applied sciences

MDPI

*Article*

# Expressway Vehicle Trajectory Prediction Based on Fusion Data of Trajectories and Maps from Vehicle Perspective

**Yuning Duan [1], Jingdong Jia [1], Yuhui Jin [1], Haitian Zhang [1] and Jian Huang [1,2,*]**

[1] School of Software, Beihang University, Beijing 100191, China; sy2121102@buaa.edu.cn (Y.D.)
[2] Hangzhou International Innovation Institute, Beihang University, Beijing 100191, China
* Correspondence: hj@buaa.edu.cn

**Abstract:** Research on vehicle trajectory prediction based on road monitoring video data often utilizes a global map as an input, disregarding the fact that drivers rely on the road structures observable from their own positions for path planning. This oversight reduces the accuracy of prediction. To address this, we propose the CVAE-VGAE model, a novel trajectory prediction approach. Initially, our method transforms global perspective map data into vehicle-centric map data, representing it through a graph structure. Subsequently, Variational Graph Auto-Encoders (VGAEs), an unsupervised learning framework tailored for graph-structured data, are employed to extract road environment features specific to each vehicle's location from the map data. Finally, a prediction network based on the Conditional Variational Autoencoder (CVAE) structure is designed, which first predicts the driving endpoint and then fits the complete future trajectory. The proposed CVAE-VGAE model integrates a self-attention mechanism into its encoding and decoding modules to infer endpoint intent and incorporate road environment features for precise trajectory prediction. Through a series of ablation experiments, we demonstrate the efficacy of our method in enhancing vehicle trajectory prediction metrics. Furthermore, we compare our model with traditional and frontier approaches, highlighting significant improvements in prediction accuracy.

**Keywords:** conditional variational autoencoder; variational graph auto-encoders; trajectory prediction



**Citation:** Duan, Y.; Jia, J.; Jin, Y.; Zhang, H.; Huang, J. Expressway Vehicle Trajectory Prediction Based on Fusion Data of Trajectories and Maps from Vehicle Perspective. *Appl. Sci.* **2024**, *14*, 4181. https://doi.org/10.3390/app14104181

Academic Editor: Antonio Fernández-Caballero

Received: 6 April 2024
Revised: 7 May 2024
Accepted: 14 May 2024
Published: 15 May 2024



## 1. Introduction

Currently, research on vehicle trajectory prediction, powered by artificial intelligence technology, has made significant progress. Video data collected from expressway monitoring cameras serve as a valuable resource for predicting vehicle trajectories. This facilitates applications such as collision warnings and other vehicle–road collaborative functionalities, effectively enhancing driving safety [1,2].

The actual trajectory of a vehicle is influenced by various factors such as road conditions, the positions of other vehicles, and the driving habits and intentions of the driver. Early vehicle trajectory prediction methods has thus far relied solely on historical data, using only a vehicle's past driving data to learn and predict a certain future trajectory. However, in reality, the behavior of drivers is influenced by their driving intentions. For example, different destinations can result in significantly different future trajectories. Therefore, inferring driving intentions is crucial in trajectory prediction. Recent research has proposed incorporating intent inference modules to prevent models from converging on conservative results [3,4]. Instead of predicting a single outcome, the model generates multiple possible trajectory results based on potential driving intentions.

However, current studies have not thoroughly considered the mechanisms through which road structure and traffic environment influence drivers' intentions. Most studies simply utilize global road structure and the spatial distribution of vehicles from a bird's-eye view as inputs to the prediction model. However, in reality, each driver observes different road information. They can only perceive the road structure and traffic environment from the perspective of the vehicle's current location, and plan their route accordingly.

To address the above-mentioned issue, this paper proposes a novel trajectory prediction approach termed the CVAE-VGAE model. The model consists of a main model responsible for inferring driving destination intent and predicting trajectories, and a sub-model responsible for extracting road map information. The main model improves the encoding and decoding structure of the CVAE model by incorporating self-attention mechanisms. In order to enhance the model's perception of road environment data, a sub-model for road map information extraction and based on the VGAE is implemented. This sub-model reduces the dimensionality of graph-structured data and extracts environment feature variables. Unlike previous studies that have dealt with data from a global perspective, the sub-model in our paper will independently compute the spatial relationship between a vehicle and key points on the road from the perspective of each vehicle. This ensures that the final trajectory predictions fully consider the road information based on each driver's own perspective. Compared to models in previous studies, our model, by integrating environmental information, significantly improves the accuracy of simultaneous prediction of multiple vehicle trajectories in complex scenarios.

Section 1 of this paper introduces the background of the research topic. It highlights the significance of the research based on the existing problems in the current studies and outlines the research objectives and content. Section 2 provides a summary of the relevant research work and the development process in this field. Section 3 focuses on the preprocessing of the dataset, with a particular emphasis on the comprehensive processing of map data. Section 4 describes the design of the model, which includes the main prediction model and the map processing sub-model. Section 5 analyzes the experimental data and presents the findings of this study. Finally, Section 6 concludes the paper and proposes directions for future improvements.

## 2. Related Work

Earlier trajectory prediction research mainly focused on designing algorithms based on the principles of physical kinematics. Barth et al. applied Kalman filtering and Monte Carlo method into trajectory prediction and achieved good results in short-distance prediction [5]. Houenou et al. established a vehicle motion model using polynomial planning and completed future trajectory prediction based on the principles of acceleration and constant yaw rate [6]. Danielsson et al. proposed an improved road trajectory evaluation algorithm that used Monte Carlo simulations and improved dynamic models to identify threats in road scenes [7].

In the past decade, machine learning and related technologies have been widely applied in the field of vehicle trajectory prediction. And, compared to traditional physics-based model methods, prediction models based on machine learning have achieved significant breakthroughs in key indicators such as prediction accuracy. Classic machine learning methods and models such as support vector machine (SVM), multi-layer perceptron (MLP), and dynamic Bayesian network (DBN) were first applied in early trajectory prediction research. For example, Tomar et al. proposed a prediction algorithm based on an MLP model, but the algorithm could only predict discrete segments of trajectories rather than complete future vehicle trajectories [8]. Gao et al. proposed a hybrid model based on physical vehicle models and operation recognition models in which the operation recognition model was implemented through hidden Markov models [9].

With the rapid development of deep learning, deep learning models that can better predict complex scenarios have gradually replaced classical machine models. Among the many deep learning models, recurrent neural networks (RNNs) [10–12] were originally proposed to solve sequence learning problems, and so are naturally advantageous for tasks like trajectory prediction. Long short-term memory (LSTM) [13–16], building upon RNN models, incorporates a cell state representing long-term memory, and thus its ability to handle long-term dependencies is enhanced. Recent research is moving towards multi-modal trajectory prediction and exploring generative models. Gupta et al. addressed the multi-modal prediction problem by combining self-sequential prediction and GAN [3,17].

Lee incorporated the idea of CVAE into the LSTM prediction model, enabling the hybrid model to achieve multi-modal trajectory prediction [18]. Vaswani et al. proposed Transformer [19], which can better handle long-range dependencies compared to RNN models. Therefore, many studies have also attempted to apply Transformer to vehicle trajectory prediction [20–23]. In recent years, reinforcement learning has also been gradually applied to vehicle trajectory prediction research, in which a Markov decision process has been applied to a greater degree and has achieved good results [24].

With the continuous advancement of traffic information collection technology, most newer public datasets provide high-definition maps of predicted scenarios. How to obtain effectively characterized environmental feature information from high-definition maps and how to incorporate it into a model training process are the questions at the forefront of research. Early studies on environmental information processing mainly used convolutional neural networks (CNNs) to learn from grid maps of roads. Huang et al. used two CNNs to encode information about the environment around a predicted object and combined reward graphs representing probability scores and direction information to derive the most likely path [25]. ChauffeurNet annotated original aerial maps of roads with lane lines, traffic lights, etc., to obtain a series of grid maps. These grid maps were input into a convolutional network to extract contextual features and shared with other network modules during training [26]. Cai et al. proposed an environmental attention network model called EA-Net, which focused on learning the connection between vehicles and their driving environment [27]. VectorNet treated both vehicles and the environment as sets of vectors, integrated them into a vector map, and thus the spatial relationships between different elements was better represented [28]. This approach effectively avoids information loss during CNN convolution calculations.

