STS-DGNN: Vehicle Trajectory Prediction via Dynamic Graph Neural Network With Spatial–Temporal Synchronization

Feng-Jie Li, Chun-Yang Zhang, and C. L. Philip Chen, *Fellow, IEEE*

***Abstract*— Accurate prediction of vehicle trajectories is crucial to the safety and comfort of autonomous vehicles. Although several graph-based models have exhibited substantial progress in acquiring spatiotemporal dependencies among vehicles in the driving environment, the potential for additional exploration in this domain persists. The main reason is that they concentrated on independently capturing the spatial relations and temporal dependencies, neglecting to incorporate the temporal feature into the spatial feature for co-training, which limits their ability to yield satisfactory predictive accuracy. Typically, spatial and temporal correlations are coupled and should be modeled jointly. Inspired by this, a novel dynamic graph neural network with spatial–temporal synchronization (STS-DGNN) for vehicle trajectory prediction is proposed, which constructs the driving scene as dynamic graphs and can jointly extract spatial–temporal features. Specifically, low-order and high-order dynamics of vehicle trajectories are considered collaboratively in a one-stage framework rather than independently modeling the spatial relationship and temporal correlations of vehicles in two-stage models. The proposed model also considers the dynamic nature of graph sequence by utilizing gate recurrent unit (GRU) to update the graph neural network (GNN) parameters dynamically. The spatial–temporal features are subsequently conveyed to convolutional neural networks (CNNs) and processed by a multilayer perceptron (MLP) to generate the ultimate trajectories. Finally, to illustrate the effectiveness of the STS-DGNN model, the model is assessed on three well-known datasets, namely highD, EWAP, and UCY. The results confirm that our model performs better at making predictions than cutting-edge models. The visualization results intuitively explain that our method can extract sophisticated and subtle multivehicle interactions, resulting in accurate predictions.**

***Index Terms*— Autonomous driving, dynamic graph, graph neural network (GNN), spatial–temporal dependencies, vehicle trajectory prediction.**

I. Introduction

IN RECENT years, with the rapid developments of technologies in autonomous driving, autonomous vehicles are providing more convenient services to our life, such as safety warnings, motion planning, and driving decision-making [1]. However, based on the inquiry conducted by the NHTSA, there were 367 self-driving traffic accidents between July 2021 and May 2022. Such developments have instigated apprehensions about the safety of driverless vehicles. Accurate trajectory prediction has the potential to enhance the security of autonomous cars and is instrumental in advancing the prospects of autonomous driving. For instance, by predicting the trajectory of surrounding vehicles to recognize possible future collisions, vehicles can make reasonable driving decisions in advance to avoid collisions, thus reducing traffic accidents. Consequently, enhancing the precision of vehicle trajectory forecasting holds significant importance.

Vehicle trajectory prediction is a challenging task in practice due to the intricate dynamics of driving scenarios and complex spatial–temporal dependencies of vehicle trajectories. Early works usually predict vehicle trajectories in a noninteraction way, such as physics-based models [2], maneuver-based models [3], or combinations of the two [4], [5]. Besides, some RNN-based techniques, such as gate recurrent unit (GRU) and long short term memory (LSTM), have been implemented for identical tasks. However, these methods neglect the interactions between vehicles and can only achieve short-term predictions. Many existing works have confirmed that modeling intervehicle interactions is essential to enhance trajectory prediction accuracy in highly dynamic interaction driving scenarios. As a result, the above approaches were quickly substituted by interaction methods that considered the mutual influences among vehicles. More recently, methods based on attention mechanisms [6], [7], [8], [9] have been widely used for modeling intervehicle interactions. The incorporation of the attention mechanism has been proven to effectively mitigate the impact of extraneous information and amplify the influence of pertinent information, thereby leading to the superior predictive performance of the attention-based models compared to the methodologies that overlook intervehicle interactions. Currently, numerous powerful methods, such as graph neural network (GNN) and its variants [10], [11], [12], [13], [14], [15], have attracted much attention for their ability to model traffic scenes well. Pareja et al. [12] develop an approach called EvolveGCN, which integrates LSTMs and GRUs with graph convolutional networks (GCNs) to deal with dynamic graphs. Moreover, various other approaches [16], [17], [18], [19], [20], [21], [22], [23], [24], [25] have also been proposed for trajectory prediction tasks.

Manuscript received 3 May 2023; revised 19 July 2023; accepted 7 August 2023. Date of publication 23 August 2023; date of current version 7 September 2023. This work was supported in part by the National Natural Science Foundation of China under Grant 62076065 and in part by the Natural Science Foundation of Fujian Province under Grant 2020J01495. The Associate Editor coordinating the review process was Dr. Manyun Huang. *(Corresponding author: Chun-Yang Zhang.)*

Feng-Jie Li and Chun-Yang Zhang are with the School of Computer and Data Science, Fuzhou University, Fuzhou 350025, China (e-mail: zhangcy@fzu.edu.cn).

C. L. Philip Chen is with the School of Computer Science and Engineering, South China University of Technology, Guangzhou, Guangdong 510006, China.

Digital Object Identifier 10.1109/TIM.2023.3307179

1557-9662 © 2023 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission.
See https://www.ieee.org/publications/rights/index.html for more information.

Authorized licensed use limited to: UNIVERSITY OF NOTTINGHAM. Downloaded on May 16,2024 at 23:02:00 UTC from IEEE Xplore. Restrictions apply.

Despite the notable accomplishments of prior research in enhancing the precision of vehicle trajectory prediction, two key issues urgently need to be solved. First, the existing graph-based approaches build the traffic scene as graphs and treat vehicles and their spatial relationships as nodes and edges, respectively. However, the graphs constructed by the above methods are mostly static graphs with node features changing dynamically and edges fixed. These static graphs usually fail to simulate highly dynamic driving scenarios. Second, although many prior works have considered the extraction of spatiotemporal features, they all adopt two networks to capture the spatial relations and temporal dependencies independently rather than to extract spatial–temporal dependencies collaboratively. Therefore, these methods can not obtain high-order spatiotemporal features, and the prediction accuracy is limited.

