Trajectory Prediction Neural Network and Model Interpretation Based on Temporal Pattern Attention

Hongyu Hu, *Member, IEEE*, Qi Wang, Ming Cheng, and Zhenhai Gao

***Abstract*— High-precision vehicle trajectory prediction can enable autonomous vehicles to provide a safer and more comfortable trajectory planning and control. Unfortunately, current trajectory prediction methods have difficulty extracting hidden driving features across multiple time steps, which is important for long-term prediction. In order to solve this shortcoming, a temporal pattern attention-based trajectory prediction network, named TP2Net, was proposed, and vehicle of interest inception was established to construct an interaction model among vehicles. Experimental results show a 15% improvement in predictive performance over the previous best method under a 5-s prediction horizon. Moreover, in order to explain why temporal pattern attention was adopted and demonstrate its ability to extract hidden features that are intuitive to human beings, a layer interpretation module was included in TP2Net to quantify the mutual information contained between the input and the intermediate layer output tensor. The results of experiments using naturalistic trajectory datasets indicated that temporal pattern attention can extract three important stages in lane changing, showing that temporal pattern attention can effectively extract hidden features and improve prediction accuracy.**

***Index Terms*— Trajectory prediction, hidden driving features, temporal pattern attention, model interpretation.**

I. Introduction

AN AUTONOMOUS vehicle is understood to have the ability to perceive more comprehensive social knowledge in the driving context than human drivers through its installed sensors. Based on the perception, the autonomous vehicle needs to infer the intention of surrounding agents and make high-precision estimates of their future trajectories [1], [2]. Trajectory prediction can generally help autonomous vehicles to better understand the future driving environment and determine the subsequent tactical maneuver [3]. However, the multi-modality of driving behavior and the complexity of the driving context make the task of trajectory prediction a considerable challenge [4].

In fact, in a specific driving context, the tactical intention of a vehicle driver has a significant influence on their multi-modal driving behavior [5], [6]. The tactical intention is defined as the quick decision made by a driver to achieve the corresponding goal through a series of driving operations [1], [7]. In other words, the driver of a vehicle must maintain an understanding of the driving context in order to perform reasonable and comfortable driving operations. Therefore, the driving intention and maneuvering pattern of a driver, extracted from their interaction with the surrounding agents and the mobility of the target vehicle, may be helpful for enabling autonomous vehicles to make longer horizon predictions.

Vehicle trajectory prediction is a multivariable time series (MTS) task for which recurrent neural networks (RNNs) are often used to predict the sequence of events [8]. Unfortunately, due to a lack of long-term dependency management [9], there are several weaknesses in the extraction of long-term dependency relationships using RNNs that may affect trajectory prediction. Generally, a driver does not execute their intention immediately when it appears but executes the previous driving intention after preparing the driving control action [7]. It takes an average of 1 to 4 s to execute a driving maneuver after the appearance of a driving intention [10], [11]. Existing RNNs generally have weak recognition ability across multiple time steps, so it is difficult to capture the hidden long-term pattern between the driving intention and subsequent maneuver [9], [12]. In order to address these problems, a Temporal Pattern attention-based Trajectory Prediction Network (TP2Net) was proposed in this study and introduced into the trajectory prediction task to extract the hidden driving features pertaining to the target vehicle. This attention mechanism is effective because it can span multiple time steps, which is particularly suitable for the extraction of hidden driving intention and maneuver patterns. Moreover, considering that the interaction between the target vehicle and the surrounding agents, the use of vehicle of interest (VOI) inception was proposed based on the GoogLeNet [13], [14], simplifying the input to focus on the surrounding vehicles with sufficient interaction influence.

Although trajectory prediction neural networks, similar to TP2Net, have been shown to provide better performance and accuracy. Such deep learning models are often considered to be black boxes because they cannot provide a meaningful explanation of how a prediction or decision was made [15], [16]. Indeed, trust increases when a model is shown to base its decisions on environmental aspects that appear reasonable to a human [17]. The proposed TP2Net extracts hidden driving features through temporal pattern attention (TPA), but without a sufficient transparency, the reliability of this method would

Manuscript received 21 January 2021; revised 21 April 2022 and 25 June 2022; accepted 2 November 2022. Date of publication 10 November 2022; date of current version 1 March 2023. This work was supported in part by the National Natural Science Foundation of China under Grant 52272417, in part by the Natural Science Foundation of Jilin Province under Grant 20210101064JC, and in part by the Science and Technology Development Project of Jilin Province under Grant 20210301030GX. The Associate Editor for this article was G. Ostermayer. *(Corresponding author: Zhenhai Gao.)*

The authors are with the State Key Laboratory of Automotive Simulation and Control, Jilin University, Changchun 130022, China (e-mail: huhongyu@jlu.edu.cn; dfefqbnb2@hotmail.com; chengming20@mails.jlu.edu.cn; gaozh@jlu.edu.cn).

Digital Object Identifier 10.1109/TITS.2022.3219874

be considered relatively poor. Therefore, a trajectory prediction network interpretation module was proposed to quantify the relative importance between the input and output tensors of the TPA model. Neural network interpretation methods are divided into active approach (actively changes the network architecture or training process) and passive approach (post hoc explain trained neural networks). The proposed interpretation method belongs to passive approach [18]; that is, it will not affect the training of the model. By using information entropy theory [15], the importance of each time step of the trajectory to the prediction can be quantified through the trained TP2Net. Although it is difficult to directly interpret this importance as the driver's intention–maneuver pattern, improving the interpretability of the trajectory prediction network can quantitatively represent the importance of each time step before a specific driving maneuver, providing a theoretical basis for the study of personalized and anthropomorphic autonomous driving.

The main contributions of this study can be summarized as follows:

1) A trajectory prediction network, TP2Net, was proposed based on TPA, which is more suitable for extracting hidden multimodal driving features and solves the weakness of RNNs in managing short-term dependencies. In addition, the Sigmoid function was adopted to replace the Softmax function in order to improve the multiple temporal pattern extraction ability. Finally, VOI inception was used to better extract the social context knowledge.
2) An interpretation module was proposed to explain the mutual information between TPA input and output tensors. The theory of information entropy was adopted, and the perturbation method was then used to measure the information extracted by TPA under specific driving maneuvers.
3) The relative importance of each time step in the process of left and right lane changing was studied, and multiple response peaks were extracted. Three important stages of lane changing were identified using a relative importance line chart of lane change data, and were shown to correspond to the statistical data describing vehicle driving dynamics. It was thus proven that TPA can effectively extract hidden driving features across multiple time steps.

The remainder of this paper is organized as follows. **Section II** provides a review of literature pertaining to the existing research on vehicle trajectory prediction and network interpretation. **Section III** describes the methodology of the present study, including the overall framework and details. The experimental settings, results, and evaluations are described in **Section IV**. Finally, **Section V** presents concluding remarks and highlights the scope for future work.

II. Related Research

Many researchers have investigated trajectory prediction from different perspectives, including conventional machine learning methods and deep learning methods. The typically employed neural network interpretation methods are the gradient method and perturbation method. This section briefly discusses previous research into trajectory prediction related to this study as well as the interpretation of neural networks.

A. Methods of Trajectory Prediction

1) Conventional Machine Learning Methods: Kinematic and dynamic parameters were often used in early vehicle trajectory prediction tasks. For example, the constant yaw rate and acceleration model has often been employed because it assumes that the yaw rate and acceleration will not change suddenly in a short period of time during smooth driving [19]. Bayesian filters such as the unscented Kalman filter and extended Kalman filter have also been used as models for trajectory prediction. However, these models perform poorly on long-term prediction tasks because the short-term assumptions no longer hold for the long-term prediction horizon [20].

Because driving maneuvers can better reflect a driver's long-term intentions, many researchers have proposed models based on driving maneuvers to improve the accuracy of long-term horizon trajectory prediction. Driving maneuver recognition is mainly based on historical vehicle trajectories and vehicle motion states. For example, Houenon et al. [21] proposed a model combining a constant yaw rate and acceleration model with driving maneuver recognition. The optimal trajectory was selected from the trajectory cluster predicted by this model according to the recognized driving maneuver. The hidden Markov model (HMM) [22], Gaussian process regression [23], support vector machine [26], and probabilistic finite state machine [24] have also been used for driving maneuver recognition. These methods mainly focus on the driving state and maneuvering of the target vehicle. Unfortunately, they ignore any interaction with surrounding vehicles, and exhibit a large prediction error in complex traffic contexts.

Combining the interaction model with driving maneuver recognition solves the problem of accounting for the interaction between the target vehicle and surrounding vehicles. Such combined methods mainly assign different weights according to the degree of influence of each surrounding vehicle in order to focus on those that are more important to the target vehicle under a specific driving context [25]. For example, Deo et al. [20] combined the confidence pertaining to each driving maneuver using the HMM with the feasibility of each trajectory to obtain an energy function that was minimized to obtain the optimal trajectory predicted by the interactive model. However, the hand-crafted features used by conventional methods have difficulty characterizing multimodal driving patterns, so the accuracy of their long-term horizon trajectory predictions remains relatively low.

2) Deep Learning Methods: Most recent studies of trajectory prediction have been based on network structures such as the RNN or long short-term memory (LSTM) due to their ability to extract hidden dependencies from the context time steps. These structures perform the same operation on each input item of the sequence while considering the calculation of the previous input item [9]. Since vehicle trajectory prediction is a sequence-to-sequence task, it is very

common to adopt the LSTM encoder/decoder framework for this task [27]. For example, Deo and Trivedi [28] embedded convolutional social pooling into this framework and used the social tensor to encode the past motion state of the surrounding vehicles; Lee et al. [29] proposed a conditional variational auto-encoder framework that produced a set of different prediction hypotheses given observations of past trajectories to capture the multi-modality of driving behaviors; Hou et al. [30] proposed a structural-LSTM to capture the high-level dependencies between multiple interactive vehicles. These LSTMs share their cell states and hidden states with the spatial-adjacent LSTM through radial connection, and repeatedly analyze their own and other output states in a deeper layer. In another approach, Messaoud et al. [3] proposed a multi-head attention-based LSTM encoder/decoder framework to quantify the relative importance of surrounding vehicles during driving. In addition, generative methods such as the generative adversarial imitation learning (GAIL) [31] and generative adversarial network (GAN) [32] models have also been used for trajectory prediction.

There has also been a great deal of research focusing on the coding of the target vehicle and surrounding vehicles to improve the accuracy of trajectory prediction. The occupancy grid map has been widely adopted for probabilistic localization and mapping in robotics, as it can analyze trajectory uncertainty [33]. Zhao et al. [32] accordingly proposed multi-agent tensor fusion (MATF), which encodes the scene context and multiple agents into spatial feature maps, and can be trained and represented in an end-to-end manner through a flexible network. Furthermore, graph convolution network (GCN)-based methods are increasingly being applied to model interactions. Jeon et al. [34] adopted graphs to encode the behavior of surrounding vehicles in a manner fully scalable to the number of surrounding vehicles, minimizing the coding time complexity of each scene. Shi et al. [35] proposed the Sparse GCN (SGCN) to model interactions for pedestrian trajectory prediction. Redundant and useless interactions are explicitly eliminated, which improves the efficiency and performance of interaction extraction.

B. Interpretation of Neural Networks

Among the existing model interpretation methods, instance level interpretation is most suitable for explaining what features activate the specific neurons of a neural network to cause a specific prediction [15], allowing for study of the attributes of the trajectory prediction model in this study. There are two main instance level interpretation methods: the gradient method and the perturbation method. The gradient-based method uses backpropagation to calculate the partial derivatives of each class relative to the input of the neural network. By calling the gradient operator several times [36], the integrated gradient evaluates the global importance of each feature to the prediction rather than the local sensitivity. In order to reduce the influence of high-frequency noise in backpropagation, Smilkov et al. [37] proposed the use of a smooth gradient. The average image formed by many small disturbances of a given image has a significant smoothing effect that can allow for more intuitive interpretations of the importance of an image. The smooth gradient method ignores the intermediate layer of the neural network, but this layer could contain a great deal of valuable information. Therefore, Du et al. [38] proposed a guided feature inversion method by adding class-related constraints that provide class discrimination capabilities for a more refined interpretation.

