A Novel Generation Adversarial Network-based Vehicle Trajectory Prediction Method for Intelligent Vehicular Networks

Liang Zhao, *Member, IEEE,* Yufei Liu, Ahmed Al-Dubai, *Senior Member, IEEE,* Albert Y. Zomaya, *Fellow, IEEE,* Geyong Min, and Ammar Hawbani, *Member, IEEE*

***Abstract*—Prediction of the future location of vehicles and other mobile targets is instrumental in intelligent transportation system applications. In fact, networking schemes and protocols based on machine learning can benefit from the results of such accurate trajectory predictions. This is because routing decisions always need to be made for the future scenario due to the inevitable latency caused by processing and propagation of the routing request and response. Thus, to predict the high-precision trajectory beyond the state-of-the-art, we propose a Generative Adversarial Network-based VEhicle trajEctory Prediction method, GAN-VEEP, for urban roads. The proposed method consists of three components, 1) vehicle coordinate transformation for data set preparation, 2) neural network prediction model trained by GAN, and 3) vehicle turning model to adjust the prediction process. The vehicle coordinate transformation model is introduced to deal with the complex spatial dependence in the urban road topology. Then, the neural network prediction model learns from the behavior of vehicle drivers. Finally, the vehicle turning model can refine the driving path based on the driver's psychology. Compared with its counterparts, the experimental results show that GAN-VEEP exhibits higher effectiveness in terms of the Average Accuracy, Mean Absolute Error, and Root Mean Squared Error.**

***Index Terms*—Vehicle trajectory prediction, generative adversarial network, behavior model of vehicle drivers, spatial dependence.**

I. Introduction

With the development of the intelligent transportation systems (ITS), the demand for various applications that are dependent on vehicle trajectory continues to expand, such as travel planning, urban traffic congestion mitigation, and urban traffic management. Vehicular Networking is the key enabler for ITS applications that are on-board and which are expected to be utilized to collect vehicular data in real-time. Besides, a variety of services for passengers and drivers can be made possible by on-board applications, such as vehicle road safety and travel experience and efficiency. Although vehicular networking has attracted a lot of attention from both academia and industry, current forms of networking schemes still suffer from the impassable ceiling of QoS parameters. This is partly because the simulation and design of these schemes are mostly based on historical data [1]. An extreme case is packet routing for highly dynamic vehicles. The routing decision is made to respond to an early-time request, while the route is for guiding data messages in the future network. This does exist in all types of mobile multi-hop networks. However, it can cause the most serve problem in vehicular networks as it is of vital importance for driving applications to exchange data with minimum latency and data drops. On the other hand, when it comes to discuss the next-generation vehicular networks, learning new routing schemes is an inevitable ability. Although Software-Defined Vehicular Network (SDVN) is a feasible networking paradigm for learning, the existing data to feed the machine is still historical where the road information are collected at an earlier time due to the transmission delay [2].

As a solution, traffic prediction is a key component to realize ITS and vehicular networks by expecting the foreseen future on the road. The traffic prediction can be divided into two categories, macroscopic traffic flow prediction [3], [4], [5] and microscopic vehicle trajectory prediction [6], [7], [8]. The macroscopic traffic flow reflects the vehicle flow, speed, and density of each road section. By predicting the traffic flow, the traffic pattern and trend can be analyzed. On the other hand, the microscopic vehicle trajectory prediction is to study the data of the historical movement state and position distribution of each vehicle and develop the future trajectory. However, due to the rapid change of vehicle motion and the complex spatial-temporal dependence of vehicle trajectories, high-precision traffic prediction has always been a challenging problem.

Existing traffic prediction models have applied a variety of techniques including Hidden Markov model (HMM) [9], Neural Network model [10], Bayesian inference model [11], Kalman Filter model [12], and Autoregressive Integral Moving Average (ARIMA) model [13], [14]. Most of these methods are to predict the macroscopic traffic flow, where they ignore the motions of individual vehicles. However, the traffic flow prediction is not helpful to vehicular networking [15] as networking is more about individuals instead of groups. In this context, we should study the vehicle trajectory prediction to target individual vehicles. Due to the diversity and complexity of vehicle traffic events and the rapid change of vehicle

Liang Zhao (lzhao@sau.edu.cn) and Yufei Liu (liuyufei1119@163.com) are with the School of Computer Science at Shenyang Aerospace University, China.

Ahmed Al-Dubai (a.al-dubai@napier.ac.uk) is with the School of Computing at Edinburgh Napier University, UK.

Albert Y. Zomaya (albert.zomaya@sydney.edu.au) is with the Faculty of Engineering at University of Sydney, Australia.

Geyong Min (g.min@exeter.ac.uk) is with the Department of Computer Science at University of Exeter, UK.

Ammar Hawbani (anmande@ustc.edu.cn) is with the School of Computer Science and Technology at University of Science and Technology of China, China.

itself, high-precision vehicle trajectory prediction is a challenging problem. Hence, this paper introduces a short-term and high-precision vehicle trajectory prediction method, namely, Generative Adversarial Network-based VEhicle trajEctory Prediction method, GAN-VEEP, for urban environments. The contributions of the paper are summarized as follows.

- Generative Adversarial Network (GAN) is employed to train and learn the driver's behavior from the historical trajectory data. To the best of our knowledge, this paper is the first work of its kinds that adopts GAN to produce accurate vehicle trajectory. Different from other GAN models, our discriminant network consists of two independent networks, which are applied to receive the input and output data of the generated network, respectively, to achieve the better prediction precision.
- To better prepare the training data, we propose the coordinate transformation model to pre-process the vehicle trajectory data. This model can reduce the complexity of trajectory prediction caused by spatial dependence of the historical data and improve the accuracy of trajectory prediction.
- Extensive experiments have been conducted by our simulation tool, which is available online (See Section IV for the link). We apply the vehicle trajectory data set generated by SUMO [16] to evaluate the proposed trajectory prediction method. The results prove the superiority of GAN-VEEP in vehicle trajectory prediction.

The rest of the paper is organized as follows. Section II presents related work. Section III formulates the problem. In Section IV, we prepare the data set. We present the trajectory prediction method further in Section V. In Section VI, we evaluated the performance of GAN-VEEP. Section VII concludes the paper.

II. RELATED WORK

Existing traffic prediction research can be divided into two categories, macroscopic traffic flow prediction and microscopic vehicle trajectory prediction. Here, traffic flow prediction is mainly the specific analysis of traffic volume, speed and density in urban road traffic. On the other hand, vehicle trajectory prediction mainly studies the specific performance of each vehicle in traffic.

A. Traffic Flow Prediction

With the continuous advancement of research on traffic prediction, various traffic flow prediction models have emerged, which can be divided into two categories, parametric and non-parametric model. The parametric model determines the parameters by processing the original data, and then realizes the traffic forecast based on the regression function. The Kalman filter model and the Autoregressive Integrate Moving Average Model (ARIMA) are common methods. Alghamdi et al. [13] leverages ARIMA-based modeling to study some factors that significantly affect the rate of traffic congestion. A proposed short-term time series model for non-Gaussian traffic data to predict the abnormal state of traffic, which can help traffic coordinators to better manage the congestion. Seasonal ARIMA (SARIMA) has been proposed and compared with Support Vector Regression (SVR) model [17], in which SARIMA performs better in predicting traffic congestion. However, the drawback of SARIMA is, when there is a large amount of data or non-linear relationship between time and traffic flow during prediction, it always produces a doubtful prediction. In short-term traffic flow forecasting, Kalman filter model has been applied to predict the future traffic information according to the traffic state of the previous time and the current time. [12] proposed a hybrid dual Kalman filter to predict accurate and timely short-term traffic flow. However, this model relies on the assumption that all parameter of the system are static. It cannot reflect the nonlinearity and uncertainty of traffic data, and cannot overcome the interference of other random events such as traffic accidents.

The non-parametric model can solve these problems and learn statistical rules from traffic data automatically, but only there is enough historical data. Markov model, K Nearest Neighbor (KNN), SVR, Bayesian network model, and neural network model are commonly non-parametric models. In [11], Bayesian inference method is adopted to estimate the parameters of the model. Then, a Markov chain Monte Carlo (MCMC) simulation is used to obtain the optimal model. The model has higher prediction accuracy and stronger interpretability, which is of great significance for traffic managers to judge the future traffic status and traffic trend. In traffic flow prediction, Markov model is usually used to describe the statistical relationship between vehicle movement and traffic flow. [3] considers that the Markov model has a high degree of consistency with the vehicle motion mode, which is in line of the actual vehicle motion scenario and can improve the accuracy of prediction. The minimum order of Markov model is determined by calculating the cumulative distribution function of vehicle motion entropy. Finally, the author uses the established motion model to predict the traffic flow in different periods of the holiday. Zhao et al. [9] applies the sliding window model to extract real-time traffic data flow, and combines with HMM to predict short-term (i.e., relatively real-time) traffic flow state of road sections.