To overcome the two limitations mentioned above, we propose a new model called dynamic GNN with spatial–temporal synchronization (STS-DGNN), which manages to build the traffic scene as dynamic graphs. Specifically, vehicles and their connection relationships are described accordingly as nodes and edges. Unlike most previous works, the nodes and edges here are dynamic over time rather than fixed. In addition, we conduct GNN on the 3-D dynamic graph tensor and design a spatiotemporal synchronization module. This module explores dynamic graphs by combining GRUs and GNNs, taking spatiotemporal features from each layer of GNN, which are then fed into convolutional neural network (CNN) and multilayer perceptron (MLP) to generate future trajectories. To be more specific, MLP is used to ensure that the output of the model has the expected dimension. Following is an overview of the main contributions.

1) An innovative model (STS-DGNN) for predicting vehicle trajectories is proposed, which successfully models the traffic scene with dynamic graphs. Besides, it is suitable for different graph encoders, such as GCN and graph attention networks (GATs).
2) To the best of our knowledge, this article is a unique work that can integrate GNN and GRU into the same one-stage framework to jointly capture spatiotemporal features directly rather than extracting spatial and temporal features separately. In this way, high-order features, such as vehicle driving patterns and the driving law of the surrounding vehicles, can be obtained, thus further enhancing the prediction accuracy of the model.
3) Extensive experimentation is carried out on three prominent datasets for trajectory prediction to confirm the validity of the proposed model. The findings indicate that our approach is productive and surpasses existing methods in forecasting accuracy.

The subsequent sections of this manuscript are structured as follows. Section II provides a concise survey of pertinent literature, while Section III outlines the problem formulation. Section IV presents the framework and implementation details, followed by the presentation of experimental results in Section V. Finally, Section VI concludes this work and points out potential future directions for research.

II. Related Works

As an imperative task in the autonomous vehicles field, the problem of vehicle trajectory prediction has attracted increasing attention in both industry and academia [26]. Thus far, numerous approaches have been proposed for the trajectory prediction problem. This section briefly reviews the existing methods, primarily focusing on interaction-aware-based models.

A. Noninteraction-Aware Prediction Methods

Early studies on vehicle trajectory forecasting mainly adopt physics-based models and maneuver-based models. Specifically, the former usually considers the dynamic and kinematic constraints (position, angle velocity, etc.) of the vehicle to predict its future trajectories. This type of model includes two main trajectory models, namely the Monte Carlo method [2] and the Kalman filter [27]. Since these methods only consider the low-order motion characteristics (dynamics and kinematics), these models have demonstrated efficacy in short-term prediction, limited to timeframes of 1 or 2 s. However, they prove inadequate for achieving satisfactory accuracy in long-term forecast. The maneuver-based models further improve the physics-based models by using the maneuver of the vehicle (turn around, or go straight, etc.) to predict future trajectories. Deo et al. [28] design a unified architecture for classifying vehicle maneuvers and predicting their movement. However, these maneuver-based models cannot accurately predict the trajectories of vehicles when they incorrectly identify the type of maneuver.

With the rapid development of deep learning, LSTM has been widely used in natural language processing, speech recognition, and other fields. Inspired by this, numerous researchers attempt to employ LSTM as a technique to predict vehicle trajectories. Park et al. [29] propose an LSTM-based encoder–decoder model, which employs the LSTM to encode previous trajectories and uses the decoder to produce vehicle trajectories. Similarly, a model based on LSTM [30] is developed to forecast future trajectories. Particularly, the data employed for the purpose of prediction corresponds to the tracking system based on Lidar technology. Although diverse factors of the vehicle itself are considered in the above approaches, the scope for improvement is limited as they fail to account for the crucial spatial interactions between cars, which have a significant impact on the accuracy of prediction.

B. Spatial Interaction Prediction Methods

Subsequently, various vehicle trajectory prediction techniques that consider spatial interactions with nearby vehicles are proposed. An integrated approach, devised by Xie et al. [4], combines physics-based and maneuver-based techniques to forecast vehicle trajectory. Deo and Trivedi [31] use a convolutional social pooling network (CSPN) with LSTMs to model the spatial interrelationships between cars within a given scene for vehicle trajectory prediction. Similarly, Hou et al. [22] and Liu et al. [23] adopt a structural-LSTM network, in which only the effects of the five

Authorized licensed use limited to: UNIVERSITY OF NOTTINGHAM. Downloaded on May 16,2024 at 23:02:00 UTC from IEEE Xplore. Restrictions apply.

nearest cars around the ego vehicle are considered. Later, Zhao et al. [32] design a MATF model incorporates a spatial tensor to represent spatial interrelationships between cars. However, the spatial tensor in MATF exclusively preserves spatial relationships at the last timestamp of past trajectories, disregarding any spatial–temporal reliances. Moreover, Fei et al. [33] propose a multi-LSTM encoder–decoder model. The encoder comprises seven LSTMs, one for the target vehicle and six for neighbor vehicles. In comparison, the decoder is included four additional LSTMs.

Besides the above methods, numerous investigators adopt attention mechanisms as a means to enhance predictive accuracy. Yan et al. [34] use two attention mechanisms (context attention and lane attention) specifically for trajectory prediction. Some techniques [35], [36] incorporate multihead attention into an encoder–decoder architecture based on LSTM, to account for spatial interactions when forecasting the vehicle trajectories. Cai et al. [37] propose an environment-attention network model to acquire helpful interactive information between a vehicle and its driving environment. Furthermore, a unified prediction framework is studied in [38] with a dual graph attention module for modeling. A trajectory forecast approach [39] integrating the double attention mechanism with the motion tendency of the ego car is proposed. Although the aforementioned spatial interaction methods have made significant progress, two limitations remain. One is that they solely predict the trajectory of a particular vehicle. To forecast trajectories for all cars, extensive computation resources are necessary. The other is that they solely focus on spatial interactions of vehicles but neglect the spatial–temporal interactions. The neglect of spatiotemporal dependencies in these previous approaches is apparent, resulting in unreliable long-term predictions. Alternatively, there are some pedestrian trajectory prediction models. For example, Yuan et al. [40] employ a novel transformer framework based on the agent-aware attention mechanism. Yue et al. [41] design a new model that combines social force models and neural networks for trajectory prediction. A novel model called SEEM, which includes a generator network and an energy network, is proposed by Wang et al. [42] and is based on sequence entropy energy.