The trajectory prediction model constructed in this paper is based on the generative model PECNet [29] in the pedestrian prediction domain. The PECNet model as a whole uses the autoencoder architecture of CVAE, which achieved the best prediction accuracy at that time. Due to the significant similarity between pedestrian trajectory prediction and vehicle trajectory prediction, the PECNet model is also suitable for the transfer learning tasks for vehicle trajectory prediction. However, there are some areas for improvement when using PECNet to predict trajectory. Firstly, PECNet completely ignores the influence of environmental and road factors, which means that the model only inputs the historical trajectories of the prediction target without perceiving environmental conditions of the actual scene. Secondly, the encoder and decoder of PECNet are relatively simple and are composed of multiple multi-layer perceptrons (MLPs). Thus, room for further improvement exists in terms of prediction effectiveness.

Due to a lack of high-precision scene map data in previous public datasets, current research on vehicle trajectory prediction mostly lacks consideration of the road environment in which vehicles are situated. Additionally, current map data processing primarily calculates spatial relationships from an overhead global perspective, while drivers collect information and make decisions from their own perspectives during actual driving. In the environmental information processing of this paper, the spatial relationships between each predicted vehicle's perspective and key road points are calculated and used as inputs of our model. This ensures that the final predicted trajectory is fully referenced to road information based on each vehicle's own perspective. In terms of model construction, this study adopts the idea of multimodal trajectory prediction by adding a multi-head self-attention mechanism to the encoding and decoding structure of CVAE. Additionally, a sub-model based on the principle of the VGAE model is designed, which takes the road data represented by graph structure obtained from dataset preprocessing as an input and effectively extracts the road environment features that can be embedded in the main model.

## 3. Description of the Dataset and Data Processing

The final prediction performance of a vehicle trajectory prediction model is heavily influenced by the quality of a dataset. In order to achieve simultaneous prediction for

all vehicles in the same scene, the required dataset should include complete trajectory time series data for all vehicles in specific road scenarios. Additionally, the road scenarios should be rich enough and encompass different types of road conditions, so that the trained model can have good prediction capabilities in various complex scenarios. Furthermore, in order to achieve the research goal of integrating road environment information during the training process, the dataset should be equipped with high-precision road maps for each scene and complete annotations of key road elements in the scene such as lane lines.

Argoverse 2 is a collection of open-source autonomous driving data and high-definition maps sourced from six cities in the United States [30]. This paper focuses on using the motion prediction dataset in Argoverse 2. This dataset has been improved based on the dataset in Argoverse 1, and it includes 250,000 scenes with trajectory data for multiple vehicles. Each scene lasts for 11 s with a sampling frequency of 10 Hz. Scene data includes the center and heading of each tracked object in a 2D aerial view. A key feature of this dataset is that each scene is paired with a local map. This map contains rich geometric and semantic metadata to better understand the 3D scene, such as lane boundaries, drivable area polygons, and intersection annotations, aligning with the research objectives of this paper.

The acquired public dataset cannot be directly used for model training, and needs to undergo some data preprocessing work. Raw data contains attributes unrelated to vehicle driving, such as the city of origin, and also records trajectories of other types of entities like pedestrians and motorcycles. Firstly, attribute filtering work is performed on the dataset to retain only the valid information related to a vehicle's trajectory. The retained attributes are the coordinates representing the current position of a vehicle, the velocity in the direction of the coordinate axis, the angle of orientation, the corresponding vehicle ID and the scene ID. The first 5 s of each trajectory record are separated off as inputs for the model (x), while the following 6 s are separated off as content to be predicted (y). As some vehicle trajectories in the scene may not completely cover the first 5 s, data cleaning is performed on these incomplete data. Finally, the dataset is augmented through two methods: translation augmentation and reverse trajectory augmentation.

In addition to the vehicle trajectory data, this paper also extensively processed the map data provided by Argoverse 2 in JSON format. Since the map data files are complex, and the amount of environmental information input into the deep learning model should not be excessive, it is necessary to select the most relevant information. The map data provided by Argoverse 2 includes three types of data most relevant to highway scenes: lane centerlines, left lane boundaries, and right lane boundaries. This dataset provides sets of points for these boundaries and records the two-dimensional coordinates of these points, as shown in Figure 1. The lane centerline best represents the complete topological information of the road where the vehicle is located, while the left and right lane boundaries emphasize changes in the road during abrupt behavior such as turning.

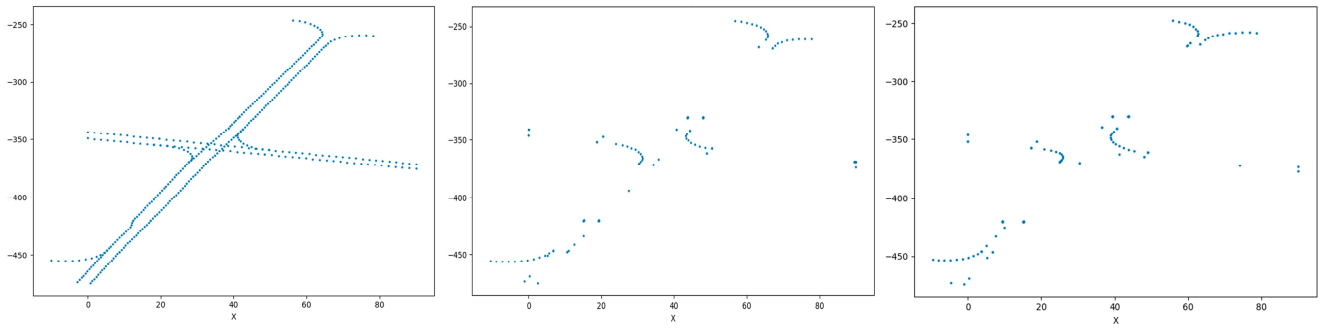


**Figure 1.** Display of the points in the map data of a scene from the Argoverse 2 dataset in plan view according to their coordinates. From (**left)** to (**right**) are the lane centerline, left boundary, and right boundary.

Extracted point collection will go through the data processing shown in Figure 2 below and finally generate inputs based on the VGAE model.

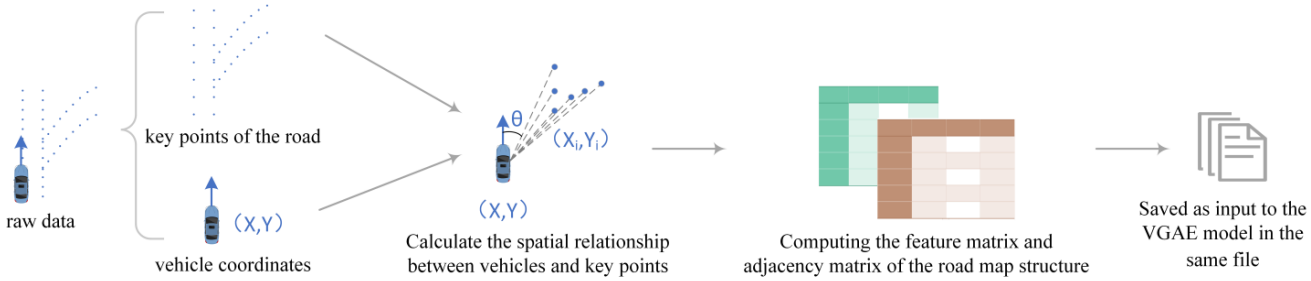


**Figure 2.** Road data preprocessing process.

The first step is the filtering of key points of the road. The reason for this step is that the length of environmental features ultimately input to the model is fixed, while the number of points recorded in different scenes varies greatly. Therefore, it is necessary to extract a fixed number of key points to eliminate the data source difference between different scenes. The main selection idea is to retain all points for the left and right lane boundaries because these points are relatively few in number and have a significant influence on drivers' decisions such as turning and lane changing. For the lane centerline, points are selected at intervals, with the interval length calculated based on the remaining available key points that can be selected.

In the second step, the spatial relationships between vehicles and all key points in the scene are calculated, including relative coordinates (Equations (1) and (2)), distances (Equation (3)), and the angles (Equation (4)) between them. It is worth stating that this process is performed on all vehicles in the scene. Thus, it ensures that all final output trajectories are fully referenced to the road information based on each vehicle's own perspective.