In recent years, with the rapid development of machine learning, neural network model can capture the dynamic characteristics of traffic data and achieve the best effect at present. Oh et al. [4] combines artificial neural network (ANN) with Gaussian mixture model, proposed a method of urban traffic flow prediction based on multi-factor pattern recognition model. This method can combine road environmental factors and dynamic traffic flow attributes from intelligent transportation system (ITS) to predict traffic flow. Bartlett et al. [5] compare the short-term traffic flow prediction effects of three machine learning models (K Nearest Neighbor (KNN), SVR and ANN) on the urban arterial road between Manchester and Liverpool. They found that ANN is more suitable for short-term traffic prediction of urban trunk roads and performs well for heterogeneous traffic flows. In addition, Recurrent Neural Network (RNN) is widely used in time series data because of its good performance in processing time series information. As a variant of RNN, Long Short-Term Memory (LSTM) network solves the long-term dependence problem of RNN.

In the online learning mode, LSTM has superior predictive performance in short-term traffic flow prediction, and can adapt to the dynamic changes of traffic flow distribution [10].

B. Vehicle Trajectory Prediction

Currently, the study of Microscopic vehicle trajectory prediction mainly focuses on automatic driving vehicle control [18], collision detection [19], traffic data mining [8] [20] [21], and vehicle network dynamic planning [22]. Vehicle trajectory prediction methods can be roughly divided into two categories, physical/maneuver-based models and interaction awareness-based models.

The physical-based method can only predict the vehicle trajectory in a short time (less than 1 second) with high prediction accuracy. These methods assume that the vehicle has a constant speed and direction. Thus, they have the lowest abstraction in trajectory prediction [6]. In contrast, the maneuver-based prediction method needs to identify the vehicle operation first, and then uses the identified operation to predict the future trajectory of the vehicle. [8] proposes a vehicle turning trajectory tracking control algorithm based on vehicle dynamics constraints, which can make the trajectory tracking of intelligent vehicles meet the requirements of urban road level accuracy. Schreier et al. [23] apply the Bayesian inference network to estimate the current driving action of each vehicle. It combines with the current physical information of the vehicle to predict the future trajectory. [7] believes that the trajectory tracking control strategy should be human centered. This strategy combines the driver behavior prediction with the transient process of cut-in scene, which can make the automatic driving vehicle obtain a smooth transition process.

The interaction awareness-based models mainly consider the influence of interaction between vehicle and vehicle, between vehicle and road environment on vehicle trajectory prediction. These methods can provide long-term prediction results. Jeong et al. [21] use vehicle speed, acceleration, yaw rate, steering, and road curvature as inputs for training the neural network model, which can produce more accurate trajectory information of vehicles in the next few seconds. In [20], the Traj-clusiVAT algorithm is used to cluster a large number of overlapping tracks in dense road network, in which the results are used to train Markov model. This algorithm achieves good performance in both short-term and long-term trajectory prediction.

Since the interactive model can track and predict the vehicle trajectory in real-time, it is widely used in the trajectory prediction of autonomous driving (i.e., collision prediction). Collision prediction is one of the key technologies of autopilot system and other driver assistance systems. [19] introduces a collision prediction algorithm, which can accurately predict Time-to-Collision (TTC) in a longer time horizon according to the future trajectory of surrounding vehicles. Chen et al. [18] present an input-output HMM to predict the speed of leading vehicles. Then, a lane change strategy for autonomous vehicles is developed, which integrates speed prediction, motion planning and trajectory tracking control. The major drawback of most trajectory prediction methods mentioned above is that they can only predict the future trajectory of vehicles in a short time (several seconds).

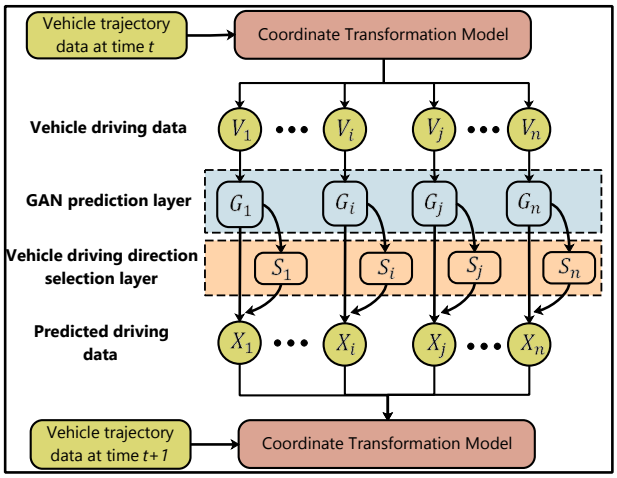


Fig. 1. Systematic overview of GAN-VEEP

III. PROBLEM FORMULATION

Vehicle trajectory is the deliverable of the complex interaction between vehicle and road. Due to the dynamicity of vehicles and the diversity of driving behavior, it is difficult to foresee the exact movements of vehicles. The goal of this work is to predict the future trajectory according to the current driving condition of the vehicle. The driving condition data of the vehicle at time t is defined as $V_t = \{V_{ts} \bigcup V_{to}\}$, where V_{ts} represents the state set of the vehicle. Here we have $V_{ts} = \{x, y, s, des\}$, where x and y represent the position coordinate of a vehicles; s denotes the instantaneous speed of vehicles at time t, and des represents the destination of vehicles at this time. Also, V_{to} represents the set of information of the road environment, where $V_{to} = \{f_h, f_d, f_s, i_d, t_l\}$. f_h represents whether there are other vehicles driving in front of the vehicle at time t; f_d is the distance from the vehicle ahead; f_s denotes the speed of the vehicle ahead; i_d is the distance between the vehicle and the next intersection; t_l represents the status of the traffic light at the intersection is a boolean variable. If there is no vehicle running in front of the vehicle, f_h, f_d, and f_s will be set to 0. As shown in Figure 1, to improve the prediction accuracy of vehicle trajectory, we first preprocess the vehicle data. This operation is achieved by the vehicle coordinate transformation model, which deals with the complex spatial dependence among urban lanes. The vehicle coordinate transformation model transforms the original vehicle information into the state set of the vehicle V_{ts}, in which this module will be introduced in Section IV.

The neural network is usually used to deal with the complex mapping between input and output. Thus, we use the neural network model to learn the driver's behavior model to predict the future vehicle trajectory. The prediction of vehicle trajectory using the neural network is defined as $V_{ts} = G(V_{t-1})$. Here, V_{t-1} represents the driving condition data of the vehicle at the $t-1$ moment. V_{ts} denotes the state set of the vehicle at t moment. G is the mapping function from V_{t-1} to V_{ts}. In this

paper, we divide the road between two adjacent intersections into two $road\ segments$ (RSGs) with opposite directions, where a road network can be divided into multiple RSGs. The neural network prediction model can only predict the situation that the vehicle does not drive out of the current RSG, so we design the vehicle turning model based on the driver's psychology to deal with the problem of choosing the driving direction when the vehicle moves to the next RSG (i.e., the vehicle can turn or move straight to the next RSG). The turning strategy of the vehicle is defined as $C = \{c_1, c_2, \cdots, c_b\}$, where C represents the turning strategy set of the vehicle; c denotes the driving direction of the vehicle at the intersection, and b is the number of intersections the vehicle passes. As shown in Figure 1, when the vehicle needs to select the driving direction, the vehicle driving direction selection layer will be used to help the vehicle select the correct driving direction. If the vehicle is driving straight and does not drive out of the current RSG, the prediction layer directly output the predicted results to the vehicle driving data without passing through the vehicle driving direction selection layer. These two layers are tightly coupled together, which determine the predicted driving data of the vehicle together. The neural network prediction model and vehicle turning model will be introduced in Section V.

In general, the vehicle trajectory prediction problem can be considered as learning the mapping function F under the premise of vehicle driving conditions and turning strategy set. Then, we have the trajectory information at the next moment, which is defined as follows:

$$\left[\hat{V}_{t+1}, \hat{V}_{t+2}, \cdots, \hat{V}_{t+T}\right] = F\left(V_t; C\right) \tag{1}$$

where T is the length of the vehicle trajectory time series to be predicted; $\hat{V}_{t+1}$ represents predicted vehicle driving conditions. To measure the performance of the mapping function, we define the trajectory error as follow:

$$l = [V_{t+1}, V_{t+2}, \cdots, V_{t+1}] - \left[\hat{V}_{t+1}, \hat{V}_{t+2}, \cdots, \hat{V}_{t+T}\right] \tag{2}$$

where l is the error between the predicted vehicle trajectory and the actual value. Our goal is to minimize l to achieve high prediction accuracy.

