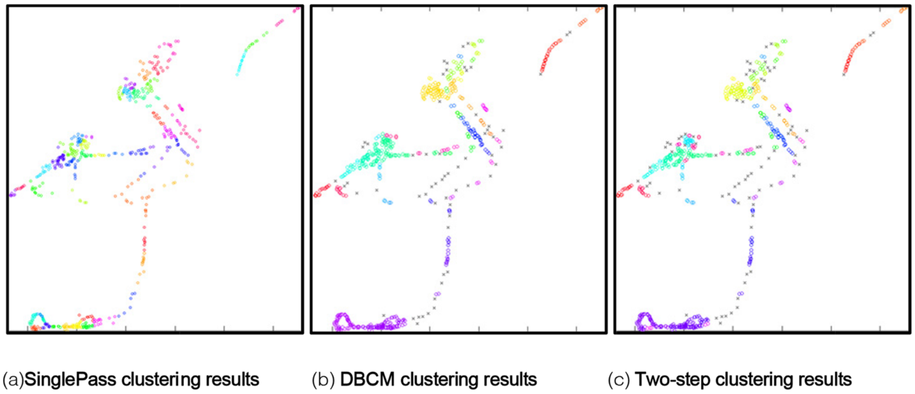
(a)SinglePass clustering results (b) DBCM clustering results (c) Two-step clustering results

Fig. 5 Clustering effect of the three clustering methods

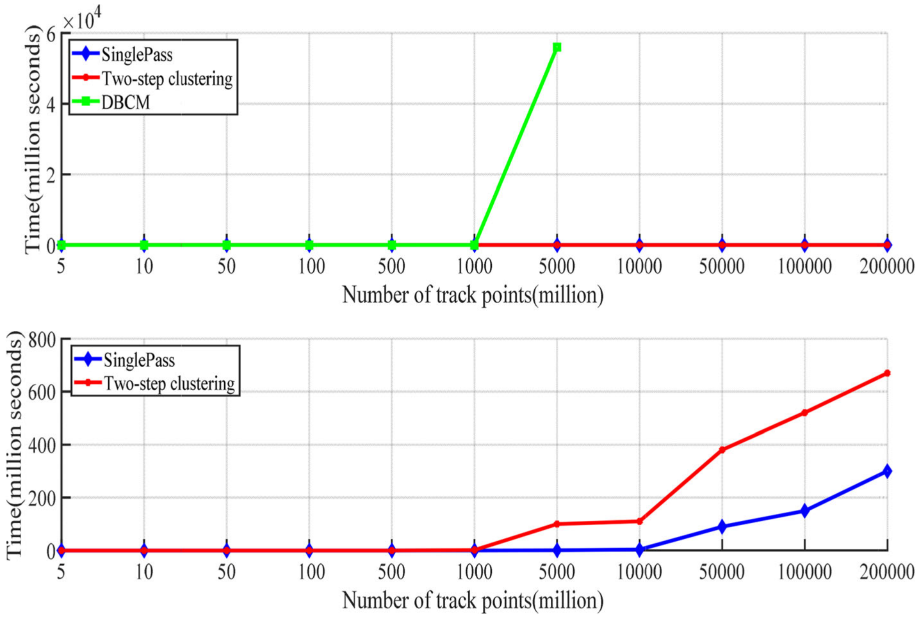

Fig. 6 Time-consuming comparison of three clustering algorithms

predictive deviation degree of the two models will decrease, and the trend will decrease after the data volume reaches 16 million. At the same time, the forecasting accuracy shows the opposite trend.

In addition, the predictive accuracy of the improved SATP model is also affected by the degree of reduction of the hidden state after clustering.

It can be seen from Fig. 9 that with the increasing of the number of hidden states after clustering, the predictive accuracy obtained by the improved SATP model shows a trend of rising first and then decreasing, and it reaches the extreme value when the number of hidden states reaches around 1000 and when the number of hidden states exceeds 1000, due to the possibility of overtraining, the accuracy rate decreases. When the number of hidden states is about 50 or 2500, the accuracy rate drops to around 0.6.