First, the coordinates of the last frame of a vehicle in the scene $(X, Y)$, and the coordinates of the key point of the road $(X_i, Y_i)$ are read. $N$ represents the total number of key points in this scene. The relative coordinates $\left(X_i^r, Y_i^r\right)$ are calculated by the following equation:

$$X_i^r = X - X_i,\ i \in [0, N) \tag{1}$$

$$Y_i^r = Y - Y_i,\ i \in [0, N) \tag{2}$$

Relative coordinates can also avoid the situation where the large difference in coordinate values in different scenes leads to gradient explosion in the training process. The distance and angle between the vehicle and the key road point are also calculated as follows:

$$Distance_i = \sqrt{(X - X_i)^2 + (Y - Y_i)^2},\ i \in [0, N) \tag{3}$$

$$Angle_i = arctan\left(\frac{Y - Y_i}{X - X_i}\right),\ i \in [0, N) \tag{4}$$

Finally, the features of each node obtained above are integrated into a feature matrix, and the adjacency matrix is established based on the relationship between key road points. The value of the adjacency matrix is determined based on whether the key road points belong to the same lane, and its equation is as follows:

$$adj\_norm[i][j] = \begin{cases} 1, & if\ i \neq j\ and\ lane\_id[i] = lane\_id[j] \\ 0, & otherwise \end{cases},\ i, j \in [0, N) \tag{5}$$

where $i$ and $j$ are any two key road points in a scene. $lane\_id[i]$ represents the ID of the lane where key point $i$ is located, and the same goes for $lane\_id[j]$.

In addition to the historical trajectory of a vehicle and the road information of the scene where the vehicle is situated, our model also needs to fully consider the relationships between different vehicles. This part of the work can also be completed in the data preprocessing stage, which has the benefit of avoiding repetitive calculations in each training round, and thus training time is reduced. The calculation method of the relationship matrix $R$ is improved on the PECNet model, which calculates $R$ based on whether vehicles are temporally and spatially adjacent [28]. The calculation equation for $R$ is as follows:

$$R[A][B] = \begin{cases} 0, \ if\ A = B\ or\ min_{1 \le i,j \le t_{max}} \sqrt{\left(X_i^A - X_j^B\right)^2 + \left(Y_i^A - Y_j^B\right)^2} > dis\_thresh \\ 0, \ min\left(min_{1 \le t \le t_{max}} \left|F_t^A - F_1^B\right|, min_{1 \le t \le t_{max}} \left|F_t^B - F_1^A\right|\right) > time\_thresh \\ 1, \ otherwise \end{cases} \quad (6)$$

$A$ and $B$ represent the serial numbers of any two vehicles in a scene, $t_{max}$ represents the maximum sequence number of historical trajectories (i.e., 50), and $F$ represents the corresponding timeframe in the entire scene record. After several experiments and comparisons, the maximum frame difference parameter *time_thresh* is set to 20 frames (2 s), and the maximum distance threshold *dis_thresh* is set to 1000.

## 4. Model Building

After data preprocessing in the previous section, the original dataset was processed into three parts: temporal vehicle trajectory data, the vehicle relationship matrix, and road map data. Then, the overall structure of the framework of our trajectory prediction model is shown in Figure 3:

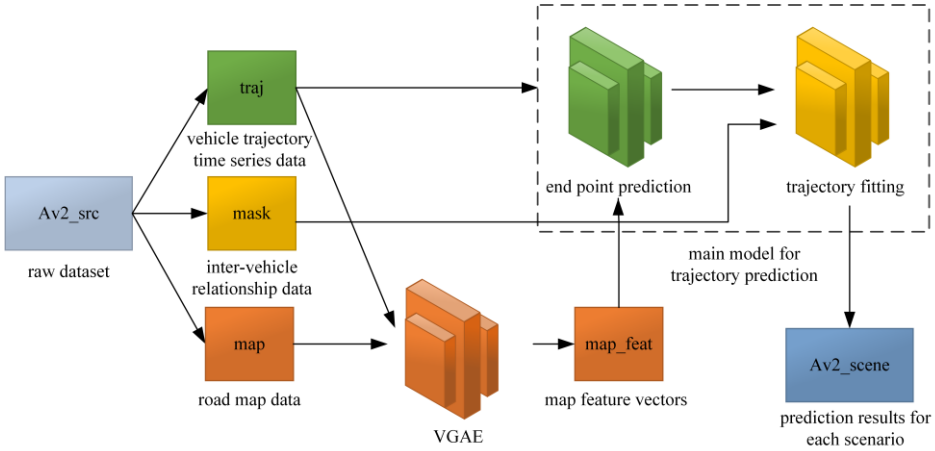


**Figure 3.** Overall structure of the framework of the trajectory prediction model.

The overall model is divided into VGAE and the main trajectory prediction model based on CVAE. Since it is necessary to generate individual feature vectors based on the perspective of each driver, the input of VGAE includes both map data and vehicle trajectory data. The main model consists of two modules. The first one is predicts the driver's destination, and then the second fits the complete trajectory based on the most probable destination. In order to fully consider road information during the training process, inputs of the destination prediction module will also include map features extracted by VGAE. Vehicle interactions are only considered in the trajectory fitting module. This design of the main model is intended to reflect the fact that a driver's final destination will not be significantly altered by the presence of surrounding vehicles, but their specific driving details are greatly influenced by them.

### *4.1. Main Model for Trajectory Prediction*

The main model structure for trajectory prediction is shown in Figure 4. The model consists of four parts. Firstly, during training, the vehicle trajectory data are divided into historical trajectory and endpoint. And they are input to two encoders, respectively, $E_{past}$ and $E_{end}$, to generate corresponding embedding representations. Secondly, the map road

data are handed over to the VGAE sub-model for specialized processing and obtains the embedding representation of the road. Thirdly, the three feature vectors obtained are concatenated and used as inputs to the endpoint prediction module. The encoder $E_{latent}$ of the CVAE model learns the distribution of the driving endpoint latent space, and the most likely trajectory endpoint is generated by the decoder $D_{latent}$. Effective improvements are made to the encoder structure of $E_{latent}$ and $D_{latent}$, including the addition of multi-head self-attention mechanisms. During the prediction phase, the generated trajectory endpoint is input to $E_{end}$ to generate the embedding representation corresponding to the predicted endpoint. Finally, outputs of the third part, together with the vehicle relationship matrix and the embedding representation of past trajectories, are input to the Social Pooling layer structure of PECNet for learning the vehicle relationships. Finally, the Predictor layer generates the complete predicted trajectory.

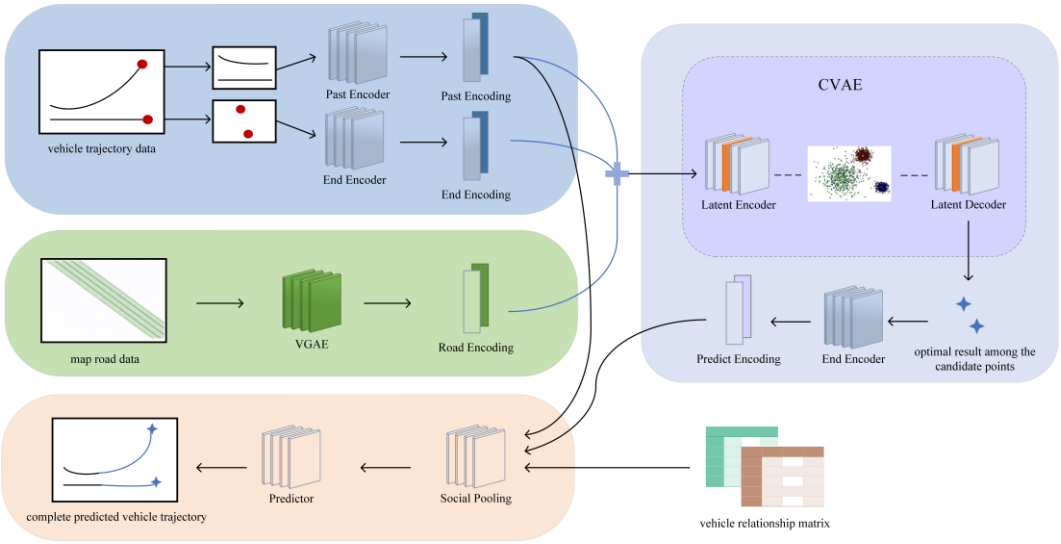


**Figure 4.** The structure of the main model for trajectory prediction.

This paper has made several significant modifications to the endpoint prediction module. Firstly, the tensors of all encoders and decoders have been enhanced with an additional "seq_len" dimension. This enhancement helps the model learn relationships between different sequence frames and also caters to the data requirements of the multi-head self-attention mechanism. Secondly, during training, the environment feature tensors processed by VGAE are integrated. Finally, after the decoder processing, a fully connected layer is added to reduce the dimensionality of the data and obtain the final predicted destination coordinates. The complete details of the training process are shown in Figure 5.