C. Graph-Based Spatial–Temporal Interactions Prediction Methods

Recently, researchers try to exploit graph-based methods for learning the spatiotemporal dependencies of vehicles in the driving scene. For instance, Li et al. [43] develop a GRIP model, which combines convolution and graph operations to extract temporal features. Zhao et al. [44] propose a T-GCN for traffic prediction. However, the node characteristics change dynamically and the edges are fixed. An innovative approach [45] is proposed, leveraging a multivehicle collaborative learning framework incorporating spatial–temporal tensor fusion. This model uses an auto-encoder social convolution and cyclic social mechanisms to capture the spatial and temporal relationships between cars.

Later on, Wu et al. [46] propose a HSTA model, which uses a GAT and multihead attention mechanisms to model the topological and time-dependent relationships between vehicles. Sheng et al. [47] design a GSTCN model to produce ultimate trajectories. The network utilizes a GCN to learn spatial relationships among the cars in the graph. To date, only a few kinds of literature have adopted dynamic graphs to capture spatiotemporal features [15], [48]. Nevertheless, existing graph-based spatial–temporal interaction approaches utilize separate networks to learn spatial relationships and temporal dependencies, ignoring incorporating temporal features into spatial features for joint extraction to acquire higher-order features, which fails to obtain the ideal prediction accuracy. Thus, a comprehensive approach to jointly capturing spatial and temporal features still needs to be discovered. To tackle this issue, a novel STS-DGNN model is proposed, which can construct the driving environment as dynamic graphs and concurrently capture the spatial–temporal interactions.

III. Problem Formulation

Existing studies describe vehicle trajectory prediction tasks as the problem of estimating future vehicle trajectories based on given past trajectory information [43], [47]. The inputs $X \in \mathbb{R}^{t_h \times n \times 2}$ to the model consists of the position coordinates of all cars in the past period (past t_h time steps)

$$X = \left[P^1, P^2, \ldots, P^{t_h}\right] \tag{1}$$

where

$$P^t = \left[\left(x_1^t, y_1^t\right), \ldots, \left(x_n^t, y_n^t\right)\right] \in \mathbb{R}^{n \times 2}, \quad t = 1, 2, \ldots, t_h \tag{2}$$

are the position coordinates (x and y are the horizontal and vertical coordinates, respectively) of vehicles appearing at time t, and n indicates the number of cars. Since predicting the dynamic variation value of vehicle coordinates is generally easier than predicting their absolute coordinates, we employ the variation values ($P^t - P^{t-1}$) as input instead of absolute coordinates.

The output of the model is defined as $\hat{Y} \in \mathbb{R}^{t_f \times n \times 2}$, which represents the position coordinates of all cars in the forthcoming period $[t_h + 1, t_h + t_f]$

$$\hat{Y} = \left[\hat{P}^{t_h+1}, \hat{P}^{t_h+2}, \ldots, \hat{P}^{t_h+t_f}\right] \tag{3}$$

where t_f represents the prediction horizon.

IV. Proposed Model

The overarching structure of the presented model, displayed in Fig. 1, comprises three primary modules, namely: 1) the data preprocessing module (DPM) converts the raw data into a specific format, making the following computation more efficient; 2) the spatial–temporal synchronization module (STSM), through the joint extraction of spatial features and temporal features, to obtain higher-order features; and 3) trajectory prediction module (TPM) for predicting future trajectories. More specifically, the DPM mainly includes two operations: trajectory data preparation and dynamic graph construction. More details of each module in the framework are presented in Sections IV-A–IV-C.

Authorized licensed use limited to: UNIVERSITY OF NOTTINGHAM. Downloaded on May 16,2024 at 23:02:00 UTC from IEEE Xplore. Restrictions apply.

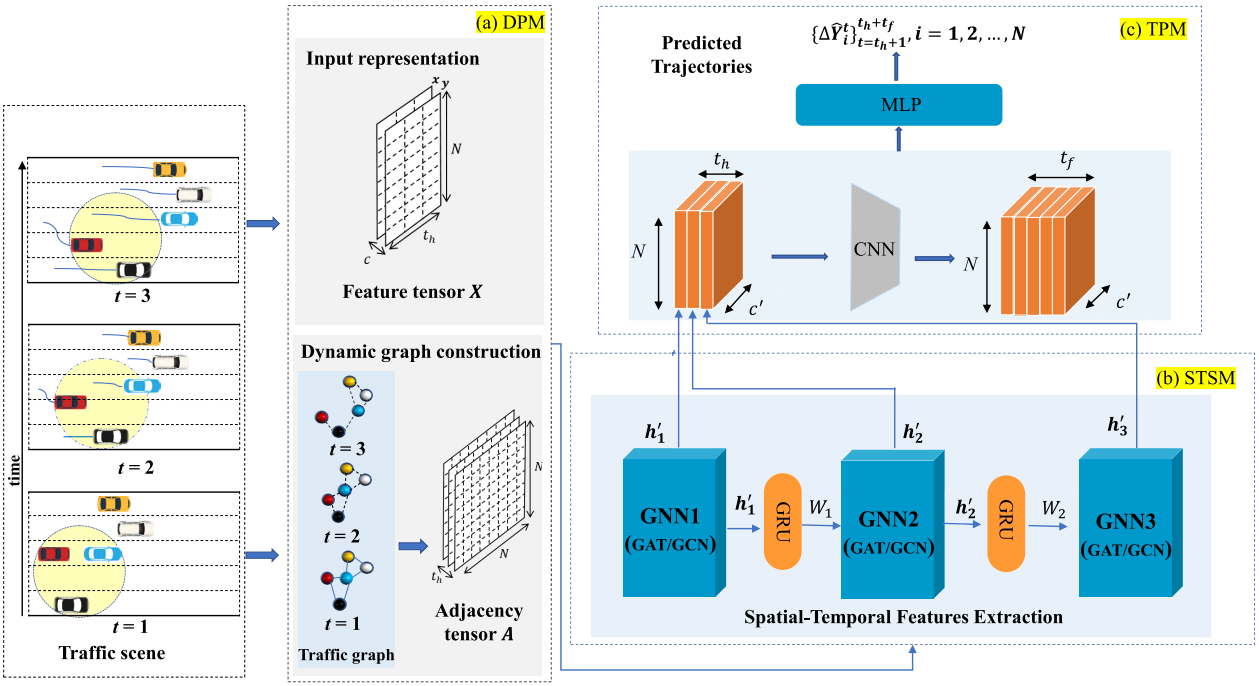

Fig. 1. Illustration of the STS-DGNN model. The model has three modules. (a) DPM receives the historical trajectories of cars as inputs and outputs the corresponding trajectories representation and adjacency matrix. (b) STSM captures the spatial–temporal features simultaneously. (c) TPM generates future trajectory information using the extracted spatial–temporal feature.

