sustainability

MDPI

Article

Vehicle Intersections Prediction Based on Markov Model with Variable Weight Optimization

Zhihui He [1,2], Lei Ning [1,2,*], Baihui Jiang [1,2], Jiajia Li [1,2] and Xin Wang [3]

[1] College of Big Data and Internet, Shenzhen Technology University, Shenzhen 518118, China; 2110416002@stumail.sztu.edu.cn (Z.H.)
[2] College of Applied Technology, Shenzhen University, Shenzhen 518118, China
[3] eSix Technology (Guangdong) Co., Ltd, Shenzhen 518052, China
* Correspondence: ninglei@sztu.edu.cn

Abstract: In this study, a new algorithm for predicting vehicle turning at intersections is proposed. The method is based on the Markov chain and can predict vehicle trajectories using GPS location sequences. Unlike traditional Markov models, which use preset weights, we created the Markov model using a data-driven weight selection method. The proposed model can dynamically adjust the weights of each intersection's influence on current trajectories based on the data, in contrast to the fixed weights in traditional models. The study also details how to process trajectory data to identify whether a vehicle has passed through an intersection and how to determine the adjacency relationship of intersections, thus providing a reference for implementing a model of the classification problem. The data-driven algorithm was applied and compared to the fixed-weight algorithm on the same trajectory dataset, and the superiority of the weight selection algorithm was proven. The prediction accuracy of the traditional method was 49.61%, while the proposed method achieved a prediction accuracy of 60.66% for 100,000 trajectory datasets, nearly an 11% increase. Volunteer participation in the second dataset collected on the university campus showed that the accuracy of the proposed method could be further improved to 79.31% as the GPS sampling frequency increased. Simulation results show that the algorithm provides accurate prediction and that the prediction effect is improved with the expansion of the trajectory data set and the increase in GPS sampling frequency. The proposed algorithm has the potential to provide a location-based optimization of network resource allocation.

Keywords: internet of things; cyber–physical systems; intersection prediction; Markov chain

Citation: He, Z.; Ning, L.; Jiang, B.; Li, J.; Wang, X. Vehicle Intersections Prediction Based on Markov Model with Variable Weight Optimization. *Sustainability* **2023**, *15*, 6943. https://doi.org/10.3390/su15086943

Academic Editor: Maxim A. Dulebenets

Received: 28 March 2023
Revised: 16 April 2023
Accepted: 17 April 2023
Published: 20 April 2023

1. Introduction

The vigorous development of communication technology and the Internet of Things (IoT) has brought us opportunities and challenges [1]. With the development of the Internet of Things, more and more vehicles begin to connect to the Internet, forming a huge network of car networking systems [2,3]. This technology facilitates the monitoring of environmental data, enabling extensive analysis, optimization, and control. Now we live in the era of car networking, thousands of connected computing devices surround us, which makes these devices more convenient and smarter [4,5]. The present rapid advancement of IoT owes much to its capacity to bolster diverse areas, including transportation and agriculture [6,7].

In the era of Industry 4.0, technological advancements have spurred the widespread adoption of IoT technology within the intelligence industry [8]. By enabling the integration of virtual computing systems with the physical environment of industry, IoT technology has engendered a paradigm shift and a novel technological paradigm, characterized by cyber–physical systems (CPSs) [9]. The realm of CPSs harbors numerous novel functionalities that can be actualized. For instance, in the transportation system, we stand to

gain considerably from enhanced embedded intelligence within automobiles [10]. Networked autonomous vehicles hold immense potential to elevate traffic efficiency, safety, and efficacy [11].

Presently, the quantity of connected vehicles situated at the Smart Connection Level is on the rise, and this trend shows no signs of abating. Concomitantly, the magnitude of trajectory data produced is also escalating [12,13]. In recent years, how to mine meaningful information from massive amounts of trajectory data has drawn the attention of scholars around the world [14]. In this study, we focus on one type of trajectory data, which is urban vehicle trajectory data. By urban vehicle trajectory data we mean the trajectory car data describing vehicle motion in an urban traffic network. In many applications of trajectory data mining, this study mainly focuses on the location prediction problem based on historical trajectory data, that is, analyzing a large number of vehicle trajectories moving in the city to predict the next location that they will arrive at. Vehicle trajectory prediction is an important branch of trajectory data mining; it aims to predict the next destination based on the initial trajectory segment [15]. This work is of great significance for trajectory-based user services and changing user lifestyles [16].

The task of vehicle trajectory prediction has been addressed in the literature from different perspectives. Therefore, a considerable number of vehicle trajectory prediction methods have recently been proposed. Here, we give an overview of the methods. In [17], the author proposed a new trajectory prediction method which combines trajectory prediction based on the Constant Yaw Rate and Acceleration motion model and a trajectory prediction based on maneuver recognition. In [18], a method which combines road recognition and the hypothesis of steady preview and dynamic correction for trajectory prediction has been proposed. In this algorithm, both methods of Kalman Filter and Recursive Least-Square work well to estimate the road slope and road friction coefficient. In [19], a vehicle trajectory prediction method based on motion model and maneuver model fusion with Interactive Multiple Model (IMM) was proposed. In the whole prediction range, this method not only has good prediction accuracy, but also has appropriate prediction uncertainty. In [20], the authors built the train control security state transition probability model under jamming attacks and proposed a cross layer defense scheme. Convolutional Neural Network (CNN) is used in [21], it proposed a new prediction algorithm, which models the trajectory as a two-dimensional image, and then feeds it into CNN architecture to extract multi-scale patterns for accurate destination prediction. In [22], the article presents a new collaborative approach for predicting vehicle trajectories. This approach incorporates a bivariate Gaussian model and a specially designed Kalman filter, leading to improved short- and long-term predictive performance.

The work in [23] considers a lane crossing and final points generation model-based trajectory prediction approach for preceding target vehicles, in which the key influence factors, such as the driver's intention and the mixed driving style, are included. The authors in [24] introduce and analyze trajectory prediction methods based on how they model the vehicles interactions. Inspired by human reasoning, they use an attention mechanism that explicitly highlights the importance of neighboring vehicles with respect to their future states. Wang in [25] states that we can extract valuable user behavior information through the trajectory data mining and use a trajectory prediction algorithm based on deep learning that fuses the convolution neural network and deep bidirectional long-term memory network to learn the local and global information of vehicle trajectory, so that we can accurately analyze the vehicle motion law and predict the vehicle motion trajectory in the future.

The prediction method proposed in other papers is based on Markov Chain (MC). The Markov-Based methods are based on the Markov property, which states that the probability of traveling to a future position depends only on the current one. In [26], the work proposed a Long Short-Term Spatio-Temporal Aggregation (LSSTA) network for human trajectory prediction. Compared to recurrent neural networks or convolutional neural networks, the model in the article has excellent scalability for long sequences, considering

not only fixed features but also dynamic interactions between pedestrians. Targeting on sparse historical trajectory data, an Individual Trajectory–Group Trajectory (ITGT) location prediction model by utilizing the pattern of group travels is proposed in [27]. It has achieved good prediction accuracy and good performance improvement. In [28], this paper proposed a real-time trajectory prediction method for ICV based on vehicle to object (V2X) communication, which considers more dynamic spatial environments and improves the accuracy of trajectory prediction. Ref. [29] proposed a Spatio-Temporal Multigraph Convolutional Network (STMGCN)-based trajectory prediction framework using the Mobile Edge computing (MEC) paradigm and it achieves excellent prediction performance.

In recent years, a methodology based on MC with weight algorithm, called Markov model-based Trajectory Prediction, which predicts the next location of a vehicle has been proposed [30], the scale of trajectory data involved is 100,000. It makes full use of historical trajectory information, the average prediction time is short and the accuracy is high. However, the weight is determined artificially and the data scale is small, which leads to the loss of prediction accuracy. In addition, ref. [30] does not describe the specific implementation methods, including how to judge whether the vehicle passes through an intersection and how to obtain the adjacent relationship of the intersection.