IV. DATA PROCESSING

A. Parameters

When the vehicle is driven on an RSG, the motion state of a vehicle will be directly affected by its adjacent vehicles in the same RSG. Hence, we divide the whole topology into multiple independent RSGs and analyze the driving state of vehicles on each RSG. To describe RSG adequately, it will be shown in the shape of a rectangle, as shown in Figure 2. Next, we model RSG and vehicle. The characteristic parameters of RSG can be expressed as follows:

$$R = (P_1, P_2, P_3, P_4, P_{ori}, P_{des}) \tag{3}$$

where P represents a two-dimensional (2D) point (x, y). P_1, P_2, P_3, P_4 are four vertices of the rectangle, which are vertices of the lower-left corner, the upper-left corner, the lower-right corner, and the upper-right corner of the rectangle,

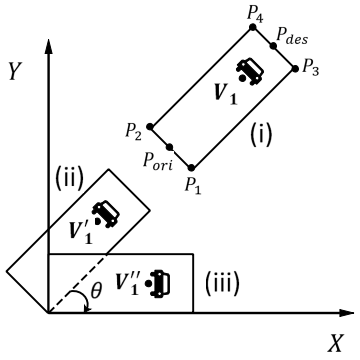


Fig. 2. Vehicle coordinate conversion

respectively. P_{ori} and P_{des} represent the starting point and the ending point of the rectangle, respectively. Since each road has a driving direction, the two points of P_{ori} and P_{des} are used to show the direction of the rectangle. In this context, the vehicles on the road can only move from P_{ori} to P_{des}. These parameters will be used to normalize the vehicle position coordinates in data preprocessing.

We refer to the RSG whose starting point is at the coordinate origin, and the driving direction of the road is parallel to the X coordinate axis as $normalized\ RSG$. For example, in Figure 2, the RSG marked as (iii) is a $normalized\ RSG$. The coordinate of the vehicle running on the $normalized\ RSG$ is called a $normalized\ coordinate$. The vehicle features are defined as follows.

$$V = (id, x, y, s, t) \tag{4}$$

where id is the number of the vehicle; (x, y) represents the location of the vehicle; t denotes the sampling time of the vehicle information; s denotes the instantaneous speed of the vehicle at time t. The function of the vehicle coordinate transformation model is to convert $non-normalized\ coordinates$ into $normalized\ coordinates$.

B. Vehicle Coordinate Transformation Model

The topological structure of the urban road network is arranged in a crisscross pattern, which constrains the vehicle's moving path. Generally speaking, vehicles always travel from the upstream to the downstream of the lane following the direction, and the connectivity between roads will also affect the next direction of vehicles. The complex spatial correlation of the urban road network makes the vehicle trajectory challenging to predict. We know that the coordinate range of each RSG of a road network is different, which can lead to a significant difference in the coordinate values of vehicle positions on different RSGs. Hence, it is complicated to train a general prediction model, which can adapt to all RSGs, especially for the vehicle position, which requires high precision. For instance, the vehicle position predicted by the neural network may deviate from the road where it locates. To solve the above problem, we propose the vehicle coordinate transformation model to deal with the complex spatial dependence in urban roads, which can improve the prediction accuracy of vehicle

trajectory and reduce the prediction error of the neural network prediction model.

We would like to present the two motivations to transform the vehicle coordinate on the general road into the normalized coordinate through the vehicle coordinate transformation model in detail. First, since the normalized RSG is parallel to the X coordinate axis, the Y coordinate value of the vehicle is kept unchanged when the vehicle is driving on the normalized RSG, and only the value of X coordinate changes continuously along with speed. By considering this feature of the normalized RSG, the change of vehicle motion initially required 2D X and Y coordinate to be represented. Nevertheless, now, it can be expressed by using one-dimensional (1D) of X coordinate. Transforming 2D coordinate data into 1D X data can significantly reduce the complexity of data, and is helpful to predict the location of vehicles. Second, all RSGs are converted to the position of normalized RSGs through the vehicle coordinate transformation model, which means that the vehicle coordinates on all RSGs in the road topology are converted to $normalized\ coordinates$. Thus, it is only necessary to train a model that can predict the location of vehicles in the $normalized\ RSG$, rather than predicting the location of vehicles on all RSGs. This method reduces the generalization ability of the neural network model but improves the prediction accuracy of the model.

Next, the vehicle coordinate normalization model is described as an example shown in Figure 2. Without loss of generality, we use the RSG at State (i) to represent the urban road to show the coordinate transformation process. V_1 is the original position of the vehicle. For the RSG translation from the State (i) to State (ii), we define transformation as below.

$$\begin{cases} V_{1x}^{'} = V_{1x} - P_{1x} \\ V_{1y}^{'} = V_{1y} - P_{1y} \end{cases} \tag{5}$$

where V_1 represents the point of vehicle V at State (i), and $V_1^{'}$ represents the point of the vehicle at State (ii) after translation. V_{1x} and V_{1y} are X and Y coordinates of V_1 point, respectively, while $V_{1x}^{'}$ and $V_{1y}^{'}$ are X and Y coordinates of $V_1^{'}$ point, respectively. To convert the RSG to the position of the $normalized\ RSG$, the RSG is needed to rotate a certain angle around the coordinate origin to the corresponding position. This angle of RSG rotation is defined as follows.

$$\theta = sin^{-1} \frac{P_{3y} - P_{1y}}{\sqrt{\left(P_{3x} - P_{1x}\right)^2 + \left(P_{3y} - P_{1y}\right)^2}} \tag{6}$$

where θ represents the angle of rotation required for RSG from the State (ii) to State (iii). Then, the coordinate value after vehicle V rotates θ around the coordinate origin is:

$$\begin{cases} V_{1x}^{''} = \left(P_{3x} - P_{1x}\right) cos\theta + \left(P_{3y} - P_{1y}\right) sin\theta \\ V_{1y}^{''} = -\left(P_{3x} - P_{1x}\right) sin\theta + \left(P_{3y} - P_{1y}\right) cos\theta \end{cases} \tag{7}$$

where $V_{1x}^{''}$ and $V_{1y}^{''}$ are the coordinate values of vehicle V in the State (iii). The vehicle coordinate normalization model transforms the original vehicle coordinate into the $normalized\ coordinate$, which brings great convenience to predict the vehicle position and speed with the neural network.

V. TRAJECTORY PREDICTION METHOD

In this section, we will introduce the neural network prediction model and vehicle turning model for vehicle trajectory prediction. The neural network prediction model predicts the driving state of the vehicle at the next time-window. In addition, the vehicle turning model selects the next driving direction (i.e., straight, or left, or right, or other directions) when the vehicle leaves the current RSG. In general, the prediction model is for predicting the driver's behavior, and the turning model is for predicting the trajectory.

A. Prediction Model of GAN

GAN is a generative model [24]. Compared with other generative models (including Boltzmann machine [25] and GSNs [26]), GAN only employs backpropagation and does not need complex Markov chains. Generally speaking, GAN can produce more precise, artificial samples. Inspired by the zero-sum game in game theory, GAN will consider this generation problem as the confrontation and game between the two networks of the discriminator and the generator. The generator tries to produce more real data to fool the discriminator while the discriminator tries to distinguish real data from generated data more perfectly. The discriminator distinguishes the output of the generator and real data. The generator outputs the composite data from the given noise (generally under a uniform distribution or a normal distribution). As a result, the two networks make progress in the confrontation and continue to fight after the progress as shown in Figure 3. This results that the data obtained from the generative network will become perfection, by approaching the real data, so that the demanded trajectories can be generated. The major advantage of GAN is that it can automatically learn the data distribution of the original real sample set. No matter how complicated the distribution is, it can learn as long as the training is good enough. Besides, GAN automatically defines the potential loss function.