A. Data Preprocessing Module

1) Trajectory Data Preparation: This study treats vehicle trajectory data captured across various time steps as raw input to forecast prospective trajectories. Before incorporating the unprocessed trajectory data of the vehicles into our model, undergoes an initial conversion process to facilitate efficient computation in subsequent stages. Assume there are N cars in a driving scene that lasts t_h time steps. We represent input data by a 3-D feature tensor $X \in \mathbb{R}^{t_h \times N \times c}$ (see Fig. 1), where $c = 2$ represents the dimension of the input feature (that is, variation values of coordinates x and y).

2) Dynamic Graph Construction: In recent years, dynamic graph-based methods have become a popular solution for trajectory prediction due to their ability to capture complex vehicle interactions. In the practical application of autonomous driving, the motion of vehicles is usually affected by the movements of their adjacent vehicles, which is likeness to the behavior of people on social networks [43]. For example, a person is undoubtedly influenced by her/his surrounding friends. This inspires us to utilize spatial–temporal dynamic graphs $G = \{V, E\}$ to model driving scenarios, like what investigators have done for social networks. At each historical time step t, we can construct a traffic graph G_t (see the dynamic graph construction section in Fig. 1, from bottom to top). The spatial–temporal dynamic graph G, serving as the input of our model, can be obtained through the stacking $\{G_1, \ldots, G_{t_h}\}$. It is noteworthy that the topology of $\{G_1, \ldots, G_{t_h}\}$ is not fixed because the number of nodes at different time is not the same. Similar to [12], this article supposes that all graphs are constructed on a common node set with N nodes. An existing node is taken as an actual node, while a nonexistent node is considered a virtual node. The process of spatial–temporal dynamic graph construction is shown in Fig. 2.

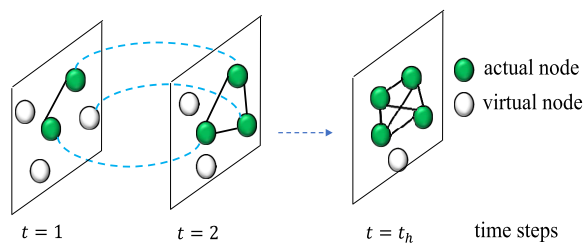

Fig. 2. Example of the construction process of spatial–temporal dynamic graphs. The nodes linked with dotted lines stands for the identical vehicle in distinct time steps, while nodes connected by solid lines indicate neighbor vehicles at the same time step. Notably, the dotted line, which connects a virtual node at $t = 1$ and an actual node at $t = 2$, means that the vehicle does not exist at $t = 1$ and appears at $t = 2$.

In this graph, nodes belonging to the node set V and edges of the edge set E are represented as cars and their interactions, respectively. Considering that the state of each vehicle is not the same at different time steps, we define $V = \{v_i^t | i = 1, \ldots, n; t = 1, \ldots, t_h\}$. And the feature vector v_i^t is the variation values of coordinates $(x_i^t - x_i^{t-1}, y_i^t - y_i^{t-1})$ of vehicle i at time t.

In each driving scenario, vehicles that have interactions are connected by edges. Specifically, such interaction only occurs when the distance between two vehicles is shorter than a threshold D of neighborhood distance. In this article, D defaults to 10 m. As shown in Fig. 1, we take light yellow circles with a diameter of D on a "traffic scene" as an

Authorized licensed use limited to: UNIVERSITY OF NOTTINGHAM. Downloaded on May 16,2024 at 23:02:00 UTC from IEEE Xplore. Restrictions apply.

example to demonstrate this threshold. In other words, vehicles that simultaneously appear in the same circle are considered neighbor vehicles to each other. The vehicles located within the circle are treated as neighbors to each other. Consequently, the E is defined as $E = \{v_i^t v_j^t | d(v_i^t, v_j^t) < D\}$, where $d(v_i^t, v_j^t)$ is Euclidean distance between vehicle i and j. More precisely, if $d(v_i^t, v_j^t) < D$, then an edge between them is denoted as $\langle v_i^t, v_j^t \rangle$.

We calculate the edge weights according to Euclidean distance initialization as follows:

$$w_{ij} = e^{-d\left(v_i^t, v_j^t\right)}, \quad \text{if } d\left(v_i^t, v_j^t\right) < D \tag{4}$$

where v_i^t and v_j^t are the variation values of coordinate of cars i and j at time t, respectively.

The adjacency matrix A_t is introduced to represent G_t

$$A_t = \begin{cases} w_{ij}, & \text{if edge } \left\langle v_i^t, v_j^t \right\rangle \in E \\ 0, & \text{otherwise} \end{cases} \tag{5}$$

where $A_t \in \mathbb{R}^{N \times N}, t = 1, \ldots, t_h$. The $A \in \mathbb{R}^{t_h \times N \times N}$ of G can be obtained through the stacking $\{A_1, \ldots, A_{t_h}\}$. Generally, the degree of interactions between any two vehicles is not all equal, necessitating a corresponding weight to be assigned to each edge in E_t. Without loss of generality, vehicles with a farther distance are assigned a lower weight, while vehicles at a closer distance are allocated a higher weight. Since the topological structure among vehicles is similar to the graph structure, previous work [46] has shown that GAT has been successfully used to extract spatial dependencies between neighbor vehicles. Inspired by this, we use GAT vehicle to acquire spatial interrelationships, which Section IV-B illustrates in detail.