To summarize, our contributions are as follows: compared with the tradition Markov model with preset model, a data-driven weight selection method is used in this paper to create Markov model, which effectively improves the prediction accuracy and shortens the prediction time. In addition, we describes the specific implementation methods, including how to judge whether a vehicle passes through an intersection and how to obtain the adjacent relationship of the intersection. The trajectory prediction of the vehicle based on a huge trajectory dataset on the intersection will help the planning framework of the intersection, which will help alleviate traffic congestion problems and improve cross-intersection efficiency.

The rest of the paper is organized as follows. In Section 2, we introduce the MC and create the vehicle trajectory prediction model according to real vehicle data. Section 3 evaluates the performance of the model. The last section concludes the paper.

2. Methods

2.1. Markov Chain

MC can be applied to Monte Carlo method to form MC Monte Carlo. It can also be used for mathematical modeling of dynamic system, chemical reaction, queuing theory, market behavior, and information retrieval. MC or Markov process is a stochastic model that describes a series of possible events in which the probability of any event depends only on the state reached in the previous event. In math, it can be expressed as the following:

$$p(X_{t_n+k} = x \mid X_{t_1} = x_1, X_{t_2} = x_2, \ldots, X_{t_n} = x_n) = p(X_{t_n+k} = x \mid X_{t_n} = x_n) \quad (1)$$

In the above formula, $\{X_{t_n}, n = 0, 1, 2, \cdots\}$ is a sequence of random variables, x belongs to the discrete state space of the random sequence $\{X_{t_n}, n = 0, 1, 2, \cdots\}$, k is any natural number. If $\{X_{t_n}, n = 0, 1, 2, \cdots\}$ satisfies the expression above, we call it MC.

If t_n represents the current,$t_1, t_2, t_3 \ldots, t_{n-1}$ represent the past and t_{n+k} represents the future in the Formula (1), that indicates that the state x at time t_{n+k} in the future only depends on the current state x_n at time tn, and has no relation to the past state at time $t_1, t_2, t_3 \ldots t_{n-1}$, we call this feature Markov property or non-after-effect property.

There are two commonly used Markov chains: one is called a homogeneous Markov Chain (HMC) and the other is called a non-homogeneous Markov chain (NMC). The main difference between them is that the former has the same k-step transition probability matrix at any time, while the latter's k-step transition probability matrix changes with time. This paper mainly discusses HMC.

In MC theory, $p(X_{t_n+k} = x \mid X_{t_n} = x_n)$ is called k-step transition probability at time t_n. The transition probability represents the probability of being in state x at time t_n and

transferring to the state x_n at time x_n after k time units. If the MC does not depend on n and is only related to the initial state x, the final state x_n and the step k, the MC is called a HMC and the k-step transition probability can be regarded as $p_{x_n x}(k)$. It can be expressed as:

$$p_{x_n x}(k) = p_{x_n x}(n, n+k) = p(X_{t_n+k} = x \mid X_{t_n} = x_n), k > 0 \tag{2}$$

In Formula (2), $0 \leq p_{x_n x}(k) \leq 1, \sum_{x \in E} p_{x_n x}(k) = 1$.

According to the operational definition of the transition matrix of the homogeneous MC, there is the following formula:

$$P = \begin{bmatrix} p_{11} & p_{21} & \cdots & p_{n1} \\ p_{12} & p_{22} & \cdots & p_{n2} \\ \vdots & \vdots & & \vdots \\ p_{1n} & p_{2n} & \cdots & p_{nn} \end{bmatrix} \overset{k \text{ times}}{\Rightarrow} P(k) = \begin{bmatrix} p_{11}^k & p_{21}^k & \cdots & p_{n1}^k \\ p_{12}^k & p_{22}^k & \cdots & p_{n2}^k \\ \vdots & \vdots & & \vdots \\ p_{1n}^k & p_{2n}^k & \cdots & p_{nn}^k \end{bmatrix} \tag{3}$$

2.2. Data-Driven Weight Selection Markov Model

2.2.1. MC in Model

The goal of this part is to predict the next target intersection when the vehicle turns at the intersection, based on its past trajectories and the past trajectories of other vehicles here. Intersection prediction implies forecasting the movement of objects between intersections. The intersection of the predicted object is the present state of the predicted object in MC. Of course, the predicted intersection can also be some iconic edifices or abstract destinations, such as the transition between administrative regions, which are also the transmutation of object state.

Vehicle trajectory composed of m number of data points of longitude(x) and latitude(y) is expressed as $l = [(x_1, y_1), (x_2, y_2), \ldots, (x_m, y_m)]$.In urban traffic network, trajectory prediction is actually only needed at intersections. At this time, the vehicle trajectory composed of longitude and latitude can be transformed into a directed sequence composed of k number of junctions, represented as $l = [j_1, j_2 \ldots, j_k]$, as shown in the Figures 1 and 2.

The vehicle trajectory is defined as ordered sequence of intersections, so the junction transfer problem can be regarded as the state transfer problem. In this paper, it is assumed that the turning probability of each intersection is the same at any time, The vehicle trajectory sequence satisfies the Markov property and it is a HMC.

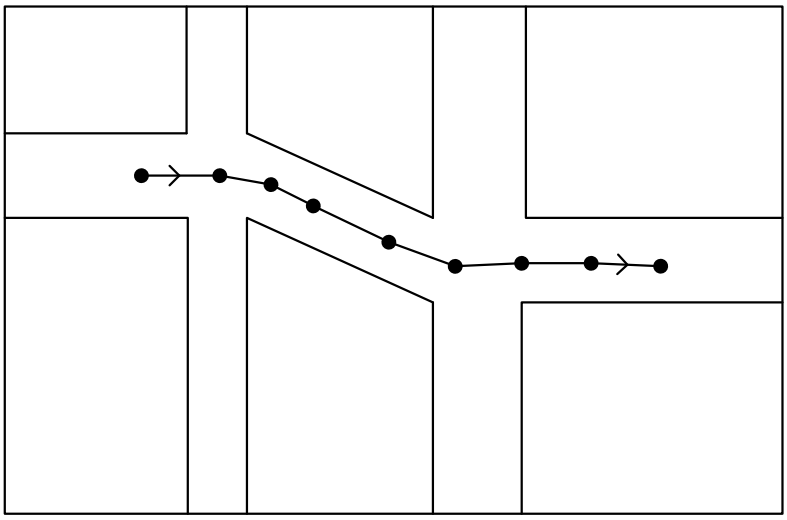

Figure 1. Schematic diagram of original trajectory data.

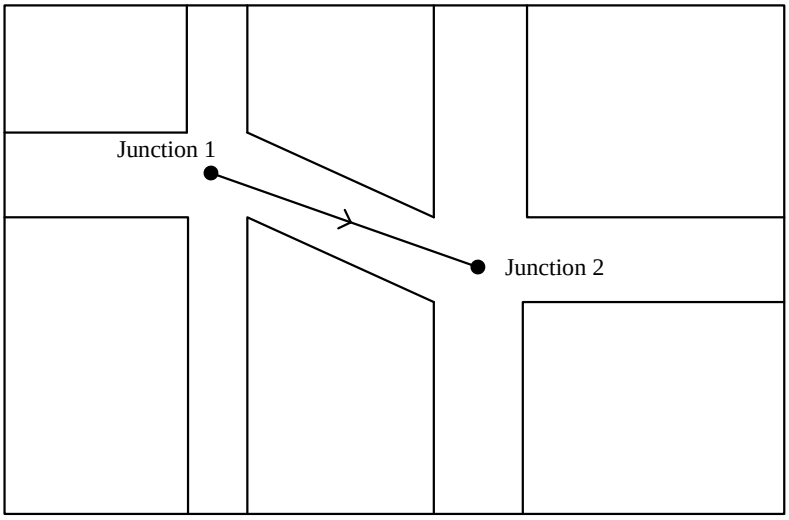

Figure 2. Schematic diagram of converted trajectory data.