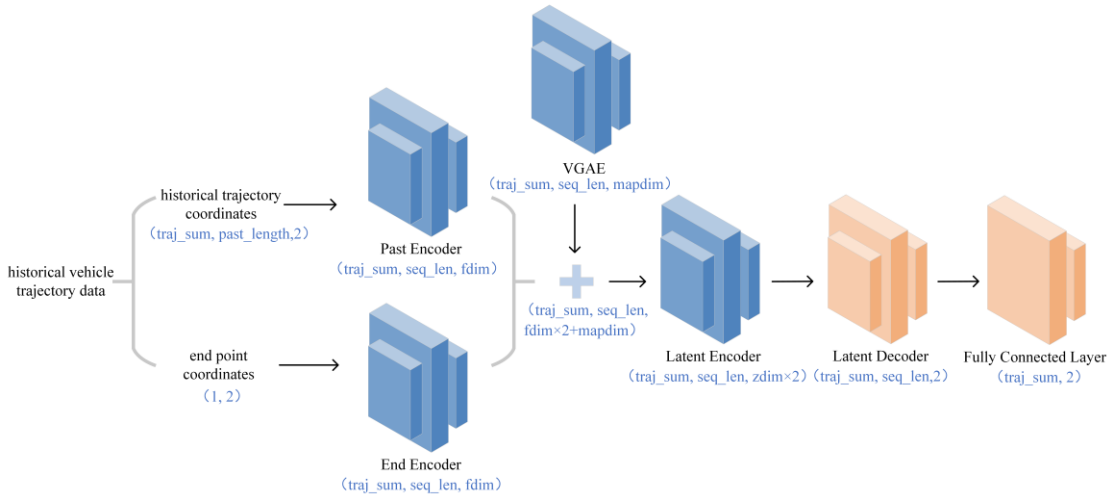


**Figure 5.** Training details of the improved endpoint prediction module.

The original encoder and decoder of PECNet were both MLP models. However, the number and duration of targets in expressway scenes have significantly increased compared to the pedestrian scenes, and the MLP alone may not be able to achieve good prediction accuracy. For this reason, the coding and decoding structure in this topic is completely revamped to include the multi-head self-attention mechanism of Transformer. The original MLP structure is used as the final part of the new autoencoder to obtain the ultimate output. Three additional layers are added before it, including the positional embedding layer, the multi-head self-attention layer, and the dropout layer. The encoder–decoder structures before and after the modification are shown in Figure 6.

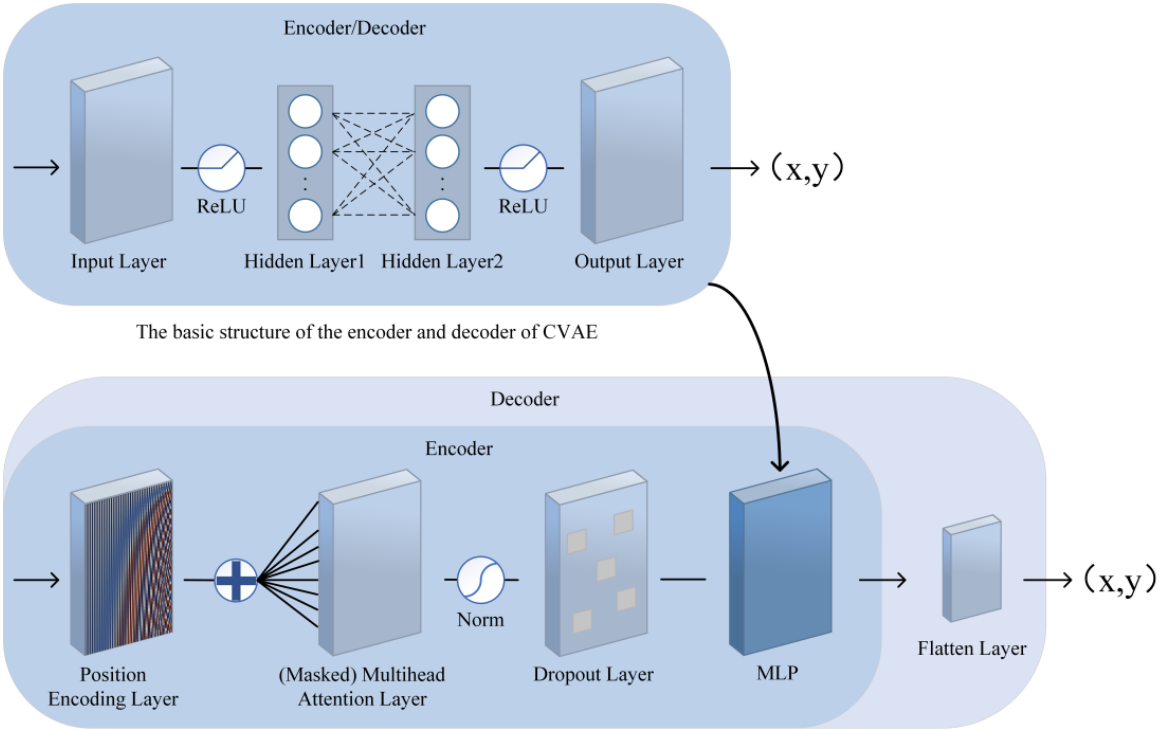


**Figure 6.** Structural details of the improved endpoint prediction module.

The position encoding layer adds the position vectors computed for each position in the input sequence to the feature vectors at the same position to better capture the dependencies between positions in the sequence. The calculation equation is as follows:

$$Position_k^i = \begin{cases} sin\frac{i}{10000^{\frac{2n}{d}}}, & k = 2n,\ n \in \mathbb{N} \\ cos\frac{i}{10000^{\frac{2n}{d}}}, & k = 2n+1,\ n \in \mathbb{N} \end{cases},\ i \in [0, N) \tag{7}$$

In Equation (7), $i$ represents the position, $k$ represents the index of the feature dimension, and $d$ represents the dimension. $N$ represents the length of the feature vector. The position encoding layer calculates position embedding vectors using trigonometric functions and adds them to the feature vectors to encode positional information. This helps the model handle sequence data more effectively without relying on a recurrent structure.

The multi-head attention layer takes a sequence with positional information as an input. The multi-head self-attention mechanism [19] calculates attention scores between each position and the other positions. Based on the attention scores, it weights the feature vectors at each position and obtains the context for each position. The attention calculation formula is as follows:

$$Attention(Q, K, V) = softmax\left(\frac{QK^T}{\sqrt{d_k}}\right)V \tag{8}$$

$K$ denotes the key vector used to compute the attention weight, $Q$ denotes the query vector, $V$ denotes the value vector, and $d_k$ denotes the dimension of the key vector. In the model, the dimensions of $K$, $Q$, and $V$ are equal. The multi-head self-attention mechanism enables the model to capture dependencies between different positions in the input sequence. This helps the model understand long-range dependencies and effectively acquire contextual information. Additionally, in practical applications, self-attention allows for parallel computation, greatly speeding up both the training and inference processes compared to sequential models like RNN.

The dropout layer normalizes and randomly drops out based on the output of the previous layer to prevent issues of vanishing or exploding gradients during multi-layer computations.

Through the above optimizations, the main model has not only achieved significant improvements in terms of parameter quantity and learning ability, but also its endpoint prediction module takes into full consideration of the road environment information collected from the vehicle's driving perspective. Comprehensive comparison and ablation experiments are set up in the next section to demonstrate the effectiveness of these improvements.