B. Spatial–Temporal Synchronization Module

To simultaneously extract spatial–temporal features between multiple vehicles, a new called STSM is proposed in our model, see Fig. 1. The input data of the model contains 3-D feature tensor X and adjacency matrix tensor A. The STSM first maps the feature tensor into a higher-order dimension to make the model learn the trajectory features more adequately. Therefore, the shape of the output data of the STSM is $(t_h \times N \times C')$. In this work, the STSM can use two different graph encoders, that is GCN and GAT. One way is to conduct GAT on the 3-D dynamic graph tensor and incorporate the dynamic nature of the graph sequence by utilizing GRU to dynamically update the GAT parameters. In this way, we can obtain spatial–temporal features at the same time. To be more specific, at any time t, the GAT employs the self-attention mechanism to fuse information of neighboring vehicles, and then form new node embeddings (i.e., from h_i to h_i'). The GRU performs at the same layer but updates the weight matrix (i.e., from W_t to W_{t+1}) of GAT along the time from left to right. More details of STSM are described as follows.

At any time t, GAT takes the node features sequence h_i as input and produces a new node feature sequence h_i' as output. First of all, to more fully represent the characteristics of the node, the feature transformation Wh_i is performed on the node h_i. Then, a self-attention operation is performed on each node in the graph to calculate the attention coefficient between any two neighbor nodes

$$e_{ij} = \text{LeakyReLU}\left(\vec{a}^T \left[Wh_i \| Wh_j\right]\right) \tag{6}$$

that represents the importance of features of vehicle j to vehicle i. Notably, we only calculate the e_{ij} of node j in N_i, which is the set of first-order neighbor vehicles of vehicle i. To this end, the nonneighbor attention coefficient of each node is set as a small value ϵ through the following operation:

$$e_{ij} = \begin{cases} e_{ij}, & w_{ij} > 0 \\ \epsilon, & w_{ij} = 0 \end{cases} \tag{7}$$

where ϵ defaults set to $-10e^9$. The normalization of coefficients is performed by applying the softmax function to make the weight allocation between nodes more efficient

$$\alpha_{ij} = \text{softmax}_j\left(e_{ij}\right) = \frac{\exp\left(e_{ij}\right)}{\sum_{k \in N_i} \exp(e_{ik})}. \tag{8}$$

Additionally, to enhance the generalization ability of attention mechanisms, we chose to use multihead attention layers, which involve employing K groups of single-head attention layers that are independent of one another, and subsequently concatenating their outcomes together as

$$h_i' = \|_{k=1}^{K} \sigma\left(\sum_{j \in N_i} \alpha_{ij}^k W^k h_j\right) \tag{9}$$

where $\|$ indicates the concatenation operation, α_{ij}^k represents the weight coefficient computed by the group k attention mechanism, W^k denotes the corresponding weight matrix, and σ refers to the activation function.

The second way is to carry out GCN on the 3-D dynamic graph tensor and consider the dynamic nature of the graph sequence by utilizing GRU to dynamically update the GCN parameters. To be more specific, GCN conducted a particular time step but generate new node embedding (i.e., from $h_t^{(l)}$ to $h_t^{(l+1)}$) along the layers from bottom to top. The GRU performs at the same layer but updates the weight matrix (i.e., from $W_t^{(l)}$ to $W_{t+1}^{(l)}$) of GCN along the time from left to right.

This article uses superscript l and subscript t to denote the network layer index and time index, respectively. Notably, in this work, the number of convolutional layers is set to 2. The outputs of the first-layer convolutional are passed to the second-layer network to acquire high-order features. The GCN can be expressed as

$$h_t^{(l+1)} = \sigma\left(\widetilde{D}_t^{-1/2} \tilde{A}_t \widetilde{D}_t^{-1/2} h_t^{(l)} W_t^{(l)}\right) \tag{10}$$

where $\widetilde{A}_t = A_t + I, I \in \mathbb{R}^{N \times N}$ is the identity matrix, $\widetilde{D}_t$ is the degree matrix of $\widetilde{A}_t$, $h_t^{(l)}$ represents the node embedding, where superscript and subscript are the corresponding network layer number l and the time step t. It is worth noting that the shape of the node embeddings before and after the graph convolution operation remains the same. Additionally, $W_t^{(l)}$ denotes the learnable weight matrix, and the purpose of calculating $\widetilde{D}_t^{-1/2} \tilde{A}_t \widetilde{D}_t^{-1/2}$ is to symmetrically normalize the adjacency matrix, which is used to ensure that the value range of features remains the same.

Authorized licensed use limited to: UNIVERSITY OF NOTTINGHAM. Downloaded on May 16,2024 at 23:02:00 UTC from IEEE Xplore. Restrictions apply.

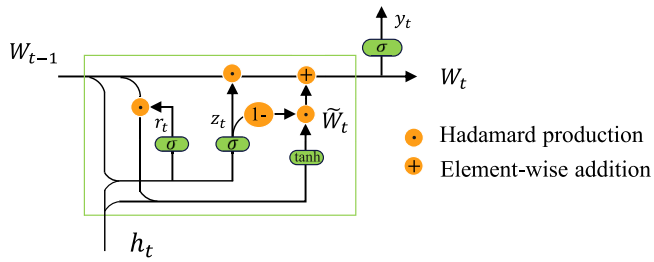

Fig. 3. Architecture of GRU.

The idea of the proposed STSM is to modify the weight matrix W_t of GAT (take GAT as an example. GCN is similar) by utilizing both the current input characteristics and the historical hidden state. The primary challenge is to incorporate temporal dynamics into the parameters of GATs and construct an evolutionary graph sequence. Luckily, this demand can be naturally satisfied by GRU. Fig. 3 shows the general structure of the GRU, where the node embedding h_t is regarded as the GRU current input, and the default initial node embedding matrix originates from the node feature matrix, e.g., $h_t = X_t$. The weight matrix W_{t-1} is taken as the GRU hidden state. Then, the current hidden state W_t of the GRU can be calculated by

$$\begin{aligned} r_t &= \sigma(h_t W_r + W_{t-1} U_r + b_r) \\ z_t &= \sigma(h_t W_z + W_{t-1} U_z + b_z) \\ \widetilde{W}_t &= \tanh\big(h_t W_q + (r_t \odot W_{t-1}) U_q + b_q\big) \\ W_t &= z_t \odot W_{t-1} + (1 - z_t) \odot \widetilde{W}_t \end{aligned} \tag{11}$$

where W_r, W_z, W_q, U_r, U_z, and U_q are trainable parameters matrices, b_r, b_z, and b_q are learnable bias, and $\odot$ represents the Hadamard production. Furthermore, r_t, and z_t are gated hidden states, $\widetilde{W}_t$ mean candidate hidden state. With (7), all the weight matrix of GAT can be obtained by recurrent updates.