The perturbation method adds a certain amount of noise to the original input and measures the importance of each feature therein by observing the corresponding changes in the hidden layer [15]. For example, by introducing noise into the image for super pixel occlusion, Ribeiro et al. [16] proposed the local interpretable model-agnostic explanations method to obtain the contribution of each pixel to the prediction. Guan et al. [39] defined a method based on information measurement; by introducing noise, the information lost in the intermediate layer was obtained to quantitatively measure the importance of each word in natural language processing.

III. Development of the Proposed Model

In this section, we first formulate the trajectory prediction problem and describe the input and objective. The proposed model and loss function is presented in Section III.B. Then, a schematic diagram of the model structure is given. In the last part of this section, we propose the interpretation module for TP2Net and present the interpretation process.

A. Problem Formulation

In this study, the input was the historical trajectory data $\boldsymbol{a}_T$ of the target vehicle within time t_{hst} ($t_{hst} = -w_h, \ldots, -2, -1, 0$). The origin of the coordinate system was defined as the current position of the target vehicle, the x-axis was its longitudinal direction (parallel to the lane) and the y-axis was its lateral direction (perpendicular to the lane).

In addition, considering the perceptual range of the target vehicle and appropriately simplifying the input, VOI inception was adopted [40] to consider the preceding vehicle, left preceding vehicle, right preceding vehicle, left alongside vehicle, right alongside vehicle, left following vehicle, right following vehicle, and following vehicle as $\mathcal{A} = \{\boldsymbol{a}_c\}, c = 1, 2, \ldots, N$, where $N = 8$. The left/right alongside vehicle refers to the closest vehicle driving along the left/right side of the target vehicle, and its longitudinal position is within a certain distance in front and rear of the target vehicle, which may directly affect the lateral driving maneuver of the target vehicle. For each vehicle's historical trajectory, $\boldsymbol{a} = [\boldsymbol{a}^{-w_h}, \boldsymbol{a}^{-w_h+1}, \ldots, \boldsymbol{a}^{-1}, \boldsymbol{a}^{0}]$, where $\boldsymbol{a}^{t_{hst}} = (x^{t_{hst}}, y^{t_{hst}}, v_x^{t_{hst}}, v_y^{t_{hst}}, a_x^{t_{hst}}, a_y^{t_{hst}}, class)$. Additional parameters not considered in previous studies were provided, including speeds v_x and v_y, accelerations a_x and a_y, and vehicle class, as they could reflect driving intentions and thus help to extract hidden driving features. The vehicle class was also taken into account because there are significant differences in the kinematic characteristics and driving features according to vehicle type. If there is no vehicle at a given position c, $\boldsymbol{a_c} = \boldsymbol{0}$. Additionally, it should be noted that the x_c and y_c of each surrounding vehicle was aligned with the origin of

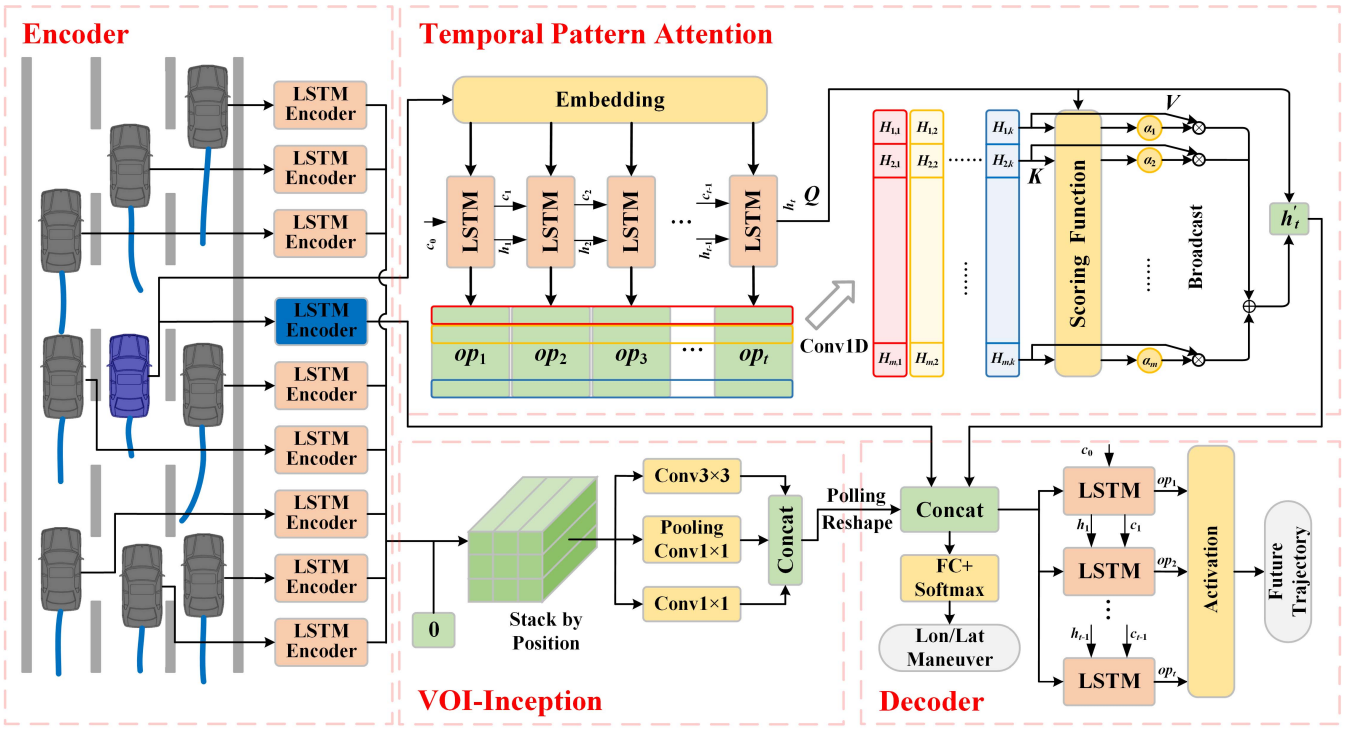


Fig. 1. Schematic diagram of the proposed TP2Net structure.

the defined coordinate system. In order to recognize driving maneuvers, the current driving maneuver classes M_{lon} and M_{lat} were defined respectively representing longitudinal (normal driving (ND), hard braking (HB), and rapid acceleration (RA)) and lateral (lane following (LF), left lane change (LLC), and right lane change (RLC)) maneuvers. These driving maneuvers were extracted using Deo's [28] method.

The objective of this study was to predict the trajectory of the target vehicle T in the future time t_{fut} ($t_{fut} = 1, 2, \ldots, w_f$). The predicted trajectory was defined as $\boldsymbol{a}^{t_{fut}} = (x^{t_{fut}}, y^{t_{fut}})$, where $x^{t_{fut}}$ and $y^{t_{fut}}$ are also aligned with the origin of the defined coordinate system. In addition, the hidden features extracted by the TPA intermediate hidden layers were explored. More specifically, the importance of each time step before a specific driving maneuver was quantitatively calculated.

B. Proposed Model

As shown in Fig. 1, the proposed model is composed of four parts: the Encoder, Decoder, TPA, and VOI inception, which are respectively used to process input coding, output decoding, the target vehicle information, and surrounding vehicle information. Each part is described in detail as follows:

1) Encoder: The function of the encoder is mainly to model the vehicle trajectory, including different historical trajectory patterns and driving maneuver patterns. Both the target vehicle and VOI are encoded. The encoder pertaining to the VOI shares parameters, but not the encoder pertaining to target vehicle. For each input $\boldsymbol{a}_c \in \mathcal{A}$ and $\boldsymbol{a}_T$, an embedding vector $\boldsymbol{e}_c$ is first formed through a fully connected (FC) layer. Subsequently, the embedding vector $\boldsymbol{e}_c$ is fed to the LSTM, and the hidden state vector $\boldsymbol{h}^t$ of the last unit is taken as the hidden driving feature. The encoding tensor is obtained after a linear transformation and leaky rectified linear unit (LeakyReLU) activation. The above operation is as follows:

$$\boldsymbol{e}_c = \phi\left(\boldsymbol{a}_c; \boldsymbol{W}_{emb}\right) \tag{1}$$

$$\boldsymbol{h}_c^t = \phi\left(\text{LSTM}\left(\boldsymbol{e}_c, \boldsymbol{h}_c^{t-1}; \boldsymbol{W}_{enc}\right); \boldsymbol{W}_{lin}\right) \tag{2}$$

where t is the number of hidden cells in the LSTM, $\boldsymbol{W}_{emb}$ is the embedding weight, $\boldsymbol{W}_{enc}$ is the weight of the LSTM encoder, and $\boldsymbol{W}_{lin}$ is the linear layer weight. These encoding tensors will be used in subsequent parts.

2) TPA: A typical attention mechanism is more inclined to select time steps that are more relevant to time series prediction. This is particularly suitable for tasks in which each time step contains a piece of information. However, for MTS prediction tasks, this approach may introduce extra noise [9]. As the TPA is a weighted summation of line vectors containing information across multiple time steps, it can better capture temporal information and span multiple time steps, making it possible to extract the temporal patterns of driving intentions and maneuvers, improving prediction accuracy. Therefore, this study optimized the attention mechanism for the task of trajectory prediction, as shown in Fig. 1.

Similarly, after embedding the history trajectory of the target vehicle $\boldsymbol{a}_T$, input $\boldsymbol{e}_T$ into one layer of the LSTM to obtain $\boldsymbol{H}_{op} = [\boldsymbol{op}^1, \boldsymbol{op}^2, \ldots, \boldsymbol{op}^t]$ and $\boldsymbol{h}^t$ as follows

$$\boldsymbol{H}_{op}, \boldsymbol{h}_T^t = \text{LSTM}(\boldsymbol{e}_T, \boldsymbol{h}_T^{t-1}, \boldsymbol{W}_{TPA}) \tag{3}$$

When one layer of the LSTM is used, $\boldsymbol{op}^t = \boldsymbol{h}^t$. As the last hidden unit of LSTM, $\boldsymbol{h}^t$ is not only used as the hidden state feature of the target vehicle, but also as the *Query* in the

attention, i.e. query sequence, to determine which time steps exert greater influence under the current hidden state.

Because a convolutional neural network can extract many different patterns from feature vectors, different temporal patterns may be captured when extracting features among different time steps. Therefore, m convolution filters $\boldsymbol{C}_m \in \mathbb{R}^{1\times t}$ are used to perform one-dimensional convolution in the direction of the hidden layer time step. For simplicity, let $k = t^{hst}$. The above operation is as follows:

$$H_{i,j} = \boldsymbol{H}_{op}(:, j) \otimes \boldsymbol{C}_i \tag{4}$$

where $i = 1, 2, \ldots, m$, $j = 1, 2, \ldots, k$, and $\otimes$ is the convolution operation.

The obtained convolution vector is used as the *Key* in the TPA mechanism. The aforementioned $\boldsymbol{h}_t$ is then used as a query sequence to map the importance of each hidden feature at a specific time step. With the aid of the weight matrix $\boldsymbol{W}_{sf}$, the score function $\psi : \mathbb{R}^{m\times k} \times \mathbb{R}^t \mapsto \mathbb{R}^k$ with Sigmoid activation is calculated according to the weight sum of each row:

$$\boldsymbol{\alpha} = \psi\left(\boldsymbol{H}^T \boldsymbol{W}_{sf} \boldsymbol{h}_t\right) \tag{5}$$

where $\boldsymbol{W}_{sf} \in \mathbb{R}^{m\times t}$ and $\alpha = (\alpha_1, \alpha_2, \ldots, \alpha_k)$. It should be noted that, unlike other attention methods, Softmax activation is not used here because more than one variable may be helpful for capturing hidden driving features.