The vehicle trajectory prediction at the intersection is jointly determined by the historical records of the current vehicle and other vehicles before. In this model, the one-step transition probability is the probability of vehicles transferring from the current intersection to other intersections. If we suppose there are currently n intersections, numbered from 1 to n, the one-step transition probability can be calculated by the following formula:

$$p_{ij} = \frac{S_{ij}}{\sum_{j=1}^{n} S_{ij}} \quad (1 \leq i, j \leq n) \tag{4}$$

In the above formula, p_{ij} is the the probability of vehicles transferring from the intersection i to intersection j. S_{ij} the number of times the vehicle is transferred from intersection i to intersection j.

Obviously, the closer to the current state, the greater the impact on the decision-making of the next transfer intersection, so the historical trajectory of k steps can be retained, and the impact of historical states other than k steps on the decision-making can be ignored. Based on MC, the predicted next intersection can be obtained by weighting. The formula for calculation is as follows:

$$A(t) = w_1 I(t-1)P + w_2 I(t-2)P(2) + \cdots + w_k I(t-k)P(k) \tag{5}$$

In Formula (5), t presents the moment of the next junction, $t-1$ is the moment of the previous junction of the next junction, and so on. $I(t-i)$ is the state of the first i intersection of the next intersection at time t. $a_1, a_2..a_k$ stand for the weight values. The weight values indicate that the degree of influence of the previous three intersections on the current decision-making countermeasures. $w_1 \geq w_2 \geq w_3 \geq ... \geq w_k$. $A(t)$ is a matrix with one row and N columns of the the predicted value of the current junction turning to each junction. If A_n is the column n in $A(t)$, $A_n$represents the predicted value of the current intersection transferred to intersection n.

2.2.2. Weight Selection Algorithm and Associated Algorithms

In the previous research work, w in Formula (5) is determined by experience, and it is the empirical value. In this paper, a weight determination method based on trajectory big data is proposed. We use the historical vehicle movement trajectory in the previous area, constantly adjust the weight parameters, record the training prediction results, and take a group of weights with excellent training results for the prediction of the test set.

In the original algorithm, the matrix weighting used to predict trajectories is fixed and determined by experience. Obviously, this has its shortcomings. When facing different

movement trajectory models in different regions, using a fixed weighting will inevitably lead to a decrease in prediction accuracy. The trajectory prediction formula used is shown as Formula (6).

$$A(t) = 4I(t-1)P + I(t-2)P(2) + 0.25I(t-k)P(3) \quad (6)$$

According to the historical trajectory and life experience, when k is greater than 3, the historical intersection choice has little impact on the current intersection choice, k in Formula (3) is set to 3 in this paper. According to previous studies, The best interval of w_1, w_2, w_3 can be obtained, $1 \leq w_1 \leq 10, 1 \leq w_2 \leq 5, 0.1 \leq w_3 \leq 1$. Since keeping the weight value to two decimal places does not improve the prediction accuracy and actually increases the operating cost, the weight values for w_1, w_2, w_3 are discrete values. w_1 = 1,2,...,10, w_2 = 1, 2, ..., 5, w_3 = 0.1, 0.2, ..., 1.0. The size of the search space is 500. Therefore, the initial value of the weight is set, $w_1 = 1, w_2 = 1, w_3 = 0.1$. Supported by the huge historical trajectory data, the selected weight will make the prediction accuracy high. The specific process is shown in the Figure 3.

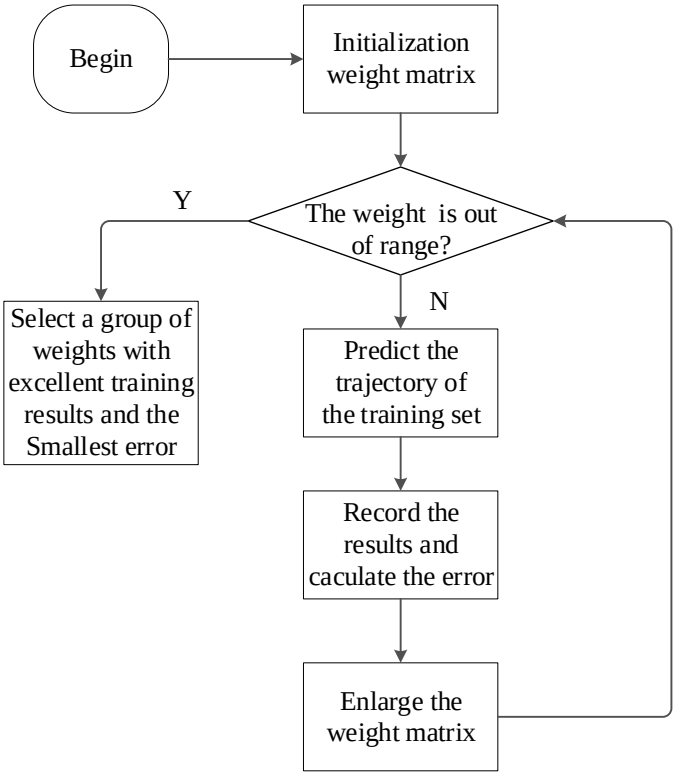

Figure 3. Main framework of data-driven weight selection method.

In accordance with $A(t)$, we set the corresponding items that are not adjacent to the intersection where the vehicle is currently situated to 0, then the next intersection predicted is the one with the highest predicted value among the adjacent intersections.

Therefore, using the existing historical trajectory statistics, the k-step transfer matrix can be obtained, as shown in Algorithm 1. Therefore, using the existing historical trajectory statistics, the k-step transfer matrix can be obtained, as shown in Algorithm 1.

Algorithm 1 Get P(k)

Require: Transaction Database L; Intersection Database I, contains historical trajectory data and involves the longitude and latitude coordinates of the intersection; number of the intersection SN.
Ensure: the k-step transfer matrix$P(k)$

```
Scan database L once;
Get the Number of trajectory sequences TN
for i ← 1 to SN do
  for j ← 1 to SN do
    for n ← 1 to TN do
      if the vehicle passes through two intersections i and j in sequence in l_n then
        S_ij ++;
        Sum+ = S_ij;
      end if
    end for
  end for
end for
Calculate the one-step transition probability p using (4);
Calculate the k-step transition probability matrix P(k) using (3);
```

In Algorithm 1, transaction database L should include the timestamp, GPS latitude and longitude coordinates, and license plate number. The intersection database I consists of the latitude and longitude coordinates of all intersections within the area.

In Algorithm 2, firstly, by arranging the trajectory data according to the timestamp, the trajectory sequence is obtained. If one of the trajectory points is within distance d of an intersection, the vehicle is judged to have passed through the intersection.

Algorithm 2 Judge whether the vehicle passes the intersection

Require: The current intersection of the vehicle J; Transaction Database L; Intersection Database I, contains historical trajectory data and involves the longitude and latitude coordinates of the intersection,the number of the intersection SN.
Ensure: Whether the vehicle passes the intersection

```
Scan Transaction database L and Intersection Database I once
for i ← 1 to SN do
  Filter out the data points near the intersection according to the distance d between the vehicle and the intersection i
  if The filtered vehicle data points belong to the current vehicle then
    Vehicles passing through the current intersection i
  end if
end for
```

In Algorithm 3, to derive the predicted turning intersection J, we assign a zero to the matrix element that is not adjacent to the current intersection, select the matrix's maximum element, and identify the intersection related to the maximum element's position in the matrix.

Algorithm 3 Function Get_predicted intersection

Require: The current intersection of the vehicle J; Transaction Database L; number of the intersection SN
Ensure: the predicted turning intersection J'

```
Scan database L once, and determine I(t), the matrix of the state of the vehicle;
Calculate A(t) using (5);
Select the maximum element A_max from the A(t);
A_max → J';
```

In Algorithm 4, according to the dense data points and the known intersection longitude and latitude information, create an intersection relationship matrix. According to the timestamp information in the vehicle trajectory, combined with Algorithm 2, it can be concluded that the intersections that pass successively must be adjacent, the corresponding value of adjacent intersections in the matrix is set to 1, and the non-adjacent value is set to 0.