Therefore, GAN is used to train the model and predict the vehicle trajectory in this paper. As we know, vehicle trajectory is highly time-series data. The position and speed of the vehicle are closely related to the vehicle state at the previous moment. Hence, we formalize the vehicle trajectory prediction problem as a time-series prediction process in Eq. (8).

$$V_{t+1} = f\left(V_t\right) \tag{8}$$

where V_t represents the driving condition data of the vehicle in the last time frame (as the input of the generated network); V_{t+1} represents the driving condition data of the vehicle predicted by the generated network one second later. That is to say, each prediction only predicts the driving condition of all vehicles in the current topology one second later. Then take the predicted driving condition data after one second as input, and continue to predict the driving condition of the vehicle in the next second. This prediction process is iterated until the required vehicle trajectory data is obtained. The advantage of this method is that as long as the error of each predicted data is minimized, the accuracy of the final predicted vehicle trajectory can be high. In the following

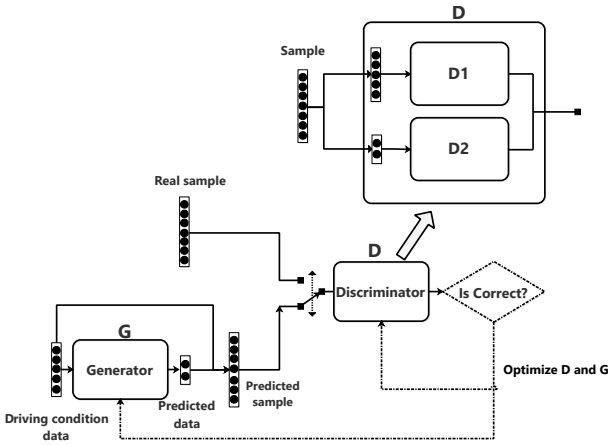


Fig. 3. The structure model of GAN

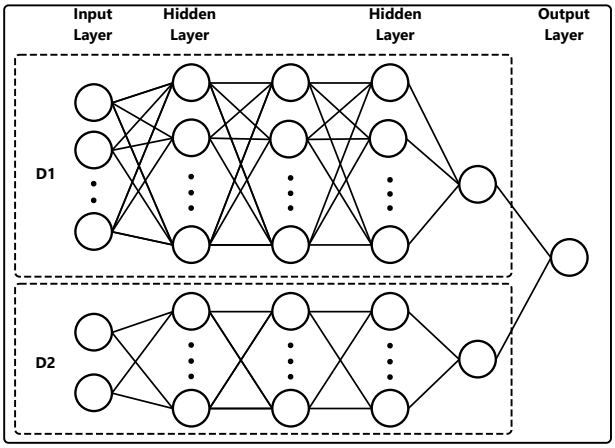


Fig. 4. The structure of the Discriminant model

subsections, the generative model and the discriminant model will be introduced, respectively.

1) Generative Model: The problem of predicting vehicle trajectory with the GAN model can be considered as making the model learn mapping function G and predict the vehicle state information in the next second under the premise of given driving condition data of the vehicle. We use the depth neural network (DNN) as the generative model. The process of generative model to predict V_{ts} can be expressed as $V_{ts} = G(V_{t-1})$, where V_{t-1} represents the driving condition data of the vehicle at the $t-1$ moment, and V_{ts} is the state set of the vehicle generated by the generated model. The generated V_{ts} can be combined with the V_{to} at that time to get the V_t of the vehicle. Then V_t is applied as the input of the generative model, while the output is the predicted state data V_{t+1s} of the vehicle at the next moment. By iterating this process, continuous vehicle trajectory data can be generated. In Algorithm 1, Line 5-8 shows the process of training the generator.

2) Discriminant model: The function of the discriminator is to distinguish the difference between the generated vehicle trajectory data and real data. The closer the generated trajectory is to the real trajectory, the closer the result of the discriminator is to 1. On the contrary, the result of the discriminator will be closed to 0. Since the generator will learn the mapping function G from V_{t-1} to V_{ts}, the discriminator cannot judge the data generated by generator only by V_{t-1} or V_{ts}. Therefore, only the vehicle trajectory data output by the generator is used as the input of the discriminator, the discriminator cannot judge whether the data is closed to the real. If the discriminator intends to judge whether the prediction result of the generator is close to the real, it needs the input of the generator to assist the discrimination. The discrimination process is shown as follows.

$$\epsilon = D(V_{t-1}; G(V_{t-1})) \tag{9}$$

where ϵ is the probability of discriminator to distinguish the data generated by generator into real data. From the above equation, it can be seen that the discriminator needs to accept two parts of data to correctly judge the correctness of the mapping function learned by the generator. The first part is the input data of the generator, while the second is the output data of the generator. As shown in Figure 3, V_{t-1} represents the driving condition data, and $G(V_{t-1})$ denotes the predicted data. Hence, in this paper, the discriminant network of the discriminant model is also composed of two parts, D1 and D2. As shown in Figure 4, D1 and D2 are both deep neural networks with three hidden layers, which jointly determine the probability of discriminator to distinguish the data generated by generator into real data. Positive and negative samples are applied in the training of discriminators. The positive sample is the complete vehicle trajectory data generated by the SUMO simulator. On the other hand, the negative sample is the forecast data generated by the generator based on the input. Then, the discriminator gives a reward to the generator according to the positive and negative samples. On this basis, the discriminator is updated by the policy gradient (Line 11-14).

In summary, Algorithm 1 shows full details of the GAN. After initializing the generator and the discriminantor, G_α and D_ρ are trained alternately. As the generator gets progressed via training on $G-steps$ updates, the discriminator needs to be retrained to keep a good pace with the generator. When training the discriminator, the positive examples are from the given dataset μ, whereas negative examples are generated from the generator. To get a more accurate generative model, the learning rate of generator is a little higher than discriminator, which also make generator converge faster.

In this way, we can use GAN to simulate the driver's behavior in order to predict the future state of the vehicle independently. However, GAN cannot choose the turning direction of the vehicle. This is mainly because the RSGs we used for training do not contain the turning directions. Therefore, we integrate the vehicle turning model in the prediction method. This model is specially designed to deal with the situation that the vehicle is at the end of the current RSG and has to choose its future moving direction.

B. Vehicle Turning Model

The prediction model introduced above is to foretell the position if the vehicle is still within the current RSG. However,

Algorithm 1: Generative Adversarial Network

Input: generator G_α; discriminator D_ρ; a dataset $\mu = \{Z_{1:n}\}$, $Z_t = \{V_{t-1} \cup V_{ts}\}$; learning rate of generator ω_g; learning rate of discriminator ω_d
Output: roll-out policy G_o

Initialize G_α, D_ρ with random weights α, ρ
$o \leftarrow \alpha$
repeat
  for *G-steps* **do**
    Generate a negative example $\hat{Z}_t = \left\{V_{t-1} \cup \hat{V}_{ts}\right\}$, $\hat{V}_{ts} = G_\alpha(V_{t-1})$
    $\bigtriangledown = log\left(1 - D_\rho\left(\hat{Z}_t\right)\right)$
    $\alpha \leftarrow \alpha - \omega_g \bigtriangledown$
    Update the generator parameters α
  end
  for *D-steps* **do**
    Generate a negative example $\hat{Z}_t$ by using current G_α
    $\bigtriangledown = log\left(D_\rho\left(Z_t\right)\right) + log\left(1 - D_\rho\left(\hat{Z}_t\right)\right)$
    $\rho \leftarrow \rho + \omega_d \bigtriangledown$
    Update the discriminator parameters ρ
  end
  $o \leftarrow \alpha$
  return o
until *GAN converges*;

if the vehicle reaches the end of the current RSG and going to move out of the current RSG, we will require the vehicle turning model to determine the next direction of the vehicle in order to put the vehicle in the next RSG (i.e., it can either move straight, or turn left, or turn right, or other directions in this case). The driving direction of a single-vehicle is not the first concern for the traffic flow prediction. Generally, the evaluation of macro traffic flow focuses on the overall performance of all vehicles in the whole topology by ignoring the differences of each vehicle. However, for fine-grained micro vehicle trajectory prediction, it is the key to predict vehicle trajectory close to the real scenario. In this study, we focus on the specific performance of each vehicle in the road topology. Hence, the vehicle turning at every intersection has a significant influence on the result of trajectory prediction. The vehicle turning model is designed to select the driving route of the vehicle by simulating the real-world drivers' behavior. The relation between the prediction model and the turning model are like navigation app and driver, in which the navigation app points out the turning direction and the the driver controls the moving of vehicle on the road. When the vehicle is turning, the turning model can select the best driving direction based on the driver's psychology, which is greedy for time and cost.

We use a directed graph $\eta = (N, E)$ to describe the topological structure of the road network, and we define each intersection as a node. N is the intersection node-set, $N = \{n_1, n_2, \cdots, n_m\}$, m is the number of nodes, and E is the road edge set; $E = \{e_1, e_2, \cdots, e_w\}$, where w is the number of edges. The adjacency matrix A is used to express the connectivity between nodes, $A \in U^{m \times m}$. The adjacency matrix contains only elements 0 and 1. If there is no link between two nodes, the element is 0. Otherwise, the element is 1. Each link represents a directed edge, where each edge is constrained by two intersection nodes, namely the start node and the end node. The direction of the edge is from the start node to the end node. We use the feature matrix $H \in R^{w \times o}$ to represent the attributes of roads, where o is the number of attributes of roads. The attribute of the road includes the length of the road and the traffic light status of the intersection.