Then, the attention mapping is broadcast on *Value*, and the more important features in the hidden variables $\boldsymbol{v}_t$ are weighted as follows:

$$\boldsymbol{v}_t = \sum_{j=1}^{k} \alpha_j \boldsymbol{H}_{:,j} \tag{6}$$

Finally, the obtained attention weight $\boldsymbol{v}_t$ and the hidden vector $\boldsymbol{h}_t$ are combined through $\boldsymbol{W}_h$ and $\boldsymbol{W}_v$, and the final hidden variable $\boldsymbol{h}'_t$ with the weight of the hidden temporal pattern is obtained:

$$\boldsymbol{h}'_t = \boldsymbol{W}_h \boldsymbol{h}_t + \boldsymbol{W}_v \boldsymbol{v}_t \tag{7}$$

where $\boldsymbol{W}_h \in \mathbb{R}^{t\times t}$, $\boldsymbol{W}_v \in \mathbb{R}^{t\times m}$, and $\boldsymbol{h}'_t \in \mathbb{R}^t$. Both $\boldsymbol{h}_t$ and $\boldsymbol{v}_t$ are multiplied by the weight matrix by broadcasting. The result is the sum of the hidden vector of the target vehicle after encoding and the attention weight across multiple time steps. With the increase in weight across multiple time steps, hidden driving features can be more easily discovered, thus improving the accuracy of long-term horizon prediction.

3) VOI Inception: When the encoder encodes the motion of the surrounding vehicles, it is difficult to capture the spatial and position features of the driving context. However, the dependence of the target vehicle on the surrounding vehicles varies [28]. If the encoded tensor of the surrounding vehicles is directly used as the input of the decoder, the spatial information will be lost. Therefore, the VOI inception is proposed. In order to maintain the spatial information of all vehicles, the encoded tensors of the target vehicle and surrounding vehicles are stacked according to their spatial positions. In particular, it not necessary to consider the problems caused by the different positions of surrounding vehicles, because their locations in the target vehicle coordinate system are provided in the input $\mathcal{A} = \{\boldsymbol{a}_c\}$. For the target vehicle, a zero tensor is used instead.

The stacked tensor is denoted as $\boldsymbol{E}_{stk}$. There are multiple patterns of interaction between the target vehicle and surrounding vehicles under different scales, but the extraction ability of a single-size convolution kernel (that is, at the single scale) is limited. Inspired by the inception module in GoogLeNet [13], the following VOI inception structure is proposed:

Due to the size limitation of $\boldsymbol{E}_{stk}$, three branches similar to the inception v1 module are adopted. The 3×3 convolution kernel is adopted by the first branch with a padding of 1. The max pooling with size $= 3$ and padding $= 1$ is used for the second branch, and then 1×1 convolution is performed. A 1×1 convolution is adopted directly by the third branch. The reason for not using a larger convolution kernel or a deeper number of convolution layers (such as inception v3, ResNet-v2) is that it easily leads to overfitting. Finally, the hidden interaction feature $\boldsymbol{h}_{\mathcal{A}}$ is obtained by connecting the outputs of the above three branches through the max pooling of size $= 3$. The above operation is as follows:

$$\begin{aligned}\boldsymbol{h}_{inc} = concat\{&\phi(\boldsymbol{E}_{stk} \otimes \boldsymbol{C}^{3\times3}), \phi[pooling(\boldsymbol{E}_{stk}) \otimes \boldsymbol{C}^{1\times1}],\\ &\phi(\boldsymbol{E}_{stk} \otimes \boldsymbol{C}^{1\times1})\}\end{aligned} \tag{8}$$

$$\boldsymbol{h}_{\mathcal{A}} = pooling(\boldsymbol{h}_{inc}) \tag{9}$$

where $\boldsymbol{C}^{1\times1}$ is the 1×1 convolution kernel and $\boldsymbol{C}^{3\times3}$ is the 3×3 convolution kernel.

4) Decoder: First, the decoder concatenates the encoded tensor and TPA tensor of the target vehicle, as well as the VOI inception tensor:

$$\boldsymbol{h}_{dec} = concat(\boldsymbol{h}_T, \boldsymbol{h}'_t, \boldsymbol{h}_{\mathcal{A}}) \tag{10}$$

Then, one branch is used to predict the distribution of driving maneuvers. It should be noted that the one-hot encoding of lateral and longitudinal driving maneuver is provided in the training process, and the probability of a vehicle lateral and longitudinal maneuver is output through the FC layer and Softmax layer. The other branch repeats the tensor t_{fut} times, and then enters the corresponding input into each LSTM unit. After the output activation layer, the output of each unit represents the target vehicle coordinate value at a certain time in the future. The above operations are shown as follows:

$$\hat{\boldsymbol{P}}_{lon} = Softmax(FC(\boldsymbol{h}_{dec}; \boldsymbol{W}_{lon})) \tag{11}$$

$$\hat{\boldsymbol{P}}_{lat} = Softmax(FC(\boldsymbol{h}_{dec}; \boldsymbol{W}_{lat})) \tag{12}$$

$$\hat{\boldsymbol{a}} = \varphi[\text{LSTM}(repeat(\boldsymbol{h}_{dec}); \boldsymbol{W}_{dec}] \tag{13}$$

where φ is the output activation.

For the vehicle trajectory prediction value, we use the mean squared error as the loss function of the regression task as follows:

$$J_{MSE} = \frac{1}{N} \sum_{n=1}^{N} \sum_{j=1}^{t_{fut}} \left\| \hat{\boldsymbol{a}}_{n,j} - \boldsymbol{a}_{n,j} \right\|^2 \tag{14}$$

where N is the size of the mini-batch and $\boldsymbol{a}_{n,j}$ is the ground truth of the target vehicle trajectory. The categorical cross entropy is used to calculate the classification loss of the

driving maneuver, as shown in the following equation (taking a longitudinal maneuver as an example):

$$J_{CE}^{lon} = -\frac{1}{N}\sum_{n=1}^{N}\sum_{j=1}^{class} \boldsymbol{P}_{lon}\log(\hat{\boldsymbol{P}}_{lon}) \tag{15}$$

Thus, the total loss of the model is as follows:

$$J = J_{CE}^{lon} + J_{CE}^{lat} + J_{MSE} \tag{16}$$

C. Interpretation Module for TP2Net

In order to ensure confidence in the results, the data under specific driving maneuvers are taken as the dataset $\boldsymbol{X}_{set}$ to be interpreted. For each input trajectory $\boldsymbol{X} = (\boldsymbol{x}_{-w_h}^{\mathrm{T}}, \ldots, \boldsymbol{x}_{-1}^{\mathrm{T}}, \boldsymbol{x}_{0}^{\mathrm{T}})$, $\boldsymbol{X} \in \boldsymbol{X}_{set}$, $\boldsymbol{X} = \boldsymbol{a}_T \cup \mathcal{A}$, and $\boldsymbol{x}_i \in \mathbb{R}^k$, where k is the dimension of inputs. Suppose that TP2Net contains L neural network layers, and the corresponding hidden state is $\boldsymbol{S} = \Psi(\boldsymbol{X})$. The goal is to quantify the amount of information at each time step $\boldsymbol{x}$ of the input trajectory contained in the hidden state $\boldsymbol{s}$ to explain which time steps are more important for trajectory prediction.

The entropy is used to quantify the information contained in the hidden state. Given trajectory $\boldsymbol{X}$, suppose the probability density function of the input feature $\boldsymbol{x}$ at each input time step is $p(\boldsymbol{x})$, then the entropy of the trajectory $\boldsymbol{X}$ is:

$$\mathcal{H}(\boldsymbol{X}) = -\int_{\boldsymbol{x}\in\boldsymbol{X}} p(\boldsymbol{x})\log p(\boldsymbol{x})d\boldsymbol{x} \tag{17}$$

Trajectory $\boldsymbol{X}$ is input into TP2Net $\Psi(\cdot)$, then all hidden states are obtained after forward passing through L layers, and the probability density function is $p(\boldsymbol{s})$; thus, the entropy of the hidden state $\boldsymbol{S}$ is:

$$\mathcal{H}(\boldsymbol{S}) = -\int_{\boldsymbol{s}\in\boldsymbol{S}} p(\boldsymbol{s})\log p(\boldsymbol{s})d\boldsymbol{s} \tag{18}$$

Let the joint entropy of trajectory $\boldsymbol{X}$ and hidden state $\boldsymbol{S}$ be $\mathcal{H}(\boldsymbol{X}, \boldsymbol{S})$:

$$\mathcal{H}(\boldsymbol{X}, \boldsymbol{S}) = -\int_{\boldsymbol{x}\in\boldsymbol{X}}\int_{\boldsymbol{s}\in\boldsymbol{S}} p(\boldsymbol{x}, \boldsymbol{s})\log p(\boldsymbol{x}, \boldsymbol{s})d\boldsymbol{s}d\boldsymbol{x} \tag{19}$$

Then, the amount of information $\mathcal{I}(\boldsymbol{X}, \boldsymbol{S})$ contained between the trajectory $\boldsymbol{X}$ and the hidden state $\boldsymbol{S}$ is:

$$\mathcal{I}(\boldsymbol{X}, \boldsymbol{S}) = \mathcal{H}(\boldsymbol{X}) - (\mathcal{H}(\boldsymbol{X}, \boldsymbol{S}) - \mathcal{H}(\boldsymbol{S})) \tag{20}$$

where the first item is the total amount of information describing trajectory $\boldsymbol{X}$ and the second item is the amount of information discarded by the neural network after training. The mutually contained information between $\boldsymbol{X}$ and $\boldsymbol{S}$ will not exceed the total amount of information, indicating that no additional information will be generated. If $\boldsymbol{X}$ and $\boldsymbol{S}$ are completely independent, that is

$$\mathcal{H}(\boldsymbol{X}, \boldsymbol{S}) = \mathcal{H}(\boldsymbol{S}) + \mathcal{H}(\boldsymbol{X}), \tag{21}$$

then $\mathcal{I}(\boldsymbol{X}, \boldsymbol{S}) = 0$; in other words, there is no mutual information between trajectory $\boldsymbol{X}$ and hidden variable $\boldsymbol{S}$, and all the information of trajectory $\boldsymbol{X}$ is lost. Given the input trajectory $\boldsymbol{X}$, the entropy $\mathcal{H}(\boldsymbol{X})$ is constant during model training. Then, the relative information $\mathcal{I}'(\boldsymbol{X}, \boldsymbol{S})$ contained between trajectory $\boldsymbol{X}$ and hidden state $\boldsymbol{S}$ is as follows:

$$\begin{aligned}
\mathcal{I}'(\boldsymbol{X}, \boldsymbol{S}) &= \mathcal{H}(\boldsymbol{S}) - \mathcal{H}(\boldsymbol{X}, \boldsymbol{S}) \\
&= (-\int_{\boldsymbol{s}\in\boldsymbol{S}}(\int_{\boldsymbol{x}\in\boldsymbol{X}} p(\boldsymbol{x}, \boldsymbol{s})d\boldsymbol{x})\log p(\boldsymbol{s})d\boldsymbol{s}) \\
&\quad -(-\int_{\boldsymbol{x}\in\boldsymbol{X}}\int_{\boldsymbol{s}\in\boldsymbol{S}} p(\boldsymbol{x}, \boldsymbol{s})\log p(\boldsymbol{x}, \boldsymbol{s})d\boldsymbol{s}d\boldsymbol{x}) \\
&= \int_{\boldsymbol{s}\in\boldsymbol{S}}\int_{\boldsymbol{x}\in\boldsymbol{X}} p(\boldsymbol{s})\frac{p(\boldsymbol{x}, \boldsymbol{s})}{p(\boldsymbol{s})}\log\frac{p(\boldsymbol{x}, \boldsymbol{s})}{p(\boldsymbol{s})}d\boldsymbol{x}d\boldsymbol{s} \\
&= \int_{\boldsymbol{s}\in\boldsymbol{S}} p(\boldsymbol{s})\mathcal{H}(\boldsymbol{X}|\boldsymbol{s})d\boldsymbol{s} \\
&= -\mathcal{H}(\boldsymbol{X}|\boldsymbol{S})
\end{aligned} \tag{22}$$

For the output $\boldsymbol{s}$ of one hidden layer, the relative information of trajectory $\boldsymbol{X}$ can be obtained as:

$$\mathcal{H}(\boldsymbol{X}|\boldsymbol{s}) = -\int_{\boldsymbol{x}_i\in\boldsymbol{X}} p(\boldsymbol{x}_i|\boldsymbol{s})\log p(\boldsymbol{x}_i|\boldsymbol{s})d\boldsymbol{x}_i \tag{23}$$

From the above equation, it can be seen that the calculation of entropy $\mathcal{H}(\boldsymbol{X}|\boldsymbol{s})$ requires the conditional probability $p(\boldsymbol{x}_i|\boldsymbol{s})$. However, this distribution is established by TP2Net, and cannot be solved directly. As a result, the information must be computed indirectly, as shown in **Algorithm 1**.