The improved SATP model proposed in this paper has simplified the training data to speed up the training, thus reducing the prediction accuracy of the model. And in this paper, the trajectory point complementation method based on Bezier curve is used to complete the prediction trajectory and minimize the prediction error as much as possible. Although the accuracy of the improved SATP model is indeed lower than that of the original SATP model, experiments show that the improved SATP

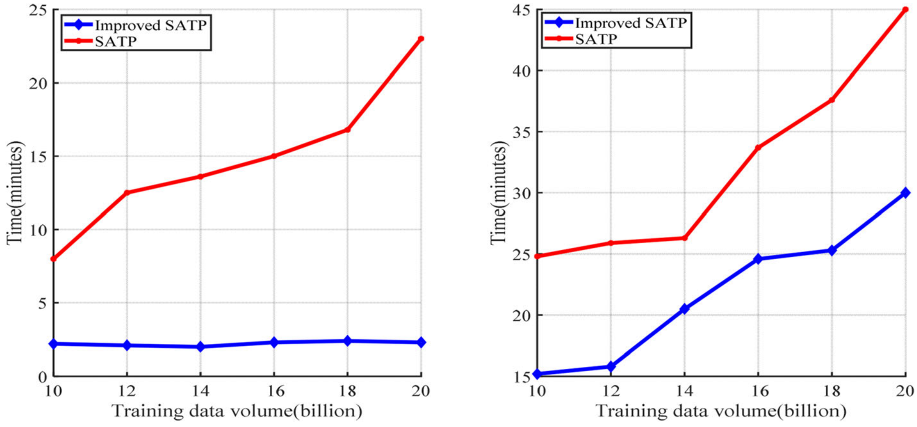

Fig. 7 Time-consuming comparison of original and improved SATP models

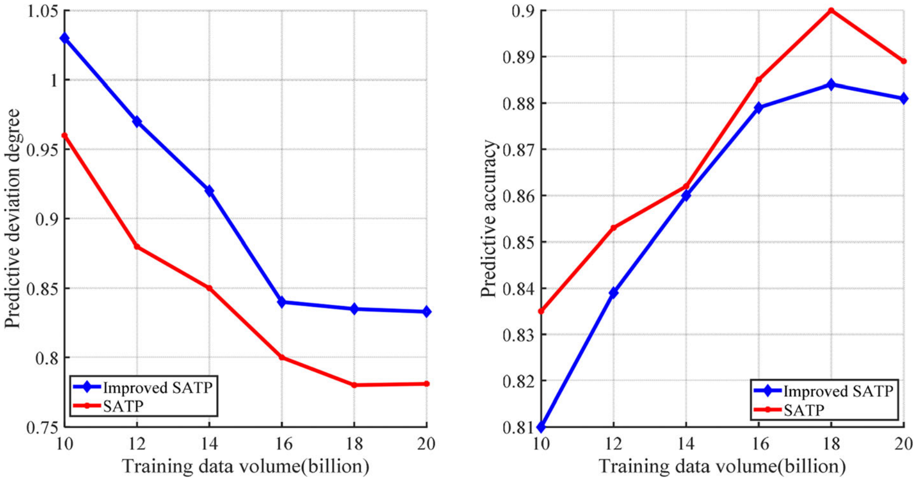

Fig. 8 Comparison of Accuracy of Original and Improved SATP Models

method has an average reduction accuracy of about 0.01, and the predictive accuracy can still reach about 0.89 when the training data reaches 18 million.

Combined with the relevant experimental results, it can be seen that when taking 1.6 billion - 1.7 billion historical data as training data and the number of hidden states is about 1000, it can meet the demand in terms of training time, predictive time, and predictive accuracy. Achieve a better prediction effect.