Algorithm 4 Get the intersection relation matrix

Require: Transaction Database L,Intersection Database I, the number of the vehicle VN
Ensure: the Intersection relation matrix R
1: Scan Transaction database L and Intersection Database I once
2: **for** $i \leftarrow 1$ **to** VN **do**
3: Using the time stamp in L and algorithm 2, sort the intersections in time order
4: With the obtained sequence, assign values to adjacent intersection values corresponding to the matrix
5: **end for**

3. System Architecture of Prediction Platform

The trajectory big data prediction platform is established through Cloudera's Distribution Including Apache Hadoop (CDH). The platform will adopt a lambda architecture. Lambda architecture integrates offline computing frameworks and real-time computing frameworks. The immutable model is used to store data, so that the task becomes traceable, which is convenient for independent analysis of trajectory data in different time stages. In massive stream data processing. Recalculation is the main factor affecting the system performance. For example, when the code changes, you can only recalculate all the data. The incremental calculation of lambda architecture can well avoid the recalculation of a large amount of stream data to improve system performance. In addition, lambda architecture adopts the basic principle of read-write separation to separate the read and write functions, so as to isolate the complexity of system design and simplify the system. In order to meet the calculation and analysis requirements of trajectory traffic data, the components adopted include

(1) Mysql, a relational database management system developed by Mysql AB company in Sweden.
(2) Hadoop Distributed File System (HDFS) [31], a distributed file system designed to run on common hardware.
(3) Spark [32], a distributed open source processing system for big data workloads.
(4) Yarn [33], a resource coordination management system provides a trajectory data analysis, processing, and development environment.
(5) ZooKeeper [34], a centralized service for maintaining configuration information, naming, providing distributed synchronization, and providing group services.

The platform data link is shown in Figure 4. In Figure 4, the traffic trajectory data of Shenzhen is temporarily stored in Mysql. Since Mysql is very slow in transmitting data to spark, the traffic trajectory data are first written into HDFS from Mysql and distributed to each datanode through the copy mechanism of HDFS. When the trajectory is predicted, spark obtains the trajectory data from HDFS for parallel calculation, and the calculation results are written back to HDFS.

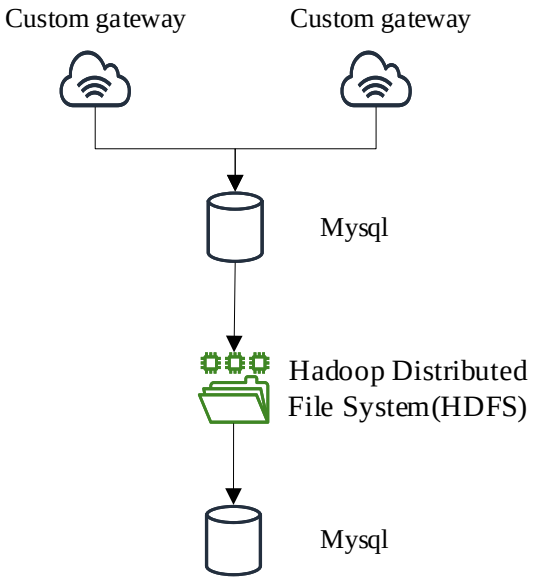

Figure 4. The trajectory big data prediction platform data link.

4. Trajectory Prediction of Shenzhen Traffic Data

4.1. Experimental Environment and Dataset

This platform uses VMware ESXi to build three virtual machine clusters. The specific server configuration is shown in the Table 1 below. Through this server, three nodes are virtualized, and big data components are built through CDH. The versions of big data components used by the platform are shown in Table 2. Our programs are written in Python. The dataset was downloaded from the Shenzhen Municipal Government Data Open Platform. The dataset contains various items of vehicle data, such as plate number, map latitude and longitude, GPS time, speed, and so on. It covers the three-month driving trajectories of operating vehicles in Shenzhen in 2018, with a total of 2,133,696 trajectories.

Table 1. Server configuration information.

Type	Specific Model
CPU	Intel(R) Xeon(R) Gold 6230R CPU @ 2.10GHz
Memory	DRAM DDR-4 2933MHz
Hard Disk	Dell Express Fla
Network Card	Broadcom Adv. Dual 10Gb Ethernet
Operating System	CentOs Linux release 7.9.2009(Core)

Table 2. Big data component version.

Name	Version
Hadoop	3.0.0-cdh6.3.2
Zookper	3.4.5-cdh6.3.2–1
Spark	2.4.0-cdh6.3.2

The trajectory dataset used in this study is obtained from the Shenzhen Transportation Bureau and comprises raw GPS data from up to five types of urban operating vehicles. The dataset spans one week, from 00:00 a.m. on 8 October 2018 to 11:59 p.m. on 14 October 2018 in the local time. In total, the dataset contains 113,503 entries, representing 29,218 vehicles. Each trajectory in the dataset is characterized by an extensive number of raw GPS coordinates, with one typical trajectory containing 295,966,347 data points. The data includes attributes such as license plate number, map longitude, latitude, timestamp, altitude, and GPS speed.

The document contains the trajectory data of vehicles, where each row represents a set of trajectory data. In each set of data, fields are separated by colons. As an example, a single set of trajectory data is listed as 68,269,849:13,839,354:20,230,301/68,266,747:13,840,980. The data can be interpreted as follows: (0) Map longitude, which is the longitude after correction, is divided by 60,000 to obtain the longitude in the WGS 84 coordinate system; (1) Map latitude, which is the latitude after correction, is divided by 60,000 to obtain the latitude in the WGS 84 coordinate system; (2) GPS time; (3) GPS longitude; and (4) GPS latitude.

4.2. Data Preprocessing

Trajectory preprocessing plays a vital role in numerous trajectory data mining tasks. This step involves solving various problems, including filtering out noise, compressing trajectories, segmenting them, detecting vehicles, and matching them to maps. Filtering out noise is a crucial step to eliminate imprecise points that arise from less than optimal signal quality in location positioning systems. This step guarantees that the trajectory data are dependable and precise. Another approach is trajectory compression, which decreases the size of a trajectory while retaining its usefulness, facilitating effective storage and processing of extensive trajectory datasets. Stay point detection can detect positions where a moving object has remained for an extended period, carrying semantic meaning, such as stopovers or points of interest.

Vehicle detection technology involves systems or methods applied to identify the position, orientation, or movement of vehicles within a specific area or environment. Tracking technology, such as vehicle kinematics and multi-sensor fusion, can be utilized to estimate parameters, such as sideslip angle [35], velocity error, and yaw misalignment [36], resulting in the improvement of the raw trajectory data's accuracy. Trajectory segmentation divides the trajectory into smaller fragments based on time interval, spatial shape, or semantic meanings, which can aid in extracting meaningful patterns or insights from the data. Map matching is sometimes used in track preprocessing, which aims to accurately project track points onto corresponding road sections, so that track data can be accurately analyzed under the background of road network [37].

In this article, the trajectory dataset was processed as follows.

1. Detection and removal of zero values in the data, where continuous zero values in a trajectory were deleted, and discrete zero values were interpolated using the average of the nearest k data points.
2. Detection and removal of duplicate data points in the trajectory data based on distance or time matching.
3. Filtering of trajectories outside the Guangdong province region based on latitude and longitude, identifying them as abnormal trajectory points and removing them.
4. Detection of noise points caused by stationary drift in the trajectory, based on Euclidean distance and time interval between consecutive points. If the average speed between two points exceeded 150 km/h, it was considered as an abnormal trajectory point. Similarly, for a few scattered noise points, we applied averaging and fitting correction based on neighboring points. For a large number of continuous noise points, the entire trajectory was discarded.