C. Trajectory Prediction Module

A 3-D node embedding $H \in \mathbb{R}^{t_h \times N \times C'}$ can be obtained by extracting the spatio-temporal features of each vehicle from STSM. Subsequently, the node embeddings are fed into the TPM, which facilitates forecasting future vehicle trajectories. Nikhil and Tran Morris [49] verify that utilizing CNN for trajectory prediction outperforms RNNs due to the robust spatiotemporal interdependencies exhibited by trajectories. Inspired by the remarkable performance of CNNs, a module based on CNN architecture is devised to predict vehicle trajectories. Compared with GRU or LSTM, this module shows fewer parameter counts and faster computation speed due to the use of convolution operation. Finally, an MLP is used to guarantee that the output of the model has the anticipated shape.

The TPM utilizes the learned motion characteristics of the vehicle, namely $H \in \mathbb{R}^{t_h \times N \times C'}$, as its input to forecast future vehicle trajectories. It is clear that H includes t_h feature maps with the same shape $N \times C'$. Similar to [47], the convolution process of the filter is illustrated in Fig. 4. A filter conducts convolution operations on past t_h feature maps to produce a new feature map. Additionally, the kernel size is set to $I \times J$, and the appropriate zero padding is adopted to obtain dynamic information on the boundary portion of the time series. To obtain the value of position (x, y) in the $(t_h + m)$th feature map, we first compute the convolution of the kth kernel by

$$H_{xy}^{(t_h+m)k} = \sum_{i=1}^{I} \sum_{j=1}^{J} v_{ij}^{k} H_{(x+i-1)(y+j-1)}^{k} \tag{12}$$

where v_{ij}^{k} is the value at position (i, j) in the kth kernel and $H_{(x+i-1)(y+j-1)}^{k}$ represents the value at position $((x + i - 1), (y + j - 1))$ in the kth feature map. Suppose in the $(t_h + m)$th feature map, where the value of the position (x, y) can be calculated based on the convolution result of each kernel

$$H_{xy}^{t_h+m} = \sum_{k=m}^{t_h+m-1} H_{xy}^{(t_h+m)k}, \quad m = 1, \ldots, t_f. \tag{13}$$

By performing (12) and (13), a filter aggregates information of past t_h feature maps together to produce a new feature map. Finally, an MLP is used to generate the delta values of the position for the next time step

$$\Delta \hat{Y}^{t_h+m} = \mathrm{MLP}\big(H_{xy}^{t_h+m}\big), \quad m = 1, \ldots, t_f. \tag{14}$$

One commonly used loss function is the mean squared error between predicted and actual trajectories, given by

$$\mathrm{MSE} = \frac{1}{n \cdot t_f} \sum_{i=1}^{n} \sum_{t=t_h}^{t_h+t_f} \big\| \hat{P}_i^t - P_i^t \big\|_2^2 \tag{15}$$

where $\hat{P}_i^t$ and P_i^t are the forecast and actual positions at time t, respectively. $\|\cdot\|_2$ represents the $L2$-norm.

Finally, the pseudo-code of the algorithm is given in Algorithm 1.

Algorithm 1 The Training of STS-DGNN Algorithm

Input: Historial trajectories of all vehicles X, Eq. (2)
Output: The predicted trajectories of all vehicles $\hat{Y}$, Eq. (3)

```
Initialize parameters learning rate ł_r, maximum episode N_e, number of attention heads K, number of samples N_s, number of hidden units in GAT N_g and GCN N_c;
for i = 1 to N_e do
    for j = 1 to N_s do
        Apply Eq. (5) to acquire adjacency tensor A;
        The spatial–temporal tensor H is obtained by using Eq. (6)-(11);
        Use Eq. (12)-(13) to acquire new feature tensor;
        The delta values of the future trajectory ΔŶ is obtained by Eq. (14);
        Add them ΔŶ to the last historical location to convert the final predicted trajectories;
        Calculate the training loss using Eq. (16);
        Update the parameters via the gradient descent method of training loss;
    end for
end for
```

Authorized licensed use limited to: UNIVERSITY OF NOTTINGHAM. Downloaded on May 16,2024 at 23:02:00 UTC from IEEE Xplore. Restrictions apply.

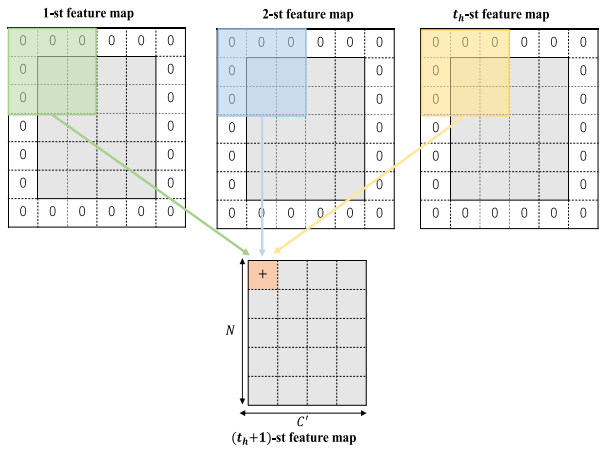

Fig. 4. Example of the convolution operation of each filter in TPM. A filter contains t_h kernels. The filter aggregates information from t_h feature maps in the past to produce a brand-new feature map. The green, blue, and yellow boxes denote three different kernels with the same shape of 3×3.

V. Experiments

This section initially presents three prominent datasets, namely highD, EWAP, and UCY datasets. Meanwhile, a comprehensive explanation of the evaluation metric employed in the experiments is presented. Subsequently, we compare our proposed approach against baselines, in addition to conducting sensitivity analysis and various ablation studies, aimed at substantiating the efficacy of the proposed model. In addition, visualization results are provided for further analysis.

A. Datasets and Metrics

1) Dataset Description: The highD encompasses driving information from over 110 500 vehicles, which is among the most comprehensive collections of vehicle trajectories and has been extensively employed in research pertaining to the prediction of vehicle trajectories [14], [35]. Each vehicle trajectory in the dataset contains a multitude of attributes, such as location, speed, and among others. The study gathered a dataset comprising 16.4 h of data obtained from six distinct road sections, each spanning approximately 420 m in length. The ratio of the training set, verification set, and test set is 7:1:2. Besides, the trajectories are discreted into several 8-s segments. The first 3 s of trajectory data are regarded as the historical input for each segment, while the last 5 s are utilized as the label value.