Section IV: conclusion

This paper proposes a moving target trajectory prediction method based on the improved SATP model. First, for millions of levels of trajectory point data, the trajectory points are reduced to small data according to the angle change and the MDL principle, respectively, thereby reducing the data to be processed to some extent. Then a two-step clustering method combining the two clustering algorithms of SinglePass and DBCM is proposed to reduce the state of the training and prediction of the model. The training time of algorithm is reduced from several hundred minutes to less than fifty minutes. Afterwards, problems such as state discontinuity that may exist in the original SATP model can be solved efficiently by adding "self- adaptive" to the model without additional judgment. Finally, the predicted trajectory point distance caused by the over-simplification of the method described in this paper is too large even deteriorated the prediction accuracy, so that this paper proposed the trajectory completion method based on the Bezier curve which solved this problem reasonably. The Predictive accuracy of the proposed method can still reach about 0.89 when the training data reaches 18 million.

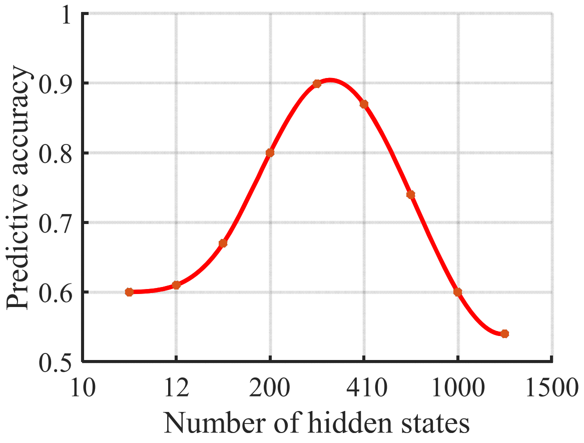

Fig. 9 Influence of the number hidden states on prediction accuracy

After detailed description of the steps and details of the moving target trajectory prediction method, this paper also tested the effect of this method through relevant experiments. By comparing the two-step clustering method with the SinglePass and DBCM algorithms, it is found that the two-step clustering can basically maintain the clustering effect and greatly reduces the time consumption of clustering at the same time. When the number of trajectories is 2 billion, the clustering time can be controlled within 20 min. Finally, by comparing the improved SATP model with the original SATP model, it is found that the algorithm proposed in this paper can significantly speed up model training and model prediction while achieving a very small decrease in accuracy, thereby meeting the demand. Furthermore, in our future work, we will consider some modern ensemble learning-based prediction methods, such as deep forest.

Acknowledgements
The authors would like to thank the editors and the anonymous reviewers for their constructive comments and valuable suggestions.

About the authors
Bin Xie received the B.S. degree from National University of Defense Technology and M.S. degree from Fudan University. He is currently pursuing the Ph.D. degree with the Nanjing University of Science and Technology, Nanjing, China. His research interests include artificial intelligence, complex networks and statistical analysis.
Kun Zhang received the B.S. degree from National University of Defense Technology and M.S. degree from Fudan University. She is currently pursuing the Ph.D. degree with the Nanjing University of Science and Technology, Nanjing, China. Her research interests include artificial intelligence, complex networks and statistical analysis.
Yi Zhao received the B.S. degree from Jiangsu University and M.S. degree from Nanjing University of Science and Technology, Nanjing, China. His research interests include artificial intelligence, complex networks and statistical analysis.
Yunchun Zhang received the B.S. degree from Jiangsu University and M.S. degree from Nanjing University of Science and Technology, Nanjing, China. Her research interests include artificial intelligence, complex networks and machine learning.
Ying Cai received the B.S. degree from Soochow University and M.S. degree from Nanjing University of Science and Technology, Nanjing, China. Her research interests include artificial intelligence and machine learning.
Tian Wang is currently working toward the PhD degree in the School of Computer Science and Engineering, Nanjing University of Science and Technology, Nanjing, China. His current research interests include the areas of cloud computing and cyber security. He is a student member of the IEEE.