### *4.2. Map Feature Extraction Sub-Model*

In Section 3, after data preprocessing, three kinds of data are obtained: vehicle trajectory data (represented by traj), road data (map), and the vehicle relationship matrix (mask). Subsequently, traj and map for all scenes will be integrated and input into the VGAE model for training. The main purpose of this training process is to reduce information loss in the graph data feature extraction process. After obtaining a well-performing model through training, traj and map will be input to the generation module of VGAE. This module will use the extraction model obtained in the previous step to generate road feature vectors for vehicles in each scene from their perspective. The output map_feat obtained by the VGAE generation module will replace the original map and will be used as part of the input for the main model. Figure 7 illustrates the detailed processing flow of the sub-model:

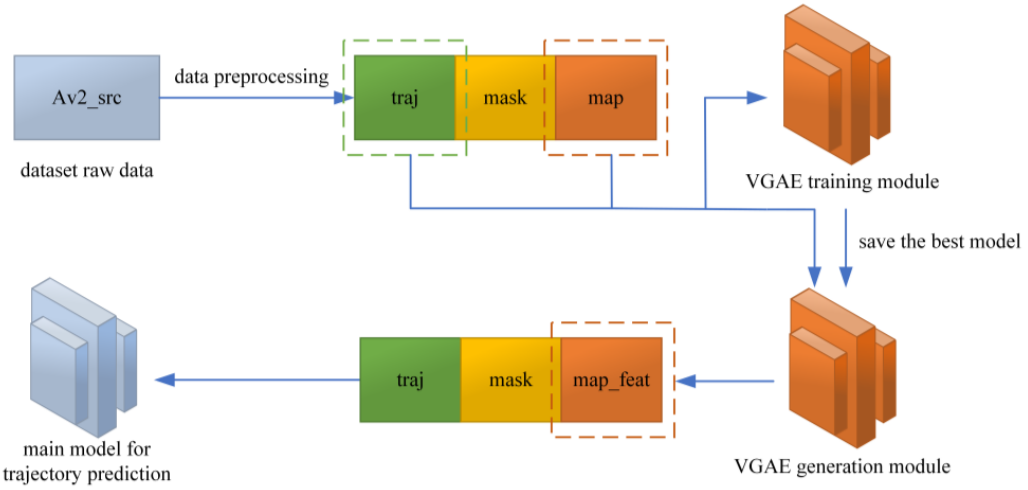


**Figure 7.** The processing flow between the map feature extraction sub-model and the main model.

Map_feat contains the feature matrix $X \in \mathbb{R}^{n \times d}$ and the adjacency matrix $A \in \mathbb{R}^{n \times n}$ for each scene, where n is the number of nodes and d is the number of features. These two matrices containing the map graph structure information will serve as inputs to the VGAE model. Prior to input, the feature matrix is normalized to scale features of different data sizes into a consistent range, and thereby the model's convergence speed is improved. Additionally, the adjacency matrix is symmetrically normalized to obtain matrix $\widetilde{A}$. The implemented VGAE model structure is as shown in Figure 8.

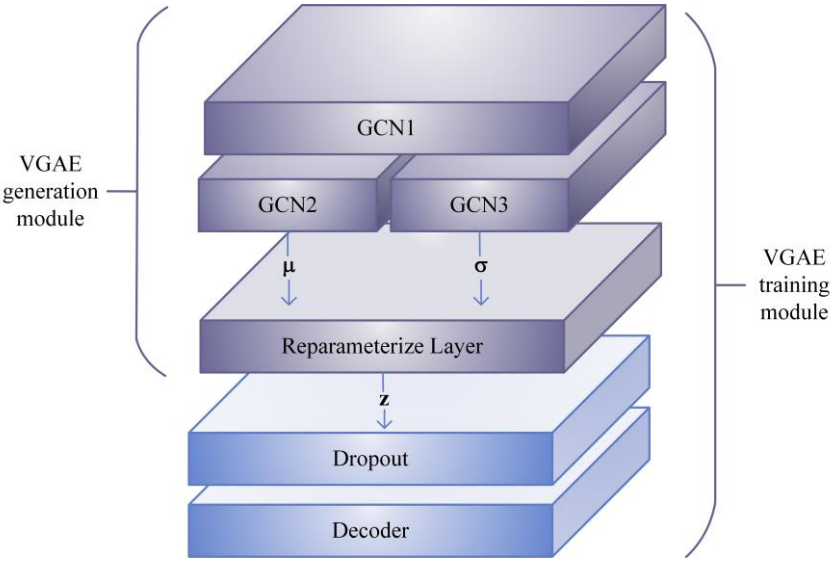


**Figure 8.** Structure of the map feature extraction sub-model based on VGAE.

The model as a whole is divided into two parts: the encoder and the decoder. The encoder is composed of a graph convolutional network (GCN) and the GCN layer computation process is as follows:

$$\mathrm{GCN}_n(X, A) = ReLU\left(\widetilde{A} X W_n\right) \tag{9}$$

$W_n$ represents the parameters learned during the $\mathrm{GCN}_n$ training process. After initial processing by $\mathrm{GCN}_1$, the intermediate tensors are given to both $\mathrm{GCN}_2$ and $\mathrm{GCN}_3$ for parallel processing to obtain the mean $\mu$ and variance $\sigma$ of the Gaussian distribution in the latent space. Subsequently, the mean and variance are input into the reparameterize layer,

which transforms the random sampling operation into a differentiable operation, enabling the effective computation and optimization of gradients through backpropagation. The equation for the reparameterization calculation is as follows:

$$z = \mu + \varepsilon \times \sigma, \ \ \varepsilon \in \mathcal{N}(0, 1) \tag{10}$$

The low-dimensional feature vector z will be used as the target output of the VGAE generation module and replace map in the original dataset as map_feat.

The decoder part first performs a dropout layer for random deactivation to reduce overfitting during training. Finally, the reconstructed adjacency matrix is output through the decoder layer. The decoder layer obtains the adjacency matrix of the reconstructed graph $A_r$ by calculating the similarity between vectors corresponding to different nodes, as shown in Equation (11). It maps the inner product value to the range of [0, 1] through the Sigmoid activation function, which represents the probability of a connection between two nodes:

$$A_r = Sigmoid\left(zz^T\right) \tag{11}$$

The decoder part will only be executed during the VGAE training process. The training process compares the difference between the reconstructed adjacency matrix and the original adjacency matrix, and continuously learns and adjusts the parameters to minimize this difference.

## 5. Experiments and Analysis

### *5.1. Training Details*

The experiments in this paper were primarily conducted on a Linux server equipped with 8 NVIDIA Titan X GPUs. The server's memory size was 64 GB, and the deep learning framework used was PyTorch (1.13.1). The dataset of Argoverse 2 is large, with a compressed file size of nearly 50 GB. Therefore, training on this huge dataset took a very long time and required good engineering optimizations. To optimize the model training speed, the approach of multi-process distributed training was chosen to fully utilize the multiple GPU resources of the server. The benchmark model PECNet itself does not support multi-process distributed training, so the entire experiment was refactored using torch.distributed for this purpose. Torch.distributed is a set of distributed training tools provided by PyTorch. It can open multiple processes on the same machine. Each process can specify the GPU to be used during training, thereby achieving the effect of accelerating training with multiple GPUs.

Large datasets pose a significant challenge to hardware devices, especially in terms of memory. It is very easy to encounter out-of-memory (OOM) errors and terminate the training program. By default, each process created by torch.distributed will read the entire dataset into memory, which leads to serious duplicate reading issues. To address this problem, the dataset is divided equally based on the number of processes in advance, and the reading logic of each process is modified to only read the corresponding subset of data. This approach ensures that no matter how many new processes are started, only one complete dataset is read.

To visually demonstrate the difference in training speed before and after optimization, a set of comparative experiments was conducted. The first experiment utilized one NVIDIA Titan X GPU without enabling multi-GPU training, while the second experiment employed all eight NVIDIA Titan X GPUs of the server and synchronized training with eight processes. Both experiments were conducted on the exact same dataset. Experimental results are shown in Table 1. From Table 1, we can see that the code runs more than 14 times faster when using the optimized method.

**Table 1.** Comparison of training duration before and after multi-process optimization.

| Training Time/s | Before (1 Titan X) | Optimized (8 Titan X) |
|---|---|---|
| 20 Epoch | 4218.33 s (1 h 10 min) | 289.73 s (4.83 min) |
| Average per Epoch | 210.92 s (3 min 31 s) | 14.49 s |

The model used the optimal hyperparameter combinations in all subsequent experiments. The learning rate was set to 0.0005. The number of features (fdim) was 64. The number of latent space features (zdim) was 64. The number of map features (mapdim) was 128. The number of heads in the multi-head self-attention mechanism was eight. The batch size was 512. The number of generated candidate endpoints was 20, and the training epoch was set to 650.