Algorithm 1 Interpretation Module for TP2Net

1: Initialization: $\Psi(\cdot; \boldsymbol{W}_\Psi)$, $\boldsymbol{X}_{set}$, λ, *epochs*, $\boldsymbol{\sigma}_I \leftarrow \emptyset$
2: $\sigma_S^2 \leftarrow \mathrm{var}(\Psi(\boldsymbol{X}_{set}; \boldsymbol{W}_\Psi))$
3: **for** $idx \leftarrow 0$ **to** $|\boldsymbol{X}_{set}|$
4: Initialization: Interpreter$(\cdot; \boldsymbol{W}_\sigma)$
5: **for** $ep \leftarrow 0$ **to** *epochs*
6: $\boldsymbol{\sigma} \leftarrow Sigmoid(\boldsymbol{W}_\sigma)$
7: $\boldsymbol{s} \leftarrow \Psi(\boldsymbol{X}_{idx}; \boldsymbol{W}_\Psi)$
8: $\tilde{\boldsymbol{X}}_{idx} \leftarrow \boldsymbol{X}_{idx} + \boldsymbol{\delta}_{\delta_i\sim\mathcal{N}(0,\boldsymbol{\Sigma}_i=\sigma_i^2\boldsymbol{I})}$
9: $\tilde{\boldsymbol{s}} \leftarrow \Psi(\tilde{\boldsymbol{X}}_{idx}; \boldsymbol{W}_\Psi)$
10: Update *loss* based on Equation (29)
11: Train the Interpreter$(\cdot; \boldsymbol{W}_\sigma)$ (*loss* backpropagation)
12: **end for**
13: $\boldsymbol{\sigma} \leftarrow Sigmoid(\boldsymbol{W}_\sigma)$
14: $\boldsymbol{\sigma}_I \leftarrow \boldsymbol{\sigma}_I \cup \boldsymbol{\sigma}$
15: **end for**
Output: $\boldsymbol{\sigma}_I$

Let a layer in the trained neural network be $\boldsymbol{s} = \Psi(\boldsymbol{X}; \boldsymbol{W}_{1:L})$, where $\boldsymbol{W}_{1:L}$ contains all the weights between layers 1–L. By adding random perturbation, if the output $\boldsymbol{s}$ of a perturbation changes considerably, then that value is more likely to have a greater impact on the model training. Gaussian noise $\boldsymbol{\delta}_i \sim \mathcal{N}(0, \boldsymbol{\Sigma}_i = \sigma_i^2\boldsymbol{I})$ is applied with a mean value of 0 and variance of $\sigma_i^2\boldsymbol{I}$. This perturbation is then added to trajectory $\boldsymbol{X}$ to obtain the input $\tilde{\boldsymbol{x}}_i = \boldsymbol{x}_i + \boldsymbol{\delta}_i \in \tilde{\boldsymbol{X}}$ with noise. According to the maximum likelihood estimation, in order to obtain the parameter variance σ_i^2 of Gaussian noise $\boldsymbol{\delta}_i$, the likelihood function must be maximized as follows:

$$L(\tilde{x}_{1:n}; \boldsymbol{\sigma}_i^2) = \max_{\boldsymbol{\sigma}_i^2}\prod_{i=1}^{n} p(\tilde{\boldsymbol{x}}_i|\boldsymbol{s})_{\boldsymbol{\delta}_i\sim\mathcal{N}(0,\boldsymbol{\Sigma}_i=\sigma_i^2\boldsymbol{I})} \tag{24}$$

The log-likelihood function is as follows:

$$\ln L = \sum_{i=1}^{n} \ln p(\tilde{\boldsymbol{x}}_i|\boldsymbol{s})_{\boldsymbol{\delta}_i \sim \mathcal{N}(0, \boldsymbol{\Sigma}_i = \sigma_i^2 \boldsymbol{I})} \tag{25}$$

Since the noise is Gaussian noise, that is:

$$-\sum_{i=1}^{n} \ln p(\tilde{\boldsymbol{x}}_i|\boldsymbol{s})_{\boldsymbol{\delta}_i \sim \mathcal{N}(0, \boldsymbol{\Sigma}_i = \sigma_i^2 \boldsymbol{I})} \propto \mathcal{N}(\Psi(\tilde{\boldsymbol{X}}); \boldsymbol{s}, \sigma_{\boldsymbol{S}}^2) \tag{26}$$

Equation (26) can be substituted into Equation (25) to yield:

$$\ln L = \lambda \cdot \frac{1}{2} \sum_{j=1}^{k} \frac{\left\| \Psi(\tilde{\boldsymbol{X}})_{\boldsymbol{\delta}_i \sim \mathcal{N}(0, \boldsymbol{\Sigma}_i = \sigma_i^2 \boldsymbol{I})} - s \right\|^2}{\sigma_{\boldsymbol{S}}^2} + C \tag{27}$$

where σ_S^2 is the variance of trajectory dataset $\boldsymbol{X}_{set}$ and λ is an approximate parameter of the log-likelihood function and Gaussian distribution.

In addition, in order to obtain the limits of noise in all perturbation directions, prior knowledge σ_i^2 shall be sufficiently trained to maximize the amount of information $\mathcal{I}' = -\mathcal{H}(\tilde{\boldsymbol{X}}|\boldsymbol{s})$. Given the trained network $\Psi(\cdot; \boldsymbol{W}_\Psi)$, the conditional distribution of $\tilde{\boldsymbol{X}}$ is approximately equal to the distribution of noise $\boldsymbol{\delta}_i \sim \mathcal{N}(0, \boldsymbol{\Sigma}_i = \sigma_i^2 \boldsymbol{I})$. According to Equation (23), the expression maximizing the amount of information about $\mathcal{H}(\tilde{\boldsymbol{X}}|\boldsymbol{s})$ can be obtained as:

$$\begin{aligned} -\mathcal{H}(\tilde{\boldsymbol{X}}|\boldsymbol{s}) &= \int_{\tilde{\boldsymbol{x}}_i \in \tilde{\boldsymbol{X}}} \frac{1}{\sqrt{2\pi}\boldsymbol{\sigma}_i} e^{-\frac{\tilde{\boldsymbol{x}}_i^2}{\sigma_i^2}} \log\left(\frac{1}{\sqrt{2\pi}\boldsymbol{\sigma}_i} e^{-\frac{\tilde{\boldsymbol{x}}_i^2}{\sigma_i^2}}\right) dx_i \\ &= -\left(\frac{K}{2}\log(2\pi) + K\log(\boldsymbol{\sigma}_i)\right. \\ &\quad \left. + \int_{\tilde{\boldsymbol{x}}_i \in \tilde{\boldsymbol{X}}} \frac{\tilde{\boldsymbol{x}}_i^2}{\boldsymbol{\sigma}_i^2} \cdot e^{-\frac{\tilde{\boldsymbol{x}}_i^2}{\sigma_i^2}} d\left(\frac{x_i}{\boldsymbol{\sigma}_i}\right) \cdot \frac{1}{\sqrt{2\pi}} \log e\right) \\ &= -K\log(\boldsymbol{\sigma}_i) + C \end{aligned} \tag{28}$$

After summarizing the previous two parts, in order to approximate the entropy $\mathcal{H}(\boldsymbol{X}|\boldsymbol{s})$, the distribution of variance in $\boldsymbol{\delta}_i \sim N(0, \boldsymbol{\Sigma}_i = \sigma_i^2 \boldsymbol{I})$ in hidden state $\boldsymbol{s}$ must be obtained. Therefore, according to Equations (27) and (28), the loss function is as follows:

$$J_{\boldsymbol{\sigma}_i} = \lambda \sum_{j=1}^{k} \frac{\left\| \Psi(\tilde{\boldsymbol{X}})_{\boldsymbol{\delta}_i \sim \mathcal{N}(0, \boldsymbol{\Sigma}_i = \sigma_i^2 \boldsymbol{I})} - s \right\|^2}{\sigma_{\boldsymbol{S}}^2} - K \sum_{i=1}^{n} \log(\boldsymbol{\sigma}_i) \tag{29}$$

According to **Algorithm 1** and Equation (29), the trained σ_i^2 represents the information utilization degree (that is, the importance of each time step) of the TP2Net intermediate layer to the specific trajectory dataset $\boldsymbol{X}_{set}$.

IV. Experiments and Evaluations

In this section, the experimental setting is first illustrated, including the datasets and experimental environment. Then, the hyper-parameter, training setting and evaluation metrics of the prediction results are provided. In addition, the results of the proposed model on the test dataset are shown in Section IV.C and compare with other excellent works. The predicted results of each driving maneuver are shown and analyzed in Section IV.D. In Section IV.E, the ablation experiments are presented to demonstrate the effectiveness of each module of the proposed model. In the last section, the output of the model interpretation module is analyzed to explain that TPA is effective.

A. Experimental Setting

The datasets used in this study were HighD [41] and NGSIM I-80 [42], [43]. The HighD dataset includes driving data of cars and trucks on the highways around Cologne, Germany, collected by the RWTH Aachen University in 2017 and 2018. All data were collected using a drone at a frequency of 25 fps. The NGSIM dataset is a public dataset that was obtained at 10 fps in 2005. Taking the HighD dataset as an example, 14 records were used, including 10 records for training, 2 records for testing, and 2 record for verification. An 8-s period was selected to describe the trajectory of each vehicle: 3 s were used as the historical input and 5 s as the trajectory to be predicted. In order to facilitate processing, the sampling frequency of the dataset was reduced to 5 fps.

All experiments were performed on an Intel Core(R) i7-7800x CPU @ 3.50 GHz (Turbo 4.00 GHz), NVIDIA GeForce(R) GTX 1080Ti 11 GB GPU with 16 GB of RAM running the Ubuntu 16.04 LTS edition. All the program tasks were conducted on Python 3.7, and the deep learning framework was based on PyTorch.

B. Training Setting and Evaluation Metrics

According to the comparison of multiple experiments, the size of the hidden layer was selected as follows: for the LSTM encoder/decoder, 128 hidden layers were selected and 32 convolution kernels were used for the TPA. After convolution, the final output dimension of the TPA was 64. Each input tensor with 64 dimensions in the VOI inception was therefore equal to the output dimension of the encoder, making a total output of 128. The Adam optimizer was employed with a learning rate of 10^{-4} and a mini-batch size of 128. In addition, the learning rate schedule method was adopted to provide more refined training. When the loss failed to decrease after the established number of consecutive "patience" times, the learning rate was reduced. The reduction factor was set to 0.8, and the minimum learning rate was set to 10^{-6}. In order to prevent the training method from affecting the generalization ability of the model, all subsequent results were performed on the validation set.