EWAP and UCY [50] are two public trajectory prediction datasets. EWAP includes two scenes (ETH and HOTEL), whereas UCY contains three scenes (UNIV, ZARA1, and ZARA2). The trajectories in these datasets are divided into several twenty-frame segments, with the first eight frames as input and the last twelve frames as predicted values.

2) Evaluation Metric: **Root mean squared error (RMSE)** is one of the most frequently used loss functions in regression tasks. Similar to previous works [51], [52], [53], the RMSE is employed as a quantitative metric for assessing the predictive performance of the STS-DGNN model. A smaller RMSE value indicates superior predictive performance. We report the results in terms of the RMSE between the predicted and ground-truth trajectories over a prediction time horizon of 5 s.

TABLE I

Hyperparameters Used in the Experiment

Symbol	Meaning	Value
K	Number of attention heads	16
l_r	Initial learning rate	0.0001
N_e	Maximum episode	100
N_g	Number of hidden units in GAT	16
N_c	Number of hidden units in GCN	32

The RMSE at time t is defined as

$$\mathrm{RMSE}_t = \sqrt{\frac{1}{n}\sum_{i=1}^{n}\left[\left(\hat{x}_i^t - x_i^t\right)^2 + \left(\hat{y}_i^t - y_i^t\right)^2\right]} \tag{16}$$

where n is the number of vehicles, $(\hat{x}_i^t, \hat{y}_i^t)$, and (x_i^t, y_i^t) are the predicted and ground-truth values of vertical and horizontal positions at predicted time step t, respectively.

1) *Average Displacement Error (ADE)*

$$\mathrm{ADE} = \frac{\sum_{i=1}^{n}\sum_{t=t_{h+1}}^{t_h+t_f}\left\|\hat{x}_i^t - x_i^t\right\|_2}{n^t\left(t_f - t_h\right)}. \tag{17}$$

2) *Final Displacement Error (FDE)*

$$\mathrm{FDE} = \frac{\sum_{i=1}^{n}\left\|\hat{x}_i^{t_f} - x_i^{t_f}\right\|_2}{n}. \tag{18}$$

B. Implementation Details and Training Process

The proposed STS-DGNN model is implemented with PyTorch. All experiments were conducted on the AutoDL AI platform, which features a 3090 GPU with 48 GB of RAM. The hyperparameters used in this work are shown in Table I.

The Adam optimizer is adopted to train the proposed model, wherein the l_r undergoes a decay of 0.8 every ten epochs, thereby expediting the rate of convergence of the loss function. The model weights are initialized with Xavier normal. The batch size is configured as 1. In TPM, the length, width, and zero padding of the kernel are 3, 3, and 1, respectively.

Fig. 5 shows the training process of the proposed model, in which the red and green curves represent the changes in

Authorized licensed use limited to: UNIVERSITY OF NOTTINGHAM. Downloaded on May 16,2024 at 23:02:00 UTC from IEEE Xplore. Restrictions apply.

TABLE II
RMSE RESULTS OF BASELINE MODELS AND THE STS-DGNN MODEL ON THE HIGHD DATASET

Prediction	CS-LSTM-M	CS-LSTM	NLS-LSTM	MHA-LSTM	MHA-LSTM(+f)	STDAN	STS-DGAT(Ours)	STS-DGCN(Ours)
1	0.23	0.22	0.20	0.19	**0.06**	0.19	0.13	0.08
2	0.65	0.61	0.57	0.55	**0.09**	0.27	0.27	0.18
3	1.29	1.24	1.14	1.10	**0.24**	0.48	0.43	0.31
4	2.18	2.10	1.90	1.84	0.59	0.91	0.63	**0.47**
5	3.37	3.27	2.91	2.78	1.18	1.66	0.87	**0.66**
Avg.	1.54	1.49	1.34	1.29	0.43	0.70	0.47	**0.34**

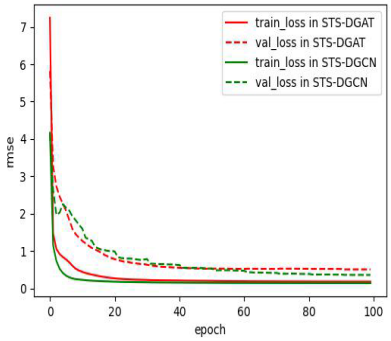

Fig. 5. Training process of the models.

RMSE in the STS-DGAT and STS-DGCN models, respectively. The solid and dashed lines indicate the training and validation losses, respectively. It is evident from the figure that the two training curves exhibit a consistent downward trend, proving the convergence of the prediction model.

C. Baselines

This section aims to provide a comprehensive evaluation of the performance of the STS-DGNN model. The following baselines are utilized for comparative analysis.

1) **CS-LSTM-M** [31] is an LSTM model with CSPN. It also includes maneuver classier.
2) **CS-LSTM** [31] combines LSTMs with CNNs to capture spatiotemporal features and accomplish the trajectory predictions on highways. Unlike **CS-LSTM-M**, this method has no maneuver classier.
3) **NLS-LSTM** [36] is an LSTM encoder–decoder model, which incorporates a social pooling mechanism to effectively capture and model the interactions with the adjacent cars.
4) **MHA-LSTM** [35] is an improved attention-based technique for learning interactive vehicle relations.
5) **MHA-LSTM(+f)** [35] is an extended version of MHA-LSTM with additional input features (velocity, acceleration, and category) and three attention heads.
6) **STDAN** [14] is a novel spatial–temporal dynamic attention network. One of the key characteristics of the network is its ability to hierarchically extract spatiotemporal features.
7) **RSBG** [54] is a new-style social behavior modeling graph representation, which uses GCNs to aggregate human social behaviors in dynamic scenes to complete prediction tasks.
8) **Tra2Tra** [55] learns the features of spatial interaction via decentralized operations and attentional strategies and then captures its temporal reliance employing LSTM to obtain a global spatiotemporal representation.
9) **SGCN** [56] utilizes spatial graphs to extract the interaction with pedestrians. In addition, sparse-directed temporal graphs are used to simulate motion trends.
10) **T-GNN** [50] performs together for trajectory prediction and domain alignment within a unified framework.
11) **AgentFormer** [40] is a novel transformer framework based on the agent-aware attention mechanism.
12) **NSP-SFM** [41] is a model that combines social force models and neural networks.
13) **SEEM** [42] a new model based on sequence entropy energy.
14) **STS-DGAT** is the model described in this work utilizing GAT as a graph encoder.
15) **STS-DGCN** is another model described in this work using GCN as the graph encoder.