Authors' contributions
Bin Xie designed the experiments, collected data for the number of trajectory points trained, performed the characterization, modeling and wrote the first draft of the paper. Kun Zhang critically reviewed the method used and contributed to structuring the paper. Yi Zhao implemented the proposed prototype, ran the experiments for the performance study. Yunchun Zhang and Ying Cai collected data for experimental comparison usage and drew the curve. Tian Wang critically reviewed the paper and contributed to the improvement on paper writing. The author(s) read and approved the final manuscript.

Funding
This work was supported in part by the Open Research Project of The Hubei Key Laboratory of Intelligent Geo-Information Processing.

Availability of data and materials
Not applicable.

Competing interests
The authors declare that they have no competing interests.

Author details
[1]School of Computer Science and Engineering Nanjing University of Science and Technology, Nanjing, China. [2]Shanghai Changjiang Intelligent Data Technology Co., Ltd., Shanghai, China.

Received: 18 August 2020 Accepted: 14 December 2020
Published online: 25 January 2021

References

1. Wang T, Zhou J, Zhang G, Hu WTS (2020) Customer perceived value- and risk-aware multiserver configuration for profit maximization. IEEE Transactions on Parallel and Distributed Systems 31(5):1074–1088
2. Zhou J, Sun J, Zhang M, Ma Y, "Dependable scheduling for real-time workflows on cyber-physical cloud systems [J], IEEE transactions on industrial informatics, in press, 2020. DOI: https://doi.org/10.1109/TII.2020.3011506
3. Zhou J, Sun J, Cong P, Liu Z, Wei T, Zhou X, Hu S (2020) Security-critical energy-aware task scheduling for heterogeneous real-time MPSoCs in IoT. IEEE Trans Serv Comput 13(4):745–758
4. Qi L, Hu C, Zhang X, M. Khosravi R, Sharma S, Pang S, Wang T. privacy-aware data fusion and prediction with spatial-temporal context for Smart City industrial environment. IEEE Transactions on Industrial Informatics in press 2020
5. Wang L, Zhang X, Wang T, Wan S, Srivastava G, Pang S, Qi L (2020) Diversified and scalable service recommendation with accuracy guarantee. IEEE Transactions on Computational Social Systems. https://doi.org/10.1109/TCSS.2020.3007812
6. Wang L, Zhang X, Wang R, Yan C, Kou H, Qi L (2020) Diversified service recommendation with high accuracy and efficiency [J]. Knowl-Based Syst 204(27):106196
7. Chang G, Xu T, Chen C, Ji B, Li S (2019) Switching position and range-domain carrier-smoothing-code filtering for GNSS positioning in harsh environments with intermittent satellite deficiencies [J]. Journal of The Franklin Institute 356:4928–4947
8. Tank DM (2014) Improved Apriori algorithm for mining association rules. Int J Information Technology Computer Sci 6(7):15–23
9. Meng XF, Ding ZM (2009) Mobile data management: concepts and techniques [M], Tsinghua University press
10. Naranjo PGV, Shojafar M, Mostafaei H et al (2017) P-SEP: a prolong stable election routing algorithm for energy-limited heterogeneous fog-supported wireless sensor networks. Journal of Supercomputing 73(2):1–23
11. Naranjo PGV, Pooranian Z, Shojafar M et al (2019) FOCAN: A Fog-supported Smart City Network Architecture for Management of Applications in the Internet of Everything Environments. Journal of Parallel & Distributed Computing 135:274–283
12. Yaghmaee MH, Leon-Garcia A (2018) A Fog-Based Internet of Energy Architecture for Transactive Energy Management Systems. IEEE Internet of Things J 5(2):1055–1069
13. Cai ZG, Jiang SW, Zhang J et al (2017) A Unified Framework for Vehicle Rerouting and Traffic Light Control to Reduce Traffic Congestion. IEEE Transactions on Intelligent Transportation Systems 18(7):1958–1973
14. Cao ZG, Guo HL, Song W et al (2020) Using reinforcement learning to minimize the probability of delay occurrence in transportation. IEEE transactions on vehicular technology 69(3):2424–2436
15. Monreale A, Pinelli F, Trasarti R et al (2009) Where next:a location predictor on trajectory pattern mining [C]. ACM SIGKDD International Conference on Knowledge Discovery and Data Mining 637-646
16. Ying J C, Lee W C, Weng T C, et al. Semantic trajectory mining for location prediction [C], ACM Sigspatial international conference on advances in geographic information systems, ACM,34–43, 2010
17. Song M B, Ryu J H, Lee S K, et al. Considering mobility patterns in moving objects database [C], international conference on parallel processing, ACM, 597, 2003
18. Ishikawa Y, Tsukamoto Y, Kitagawa H (2004) Extracting mobility statistics from indexed Spatio-temporal datasets [C]. In: Spatio-temporal database management, international workshop Stdbm'04, Toronto, Canada, August, pp 9–16
19. Ma W, Liu M, Huang HB et al (2014) Constructing a City taxi movement probability model based on historical trajectory [J]. J National University of Defense Technology 36(03):129–134
20. Asahara A, Maruyama K, Sato A et al (2011) Pedestrian movement prediction based on mixed Markov-chain model. In: ACM Sigspatial International Symposium on Advances in Geographic Information Systems, pp 25–33
21. Killijian MO (2012) Next place prediction using mobility Markov chains, the workshop on measurement, privacy and mobility, ACM, 3
22. Qiao S J, Li TR, Han N, et al. "Self-adaptive trajectory prediction model for moving objects in big data environment [J] ACM Sigcomm computer communication review, 45(4): 609–610, 2015
23. Du Y, Wang C, Qiao Y et al (2018) A geographical location prediction method based on continuous time series Markov mode [J]. PLoS One 13(11):e0207063
24. Jensen C S, Lin D, Ooi B C, et al. Effective density queries on continuously moving objects [C], international conference on data engineering, 71–71, 2006
25. Li ZH, Tang JH (2015) Weakly Supervised Deep Metric Learning for Community-Contributed Image Retrieval. IEEE Trans. Multimedia 17(11): 1989–1999
26. Li ZH, Tang JH, Mei T (2019) Deep collaborative embedding for social image understanding. IEEE trans. On pattern analysis and Machine Intelligence 41(9):2070–2083