The accuracy, precision, recall, and F1-score were used to evaluate the classification accuracy of various driving maneuvers. In addition, to measure the accuracy of trajectory prediction, the RMSE in meters was adopted to express the error between the model prediction and ground truth, as is common in trajectory prediction tasks, and is given by:

$$RMSE = \sqrt{\frac{1}{N} \sum_{n=1}^{N} \left[(\boldsymbol{x}_n - \hat{\boldsymbol{x}}_n)^2 + (\boldsymbol{y}_n - \hat{\boldsymbol{y}}_n)^2 \right]} \tag{30}$$

where $\boldsymbol{x}_n, \boldsymbol{y}_n$ is the ground truth and $\hat{\boldsymbol{x}}_n, \hat{\boldsymbol{y}}_n$ is the prediction.

TABLE I
RMSE OF EACH MODEL IN THE 5-s PREDICTION HORIZON

Dataset	Prediction horizon (s)	C-VGMMs[20]	GAIL [31]	S-LSTM [44]	MATF [32]	CS-LSTM[28]	SGCN [35]	SCALE-Net [34]	MFP [45]	MHA [3]	TP2Net
NGSIM	1	0.66	0.69	0.65	0.66	0.61	0.58	0.46	0.54	0.41	**0.30**
	2	1.56	1.51	1.31	1.34	1.27	1.18	1.16	1.16	1.01	**0.86**
	3	2.75	2.55	2.16	2.08	2.09	1.95	1.97	1.90	1.74	**1.52**
	4	4.24	3.65	3.25	2.97	3.10	3.03	2.91	2.78	2.67	**2.36**
	5	5.99	4.71	4.55	4.13	4.37	4.04	-	3.83	3.83	**3.37**
HighD	1	-	-	0.22	-	0.22	0.15	-	-	0.06	**0.05**
	2	-	-	0.62	-	0.61	0.38	-	-	0.09	**0.07**
	3	-	-	1.27	-	1.24	0.72	-	-	0.24	**0.19**
	4	-	-	2.15	-	2.10	1.16	-	-	0.59	**0.49**
	5	-	-	3.41	-	3.27	1.71	-	-	1.18	**0.98**

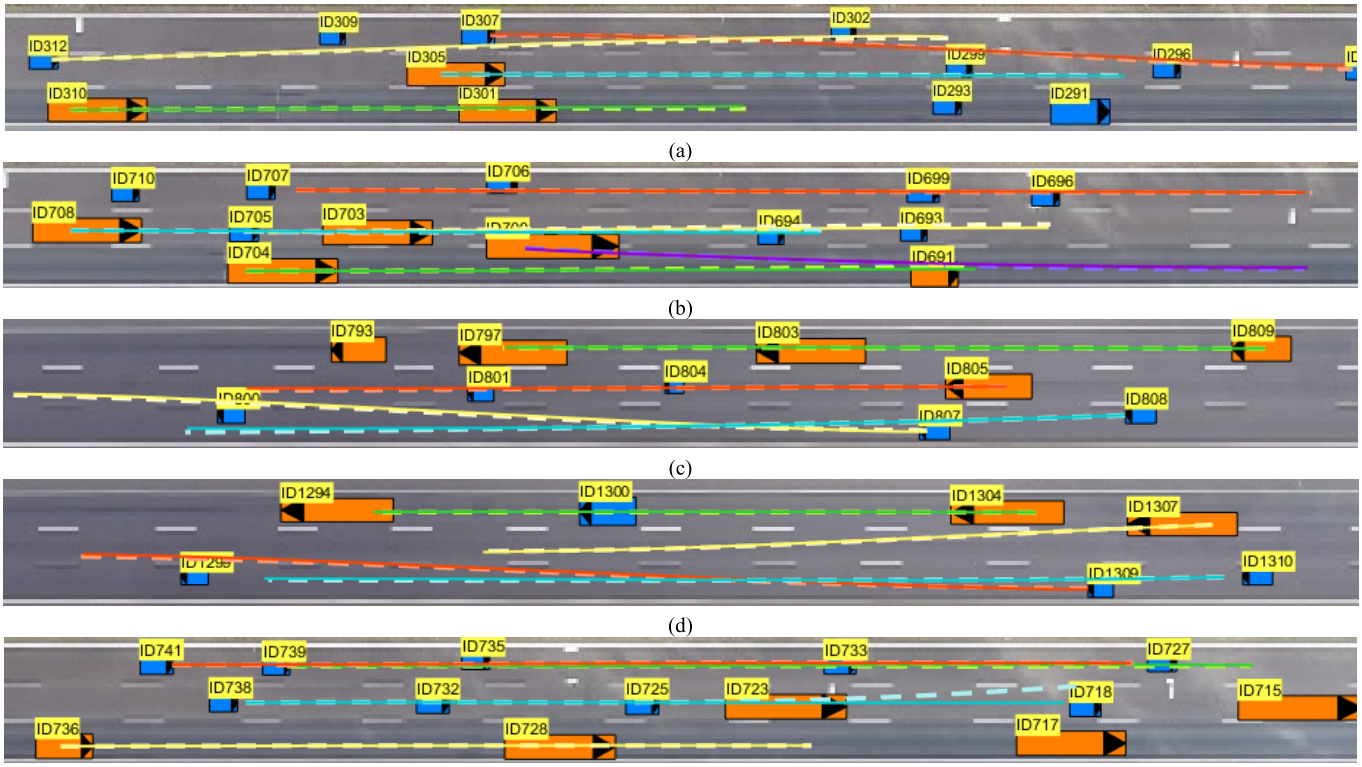


Fig. 2. Prediction instances of HighD dataset. The blue box represents a car, the orange box represents a truck, and a triangle in a box represents the driving direction of that vehicle. The light color dotted line is the ground truth, and the dark color solid line is the prediction. The vehicles in (a), (b), (e) drive from left to right whereas those in (c), (d) drive from right to left.

C. Results and Comparison

To validate the proposed model, its results were compared with those of many excellent studies conducted in recent years of the following models:

- Class variational Gaussian mixture models (C-VGMMs) [20]: Variational Gaussian mixture models with a Markov random field were used to classify the driving maneuver and predict trajectory.
- GAIL [31]: GAIL was extended into the optimization of the gated recurrent unit to retain greater policy fidelity.
- Social LSTM (S-LSTM) [44]: A fully connected pooling LSTM encoder/decoder was used to predict trajectory.
- MATF [32]: The historical trajectories of multiple agents and the scene context were encoded into a GAN with adversarial loss.
- Convolutional social pooling (CS-LSTM) [28]: Convolutional social pooling was used to encode the surrounding vehicles as social tensors, the LSTM encoder/decoder was adopted, and multi-modal driving maneuvers were considered.
- SGCN [35]: A sparse GCN that explicitly models sparse directed interactions based on sparse directed spatial graphs.
- Scalable Network (SCALE-Net) [34]: An edge-enhanced graph convolution neural network was used that was insensitive to the input data and improved prediction efficiency.
- Multiple futures prediction (MFP) [45]: Parallel RNNs with shared weight encoder was used to encode the past and future interactions of the agent and predict the trajectory.

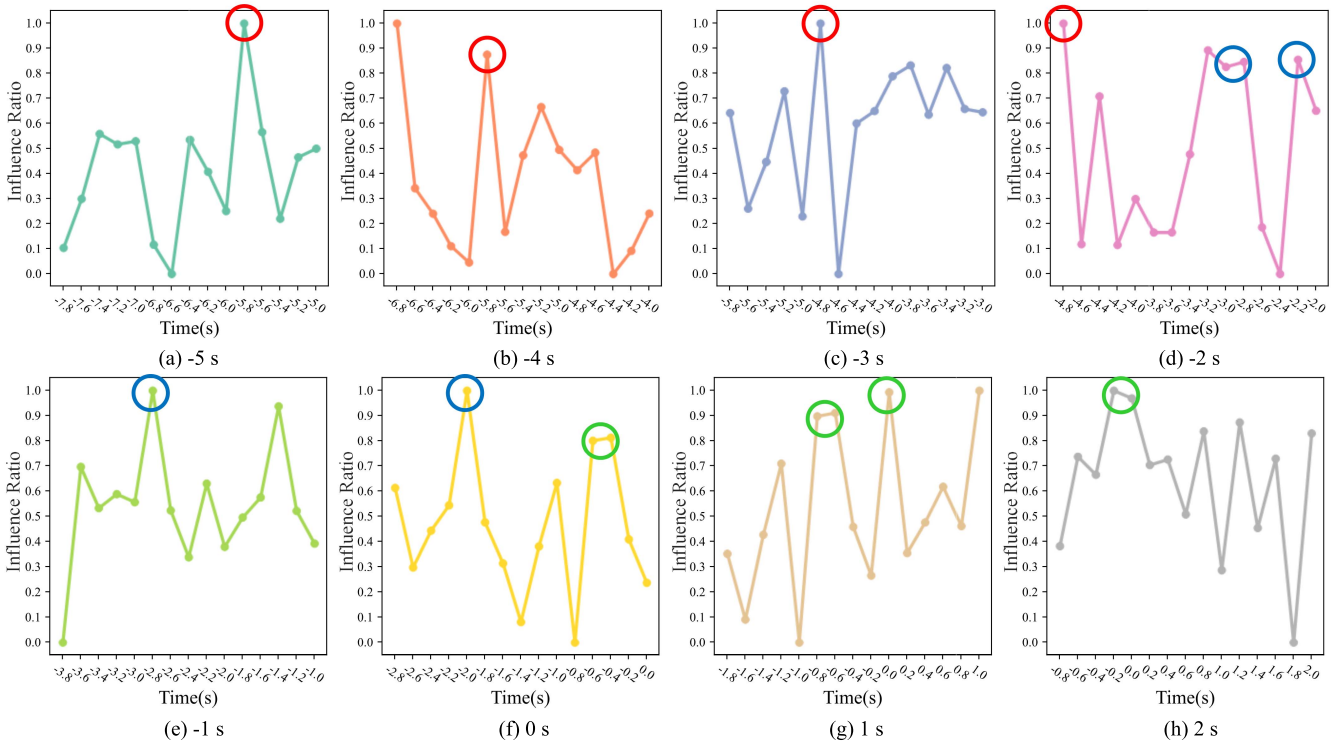


Fig. 3. Normalized relative influence between each time step and TPA output tensor for the LLC condition. The moments having obvious response peaks in at least two subfigures are circled, where the red circle represents the first stage (-4.8 s, -5.8 s, possibly an intention to change lanes), the blue circle represents the second stage ($-2.2 \sim -2.0$ s, $-3.0 \sim -2.8$ s, the driver began to turn the steering wheel), and the green circle represents the third stage (0 s, $-0.6 \sim -0.4$ s, vehicle crossing lane line).