4.3. Results

In this section, we analyze the performance of the model from three aspects: (1) the amount of track data involved in training; (2) the distance d to judge whether the vehicle passes the intersection; (3) the order of the transfer matrix, that is, the total number of intersections involved in the calculation. In this simulation, we split the dataset into a train (90%) set and a test (10%) set.

Through the model calculation, the turning prediction of vehicles at the current intersection can be obtained. By comparing the prediction results with the actual results, we

can obtain the prediction accuracy, and use this as the standard to compare the performance of algorithms. The accuracy of the model is defined as follows:

$$Acc = \frac{Sum_c}{\sum_{i=1}^{n} S_i} \quad (1 \leq i \leq n) \tag{7}$$

In the Formula (7), Sum_c represents the same sum of predicted turning intersections and actual turning intersections. $\sum_{i=1}^{n} S_i$ is the total number of predictions.

Figure 5 shows that with the increase in the amount of trajectory data involved in training, the prediction accuracy of the model has increased significantly, showing a steady upward trend. In Figure 5, the data involved in model calculation is split into a train (90%) set and a test (10%) set. The order of the transfer matrix are both 5700. The specific parameters of the model are shown in the Table 3. As the amount of trajectory data are related to the calculation of transition probability, the more the amount of trajectory data, the closer the first-order transition probability will be to the real probability of turning to each intersection, so the accuracy of trajectory prediction will increase with the increase in the amount of trajectory data.

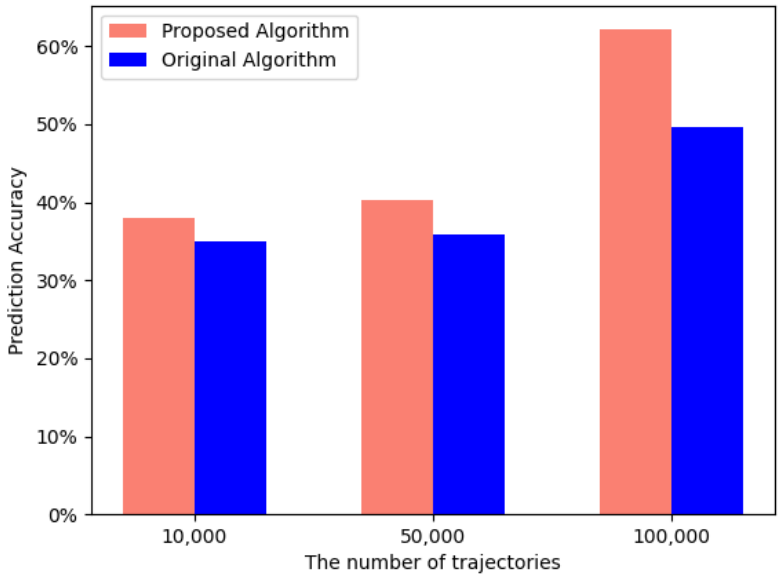

Figure 5. Influence of trajectory number on prediction accuracy.

The algorithm used in [30], adopts fixed weights, which is based on people's experience. The weight array in [30] is {4, 1, 0.25}. It can be seen that the weights obtained in this paper are basically different from those in [30], and the prediction accuracy is higher than that of the algorithm used in the first article. In addition, the first article involves at most 300 intersections, and the dataset of this article contains at most 5700 intersections, which is much larger than the first article. As shown in the Figure 6 and 7, it is a map of Shenzhen marked with the number of different intersections. From Figure 6 and 7, it can be seen that there is a broad area covered by 300 and 5000 cross intersections. When 300 crossings cover only one district in Shenzhen city, about the whole Shenzhen city can be covered by 5000 crossings.

In Figure 8, with the increase in d, the prediction accuracy fluctuates between 33% and 38%, and the upper and lower ranges are only 5%. The specific parameters of the model are shown in the Table 4. It can be seen from Table 4 that the value of d has affected the training of weight array, so it is necessary to select a reasonable value of d. If the value of d is too small, it will lead to vehicles passing through the intersection without being recorded, which will distort the transition probability and reduce the prediction accuracy. If the value of d is too large, it may be greater than the distance between intersections, which will also

lead to inaccurate transfer probability and reduce the accuracy of prediction. According to the characteristics of the dataset and the sampling interval of the trajectory dataset, it is reasonable to select 20 m as the value of d.

Table 3. The specific parameters of model.

Amount of Data	The Order of the Transfer Matrix k	Optimal Weight $\{w_1, w_2.w_3\}$	The Distance d (m)
1000	5700	$\{6, 2, 0.1\}$	20
5000	5700	$\{3, 1, 0.3\}$	20
10,000	5700	$\{9, 1, 0.1\}$	20
50,000	5700	$\{9, 1, 0.3\}$	20
100,000	5700	$\{7, 2, 0.1\}$	20

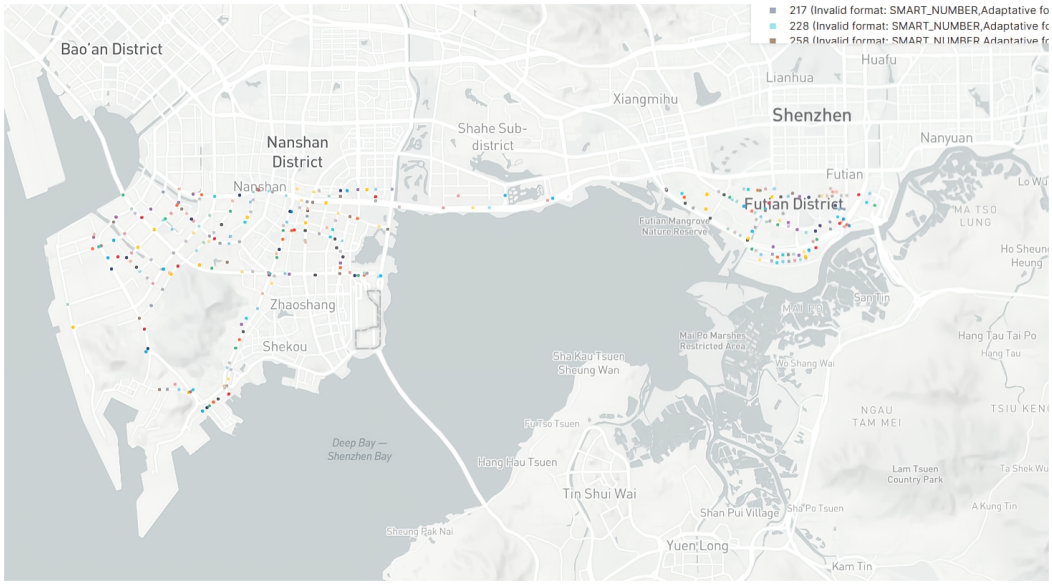

Figure 6. A map of Shenzhen marked with 300 intersections.