Besides, we should consider two contextual points for the selection of driving direction. The first is under what circumstance the vehicle needs to choose a driving direction, while the second is what direction the vehicle chooses to drive. We will discuss these two points as follows. (1) We aim to discuss the circumstances in which the vehicle adopts the turning model. If the vehicle reaches the end of RSG and the traffic light at the intersection is the green light, the vehicle will drive out of the current RSG. Thus the vehicle will adopt the turning model at this time. Another situation is that when the traffic light at the intersection is green, but the vehicle is still a certain distance from the end of the current RSG. The vehicle will drive out of the current RSG after driving at the current speed for one second, where it should also implement the turning model. When the vehicle is in the above two situations, the vehicle will execute the turning strategy. (2) Generally speaking, the driver will choose the best path (with the least travel cost) according to the road topology and destination. Since the driving route is composed of multiple RSGs, to evaluate the cost of a route, we first calculate the cost of each RSG in the route. Hence, we define the driving cost of each RSG according to the linear weighted sum of travel time and travel price [27].

$$L = \beta_1 \times time + \beta_2 \times price \tag{10}$$

where L is the cost of RSG; $time$ denotes the travel time of a vehicle on an RSG; $price$ represents the travel price, e.g., cost of oil or electricity; β_1 and β_2 represent the weight of time and price, respectively, which are equally set to 0.5 in this study. Travel time represents the time vehicles driving from the start to the end of the RSG. The more crowded the RSG is, the more time it may take a vehicle to pass the RSG. Similarly, a vehicle is needed to spend more time on a longer RSG. Hence, we use vehicle density and length of RSG to represent the travel time of each RSG, which is defined as follows:

$$time = \gamma \times den \times len + [den \div th] \times rt \tag{11}$$

where den represents the density of RSG, calculated by the number of vehicles per kilometer; γ is a constant; th is the threshold, and rt is the duration of red traffic lights. Observed from the statistics of vehicle trajectory data, without traffic lights, we find that time represents the linear ratio between vehicle density and the length of RSG; γ represents the linear ratio between travel time and density. However, the traffic lights exist in the actual urban roads. Therefore, we set a threshold, which means that when the vehicle density reaches the threshold, the vehicle will wait for a red traffic light when driving on the RSG. This threshold is determined by the average value of the statistical vehicle trajectory data, which is set to 22. Here, price is defined as:

$$cost = uc \times len \tag{12}$$

where len is the length of RSG, and uc represents the price of a vehicle per kilometer.

After calculating the L of each RSG in the current topology, we transform the vehicle routing problem into an L-based shortest path problem. Regarding the starting point and destination of the vehicle, A* [28] is applied to find the shortest path. When the vehicle needs to execute the turning model, the shortest path to the destination will be calculated according to the current topology. The vehicle then turns by following this shortest path. In summary, GAN learns the behavior model of ordinary drivers, by iteratively predicting the future driving states. The vehicle turning model makes the turning decision and couples with the prediction model to output the continuous vehicle trajectory data.

VI. EXPERIMENT EVALUATION

This section evaluates the prediction accuracy of GAN-VEEP. Although our proposal is for trajectory prediction, it can be used for forecasting traffic flow as a whole. For this, we will evaluate the prediction ability of this model regarding the macro-traffic changes (i.e., traffic speed and the traffic flow) and micro-vehicle changes (i.e., trajectory-level speed and position). The fine-grained vehicle trajectory shows more behavior patterns and activity rules of vehicles. In this context, it is evident that the accuracy of GAN-VEEP in trajectory level can determine whether we can apply such a method in the future design and evaluation of intelligent vehicular networking schemes.

A. Data Description

Due to the sparse of probing car data, we employ the SUMO to generate vehicle trajectory data for model training and evaluation. SUMO simulates the trajectory of multiple vehicles in a 1000m $\times$ 1000m map. The average vehicle density is 250/km^2. During the simulation, there will be vehicles ending their journey, and new vehicles will be generated simultaneously, to simulate the real urban scenario to the maximum extent. In SUMO, the road is defined by two nodes and a directed edge. Hence, we use adjacency matrix A to store the connectivity between nodes (i.e., the spatial relationship of roads). The feature matrix H is used to the attribute information of the road (i.e., the state of the road traffic lights, the length of the road). The vehicle turning model is based on adjacency matrix A and feature matrix H to choose the optimal route for vehicles. The vehicle trajectory information generated by SUMO is stored in matrix V. Vehicle trajectory information includes coordinate value, speed, and vehicle ID information of each vehicle. We take the matrix V as known to predict the future driving state of vehicles in road topology.

In the experiment, we apply the vehicle coordinate transformation model to normalize the coordinate information of all vehicles in the topology. The normalized data is converted for the training and testing of GAN. 80% of data is applied as a training set, while 20% is used as a test set. We predict vehicle trajectory in the next 30 seconds.

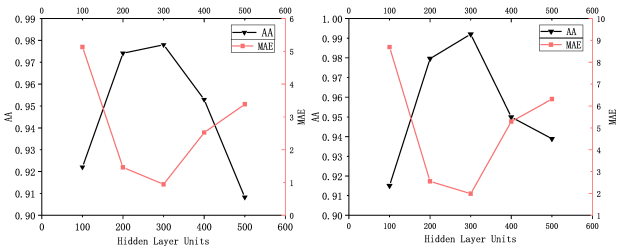


Fig. 5. Performance comparison between training Data and test data

B. Evaluation Metrics and Counterparts

We use the following metrics widely used in traffic flow prediction [29], [30] to evaluate the performance of GAN-VEEP.

(1) **Average Accuracy (AA)**: it represents the average prediction accuracy of the predicted vehicle trajectory. The bigger the values are, the better the result is.

$$AA = \frac{1}{M}\sum_{i=1}^{M}\left(1 - \frac{|\hat{y}_i - y_i|}{K}\right)$$

(2) **Mean Absolute Error (MAE)**: MAE is the average value of absolute error, which can reflect the actual situation of the error of predicted value, the smaller the value is, the better the result is.

$$MAE = \frac{1}{M}\sum_{i=1}^{M}|\hat{y}_i - y_i|$$

(3) **Root Mean Squared Error (RMSE)**: RMSE is the square root of the ratio of the square sum of the error of the prediction value to the prediction times M. RMSE and MAE are both used to measure the error. RMSE is more sensitive to the abnormal value of prediction (i.e., if there is a significant prediction error, RMSE will be very large).

$$RMSE = \sqrt{\frac{1}{M}\sum_{i=1}^{M}(\hat{y}_i - y_i)^2}$$

where y_i represents the real traffic information; $\hat{y}_i$ is the predicted value of y_i; and M is the number of vehicles; K is a constant.

We compared the performance of the proposed method with the following five well-known methods.

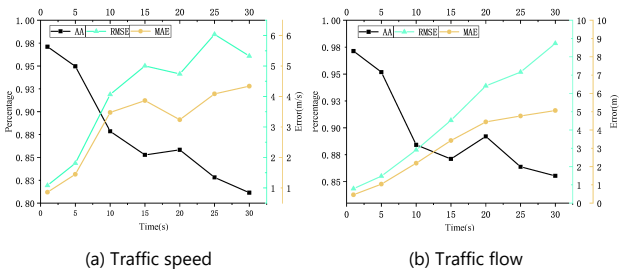


Fig. 6. Experimental results of traffic speed and traffic flow

(1) **History average model (HA)** [29]: This method predicts based on the average value of historical traffic information. Since the method proposed in [29] is more about traffic flow prediction, we then apply the HA model as one of our counterparts.

(2) **Adaptive prediction model (AP)** [31]: A dynamic OD (Origin-Destination) matrix is used to make an initial prediction while it adjusts the prediction model adaptively every once it receives new traffic data.

(3) **Markov-based trajectory prediction (Traj-clusiVAT)** [20]: It is used to cluster a large number of overlapped tracks in urban dense road network. Then the Markov model can be trained according to the clustering results in order to predict the trajectory.

(4) **DNN-based trajectory prediction (DNN)** [21]: In this method, the vehicle is modeled by speed, acceleration, yaw rate, steering, and road curvature, which are then used as the input of the deep neural network model.