TABLE II
CLASSIFICATION PERFORMANCE OF EACH DRIVING MANEUVER

Lateral accuracy: 99.22% **Longitudinal accuracy: 99.10%**	Precision (%)	Recall (%)	F1-score (%)
LF	99.60	99.58	99.59
LLC	92.05	91.61	91.83
RLC	95.09	95.73	95.41
ND	99.40	99.68	99.54
HB	85.64	80.16	82.81
RA	84.55	71.90	77.71

TABLE III
RMSE OF EACH DRIVING MANEUVER IN THE 5-s PREDICTION HORIZON

	Prediction Horizon (s)	**LF**		**LLC**		**RLC**	
		Lon	**Lat**	**Lon**	**Lat**	**Lon**	**Lat**
MHA [3]	1	0.05	**0.01**	0.20	**0.03**	0.07	**0.03**
	2	0.07	**0.02**	0.32	**0.07**	0.12	**0.06**
	3	0.22	**0.06**	0.42	0.19	0.34	0.18
	4	0.54	**0.14**	0.88	0.45	0.79	0.43
	5	1.10	**0.22**	1.74	0.78	1.43	0.76
TP2Net	1	**0.04**	0.02	**0.08**	0.07	**0.06**	0.05
	2	**0.06**	0.03	**0.12**	0.10	**0.09**	0.08
	3	**0.17**	0.08	**0.27**	**0.17**	**0.21**	**0.14**
	4	**0.44**	0.18	**0.66**	**0.41**	**0.52**	**0.34**
	5	**0.88**	0.30	**1.30**	**0.63**	**1.04**	**0.54**

- Multi-head attention social pooling (MHA) [3]: Multi-head attention with encoder/decoder was used to extract deep features of the target vehicle and surrounding vehicles with dot product attention. More input features (speed, acceleration, vehicle class) were considered.
- TP2Net: The model proposed in this study.

The RMSE values for each model are compared in Table I. Note that as many studies did not evaluate the model using the HighD dataset, the corresponding results are not provided. It can be observed in Table I that the RMSE of NGSIM was higher than that of HighD; this is likely because to the data provided by the HighD dataset were more accurate and contained fewer errors. In particular, incorrect labeling will affect the encoding of the target vehicle and surrounding vehicles, and this unreasonable coding will result in more losses to and negative effects on network prediction. In the short term (1–3 s), predictions according to the kinematic characteristics and inertia of the target vehicle show relatively small error; however, in the long term (4–5 s), predictions of driving intention have a greater influence on the future trajectory of the target vehicle, leading to a large error, so it is necessary to recognize hidden driving features to guide the trajectory prediction. Most importantly, the proposed model provides the best performance in both short-term and long-term trajectory prediction tasks. The proposed model provides an accuracy improvement of 15% over that of MHA. This demonstrates that the proposed model can better extract the

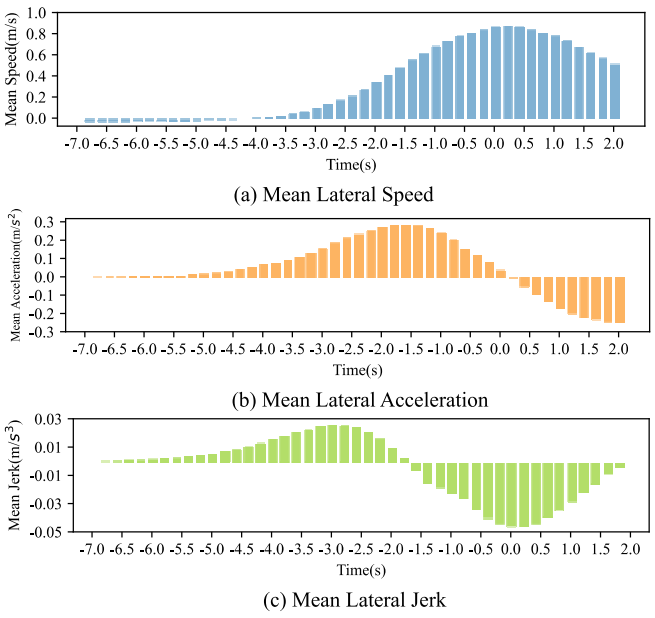


Fig. 4. Schematic diagram pertains to values of each time step for LLC.

hidden driving features of a vehicle and utilize this feature to guide the prediction of vehicle trajectory.

Figure 2 shows the prediction instances of various vehicle driving maneuvers. Figures 2 (a), (b), (c), (d) show that the model achieved good prediction accuracy on the validation set for free driving, lane following, and lane changing maneuvers. Figure 2 (e) shows a case in which the prediction error for the lane change maneuver was large. It can be observed that vehicle ID 738 made an LLC driving maneuver in the next 4 to 5 s, but the network failed to predict this maneuver. This may be because at this moment, the network detected that there was a left-front car ID 739 at a close distance that should prevent the target vehicle from changing lanes, so it predicted that the car would continue to drive straight. Thus, when the lane-changing maneuver occurred in the next 4 to 5 s, the prediction accuracy of TP2Net decreased. Improving performance in this scenario will be addressed in future research.

D. Prediction Error of Each Driving Maneuver

Taking the HighD dataset as an example, Table II shows the overall accuracy, precision, recall, and F1-score for each driving maneuver classification. It can be seen that the classification results for lateral maneuvers were satisfactory overall, but the F1-score for LLC and RLC was lower than that for LF. It was found that the driving maneuver of the target vehicle in the next 3–4 s was often incorrectly classified; improving performance in this scenario will be addressed in future work. In addition, the prediction accuracy for longitudinal driving maneuvers was relatively low, which may be due to the fuzzy classification hyperplane between different longitudinal maneuvers.

Table III shows the lateral and longitudinal prediction errors for each driving maneuver. It can be seen from the table that lateral driving maneuvers increased both the longitudinal and lateral prediction errors, but the proportion of longitudinal prediction error was higher than that of the lateral prediction error. The table also indicates that the proposed method was not as accurate as MHA in terms of the longitudinal error for LF and lateral maneuver errors in the short term, but showed considerably reduced errors otherwise. This indicates that TP2Net can better predict the trajectory in the long-term prediction horizon according to the extracted hidden features. Additionally, it can be seen in Table III that the prediction error of LLC in a longitudinal maneuver was significantly larger than that of RLC in a lateral maneuver. This error is likely larger because there were more active lane changes with RA or passive lane changes with HB associated with the LLC maneuver, and the speed variance was larger than that associated with the RLC maneuver. Similarly, the lateral error associated with LF was smaller than that associated with either LLC or RLC.

E. Comparison and Ablation Experiment

In order to verify the performance of each module of the proposed TP2Net, the following comparison and ablation experiments were conducted using the HighD dataset:

- Fewer features: only the abscissa and ordinate of the target vehicle and surrounding vehicles were used as input.
- Only $\boldsymbol{h}_T$: the prediction was only based on the target vehicle encoded tensor $\boldsymbol{h}_T$.
- No $\boldsymbol{h}_T$: removed the target vehicle encoded tensor $\boldsymbol{h}_T$.
- No $\boldsymbol{h}_t'$: removed the TPA output tensor $\boldsymbol{h}_t'$.
- No $\boldsymbol{h}_{\mathcal{A}}$: removed the VOI inception tensor $\boldsymbol{h}_{\mathcal{A}}$.
- Simple VOI: 1×1 and 3×3 convolution kernels were used to convolute the stacking tensor $\boldsymbol{E}_{stk}$ to replace the VOI inception.
- MHA: replaced the TPA mechanism with the MHA mechanism.
- GCN: a graph convolution neural network was used to model the interaction of surrounding vehicles.
- TP2Net: the model proposed in this study.

Table IV shows the RMSE of each comparison and ablation model within the 5-s prediction horizon. The following inferences can be obtained based on these results:

1) The prediction accuracy was considerably reduced when only the abscissa and ordinate of the target vehicle were used as the input for trajectory prediction. Though there can be differences in the driving features of different types of vehicles, lateral and longitudinal speed and acceleration can also reflect future driving maneuvers. Thus, it is necessary to provide additional kinematic parameters for trajectory prediction.
2) Although the tensors removed in the **No** $\boldsymbol{h}_t'$ and **No** $\boldsymbol{h}_T$ cases both extracted hidden features of the target vehicle, the lateral error was small without $\boldsymbol{h}_T$ whereas the longitudinal error was small without $\boldsymbol{h}_t'$. This indicates that TPA can better extract the lateral driving maneuvers, the target vehicle encoding can make up for the large longitudinal error, and the two tensors can complement each other.

TABLE IV
RMSE OF EACH COMPARISON AND ABLATION MODEL IN THE 5-SECOND PREDICTION HORIZON

	Prediction Horizon (s)	Less feature	Only h_T	No h_T	No h_t'	No $h_{\mathcal{A}}$	Simple VOI	MHA	GCN	TP2Net (proposed)
Longitudinal	1	0.181	0.093	0.071	0.051	0.063	0.049	0.045	0.044	**0.043**
	2	0.511	0.133	0.100	0.072	0.090	0.070	0.065	0.067	**0.063**
	3	1.055	0.264	0.209	0.199	0.218	**0.177**	0.185	0.184	**0.177**
	4	1.760	0.682	0.510	0.506	0.563	0.454	0.476	0.486	**0.453**
	5	2.645	1.377	1.028	1.022	1.164	0.943	0.973	0.987	**0.930**
Lateral	1	0.133	0.075	0.049	0.052	0.027	0.029	0.053	0.034	**0.023**
	2	0.255	0.097	0.059	0.057	0.043	0.044	0.090	0.043	**0.037**
	3	0.367	0.159	0.100	0.099	0.092	0.087	0.144	**0.081**	0.082
	4	0.480	0.284	0.193	0.206	0.196	0.189	0.240	**0.177**	0.185
	5	0.562	0.435	0.310	0.335	0.326	0.317	0.345	**0.298**	0.310

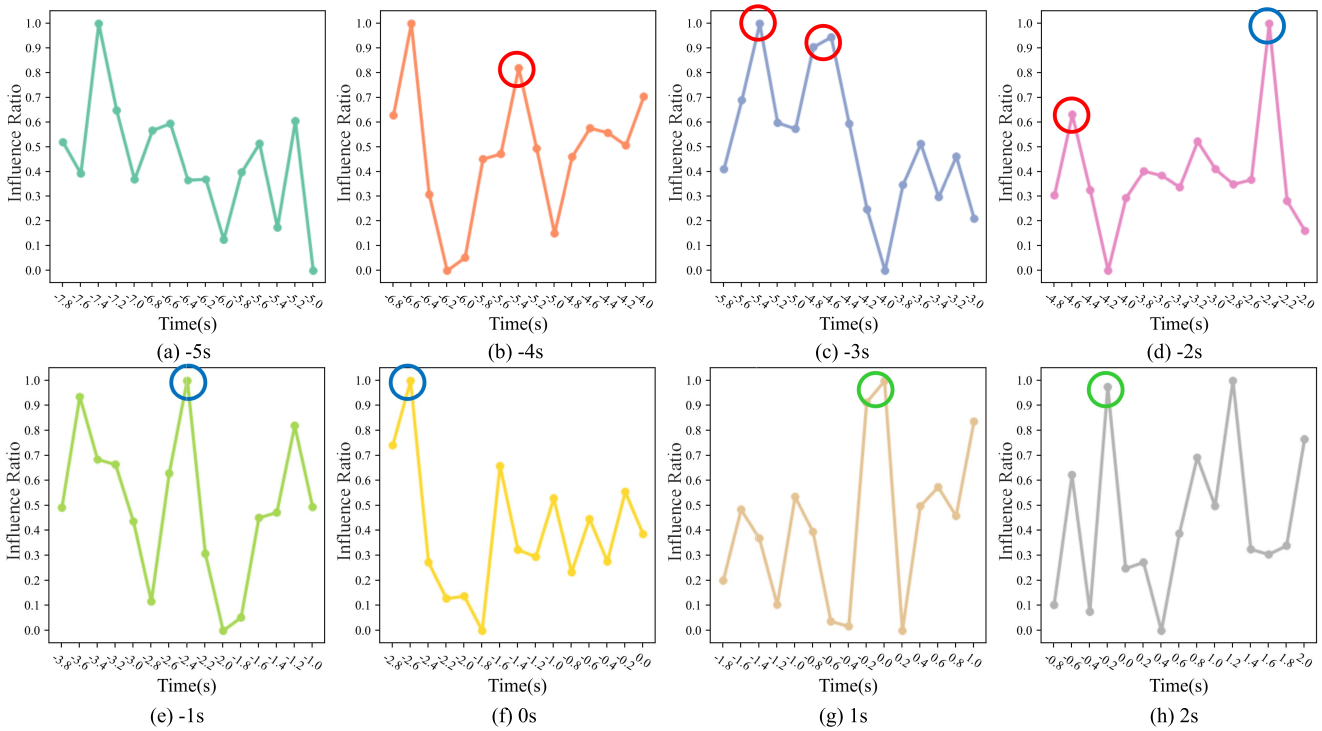


Fig. 5. Normalized relative influence between each time step and TPA output tensor of RLC condition. The moments have obvious response peaks in at least two subfigures are circled, where the red circle represents the first stage (−4.6 s, −5.4 s, possibly an intention to change lanes), the blue circle represents the second stage (−2.6 ∼ −2.4s, the driver began to turn the steering wheel), and the green circle represents the third stage (−0.2∼0s, vehicle crossing lane line).