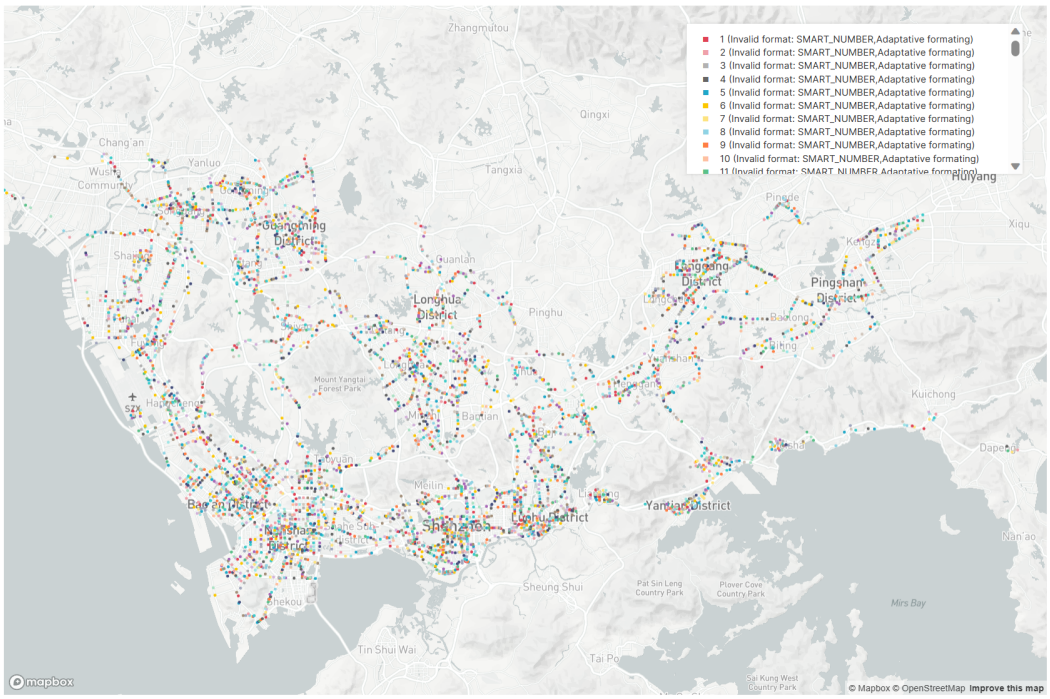

Figure 7. A map of Shenzhen marked with 5000 intersections.

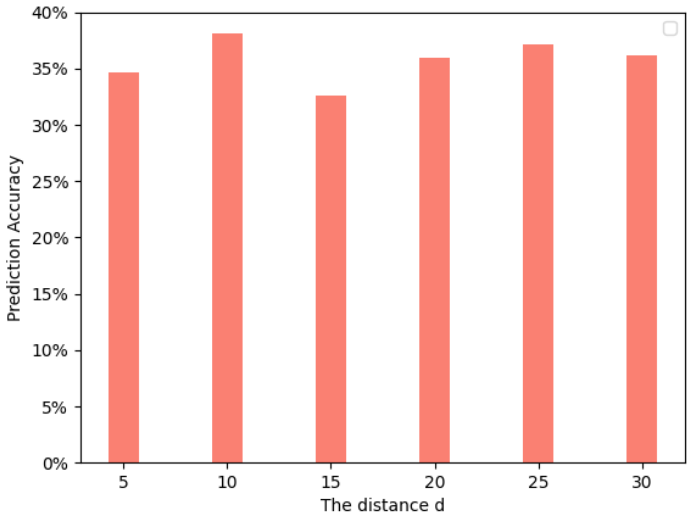

Figure 8. Influence of the distance on prediction accuracy.

Table 4. The specific parameters of model.

Amount of Data	The Order of the Transfer Matrix k	Optimal Weight $\{w_1, w_2.w_3\}$	The Distance d (m)
50,000	5700	$\{9, 1, 0.1\}$	5
50,000	5700	$\{8, 1, 0.6\}$	10
50,000	5700	$\{9, 1, 0.1\}$	15
50,000	5700	$\{9, 1, 0.3\}$	20
50,000	5700	$\{8, 3, 0.4\}$	25
50,000	5700	$\{7, 2, 0.1\}$	30

In Figure 9, with the increase in transfer matrix order, the prediction accuracy does not change linearly. The minimum prediction accuracy is 49.12% when the order is 3000. When the order is 4000, the maximum prediction accuracy is 53.21%. The specific parameters of the model are shown in the Table 5. When the amount of trajectory data is constant, the order of transfer matrix will not significantly affect the prediction accuracy, and the order of transfer matrix will significantly affect the calculation time of prediction. Because the calculation involves the multiplication of higher—order matrices. As the amount of track data increases, the total number of intersections involved will also increase, so the order of the transfer matrix will also increase.

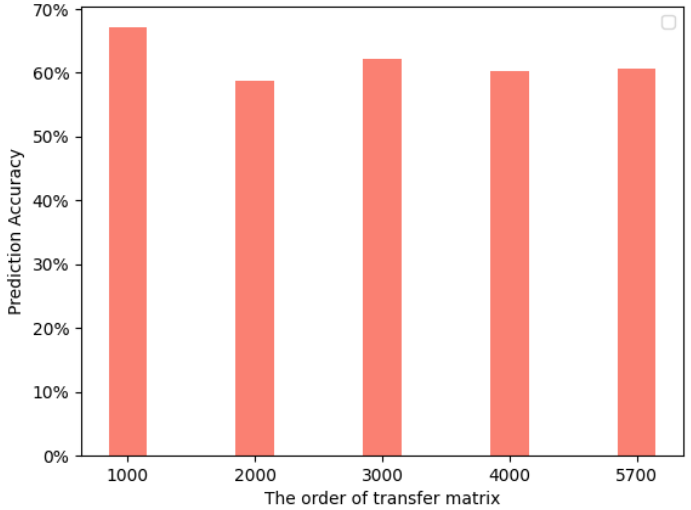

Figure 9. Influence of order of the transfer matrix on prediction accuracy.

Table 5. The specific parameters of model.

Amount of Data	The Order of the Transfer Matrix k	Optimal Weight $\{w_1, w_2.w_3\}$	The Distance d (m)
100,000	1000	$\{6, 1, 0.1\}$	20
100,000	2000	$\{6, 1, 0.2\}$	20
100,000	3000	$\{7, 1, 0.4\}$	20
100,000	4000	$\{7, 1, 0.1\}$	20
100,000	5700	$\{7, 2, 0.3\}$	20

4.4. Discussion and Summary

The above experimental results demonstrate that the extent of prediction accuracy is contingent upon the size of the dataset, achieving up to 60.66%. This is due to the fact that the sampling frequency of open traffic data in Shenzhen is relatively low, with the average GPS sampling rate being only once per minute, resulting in a track that is too coarse, with an average speed of 60 km per hour and a distance between consecutive points that is too great, thus impeding the stability of the model's prediction. In the following section, we will utilize the track data of college students' daily commuting to make predictions. The average GPS sampling frequency is once every one seconds.

5. Trajectory Prediction of Commuting of College Students

5.1. Experimental Environment and Dataset

The experimental environment is consistent with the previous part. The dataset comprises the daily activities of the volunteers on campus and the GPS coordinates gathered through the mobile GPS collection application. There are approximately 10,000 entries in the dataset.

5.2. Results

In this section, we analyze the performance of the model from The amount of track data involved in training. In this simulation, we split the dataset into a train (90%) set and a test (10%) set. Randomly partition five training and test sets to calculate the prediction accuracy rate, taking the mean value as the prediction accuracy.

Figure 10 illustrates that, as the amount of track data incorporated into the training increases, the prediction accuracy of the model is significantly enhanced, exhibiting a steady upward trend. The maximum prediction accuracy reaches 79.31%. The specific parameters of the model are outlined in Table 6. In comparison to Figure 5, the prediction accuracy of the same data volume is only 38.7%, indicating that the higher the sampling frequency, the more consistent the trajectory, the more continuous the state transition, and the better the model stability.

Table 6. Big data component version.

Amount of Data	Optimal Weight $\{w_1, w_2.w_3\}$	The Distance d (m)
1000	$\{2, 1, 0.1\}$	20
5000	$\{6, 2, 0.1\}$	20
10,000	$\{7, 3, 0.3\}$	20

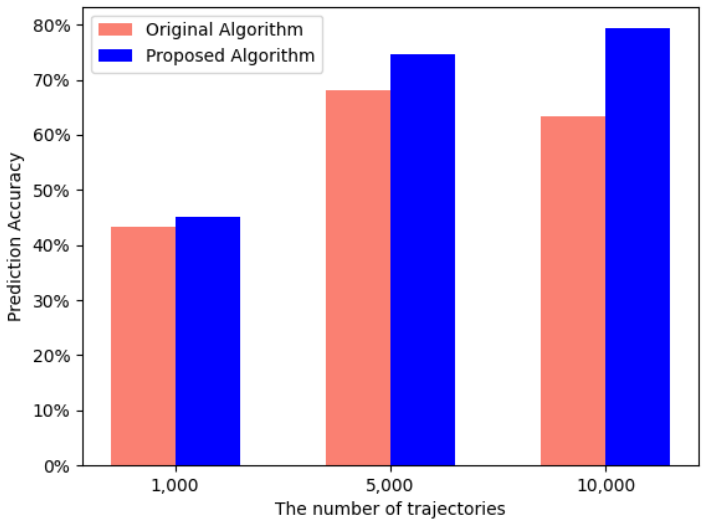

Figure 10. Influence of trajectory number on prediction accuracy.