(5) **Maneuver-based trajectory prediction (Maneuver)** [23]: This method estimates the current driving action of each vehicle by using the Bayesian inference. Then, the future trajectory of the vehicle is predicted on the basis of the current physical information of the vehicle.

TABLE I
COMPARISON OF TRAFFIC SPEED AND TRAFFIC FLOW

T	Metrics	Traffic Speed			Traffic Flow		
		HA	AP	MY	HA	AP	MY
5s	RMSE	3.801	4.309	**1.812**	1.814	4.483	**1.474**
	MAE	2.960	3.739	**1.447**	1.375	3.854	**1.043**
	AA	0.901	0.875	**0.951**	0.903	0.871	**0.949**
10s	RMSE	5.654	5.120	**4.066**	3.446	5.295	**2.907**
	MAE	4.539	4.320	**3.475**	2.875	4.436	**2.181**
	AA	0.848	0.855	**0.884**	0.852	0.852	**0.878**
15s	RMSE	6.091	6.687	**5.000**	4.659	5.819	**4.524**
	MAE	5.210	5.550	**3.866**	4.041	4.379	**3.421**
	AA	0.826	0.814	**0.871**	0.825	0.854	**0.852**
20s	RMSE	0.811	7.330	**4.737**	**5.642**	6.834	6.411
	MAE	5.668	5.785	**3.237**	5.000	5.319	**4.444**
	AA	0.848	0.807	**0.892**	0.804	0.822	**0.858**
25s	RMSE	0.753	6.967	**6.037**	**6.461**	7.518	7.166
	MAE	7.405	5.689	**4.085**	5.833	5.915	**4.764**
	AA	0.882	0.810	**0.863**	0.739	0.802	**0.828**
30s	RMSE	9.690	6.691	**5.326**	**6.958**	8.637	8.733
	MAE	8.015	5.487	**4.338**	6.416	7.024	**5.066**
	AA	0.732	0.817	**0.855**	0.717	0.765	**0.811**

C. Experimental Environment and Parameter Setting

In the experiment, we use Python 3.5 to develop the neural network prediction model, where TensorFlow 1.12 and Google's neural network algorithm library are deployed. Java is applied to preprocess the vehicle trajectory data and conduct the overall framework of GAN-VEEP. As an addition, the source code and dataset are available online at https://github.com/GANVEEP/GAN-VEEP.git. The hardware platform is a PC with Intel(R) Core(TM) i5-4210, 2.40GHz CPUs, and 12G bytes memory under Windows 8.1.

Because the vehicle data after the normalization of the coordinate transformation model has a linear correlation, the deep neural network can well represent the relationship of linear data. Hence, we use a deep neural network with three hidden layers as the generator and discriminator model. In the experiment, the learning rate of the generative model is set to 0.02, while the learning rate of the discrimination model is set to 0.01. For in adversarial learning, we need the generator to learn faster to produce a better prediction model. The number of hidden layer neurons is an essential parameter that affects the performance of the GAN model. To choose the best value, we have done incremental experiments on different numbers of hidden layer neurons and choose the value with the best prediction results.

In our incremental experiments, we choose the incremental parameters from [100, 200, 300, 400, 500] for training and select the value with the highest prediction accuracy as the final experimental parameter. As shown in Figure 5, the horizontal axis represents the change of the number of neurons in the hidden layer, and the vertical axis represents the change of the evaluation index. Figure 5 (a) shows the effect of the number of neurons on the training results. It can be seen that the error is the smallest when the number of neurons in the hidden layer is 300. Figure 5 (b) shows the changes in the results in test sets. Similarly, when the number of neurons in the hidden layer is 300, the test accuracy is the highest. Therefore, we set the number of neurons in the hidden layer to 300 in our experiment.

In training, although $Minlog(-D)$ is used in the original work of GAN [24], we apply $Maxlog(D)$ as the loss function of optimized G. This is because this function confirms that there is a gradual disappearing gradient at an early stage. Moreover, we use the ADADELTA [32] optimizer to train the model.

D. Experimental Results

1) Traffic flow and Speed: Figure 6 shows the performance of the vehicle trajectories predicted by our proposed method for 1s, 5s, 10s, 15s, 20s, 25s, and 30s, respectively, in terms of macro evaluation indexes (traffic flow and traffic speed). The traffic flow represents the number of vehicles on an RSG. The traffic speed denotes the average speed of vehicles on an RSG. In Figure 6, with the increase of prediction time, the AA decreases slowly, while the RMSE and MAE increase gradually. This is because each iteration will predict the trajectory one second later, and the error generated by each prediction in the process of continuous iteration is also accumulating. Another reason is, as a scenario construction rule of SUMO, vehicles can move in and out of the map, where this is not the issue we considered in GAN-VEEP. The AA value of traffic flow and speed is above 0.8 in the 30s, which confirms that our method still exhibits an excellent performance in the 30s. However, since the error produced by GAN-VEEP in each prediction will accumulate, GAN-VEEP is more suitable for short-term trajectory prediction tasks.

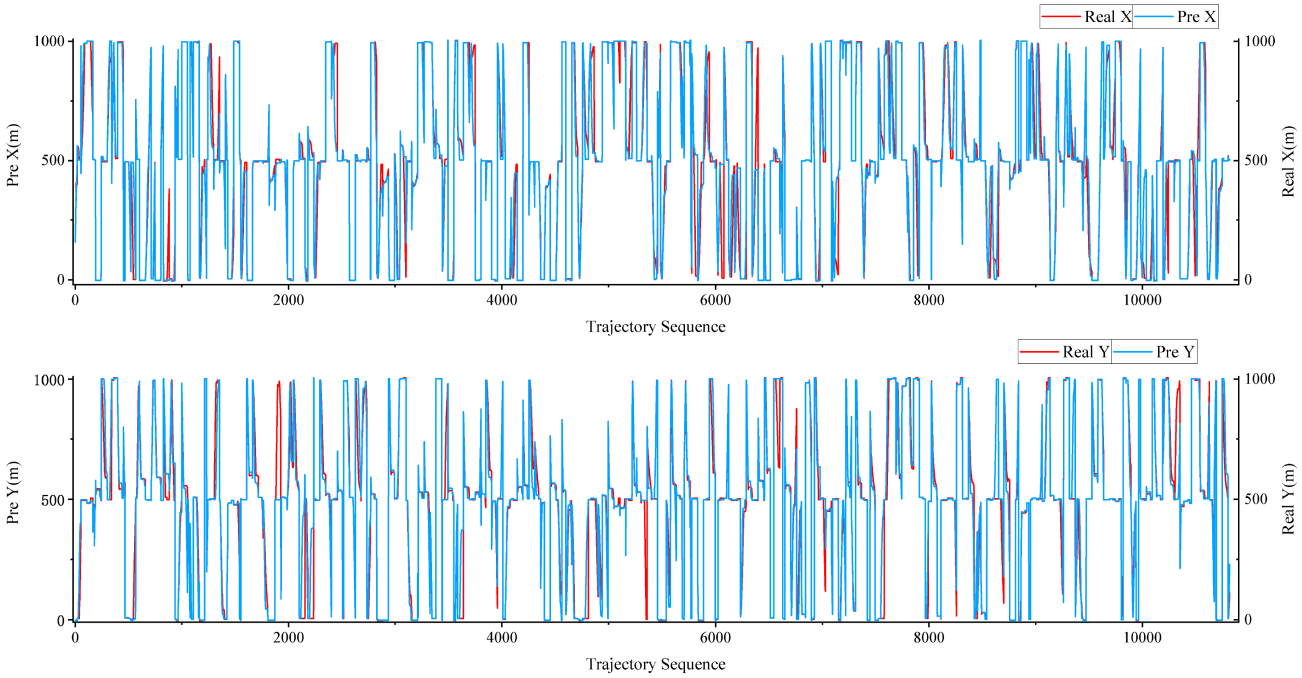
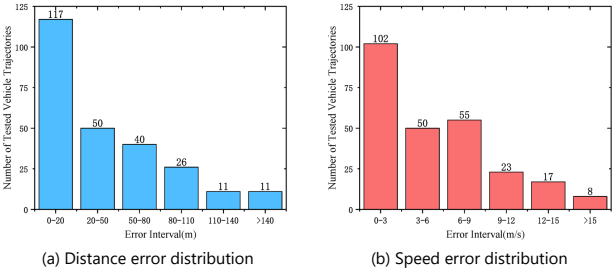
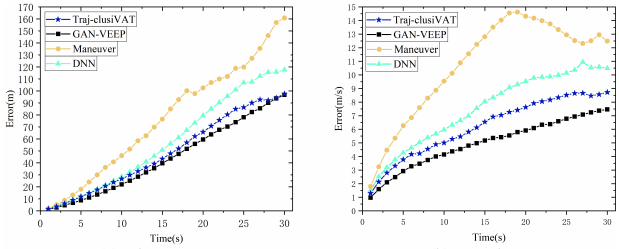