27. Kan SH, Cen YG et al (2019) Supervised Deep Feature Embedding with Hand Crafted Feature. IEEE Transactions on Image Processing 28(12):5809–5823
28. Ma C, Liu ZB, Cao ZG et al (2020) Cost-sensitive deep Forest for Price prediction. Pattern Recogn 107:107499
29. Liang Z, Zheng G, Li J (2012) Automatic parking path optimization based on Bezier curve fitting. In: IEEE International Conference on Automation and Logistics, pp 583–587
30. Rodriguez A, Laio A (2014) Clustering by fast search and find of density peaks [J]. Science 344(6191):1492–1496
31. Hu P, Wang YL, Gong B, Wang YJ, Li YC, Zhao RX, Li H, Li B (2020) A secure and lightweight privacy-preserving data aggregation scheme for internet of vehicles. Peer-to-Peer Networking and Applications 13:1002–1013
32. Hu P, Wang YL, Li QB, Wang YJ, Li QB, Zhao QB, Li H (2020) Efficient location privacy-preserving range query scheme for vehicle sensing systems. J Syst Archit 106:101714
33. Hu P, Wang YL, Xiao G, Zhou JL, Be G, Wang YJ (2020) An efficient privacy-preserving data query and dissemination scheme in vehicular cloud. Pervasive and Mobile Computing 101152
34. Wang Y, Yang Y, Han C et al (2019) LR-LRU: a PACS-oriented intelligent cache replacement policy [J]. IEEE Access 7:58073–58084
35. Grnwald PD, Myung IJ, Pitt MA (2005) Advances in minimum description length: theory and applications [M], the MIT press

Publisher's Note

Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.