5.3. Discussion and Summary

The results demonstrate that when the dataset size is increased, GPS sampling frequency can render the trajectory consistent, with the highest prediction accuracy reaching 79.31%.

Going forward, we will amalgamate the predicted track position information with the predicted signal strength, infer the current user's mobile phone signal strength by predicting the user's position information, and enhance the network retransmission algorithm. When the user's signal is about to enter a substandard environment, adjust the network packet retransmission mechanism, optimize the network allocation algorithm, and thereby attain more efficient network resource allocation.

6. Conclusions

A novel vehicle turning prediction at intersection algorithm is proposed. The algorithm utilizes the historical information and massive data in the trajectory dataset, uses a new weight selection algorithm, and compares the weight selection algorithm and the fixed weight algorithm in the same trajectory dataset. The prediction results show that the weight selection algorithm performs better. For 100,000 trajectory data, the prediction accuracy of proposed method is 60.66%, while the original method is only 49.61%.

Then, we re-selected the dataset for prediction, and recruited volunteers to collect the trajectory of volunteers in the university campus. The GPS sampling frequency was once every two seconds. The results demonstrate that when the dataset size is increased, GPS sampling frequency can render the trajectory consistent, with the highest prediction accuracy reaching 79.31%.

The simulation results demonstrate that the algorithm has a good prediction accuracy which increases with the expansion of the trajectory dataset. The results indicate that the prediction accuracy of the proposed algorithm is higher than the traditional one.

7. Further Research

Future work could combine signal prediction to conduct research. By modeling the moving objects, it is possible to predict the trajectory while also predicting the network signal. This will enable knowledge of the future network environment for the moving objects. With advanced knowledge of the network environment, the transmission mechanism can be adjusted at the network level to save network resources and achieve lower power consumption in the IoT system. However, future work should take into account the potential impact on computing resources and the amount of data. This could be achieved by

combining federated learning methodology to perform trajectory prediction. Furthermore, future research could improve the trajectory prediction model by considering two aspects: adopting more efficient trajectory preprocessing methods and adding more variables affecting prediction through machine learning. These improvements can enhance the accuracy of predictions.

In future research, it is recommended to explore the possibility of combining advanced optimization algorithms, such as hybrid heuristic and meta-heuristic adaptive algorithms, to optimize decision-making in this study. This will enable researchers to compare the efficiency and effectiveness of these algorithms with that of heuristic and meta-heuristic algorithms. Advanced optimization algorithms have proven useful in a variety of fields, including online learning, scheduling, multi-objective optimization, transportation, medicine, and data classification. In [38], The article proposes a universal island-based meta-heuristic algorithm that utilizes multiple types of meta-heuristic algorithms to address the container scheduling problem in maritime transportation. Ref. [39] proposed a learning-based algorithm that can adjust strategies to match problem features. The experimental results demonstrate that the proposed algorithm achieves satisfactory performance.The proposed algorithm outperforms several single-solution-based meta-heuristic algorithms in terms of solution quality, as it covers different regions of the search space with its diverse algorithms.

Author Contributions: Conceptualization , Z.H. and L.N.; methodology, Z.H.; software, Z.H. and J.L.; validation, Z.H. and J.L.; formal analysis, Z.H. and L.N.; investigation, Z.H.; resources, J.L.; data curation, B.J.; writing—original draft preparation, Z.H.; writing—review and editing, Z.H.; visualization, J.L.; supervision, L.N.; project administration, L.N.; funding acquisition, X.W. All authors have read and agreed to the published version of the manuscript.

Funding: This work was sponsored by General Program of Continuous Support Foundation of Shenzhen City (No. 20220715114600001), SZTU-Winoble Cooperation Research Project (No. 2021010802015), and Scientific Research Capacity Improvement Project from Guangdong Province (No.2021ZDJS109).

Data Availability Statement: The data used to support the findings of this study are available from the corresponding author upon request.

Acknowledgments: The authors would like to thank the anonymous reviewers for their valuable comments.

Conflicts of Interest: The authors declare no conflicts of interest related to this work.

Abbreviations

The following abbreviations are used in this manuscript:

IOT	Internet of Things
CPSs	Cyber–Physical Systems
CNN	Convolutional Neural Network
MC	Markov Chain
HMC	Homogeneous Markov chain
HMM	Hidden Markov Model
ITGT	Individual Trajectory-Group Trajectory
MEC	Mobile Edge computing
STMCN	Spatio-Temporal Multigraph Convolutional Network
CDH	Cloudera's Distribution Including Apache Hadoop
HDFS	Hadoop Distributed File System

References

1. Li, Y.; Zhu, L.; Wang, H.; Yu, F.R.; Liu, S. A Cross-Layer Defense Scheme for Edge Intelligence-Enabled CBTC Systems Against MitM Attacks. *IEEE Trans. Intell. Transp. Syst.* **2021**, *22*, 2286–2298. [CrossRef]
2. Tataria, H.; Shafi, M.; Molisch, A.F.; Dohler, M.; Sjöland, H.; Tufvesson, F. 6G Wireless Systems: Vision, Requirements, Challenges, Insights, and Opportunities. *Proc. IEEE* **2021**, *109*, 1166–1199. [CrossRef]
3. Yang, F.; Wang, S.; Li, J.; Liu, Z.; Sun, Q. An overview of Internet of Vehicles. *China Commun.* **2014**, *11*, 1–15. [CrossRef]
4. Xu, W.; Zhou, H.; Cheng, N.; Lyu, F.; Shi, W.; Chen, J.; Shen, X. Internet of vehicles in big data era. *IEEE/CAA J. Autom. Sin.* **2018**, *5*, 19–35. [CrossRef]