3) Comparing the case of No $\boldsymbol{h}_{\mathcal{A}}$, Simple VOI, and the proposed model. The longitudinal and lateral errors were relatively large when there was no interaction information $\boldsymbol{h}_{\mathcal{A}}$ of the surrounding vehicles. The case of Simple VOI using a simple convolution kernel to encode the interaction of the surrounding vehicles can provide an improvement in accuracy. However, after using VOI inception, the interaction features of the surrounding vehicles can be extracted on the multi-scale, obtaining even higher accuracy.
4) The TPA mechanism outperformed the MHA in both longitudinal and lateral prediction, indicating that TPA can extract hidden driving features more effectively. The GCN encoded the surrounding vehicles using graph convolution to provide better long-term lateral prediction performance, but its performance was inferior to that of the proposed network in longitudinal prediction.

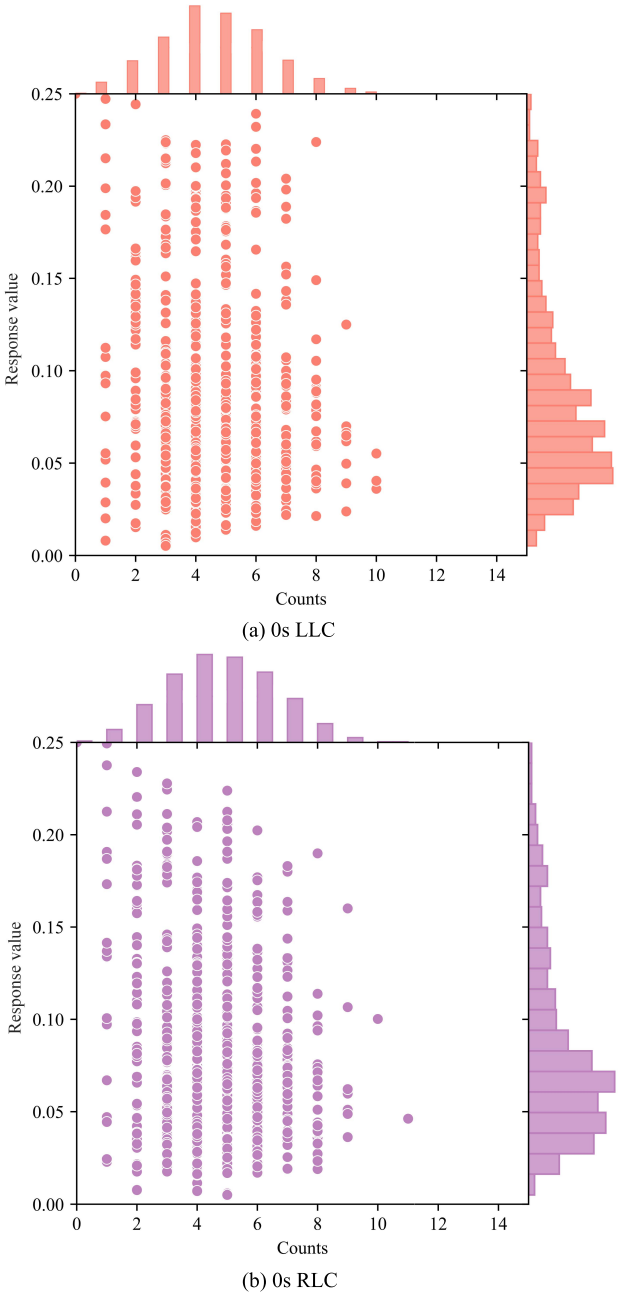
F. Output Explanation of TPA

In order to better demonstrate how the hidden driving features extracted by TPA in the proposed TP2Net network can improve prediction performance, the mutual information between the TPA output tensor and input were quantified. Only two representative driving maneuvers were analyzed: LLC and RLC. Generally, each driver has a unique "driving intention - lane changing maneuver - vehicle motion" pattern. There is also a difference between the two lane-changing maneuvers by one driver. Thus, this pattern may contain a lot of noise. Although we cannot quantify the importance value of each time step, we can clarify which is relatively important. Therefore, we hope to extract certain quantitative results from this pattern, which is a profile for most drivers. In order to reduce the impact of noise and ensure that results could be statistically significant, we analyzed 640 LLC and 640 RLC. The time at which the vehicle crossed the lane was labeled 0 s,

and the time step at which the input had the largest response to the TPA output tensor during the period from 7.8 s before crossing the lane to 2 s after crossing the lane was analyzed. In order to fully train the interpretation module, each trajectory was trained for 5000 epochs, and a learning rate of 10^{-4} was applied. After training, the relative importance of each trajectory was obtained in different time steps. Taking 0.25 as the upper limit of the importance statistics, only absolute importance values less than 0.25 were counted; the closer the value was to 0, the greater the impact it had on the output of the TPA module. To make it easier to observe the importance values of each time step, the statistical values were added according to the time step, the maximum value of the sum was subtracted from the sum and then the reciprocal was taken. After this normalization, the relative importance values for each time step were obtained.

Figure 3 shows the relative importance between each input time step and the TPA output tensor in the case of the LLC maneuver. A total of 10 s before and after the lane change was analyzed in 1-s intervals. The following time steps exhibited obvious response peaks in at least two subfigures in Figure 3: 0 s, −0.6 to 0.4 s, −2.2 to −2.0 s, −3.0 to −2.8 s, −4.8 s, and −5.8 s. These time steps can be divided into three groups according to lane change stage, described as follows:

1) 0 s and −0.6 to −0.4 s (Green circles in Figures 3 (f), (g), and (h)): Obviously, the large response values at these two time steps indicate that the actual vehicle was crossing the lane in response to the driver. At this time, the lateral speed reached its maximum value and the lateral jerk, that is, the change rate of lateral acceleration, reached its minimum value, as shown in Figures 4 (a), (c). It can be seen that the trends of the curves in Figures 3 (f) and (g) are the same except for a phase difference of 0.2 s at −1.8–0 s. This phase difference may be caused by the interval between predicted time steps.
2) −2.2 to −2.0 s and −3.0 to −2.8 s (Blue circles in Figures 3 (d), (e), and (f)): At these two time steps, the driver turned the steering wheel and began to change lanes. At −3.0 to −2.8 s, the lateral jerk of the vehicle was typically the largest, as shown in Figure 4 (c). This indicates that the driver was turning the steering wheel. At −2.2 to −2.0 s, the steering wheel rotation angle reached its maximum, and the lateral acceleration value was close to its maximum, as shown in Figure 4 (b).
3) −4.8 s and −5.8 s (Red circles in Figures 3 (a), (b), (c), and (d)): There was a long interval between these two time steps relative to the time required for lane changing, and the lateral speed, acceleration, and jerk values did not change significantly at either time step. According to [10], [11], the lane change intention will appear 1–4 s before the corresponding driving maneuver. Although it is difficult to define these two time steps showing higher response values as the time steps when the lane changing intention appeared, they at least considerably affected the trajectory predicted by the model before the LLC driving maneuver occurred. This shows that TPA can extract longer hidden driving patterns.



(a) 0s LLC

(b) 0s RLC

Fig. 6. Number and mean of high response values. The vertical axis is the mean value of the high response values extracted from each trajectory interpretation result through the sliding window. The horizontal axis is the number of high response values extracted from each trajectory.

In Figure 5, which shows the relative importance between each input time step and the TPA output tensor in the case of the RLC maneuver, there are three similar peak groups: −0.2 to 0 s, −2.6 to −2.4 s, and −4.6 s and −5.4 s. The interpretation of each group of peak values is the same as that for the LLC maneuver. There was a certain phase difference between the high response value of LLC and RLC that may be due to the different purposes and hidden features of LLC and RLC maneuvers.

Figure 6 shows the number and average of high response values of 320 trajectories each (selected randomly from 640 trajectories) at 0 s for the LLC and RLC maneuvers. The closer the value is to 0, the greater the impact of the trajectory on the prediction. The mean value was determined by a sliding window with a length of 0.1, and was recorded when the maximum number of high response values in the window was achieved. The figure shows that the LLC and RLC maneuvers are close in the number and distribution of high response values. The distribution of high response values indicates that there are about four key time steps within 3 s before a lane change, which is close to the number of blue and green circles in Figures 3 and 5. Because Gaussian noise was used as the disturbance, after sufficient training, if the final result is distributed uniformly on the vertical axis, the hidden features extracted by TPA can be considered to have little relationship with the input. It can be observed that the average value of the high response values was around 0.05, illustrating that the TPA can stably extract multiple high importance time steps. In combination, Figures 3, 4, 5, and 6 demonstrate that the TPA can effectively extract hidden driving features and use these features to make higher-precision predictions of driving maneuvers.

V. Conclusion

This study developed a vehicle trajectory prediction network called TP2Net and model interpretation based on TPA. The proposed perturbation-based method quantitatively measures the changes of input and hidden layer tensors by adding Gaussian noise, to mine the importance of certain dimensions. A total of 10 s before and after the lane change was analyzed in 1-s intervals to extract three important stages during the lane changing maneuver. Among these stages, if the vehicle crossed the lane line at about 0 s, the driver is ready to change lanes at about -2 to -3 s. Hence, the driver's intention to change lanes could have been observed by the model at about -4.5 to -6.0 s. Conclusively, the driving intention-maneuver and vehicle motion pattern are well captured by TPA. This finding indicates that TPA can effectively extract hidden driving features and use them in trajectory prediction.

Results of ablation experiments also indicated that TPA can effectively extract the lateral driving maneuvers of the target vehicle, the target vehicle encoding can compensate for the large longitudinal error, and thus the two methods complement each other. In addition, more input features and interaction between vehicles are essential. The interaction between the target vehicle and surrounding vehicles was extracted on a multi-scale to obtain greater accuracy. The experimental results demonstrated that the prediction performance of the proposed method was more than 15% greater than that of the previously best method.

The proposed model improves the accuracy of prediction in real scenarios and can output additional decision information for subsequent planning and control. The computational cost is reduced, and the inference speed is faster. The results of the interpretation module can improve the interpretability and the reliability of the network. However, there remain several important limitations of this study: the proposed method can only predict the trajectory of one vehicle at a time, which may lead to time complexities under different traffic densities. In addition, when an LC or RA and HB maneuver occurs in the future 4 to 5 s, the prediction accuracy will decline to a certain extent. Future work will focus on addressing these limitations. More surrounding vehicles will be considered, and the network can select vehicles of interest in end-to-end learning.

References

[1] Y. Xing et al., "Driver lane change intention inference for intelligent vehicles: Framework, survey, and challenges," *IEEE Trans. Veh. Technol.*, vol. 68, no. 5, pp. 4377–4390, May 2019.