### 5.2. Evaluation Measures

Average displacement error (ADE) and final displacement error (FDE) are among the most commonly used performance metrics in the field of trajectory prediction, and in this paper we will primarily use these two metrics to evaluate model prediction accuracy. ADE measures the average difference between the predicted trajectory and the ground truth trajectory. ADE is calculated as follows:

$$\mathrm{ADE} = \frac{1}{m}\sum_{i=1}^{m}\sqrt{(X_i - x_i)^2 + (Y_i - y_i)^2} \tag{12}$$

FDE measures the Euclidean distance between the predicted position and the actual position at the last time step. FDE evaluates the difference between the predicted endpoint of the trajectory and the true endpoint. FDE is calculated as follows:

$$\mathrm{FDE} = \sqrt{\left(X_f - x_f\right)^2 + \left(Y_f - y_f\right)^2} \tag{13}$$

### 5.3. Ablation Experiments

To independently verify the effectiveness of the improvements made to the original baseline model, we designed a set of comparative experiments that did not integrate the map feature extraction sub-model. A random sample of 10,000 scenes from the Argoverse 2 motion forecasting training set was selected as the training set, and 1000 scenes were used as the test set. Training was conducted for 100 epochs on both the unmodified baseline model and the improved model, and the best prediction results on the test set are summarized in Table 2 below:

**Table 2.** Comparison of predictive indicators before and after improvement of the main model for trajectory prediction. The two models used their respective best hyperparameter combinations.

| Model | Learning_Rate | Fdim | Num_Head | Test ADE | Test aveFDE |
|---|---|---|---|---|---|
| OLD | 0.0003 | - | - | 0.7256 | 1.7130 |
| NEW | 0.0005 | 64 | 8 | 0.6515 | 1.6843 |

From the experiment results, it can be seen that the predictive performance of the model improved significantly after multiple enhancements, surpassing the previous model. The overall trajectory prediction accuracy increased by over 10%. In order to further demonstrate the effectiveness of each improvement on the CVAE baseline model, comprehensive ablation experiments were designed. The ablation experiments were performed on identical small sample datasets and none of them incorporated the map feature extraction sub-model. The experiment details and results are presented in Table 3.

**Table 3.** The ablation experiment results for the improvements made to the CVAE model.

| Experiment | Test ADE |
|---|---|
| Complete improved model | 1.0127 |
| Remove the positional encoding layer alone | 1.0630 |
| Remove the dropout layer alone | 1.1340 |
| Remove the multi-head attention layer alone | 1.1544 |

The results verify that removing any additional layer added to the autoencoder will lead to a significant decrease in the predictive metrics. In particular, removing the multi-head attention layer will result in the largest decrease, indicating that the multi-head self-attention mechanism indeed effectively enhances the model's ability to capture information from sequential data.

Finally, in order to demonstrate that adding the feature vectors generated by the map feature extraction sub-model to the main model is an effective improvement, a comparative experiment was conducted. This experiment was trained for 100 epochs with and without map feature vectors, respectively, under the condition that the irrelevant factors such as the amount of data and hyperparameters were consistent. The optimal prediction performance obtained is recorded in the following Table 4:

**Table 4.** Comparison of model prediction indicators before and after adding road feature vectors.

| Experiment | Best_Epoch | RCL | KLD | ADL | Test ADE |
|---|---|---|---|---|---|
| Before | 80 | 550.53 | 0.5568 | 161.64 | 2.6188 |
| After | 78 | 299.23 | 0.3172 | 75.67 | 2.5018 |

In addition to the ADE of the entire test set, Table 4 also displays the loss results of different parts of trajectory prediction during the training process. RCL represents trajectory error. KLD represents KL divergence error, and ADL represents endpoint prediction error. It is evident that the losses in each part have been effectively reduced after incorporating the map information. This suggests that the fusion of VGAE-based sub-model has led to an effective improvement in both the prediction of driving endpoints and the fitting of intermediate trajectories.

### *5.4. Comparison with Other Existing Models*

First, a comparison of predictive metrics of several models in the Argoverse 1 motion forecasting dataset is completed. Generalized benchmark models such as LaneGCN, DenseTNT, and SceneTransformer, which are excellent performers in the field of vehicle trajectory prediction, as well as new models such as THOMAS and HcteroGCN, which have made breakthroughs in prediction metrics in recent years, were selected as the comparison targets.

LaneGCN uses a novel structured map representation instead of the traditional grid map and develops a fusion network to learn lane graph features and interactions between vehicles and maps [31]. DenseTNT transforms the trajectory prediction problem into a probability problem for predicting the future distribution of the driving space, outputting multiple possible trajectories for vehicles and the corresponding probabilities for their selection [32]. SceneTransformer focuses on the joint prediction of multiple predictive entities, incorporating attention mechanisms and employing a masked sequence modeling strategy to enhance model performance [33]. THOMAS proposes a joint multi-agent trajectory prediction framework to efficiently predict the future motion trajectories of multiple targets simultaneously [34]. HcteroGCN introduces a heterogeneous graph convolutional recursive network to learn various interactions and spatiotemporal information in the scene, achieving higher prediction accuracy on public datasets [35].

The results of the comparative experiments are shown in Table 5 below in terms of the metrics minADE (K = 1) and minFDE (K = 1). From the comparison results, it is evident that

both the minADE and minFDE of our proposed model outperform the other five compared trajectory prediction models.

**Table 5.** Comparison of metrics of six trajectory prediction models based on the Argoverse 1 motion forecasting dataset.

| Method | minADE (K = 1) | minFDE (K = 1) |
|---|---|---|
| LaneGCN | 1.71 | 3.78 |
| DenseTNT | 1.68 | 3.63 |
| SceneTransformer | 1.81 | 4.06 |
| THOMAS | 1.67 | 3.59 |
| HcteroGCN | 1.62 | 3.52 |
| CVAE-VGAE (this paper) | 1.59 | 3.44 |

Additionally, based on the Argoverse 2 motion forecasting dataset, we also completed a prediction metric comparison between our proposed model and some models that performed well on this dataset. For instance, HcteroGCN proposes a heterogeneous graph convolutional recursive network to learn various interactions and spatiotemporal information in the scene, achieving higher prediction accuracy on the Argoverse 2 dataset. The comparative results are shown in Table 6 below.

**Table 6.** Comparison of metrics for trajectory prediction models based on the Argoverse 2 motion forecasting dataset.

| Method | minADE (K = 1) | minFDE (K = 1) |
|---|---|---|
| THOMAS | 1.96 | 4.71 |
| GoReLa | 1.82 | 4.62 |
| GANet | 1.78 | 4.48 |
| BANet | 1.79 | 4.61 |
| HcteroGCN | 1.79 | 4.53 |
| CVAE-VGAE (this paper) | 1.73 | 4.21 |

The experimental results demonstrate that our model's prediction capability has significantly improved compared to THOMAS, and the final prediction performance is also superior to HcteroGCN. Additionally, thanks to multiple effective enhancements to the driving endpoint prediction module, it can be observed that the improvement of minFDE (7%) for calculating the distance between the final predicted position and the real position is significantly better than that of minADE (3.4%), which is also in line with the results of previous ablation experiments.

### *5.5. Visualization of Predictions under Different Scenarios*

In order to visualize the prediction results under different scenarios, this paper also developed a tool that can complete the visualization of road maps and all vehicle trajectories (both predicted and real values) for specified scenarios. In this section, the prediction results under several different types of typical scenarios are shown and analyzed to demonstrate the actual prediction effect of our model. The selected scenarios are mainly based on different road conditions (merging, straight lines, off-ramps, etc.) and traffic flows.

The exit ramp and its surrounding area on the expressway are accident-prone sections. Figure 9 depicts a typical exit ramp scenario. In the image, the gray lines represent known historical vehicle trajectories. The green lines represent the actual results, the red lines represent the model's prediction results. The blue dots represent the results from the endpoint prediction module. This scenario captures five vehicles traveling in the same direction, with two vehicles driving in the leftmost lane. The real trajectory of the leading vehicle in this lane indicates that the vehicle is going to exit from the ramp. The model accurately predicts the lane-changing behavior of this vehicle as it exits. The predicted future driving endpoint closely matches the actual result for this vehicle. Moreover, the

endpoint predictions for all vehicles in this scenario are relatively accurate, demonstrating the effectiveness of conducting endpoint intention inference beforehand.