Fig. 7. Error of all vehicle trajectories



(a) Distance error distribution (b) Speed error distribution

Fig. 8. Error distribution of trajectory position and speed



(a) Distance error (b) Speed error

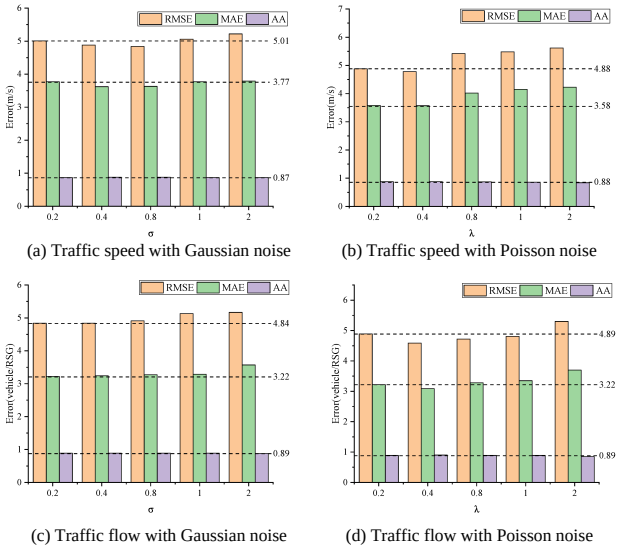
Fig. 9. Comparison of trajectory position and speed

Table I shows the comparison between our proposed method and the other two methods in the prediction performance of 5s, 10s, 15s, 20s, 25s, and 30s where we mark the best result of each row in bold. In most cases, our model achieves the best performance under all evaluation metrics, which also proves the effectiveness of the vehicle trajectory predicted by our model in the macro indicators. The RMSE value of the HA model is smaller than that of our model at 20s, 25s, 30s in the traffic flow table. This is because the vehicles in our model start slowly. It is more likely to lead to the phenomenon of vehicle crowds within an RSG. Then, the error on the RSG of the vehicle crowds will be more significant. RMSE is more susceptible to big values but less effected by the small values. This is also the reason that AA value is still better than the HA although the RMSE value of our model is higher than the HA. HA takes the mean value of the historical period as the prediction values for a period, which is more suitable for the scenario with considerable time granularity. AP builds vehicle trajectory by OD matrix without considering the current driving state of vehicles, so it does not perform well in short-term traffic prediction.

However, with the increase of prediction time, the RMSE of AP becomes stable, which is due to the adaptive acceptance of some topology information.

2) Trajectory Speed and Position: The micro-level speed and position of the individual vehicle is the key to intelligent networking. The predicted track points of each vehicle lasting for the 30s are compared with the real track data. Figure 7 shows the visual results of the predicted X and Y values of vehicle coordinates compared with the real data. In most cases, the predicted value and the real value fit well. However, sometimes there is a significant difference between the predicted value and the real value. After analysis, we find that this is a colossal error caused by tiny differences in some cases. For instance, a predicted vehicle arrives at the intersection a few seconds late and has to wait for the traffic light change from red to green. In this case, the errors between the predicted vehicle position and the real vehicle position may reach tens or even hundreds of meters. However, the probability of large errors is relatively small.

Besides, we evaluate the distance and speed error distribution of each trajectory. Figure 8 (a) shows the average



(a) Traffic speed with Gaussian noise (b) Traffic speed with Poisson noise

(c) Traffic flow with Gaussian noise (d) Traffic flow with Poisson noise

Fig. 10. The results of adding Gaussian and Poisson perturbations on the dataset

distance error distribution for each trajectory. Figure 8 (b) shows the average speed error distribution for each trajectory. From Figure 8, within the 30s, the position errors of most trajectories are less than 80m, and only 18.8% of the tracks are more than 80m. In the same aspect of speed errors, only 18.8% are more than $9m/s$.

Then, we also compare the performance of GAN-VEEP with other methods in 30 seconds in terms of position and speed. It can be seen that GAN-VEEP has almost the best prediction performance in position and speed prediction, which proves the effectiveness of the model in short-term vehicle trajectory prediction tasks. In Figure 9 (b), Maneuver, DNN, and Traj-clusiVAT reach the peak and then begin to decline. This is because some vehicles end their journey after arriving at the destination. Here, since the speed of vehicles in Maneuver is the highest, the model also reaches the peak as the earliest method. The overall trend of speed error and distance error is similar, which shows that the position of the vehicle is closely related to the speed of the vehicle. The speed of the vehicle affects the accuracy of the whole track by affecting the position of the vehicle. To conclude, the most accurate model for vehicle speed modeling can often obtain the most accurate prediction trajectory. GAN-VEEP, therefore, can get better prediction results by learning the hidden behavior of vehicle drivers from the historical data.

Last, we verify the robustness of GAN-VEEP to test the noise immunity through perturbation analysis experiments. Two types of common random noise are added to the dataset. They obey Gaussian distribution $N\left(0,\sigma^2\right)$, where $\sigma \in (0.2, 0.4, 0.8, 1, 2)$, and Poisson distribution $P\left(\gamma\right)$, and $\gamma \in (0.2, 0.4, 0.8, 1, 2)$, respectively. To illustrate the robustness of our proposal, we evaluate the stability of GAN-VEEP in terms of traffic speed and traffic flow. The results are shown as follows. Figure 10(a) and (b) show the results of the traffic speed after adding Gaussian noise and Poisson noise, respectively, where the vertical axis represents the change of each evaluation metrics, and different colors indicate different metrics. Similarly, Figure 10(c) and (b) are the results of the traffic flow after adding Gaussian noise and Poisson noise, respectively. From the above results, it can be seen that the fluctuations of metrics are relatively small whatever the noise distribution is. Therefore, the GAN-VEEP is robust as it is proved to handle high noise issues.

VII. Conclusion

In this paper, a novel short-term high-precision vehicle trajectory prediction method, GAN-VEEP is proposed. First, we use a vehicle coordinate normalization model to transform the position coordinates of each vehicle into $normalized\ coordinates$. This method improves the accuracy of the prediction model for vehicle trajectory prediction. Then, we use GAN to train a high-precision vehicle position prediction model to forecast the driving position of the vehicle on the current RSG. An cost-based vehicle turning model is introduced to select the next driving direction when the vehicle turns. According to the experimental results, compared with existing methods, GAN-VEEP has shown excellent performance in predicting vehicle trajectory on urban road scenarios.

Acknowledgment

This paper is partly supported by the National Natural Science Foundation for Young Scientists of China (61701322), the Young and Middle-aged Science and Technology Innovation Talent Support Plan of Shenyang (RC190026), and the Liaoning Provincial Department of Education Science Foundation (JYT19052).

References

[1] W. Quan, N. Cheng, P. Jing, G. Liu, and X. S. Shen, "Vedata: Promoting ai assisted autonomous vehicles," p. 771–773, 2018.

[2] L. Zhao, A. Al-Dubai, A. Y. Zomaya, G. Min, A. Hawbani, and J. Li, "Routing schemes in software-defined vehicular networks: Design, open issues and challenges," *IEEE Intelligent Transportation Systems Magazine*, 2020.

[3] D. Zhao, Y. Gao, Z. Zhang, Y. Zhang, and T. Luo, "Prediction of vehicle motion based on markov model," pp. 205–209, Dec 2017.

[4] S. do Oh, Y. jin Kim, and J. sun Hong, "Urban traffic flow prediction system using a multifactor pattern recognition model," *IEEE Transactions on Intelligent Transportation Systems*, vol. 16, no. 5, pp. 2744–2755, Oct 2015.

[5] Z. Bartlett, L. Han, T. T. Nguyen, and P. Johnson, "A machine learning based approach for the prediction of road traffic flow on urbanised arterial roads," pp. 1285–1292, June 2018.

[6] W. Xiao, J. Zou, H. Li, and K. Xu, "Smooth trajectory tracking using longitudinal distance constraint for a 4ws4wd unmanned ground vehicle*," in *2019 IEEE International Conference on Robotics and Biomimetics (ROBIO)*, Dec 2019, pp. 2105–2110.

[7] Y. Chen, C. Hu, and J. Wang, "Human-centered trajectory tracking control for autonomous vehicles with driver cut-in behavior prediction," *IEEE Transactions on Vehicular Technology*, vol. 68, no. 9, pp. 8461–8471, Sep. 2019.