[2] K. Messaoud, I. Yahiaoui, A. Verroust-Blondet, and F. Nashashibi, "Relational recurrent neural networks for vehicle trajectory prediction," in *Proc. IEEE Intell. Transp. Syst. Conf. (ITSC)*, Oct. 2019, pp. 1813–1818.

[3] K. Messaoud, I. Yahiaoui, A. V. Blondet, and F. Nashashibi, "Attention based vehicle trajectory prediction," *IEEE Trans. Intell. Vehicle*, vol. 6, no. 1, pp. 1–10, May 2020.

[4] H. Cui, V. Radosavljevic, F. C. Chou, T. H. Lin, N. Thi, and T. K. Huang, "Multimodal trajectory predictions for autonomous driving using deep convolutional networks," in *Proc. IEEE Int. Conf. Robot. Autom.*, May 2019, pp. 2090–2096.

[5] J. Li, H. Ma, W. Zhan, and M. Tomizuka, "Coordination and trajectory prediction for vehicle interactions via Bayesian generative modeling," in *Proc. IEEE Intell. Vehicles Symp. (IV)*, Jun. 2019, pp. 2496–2503.

[6] Q. Tran and J. Firl, "Online maneuver recognition and multimodal trajectory prediction for intersection assistance using non-parametric regression," in *Proc. IEEE Intell. Vehicles Symp. Proc.*, Jun. 2014, pp. 918–923.

[7] S. Lefèvre, D. Vasquez, and C. Laugier, "A survey on motion prediction and risk assessment for intelligent vehicles," *ROBOMECH J.*, vol. 1, no. 1, pp. 1–14, 2014.

[8] J. Hu and W. Zheng, "Multistage attention network for multivariate time series prediction," *Neurocomputing*, vol. 383, pp. 122–137, Mar. 2020.

[9] S.-Y. Shih, F.-K. Sun, and H.-Y. Lee, "Temporal pattern attention for multivariate time series forecasting," 2018, *arXiv:1809.04206*.

[10] P. Kumar, M. Perrollaz, S. Lefevre, and C. Laugier, "Learning-based approach for online lane change intention prediction," in *Proc. IEEE Intell. Vehicles Symp.*, Jun. 2013, pp. 797–802.

[11] A. Jain, H. S. Koppula, S. Soh, B. Raghavan, A. Singh, and A. Saxena, "Brain4Cars: Car that knows before you do via sensory-fusion deep learning architecture," 2016, *arXiv:1601.00740*.

[12] A. Vaswani et al., "Attention is all you need," in *Proc. 31st Int. Neural Inf. Process. Syst.*, Dec. 2017, pp. 6000–6010.

[13] C. Szegedy et al., "Going deeper with convolutions," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recogn.*, Jun. 2015, pp. 1–9.

[14] C. Szegedy, S. Ioffe, V. Vanhoucke, and A. A. Alemi, "Inception-v4, inception-ResNet and the impact of residual connections on learning," in *Proc. AAAI Conf. Artif. Intell.*, Feb. 2017, pp. 4278–4284.

[15] R. McAllister et al., "Concrete problems for autonomous vehicle safety: Advantages of Bayesian deep learning," in *Proc. Int. Joint Conf. Artif. Intell.*, Aug. 2017, pp. 4745–4753.

[16] M. T. Ribeiro, S. Singh, and C. Guestrin, "'Why should i trust you?': Explaining the predictions of any classifier," in *Proc. 22nd ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Aug. 2016, pp. 1135–1144.

[17] F. Doshi-Velez and B. Kim, "Towards a rigorous science of interpretable machine learning," 2017, *arXiv:1702.08608*.

[18] Y. Zhang, P. Tino, A. Leonardis, and K. Tang, "A survey on neural network interpretability," *IEEE Trans. Emerg. Topics Comput. Intell.*, vol. 5, no. 5, pp. 726–742, Aug. 2021.

[19] R. Schubert, E. Richter, and G. Wanielik, "Comparison and evaluation of advanced motion models for vehicle tracking," in *Proc. Int. Conf. Inform. Fusion*, Jun. 2008, pp. 1–6.

[20] N. Deo, A. Rangesh, and M. M. Trivedi, "How would surround vehicles move? A unified framework for maneuver classification and motion prediction," *IEEE Trans. Intell. Vehicles*, vol. 3, no. 2, pp. 129–140, Jun. 2018.

[21] A. Houenou, P. Bonnifait, V. Cherfaoui, and W. Yao, "Vehicle trajectory prediction based on motion model and maneuver recognition," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Nov. 2013, pp. 4363–4369.
[22] J. Firl, H. Stubing, S. A. Huss, and C. Stiller, "Predictive maneuver evaluation for enhancement of car-to-X mobility data," in *Proc. IEEE Intell. Vehicles Symp.*, Jun. 2012, pp. 558–564.
[23] C. Laugier, I. E. Paromtchik, M. Perrollaz, M. Yong, and J. Yoder, "Probabilistic analysis of dynamic scenes and collision risks assessment to improve driving safety," *IEEE Intell. Trans. Syst. Mag.*, vol. 3, no. 4, pp. 4–19, Oct. 2011.
[24] T. Hülnhagen, I. Dengler, A. Tamke, T. Dang, and G. Breuel, "Maneuver recognition using probabilistic finite-state machines and fuzzy logic," in *Proc. IEEE Intell. Vehicles Symp. (IV)*, Aug. 2010, pp. 65–70.
[25] G. Xie, H. Gao, L. Qian, B. Huang, K. Li, and J. Wang, "Vehicle trajectory prediction by integrating physics- and maneuver-based approaches using interactive multiple models," *IEEE Trans. Ind. Electron.*, vol. 65, no. 7, pp. 5999–6008, Jul. 2018.
[26] G. S. Aoude, B. D. Luders, K. K. H. Lee, D. S. Levine, and J. P. How, "Threat assessment design for driver assistance system at intersections," in *Proc. IEEE 13th Int. Conf. Intell. Transp. Syst.*, Sep. 2010, pp. 1855–1862.
[27] S. H. Park, B. Kim, C. M. Kang, C. C. Chung, and J. W. Choi, "Sequence-to-sequence prediction of vehicle trajectory via LSTM encoder–decoder architecture," in *Proc. IEEE Intell. Veh. Symp.*, Jun. 2018, pp. 1672–1678.
[28] N. Deo and M. M. Trivedi, "Convolutional social pooling for vehicle trajectory prediction," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. Workshops (CVPRW)*, Jun. 2018, pp. 1549–1557.
[29] N. Lee, W. Choi, P. Vernaza, C. B. Choy, P. H. S. Torr, and M. Chandraker, "DESIRE: Distant future prediction in dynamic scenes with interacting agents," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 2165–2174.
[30] L. Hou, L. Xin, S. E. Li, B. Cheng, and W. Wang, "Interactive trajectory prediction of surrounding road users for autonomous driving using structural-LSTM network," *IEEE Trans. Intell. Transp. Syst.*, vol. 21, no. 11, pp. 4615–4625, Nov. 2020.
[31] A. Kuefler, J. Morton, T. Wheeler, and M. Kochenderfer, "Imitating driver behavior with generative adversarial networks," in *Proc. IEEE Intell. Vehicles Symp. (IV)*, Jun. 2017, pp. 204–211.
[32] T. Zhao et al., "Multi-agent tensor fusion for contextual trajectory prediction," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recogn.*, Jun. 2019, pp. 12118–12126.
[33] B. Kim, C. M. Kang, J. Kim, S. H. Lee, C. C. Chung, and J. W. Choi, "Probabilistic vehicle trajectory prediction over occupancy grid map via recurrent neural network," in *Proc. IEEE 20th Int. Conf. Intell. Transp. Syst. (ITSC)*, Sep. 2017, pp. 399–404.
[34] H. Jeon, J. Choi, and D. Kum, "SCALE-Net: Scalable vehicle trajectory prediction network under random number of interacting vehicles via edge-enhanced graph convolutional neural network," 2020, *arXiv:2002.12609*.
[35] L. Shi et al., "SGCN: Sparse graph convolution network for pedestrian trajectory prediction," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recogn.*, Jun. 2021, pp. 8990–8999.
[36] K. Simonyan, A. Vedaldi, and A. Zisserman, "Deep inside convolutional networks: Visualising image classification models and saliency maps," 2013, *arXiv:1312.6034*.
[37] D. Smilkov, N. Thorat, B. Kim, F. Viégas, and M. Wattenberg, "SmoothGrad: Removing noise by adding noise," 2017, *arXiv:1706.03825*.
[38] M. Du, N. Liu, Q. Song, and X. Hu, "Towards explanation of DNN-based prediction with guided feature inversion," in *Proc. 24th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Jul. 2018, pp. 1358–1367.
[39] C. Guan et al., "Towards a deep and unified understanding of deep neural models in NLP," in *Proc. Int. Conf. Mech. Learn.*, Jun. 2019, pp. 1–10.
[40] F. Altché and A. de La Fortelle, "An LSTM network for highway trajectory prediction," in *Proc. IEEE 20th Int. Conf. Intell. Transp. Syst. (ITSC)*, Sep. 2017, pp. 353–359.
[41] R. Krajewski, J. Bock, L. Kloeker, and L. Eckstein, "The highD dataset: A drone dataset of naturalistic vehicle trajectories on German highways for validation of highly automated driving systems," in *Proc. 21st Int. Conf. Intell. Transp. Syst. (ITSC)*, Nov. 2018, pp. 2118–2125.
[42] J. Colyar and J. Halkias, "U.S. highway 101 dataset," Federal Highway Admin., Washington, DC, USA, Tech. Rep. FHWA-HRT-07–030, 2007.
[43] J. Colyar and J. Halkias, "U.S. highway I-80 dataset," Federal Highway Admin., Washington, DC, USA, Tech. Rep. FHWA-HRT-06–137, 2006.
[44] A. Alahi, K. Goel, V. Ramanathan, A. Robicquet, L. Fei-Fei, and S. Savarese, "Social lstm: Human trajectory prediction in crowded spaces," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 961–971.
[45] Y. C. Tang and R. Salakhutdinov, "Multiple futures prediction," 2019, *arXiv:1911.00997*.

Hongyu Hu (Member, IEEE) received the B.S. degree in computer science and technology in 2005 and the Ph.D. degree in traffic information engineering and control from Jilin University, Changchun, China, in 2010. Since 2020, he has been a Professor with the State Key Laboratory of Automotive Simulation and Control. From 2019 to 2020, he was a Visiting Scholar at California PATH, UC Berkeley. His research interests include connected and automated vehicles, driver behavior analysis, advanced driver assistance systems, and human–machine interaction. He is a Committee Member on Human Factors in Intelligent Transportation Systems and a member of Society of Automotive Engineers.

Qi Wang received the M.S. degree from the School of Mechanical Engineering, Beijing Institute of Technology, China, in 2019. He is currently pursuing the Ph.D. degree with the State Key Laboratory of Automotive Simulation and Control, Jilin University, China, with a focus on intelligent vehicle. His research interests include deep learning, driving risk assessment, and trajectory prediction and planning.

Ming Cheng received the B.S. degree in automotive engineering from the Harbin Institute of Technology in 2018. He is currently pursuing the Ph.D. degree with the State Key Laboratory of Automotive Simulation and Control, Jilin University, China. His research and academic interests include intelligent connected vehicle and advanced driving assistance systems.

Zhenhai Gao received the Ph.D. degree in automotive engineering from Jilin University, China, in 2000. He is currently a Professor at the State Key Laboratory of Automotive Simulation and Control, Jilin University. He was a Post-Doctoral Fellow at Xi'an Jiaotong University and a Foreign Researcher of The University of Tokyo. He has more than 20 years of research experience in a broad range of automotive engineering. In recent years, his research has been focused on connected and automated vehicles, driver behavior analysis, advanced driver assistance systems, and human–machine interface. He has more than 100 papers of his achievements have been published.