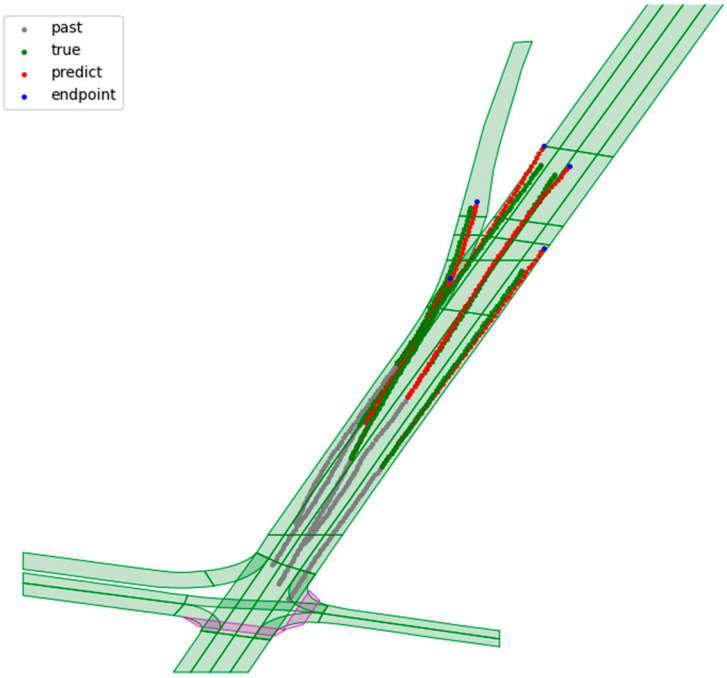


**Figure 9.** Visualization of predictions for a multi-vehicle scenario at an exit ramp.

The scenario in Figure 10 below is similar to the most common situation of multiple vehicles driving straight on expressways. For straight-driving vehicles with stable operating conditions, our model can more accurately predict the driving endpoints and fit the trajectories. At the same time, for stopped vehicles that are located in the right lane from left to right, their stopping status will also be accurately predicted.

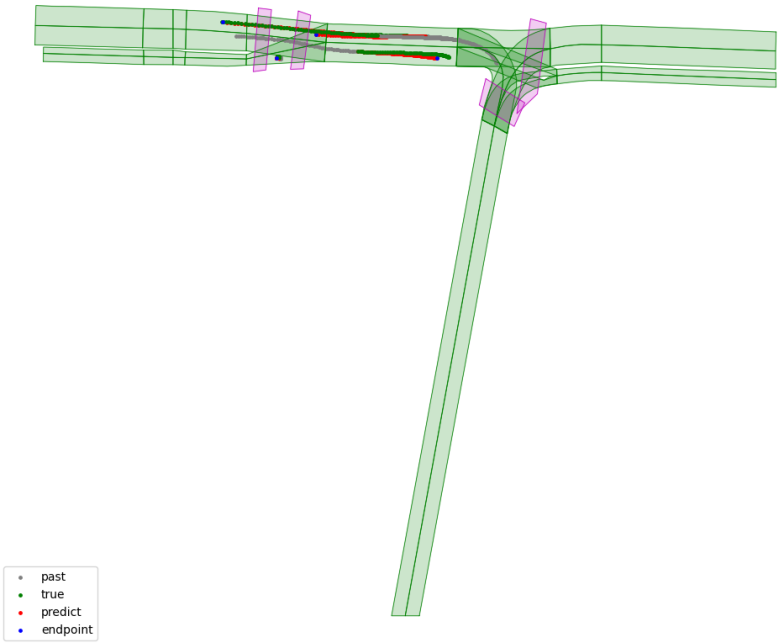


**Figure 10.** Visualization of predictions for a scenario with some vehicles traveling straight.

Predicting turns and lane changes is one of the most challenging situations in trajectory prediction. Figure 11 includes two common turn prediction scenarios. The past trajectory

of the vehicle above includes part of the turning trajectory, so it is relatively easy to predict its future trajectory. However, the past trajectory of the vehicle below does not show a clear turning process, so it is relatively difficult to make the prediction. In both cases, our model successfully predicts the driving intention, and the prediction results closely match the actual trajectories.

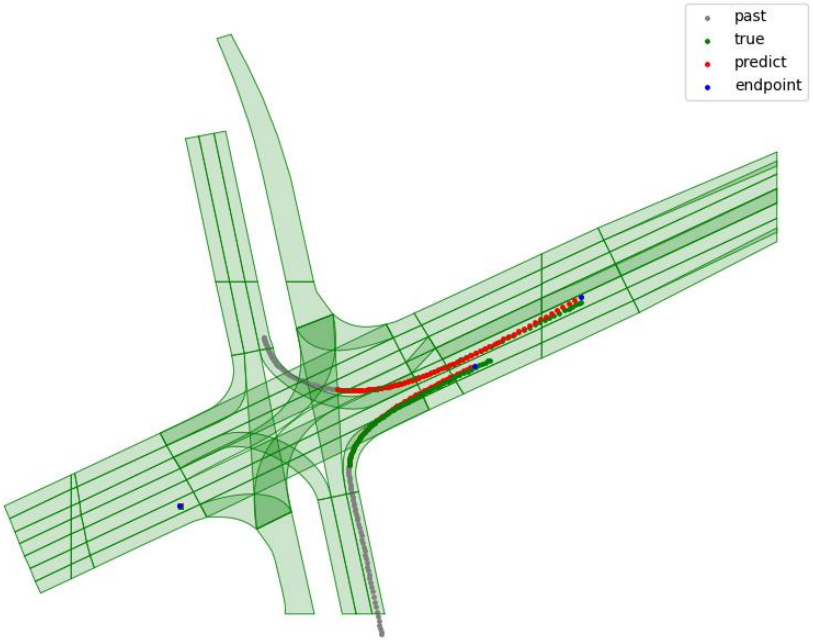


**Figure 11.** Visualization of predictions for a scenario containing vehicle turns.

## 6. Conclusions and Future Work

This paper delves into the analysis of accident-prone road sections on expressways, investigating the combined effects of historical trajectories, surrounding vehicles, and road environment on drivers' driving behavior. Building upon state-of-the-art models in the field, the paper proposes improvements in the CVAE module structure by incorporating a self-attention mechanism to enhance learning capabilities for long-term time series data. An independent environmental information extraction sub-model based on the VGAE model is also designed and integrated into the main model to address the issue that the original model did not consider the vehicle driving environment. Other work, such as data preprocessing and scenario visualization analysis, are also introduced. Through detailed and comprehensive experiments, the paper demonstrates the effectiveness of our model, showing its good predictive performance in multi-vehicle motion forecasting. Furthermore, visualization results across various practical scenarios show that the model can provide strong technical support for key applications such as collision warning in intelligent transportation systems.

There are areas for further improvement in this model, which will be the focus of our future work. The road information extraction method proposed in this paper can be improved by incorporating more attributes that reflect the spatiotemporal relationships between vehicles and road elements. The map feature extraction sub-model in this study can be further optimized by improving the model structure and map data processing methods. Meanwhile, since the training process of this model uses high-definition map data, which is often not available in practical applications, specific processing will be needed for low-precision map data in the practical application of our method.

**Author Contributions:** Conceptualization, Y.D.; methodology, Y.D.; software, Y.D.; validation, Y.D. and J.H.; formal analysis, Y.D.; investigation, Y.D., J.J. and H.Z.; resources, Y.D. and Y.J.; data curation, Y.D.; writing—original draft preparation, Y.D.; writing—review and editing, Y.D., J.J., Y.J. and J.H.;

visualization, Y.D.; supervision, Y.J., J.J. and J.H.; project administration, J.H.; funding acquisition, J.H. All authors have read and agreed to the published version of the manuscript.

**Funding:** This work is supported by the 2022–2025 National Key Research and Development Program of China, No. 2022YFB2602104.

**Institutional Review Board Statement:** Not applicable.

**Informed Consent Statement:** Not applicable.

**Data Availability Statement:** The public dataset used to support the results of this study is available from the official Argoverse 2 website at https://www.argoverse.org/av2 (accessed on 2 March 2023). This study has processed the raw data extensively, and the processed data are available from the corresponding author upon reasonable request.