[8] W. Shufeng, Z. Dawei, A. Junhui, and Z. Junyou, "Lane level turning trajectory tracking of intelligent vehicle based on drivers' manipulate habits," in *2017 29th Chinese Control And Decision Conference (CCDC)*, 2017, pp. 6873–6877.

[9] Y. Zhao and H. Zhang, "Research on short-time prediction of dynamical local replanning route guidance method based on hmm," in *2017 14th Web Information Systems and Applications Conference (WISA)*, 2017, pp. 19–22.

[10] J. Mackenzie, J. F. Roddick, and R. Zito, "An evaluation of htm and lstm for short-term arterial traffic flow prediction," *IEEE Transactions on Intelligent Transportation Systems*, vol. 20, no. 5, pp. 1847–1857, May 2019.

[11] Y. Xu, Q. Kong, R. Klette, and Y. Liu, "Accurate and interpretable bayesian mars for traffic flow prediction," *IEEE Transactions on Intelligent Transportation Systems*, vol. 15, no. 6, pp. 2457–2469, 2014.

[12] T. Zhou, D. Jiang, Z. Lin, G. Han, X. Xu, and J. Qin, "Hybrid dual kalman filtering model for short-term traffic flow forecasting," *IET Intelligent Transport Systems*, vol. 13, no. 6, pp. 1023–1032, 2019.

[13] T. Alghamdi, K. Elgazzar, M. Bayoumi, T. Sharaf, and S. Shah, "Forecasting traffic congestion using arima modeling," in *2019 15th International Wireless Communications Mobile Computing Conference (IWCMC)*, 2019, pp. 1227–1232.

[14] Z. Chi and L. Shi, "Short-term traffic flow forecasting using arima-svm algorithm and r," in *2018 5th International Conference on Information Science and Control Engineering (ICISCE)*, 2018, pp. 517–522.

[15] A. Fazekas and M. Oeser, "Performance metrics and validation methods for vehicle position estimators," *IEEE Transactions on Intelligent Transportation Systems*, pp. 1–11, 2019.

[16] SUMO, "Simulation of urban mobility," 2020. [Online]. Available: http://sumo.sourceforge.net

[17] M. Lippi, M. Bertini, and P. Frasconi, "Short-term traffic flow forecasting: An experimental comparison of time-series analysis and supervised learning," *IEEE Transactions on Intelligent Transportation Systems*, vol. 14, no. 2, pp. 871–882, 2013.

[18] Y. Chen, C. Hu, and J. Wang, "Motion planning with velocity prediction and composite nonlinear feedback tracking control for lane-change strategy of autonomous vehicles," *IEEE Transactions on Intelligent Vehicles*, vol. 5, no. 1, pp. 63–74, 2020.

[19] J. Kim and D. Kum, "Threat prediction algorithm based on local path candidates and surrounding vehicle trajectory predictions for automated driving vehicles," in *2015 IEEE Intelligent Vehicles Symposium (IV)*, 2015, pp. 1220–1225.

[20] P. Rathore, D. Kumar, S. Rajasegarar, M. Palaniswami, and J. C. Bezdek, "A scalable framework for trajectory prediction," *IEEE Transactions on Intelligent Transportation Systems*, vol. 20, no. 10, pp. 3860–3874, 2019.

[21] D. Jeong, M. Baek, and S. Lee, "Long-term prediction of vehicle trajectory based on a deep neural network," in *2017 International Conference on Information and Communication Technology Convergence (ICTC)*, 2017, pp. 725–727.

[22] L. Zhao, W. Zhao, A. Al-Dubai, and G. Min, "A novel adaptive routing and switching scheme for software-defined vehicular networks," in *ICC 2019 - 2019 IEEE International Conference on Communications (ICC)*, May 2019, pp. 1–6.

[23] M. Schreier, V. Willert, and J. Adamy, "An integrated approach to maneuver-based trajectory prediction and criticality assessment in arbitrary road environments," *IEEE Transactions on Intelligent Transportation Systems*, vol. 17, no. 10, pp. 2751–2766, 2016.

[24] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial networks," 2014.

[25] E. D. L. Rosa and W. Yu, "Restricted boltzmann machine for nonlinear system modeling," in *2015 IEEE 14th International Conference on Machine Learning and Applications (ICMLA)*, Dec 2015, pp. 443–446.

[26] G. Alain, Y. Bengio, L. Yao, J. Yosinski, S. Thibodeau-Laufer, S. Zhang, and P. Vincent, "GSNs: generative stochastic networks," *Information and Inference: A Journal of the IMA*, vol. 5, no. 2, pp. 210–249, 03 2016.

[27] Q. Xiang, Y. Ma, and J. Lu, "Optimal route selection in highway network based on travel decision making," in *2007 IEEE Intelligent Vehicles Symposium*, June 2007, pp. 1266–1270.

[28] S. Sedighi, D.-V. Nguyen, and K.-D. Kuhnert, "Guided hybrid a-star path planning algorithm for valet parking applications," 04 2019, pp. 570–575.

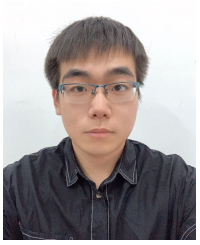
[29] L. Zhao, Y. Song, C. Zhang, Y. Liu, P. Wang, T. Lin, M. Deng, and H. Li, "T-gcn: A temporal graph convolutional network for traffic prediction," *IEEE Transactions on Intelligent Transportation Systems*, pp. 1–11, 2019.

[30] N. Deo and M. M. Trivedi, "Multi-modal trajectory prediction of surrounding vehicles with maneuver based lstms," in *2018 IEEE Intelligent Vehicles Symposium (IV)*, June 2018, pp. 1179–1184.

[31] A. Abadi, T. Rajabioun, and P. A. Ioannou, "Traffic flow prediction for road transportation networks with limited traffic data," *IEEE Transactions on Intelligent Transportation Systems*, vol. 16, no. 2, pp. 653–662, April 2015.

[32] D. Zeiler, "Adadelta: An adaptive learning rate method," *Computer Ence*, 2012.

Liang Zhao [M] is an associate professor at Shenyang Aerospace University, China. He received his PhD degree in Computing from the School of Computing at Edinburgh Napier University in 2011. Before joining Shenyang Aerospace University, he worked as an associate senior researcher in Hitachi (China) Research and Development Corporation from 2012 to 2014. He has published over 70 peer-reviewed papers, 1 monograph, and 3 chapters. His research interests include VANETs, SDN, and UAV. His postal address is 37 Daoyi South Avenue, Shenbei New Area, Shenyang, China.



Yufei Liu received the B.S. degree in Computer Science and Technology from Shenyang Aerospace University, China. Currently he is a Master student in the School of Computer Science at Shenyang Aerospace University. His research interests mainly include VANETs, SDVN, GAN, Traffic Prediction.

Ahmed Y. Al-Dubai [SM] is Professor of Networking and Communication Algorithms in the School of Computing at Edinburgh Napier University, UK. He received the PhD degree in Computing from the University of Glasgow in 2004. His research interests include Communication Algorithms, Mobile Communication, Internet of Things, and Future Internet. He received several international awards. His postal address is 10 Colinton Road, Edinburgh, United Kingdom.

Albert Y. Zomaya [F] is a Chair Professor and director of the Centre for Distributed and High Performance Computing at the University of Sydney. He has published more than 600 scientific papers and is an author, co-author, or editor of more than 20 books. He is the Editor-in-Chief of IEEE Transactions on Sustainable Computing and ACM Computing Surveys and serves as an Associate Editor for several leading journals. He is a Fellow of IEEE, AAAS, IET, and member of Academia Europaea. His address is University of Sydney, Sydney, Australia.

Geyong Min is a professor of high-performance computing and networking in the Department of Mathematics and Computer Science within the College of Engineering, Mathematics and Physical Sciences at the University of Exeter, United Kingdom. He received his Ph.D. degree in computing science from the University of Glasgow, United Kingdom, in 2003, and his B.Sc. degree in computer science from Huazhong University of Science and Technology, China, in 1995. His research interests include future Internet, computer networks, wireless communications, multimedia systems, information security, high-performance computing, ubiquitous computing, modeling, and performance engineering. His postal address is University of Exeter, Exeter, United Kingdom.

Ammar Hawbani [M] received the B.S., M.S. and Ph.D. degrees in Computer Software and Theory from the University of Science and Technology of China (USTC), Hefei, China, in 2009, 2012 and 2016, respectively. Currently, he is a postdoctoral researcher in the School of Computer Science and Technology at USTC. His research interests mainly in WSN and WBAN. His postal address is 96 Jinzhai Road, Baohe District, Hefei, China.