5. Zhou, H.; Xu, W.; Chen, J.; Wang, W. Evolutionary V2X Technologies Toward the Internet of Vehicles: Challenges and Opportunities. *Proc. IEEE* **2020**, *108*, 308–323. [CrossRef]
6. Lezoche, M.; Hernandez, J.E.; Alemany Diaz, M.d.M.E.; Panetto, H.; Kacprzyk, J. Agri-food 4.0: A survey of the supply chains and technologies for the future agriculture. *Comput. Ind.* **2020**, *117*, 103187. [CrossRef]
7. Motlagh, N.H.; Mohammadrezaei, M.; Hunt, J.; Zakeri, B. Internet of Things (IoT) and the Energy Sector. *Energies* **2020**, *13*, 494. [CrossRef]
8. Oztemel, E.; Gursev, S. Literature review of Industry 4.0 and related technologies. *J. Intell. Manuf.* **2020**, *31*, 127–182. [CrossRef]
9. Pivoto, D.G.S.; de Almeida, L.F.F.; Righi, R.d.R.; Rodrigues, J.J.P.C.; Lugli, A.B.; Alberti, A.M. Cyber-physical systems architectures for industrial internet of things applications in Industry 4.0: A literature review. *J. Manuf. Syst.* **2021**, *58*, 176–192. [CrossRef]
10. Morgan, J.; Halton, M.; Qiao, Y.; Breslin, J.G. Industry 4.0 smart reconfigurable manufacturing machines. *J. Manuf. Syst.* **2021**, *59*, 481–506. [CrossRef]
11. Rahman, M.A.; Hossain, M.S.; Showail, A.J.J.; Alrajeh, N.A.A.; Ghoneim, A. AI-Enabled IIoT for Live Smart City Event Monitoring. *IEEE Internet Things J.* **2023**, *10*, 2872–2880. [CrossRef]
12. Zheng, Y. Trajectory Data Mining: An Overview. *ACM Trans. Intell. Syst. Technol.* **2015**, *6*, 1–41. [CrossRef]
13. de Almeida, D.R.; Baptista, C.d.S.; de Andrade, F.G.; Soares, A. A Survey on Big Data for Trajectory Analytics. *ISPRS Int. J.-Geo-Inf.* **2020**, *9*, 88. [CrossRef]
14. Feng, Z.; Zhu, Y. A Survey on Trajectory Data Mining: Techniques and Applications. *IEEE Access* **2016**, *4*, 2056–2067. [CrossRef]
15. Guo, Y.; Wang, S.; Zheng, L.; Lu, M. Trajectory Data Driven Transit-Transportation Planning. In Proceedings of the 2017 Fifth International Conference on Advanced Cloud and Big Data (CBD), Shanghai, China, 13–16 August 2017; pp. 380–384. [CrossRef]
16. Lv, Q.; Qiao, Y.; Ansari, N.; Liu, J.; Yang, J. Big Data Driven Hidden Markov Model Based Individual Mobility Prediction at Points of Interest. *IEEE Trans. Veh. Technol.* **2017**, *66*, 5204–5216. [CrossRef]
17. Houenou, A.; Bonnifait, P.; Cherfaoui, V.; Yao, W. Vehicle trajectory prediction based on motion model and maneuver recognition. In Proceedings of the 2013 IEEE/RSJ International Conference on Intelligent Robots and Systems, Tokyo, Japan, 3–7 November 2013; pp. 4363–4369. [CrossRef]
18. Qing, B.S.; Guan, H.; Jia, X. Vehicle Trajectory Prediction Based on Road Recognition. *Appl. Mech. Mater.* **2014**, *3360*, 599–601.
19. Xiao, W.; Zhang, L.; Meng, D. Vehicle Trajectory Prediction Method based on Deep Learning under the background of Internet of Vehicles. *SAE Int. J. Adv. Curr. Prac. Mobil.* **2020**, *2*, 3060–3071. [CrossRef]
20. Zhu, L.; Li, Y.; Yu, F.R.; Ning, B.; Tang, T.; Wang, X. Cross-Layer Defense Methods for Jamming-Resistant CBTC Systems. *IEEE Trans. Intell. Transp. Syst.* **2021**, *22*, 7266–7278. [CrossRef]
21. Lv, J.; Sun, Q.; Li, Q.; Moreira-Matias, L. Multi-Scale and Multi-Scope Convolutional Neural Networks for Destination Prediction of Trajectories. *IEEE Trans. Intell. Transp. Syst.* **2020**, *21*, 3184–3195. [CrossRef]
22. Zhang, B.; Yu, W.; Jia, Y.; Huang, J.; Yang, D.; Zhong, Z. Predicting vehicle trajectory via combination of model-based and data-driven methods using Kalman filter. *J. Automob. Eng.* **2023**, p. 09544070231161846. [CrossRef]
23. Liu, X.; Wang, Y.; Zhou, Z.; Nam, K.; Wei, C.; Yin, C. Trajectory Prediction of Preceding Target Vehicles Based on Lane Crossing and Final Points Generation Model Considering Driving Styles. *IEEE Trans. Veh. Technol.* **2021**, *70*, 8720–8730. [CrossRef]
24. Messaoud, K.; Yahiaoui, I.; Verroust-Blondet, A.; Nashashibi, F. Attention Based Vehicle Trajectory Prediction. *IEEE Trans. Intell. Veh.* **2021**, *6*, 175–185. [CrossRef]
25. Rongxia, W. Vehicle Trajectory Prediction Method based on Deep Learning under the background of Internet of Vehicles. *J. Phys. Conf. Ser.* **2021**, *1881*, 022055. [CrossRef]
26. Yang, C.; Pei, Z. Long-Short Term Spatio-Temporal Aggregation for Trajectory Prediction. *IEEE Trans. Intell. Transp. Syst.* **2023**, *24*, 4114–4126. [CrossRef]
27. Li, F.; Li, Q.; Li, Z.; Huang, Z.; Chang, X.; Xia, J. A Personal Location Prediction Method Based on Individual Trajectory and Group Trajectory. *IEEE Access* **2019**, *7*, 92850–92860. [CrossRef]
28. Wang, P.; Yu, H.; Liu, C.; Wang, Y.; Ye, R. Real-Time Trajectory Prediction Method for Intelligent Connected Vehicles in Urban Intersection Scenarios. *Sensors* **2023**, *23*, 2950. [CrossRef]
29. Liu, R.W.; Liang, M.; Nie, J.; Yuan, Y.; Xiong, Z.; Yu, H.; Guizani, N. STMGCN: Mobile Edge Computing-Empowered Vessel Trajectory Prediction Using Spatio-Temporal Multigraph Convolutional Network. *IEEE Trans. Ind. Inform.* **2022**, *18*, 7977–7987. [CrossRef]
30. Qu, P.; Ding, Z.; Guo, L. Prediction of Trajectory Based on Markov Chains. *Comput. Sci.* **2010**, *37*, 189–193.
31. Shvachko, K.; Kuang, H.; Radia, S.; Chansler, R. The Hadoop Distributed File System. In Proceedings of the 2010 IEEE 26th Symposium on Mass Storage Systems and Technologies (MSST), Incline Village, NV, USA, 6–7 May 2010.
32. Zaharia, M.; Xin, R.S.; Wendell, P.; Das, T.; Armbrust, M.; Dave, A.; Meng, X.; Rosen, J.; Venkataraman, S.; Franklin, M.J.; et al. Apache Spark: A Unified Engine for Big Data Processing. *Commun. ACM* **2016**, *59*, 56–65. [CrossRef]
33. Cheng, D.; Zhou, X.; Lama, P.; Wu, J.; Jiang, C. Cross-Platform Resource Scheduling for Spark and MapReduce on YARN. *IEEE Trans. Comput.* **2017**, *66*, 1341–1353. [CrossRef]
34. Charapko, A.; Ailijiang, A.; Demirbas, M.; Kulkarni, S. Retroscope: Retrospective Monitoring of Distributed Systems. *IEEE Trans. Parallel Distrib. Syst.* **2019**, *30*, 2582–2594. [CrossRef]
35. Xia, X.; Hashemi, E.; Xiong, L.; Khajepour, A. Autonomous Vehicle Kinematics and Dynamics Synthesis for Sideslip Angle Estimation Based on Consensus Kalman Filter. *IEEE Trans. Control Syst. Technol.* **2023**, *31*, 179–192. [CrossRef]

36. Xia, X.; Xiong, L.; Huang, Y.; Lu, Y.; Gao, L.; Xu, N.; Yu, Z. Estimation on IMU yaw misalignment by fusing information of automotive onboard sensors. *Mech. Syst. Signal Process.* **2022**, *162*, 107993. [CrossRef]
37. Newson, P.; Krumm, J. Hidden Markov Map Matching through Noise and Sparseness. In Proceedings of the 17th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems, New York, NY, USA, 2009; pp. 336–343. [CrossRef]
38. Kavoosi, M.; Dulebenets, M.A.; Abioye, O.; Pasha, J.; Theophilus, O.; Wang, H.; Kampmann, R.; Mikijeljevic, M. Berth scheduling at marine container terminals A universal island-based metaheuristic approach. *Marit. Bus. Rev.* **2020**, *5*, 30–66. [CrossRef]
39. Zhao, H.; Zhang, C. An online-learning-based evolutionary many-objective algorithm. *Inf. Sci.* **2020**, *509*, 1–21. [CrossRef]