**Conflicts of Interest:** The authors declare no conflicts of interest.

## References

1. Abduljabbar, R.; Dia, H.; Liyanage, S.; Bagloee, S.A. Applications of artificial intelligence in transport: An overview. *Sustainability* **2019**, *11*, 189. [CrossRef]
2. Ma, Y.; Ma, C.; Lv, C.; Zhang, S.; Tian, Y.; Zhao, T.; Du, C.; Wu, J. Vehicle Trajectory Prediction in Expressway Merging Areas Based on Self-Supervised Mechanism. *J. Transp. Eng. A Syst.* **2024**, *150*, 04024013. [CrossRef]
3. Gupta, A.; Johnson, J.; Fei-Fei, L.; Savarese, S.; Alahi, A. Social gan: Socially acceptable trajectories with generative adversarial networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 2255–2264.
4. Rhinehart, N.; McAllister, R.; Kitani, K.; Levine, S. Precog: Prediction conditioned on goals in visual multi-agent settings. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Seoul, Republic of Korea, 27 October–2 November 2019; pp. 2821–2830.
5. Barth, A.; Franke, U. Where will the oncoming vehicle be the next second? In Proceedings of the 2008 IEEE Intelligent Vehicles Symposium, Eindhoven, The Netherlands, 4–6 June 2008; pp. 1068–1073.
6. 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.
7. Danielsson, S.; Petersson, L.; Eidehall, A. Monte carlo based threat assessment: Analysis and improvements. In Proceedings of the 2007 IEEE Intelligent Vehicles Symposium, Istanbul, Turkey, 13–15 June 2007; pp. 233–238.
8. Tomar, R.S.; Verma, S.; Tomar, G.S. Prediction of lane change trajectories through neural network. In Proceedings of the 2010 International Conference on Computational Intelligence and Communication Networks, Bhopal, India, 26–28 November 2010; pp. 249–253.
9. Gao, H.; Qin, Y.; Hu, C.; Liu, Y.; Li, K. An interacting multiple model for trajectory prediction of intelligent vehicles in typical road traffic scenario. *IEEE Trans. Neural Netw. Learn. Syst.* **2021**, *34*, 6468–6479. [CrossRef] [PubMed]
10. Kordmahalleh, M.M.; Sefidmazgi, M.G.; Homaifar, A. A Sparse Recurrent Neural Network for Trajectory Prediction of Atlantic Hurricanes. In Proceedings of the Genetic and Evolutionary Computation Conference, Denver, CO, USA, 20–24 July 2016; pp. 957–964.
11. Pool, E.A.I.; Kooij, J.F.P.; Gavrila, D.M. Context-based cyclist path prediction using Recurrent Neural Networks. In Proceedings of the 2019 IEEE Intelligent Vehicles Symposium (IV), Paris, France, 9–12 June 2019; pp. 824–830.
12. Sherstinsky, A. Fundamentals of recurrent neural network (RNN) and long short-term memory (LSTM) network. *Physica D* **2020**, *404*, 132306. [CrossRef]
13. Altché, F.; de La Fortelle, A. An LSTM network for highway trajectory prediction. In Proceedings of the 2017 IEEE 20th International Conference on Intelligent Transportation Systems (ITSC), Yokohama, Japan, 16–19 October 2017; pp. 353–359.
14. Park, S.H.; Kim, B.; Kang, C.M.; Chung, C.C.; Choi, J.W. Sequence-to-sequence prediction of vehicle trajectory via LSTM encoder-decoder architecture. In Proceedings of the 2018 IEEE Intelligent Vehicles Symposium (IV), Changshu, China, 26–30 June 2018; pp. 1672–1678.
15. Dai, S.; Li, L.; Li, Z. Modeling vehicle interactions via modified LSTM models for trajectory prediction. *IEEE Access* **2019**, *7*, 38287–38296. [CrossRef]
16. Wang, S.; Zhao, P.; Yu, B.; Huang, W.; Liang, H. Vehicle trajectory prediction by knowledge-driven LSTM network in urban environments. *J. Adv. Transp.* **2020**, *2020*, 1–20. [CrossRef]
17. Goodfellow, I.; Pouget-Abadie, J.; Mirza, M.; Xu, B.; Warde-Farley, D.; Ozair, S.; Courville, A.; Bengio, Y. Generative adversarial networks. *Commun. ACM* **2020**, *63*, 139–144. [CrossRef]
18. Lee, N.; Choi, W.; Vernaza, P.; Choy, C.B.; Torr, P.H.; Chandraker, M. Desire: Distant future prediction in dynamic scenes with interacting agents. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 336–345.

19. Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.N.; Kaiser, Ł.; Polosukhin, I. Attention is all you need. *Adv. Neural Inf. Process. Syst.* **2017**, *30*, 5998–6008.
20. Liu, Y.; Zhang, J.; Fang, L.; Jiang, Q.; Zhou, B. Multimodal motion prediction with stacked transformers. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Nashville, TN, USA, 20–25 June 2021; pp. 7577–7586.
21. Chen, X.; Zhang, H.; Zhao, F.; Cai, Y.; Wang, H.; Ye, Q. Vehicle trajectory prediction based on intention-aware non-autoregressive transformer with multi-attention learning for Internet of Vehicles. *IEEE Trans. Instrum. Meas.* **2022**, *71*, 1–12. [CrossRef]
22. Lv, P.; Wu, W.; Zhong, Y.; Du, F.; Zhang, L.; Sensing, R. SCViT: A spatial-channel feature preserving vision transformer for remote sensing image scene classification. *IEEE Trans. Geosci. Remote Sens.* **2022**, *60*, 1–12. [CrossRef]
23. Zhang, K.; Feng, X.; Wu, L.; He, Z. Trajectory prediction for autonomous driving using spatial-temporal graph attention transformer. *IEEE Trans. Intell. Transp. Syst.* **2022**, *23*, 22343–22353. [CrossRef]
24. Zhang, E.; Zhang, R.; Masoud, N. Predictive trajectory planning for autonomous vehicles at intersections using reinforcement learning. *Transp. Res. Part C Emerg. Technol.* **2023**, *149*, 104063. [CrossRef]
25. Huang, S.; Li, X.; Zhang, Z.; He, Z.; Wu, F.; Liu, W.; Tang, J.; Zhuang, Y. Deep learning driven visual path prediction from a single image. *IEEE Trans. Image Process.* **2016**, *25*, 5892–5904. [CrossRef] [PubMed]
26. Bansal, M.; Krizhevsky, A.; Ogale, A. Chauffeurnet: Learning to drive by imitating the best and synthesizing the worst. *arXiv* **2018**, arXiv:03079.
27. Cai, Y.; Wang, Z.; Wang, H.; Chen, L.; Li, Y.; Sotelo, M.A.; Li, Z. Environment-attention network for vehicle trajectory prediction. *IEEE Trans. Veh. Technol.* **2021**, *70*, 11216–11227. [CrossRef]
28. Gao, J.; Sun, C.; Zhao, H.; Shen, Y.; Anguelov, D.; Li, C.; Schmid, C. Vectornet: Encoding hd maps and agent dynamics from vectorized representation. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 13–19 June 2020; pp. 11525–11533.
29. Mangalam, K.; Girase, H.; Agarwal, S.; Lee, K.-H.; Adeli, E.; Malik, J.; Gaidon, A. It is not the journey but the destination: Endpoint conditioned trajectory prediction. In Proceedings of the Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, 23–28 August 2020; Part II 16. pp. 759–776.
30. Wilson, B.; Qi, W.; Agarwal, T.; Lambert, J.; Singh, J.; Khandelwal, S.; Pan, B.; Kumar, R.; Hartnett, A.; Pontes, J.K. Argoverse 2: Next generation datasets for self-driving perception and forecasting. *arXiv* **2023**, arXiv:00493.
31. Liang, M.; Yang, B.; Hu, R.; Chen, Y.; Liao, R.; Feng, S.; Urtasun, R. Learning lane graph representations for motion forecasting. In Proceedings of the Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, 23–28 August 2020; Part II 16. pp. 541–556.
32. Gu, J.; Sun, C.; Zhao, H. Densetnt: End-to-end trajectory prediction from dense goal sets. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Montreal, BC, Canada, 11–17 October 2021; pp. 15303–15312.
33. Ngiam, J.; Caine, B.; Vasudevan, V.; Zhang, Z.; Chiang, H.-T.L.; Ling, J.; Roelofs, R.; Bewley, A.; Liu, C.; Venugopal, A. Scene transformer: A unified architecture for predicting multiple agent trajectories. *arXiv* **2021**, arXiv:08417.
34. Gilles, T.; Sabatini, S.; Tsishkou, D.; Stanciulescu, B.; Moutarde, F. Thomas: Trajectory heatmap output with learned multi-agent sampling. *arXiv* **2021**, arXiv:06607.
35. Gao, X.; Jia, X.; Li, Y.; Xiong, H.; Letters, A. Dynamic scenario representation learning for motion forecasting with heterogeneous graph convolutional recurrent networks. *IEEE Robot. Autom. Lett.* **2023**, *8*, 2946–2953. [CrossRef]