The comparison results of the above baselines and the proposed STS-DGNN in different datasets are summarized in Tables II–IV, respectively. Notably, all results are in meters, and the best results are highlighted in bold.

For the highD dataset, as shown in Table II, STS-DGAT is slightly worse than the MHA-LSTM(+f) in short predictions (first 4 s) but works better than that in long-term forecasts (fifth second). STS-DGCN was slightly worse than MHA-LSTM(+f) in short-term (first 3 s) predictions, but better than MHA-LSTM in long-term (last 2 s) predictions. Thus, our algorithm is more suitable for long-term prediction than MHA-LSTM(+f). In summary, our model can be applied to different graph encoders, and all of them can achieve excellent performance.

For the EWAP and UCY datasets, refer to [50], which views each scene as one domain. Based on the differences in the tasks performed by the seven baselines (RGBG, Tra2Tra, SGCN, T-GNN, AgentFormer, NSP-SFM, and SEEM), we split them into two groups, with the former four being the first group and the final three being the second group. The first group trained the model using only one domain and tested it separately in the other four domains. Thus, there are 20 trajectory prediction tasks in total: E→H/U/Z/R, H→E/U/Z/R, U→E/H/Z/R, Z→E/H/U/R, and R→E/H/U/Z, where E, H, U, Z, and R denote ETH, HOTEL, UNIV, ZARA1, and ZARA2, respectively. Table III shows the ADE and FDE results of four existing baselines and our models in these 20 tasks. Note that our approach has a better performance than other

Authorized licensed use limited to: UNIVERSITY OF NOTTINGHAM. Downloaded on May 16,2024 at 23:02:00 UTC from IEEE Xplore. Restrictions apply.

TABLE III

ADE/FDE RESULTS OF BASELINE MODELS AND THE STS-DGNN MODEL ON THE EWAP AND UCY DATASETS. "→" DENOTES FROM SOURCE DOMAIN TO TARGET DOMAIN

Tasks	RSBG	Tra2Tra	SGCN	T-GNN	STS-DGAT(ours)	STS-DGCN(ours)
E→H	2.21 / 3.42	1.72 / 3.29	1.68 / 3.22	1.13 / 2.18	**0.77** / **1.33**	0.88 / 1.52
E→U	1.59 / 2.96	1.58 / 2.88	1.54 / 2.81	1.25 / 2.25	**0.57** / **1.05**	0.60 / 1.08
E→Z	1.48 / 2.75	1.27 / 2.66	1.26 / 2.52	0.94 / 1.78	**0.40** / 0.72	0.38 / **0.67**
E→R	1.42 / 2.75	1.37 / 2.66	1.28 / 2.52	1.03 / 1.78	0.36 / 0.67	0.35 / **0.64**
H→E	3.18 / 5.28	3.32 / 5.22	3.22 / 5.18	2.54 / 4.15	1.68 / 2.85	**1.53** / **2.68**
H→U	1.49 / 2.59	1.36 / 2.50	1.38 / 2.47	1.08 / 1.82	**1.03** / **1.79**	1.07 / 1.90
H→Z	2.72 / 5.19	2.67 / 4.89	2.62 / 4.83	2.25 / 4.04	0.85 / 1.45	**0.82** / **1.44**
H→R	1.73 / 3.10	1.58 / 2.90	1.58 / 2.85	1.41 / 2.53	0.80 / 1.37	**0.75** / **1.33**
U→E	1.23 / 2.36	1.16 / 2.29	1.14 / 2.24	0.97 / 1.91	0.96 / 1.67	**0.91** / **1.59**
U→H	0.87 / 1.55	0.70 / 1.33	0.70 / 1.32	**0.54** / **1.12**	0.87 / 1.50	0.67 / 1.15
U→Z	1.04 / 1.99	0.85 / 1.78	0.82 / 1.71	0.61 / 1.30	**0.41** / 0.74	0.42 / **0.73**
U→R	0.60 / 1.37	0.60 / 1.09	0.52 / 1.03	**0.23** / 0.87	0.39 / **0.70**	0.39 / **0.70**
Z→E	1.19 / 2.28	1.09 / 2.26	1.05 / 2.23	0.88 / 1.92	**0.72** / 1.29	**0.68** / **1.22**
Z→H	1.21 / 2.22	1.07 / 2.12	0.97 / 1.90	0.78 / 1.46	0.92 / 1.58	**0.65** / **1.15**
Z→U	0.80 / 1.77	0.81 / 1.63	0.80 / 1.48	0.59 / 1.25	**0.44** / **0.82**	0.49 / 0.90
Z→R	0.49 / 0.97	0.52 / 0.92	0.48 / 0.97	0.32 / 0.65	**0.30** / **0.56**	**0.30** / **0.56**
R→E	1.09 / 2.19	1.03 / 2.18	0.97 / 2.10	0.87 / 1.86	0.69 / 1.22	**0.67** / **1.20**
R→H	1.37 / 2.29	1.10 / 2.06	1.08 / 1.95	**0.72** / 1.45	0.79 /1.35	**0.72** / **1.28**
R→U	1.03 / 1.81	0.75 / 1.52	0.75 / 1.52	0.65 / 1.28	**0.43** / **0.80**	0.47 / 0.86
R→Z	0.78 / 1.34	0.52 / 1.17	0.51 / 0.99	0.34 / 0.72	**0.33** / **0.58**	0.34 / 0.59
Avg.	1.38 / 2.50	1.25 / 2.37	1.22 / 2.29	0.96 / 1.82	0.69 / 1.20	**0.65** / **1.16**

TABLE IV

RESULTS OF COMPARATIVE MODELS ON EWAP AND UCY DATASETS. "k" DENOTES THE NUMBER OF OUTPUT TRAJECTORIES FOR EACH AGENT PER STEP

Datasets	AgentFormer (k = 1)	NSP-SFM (k = 20)	SEEM (k = 1)	STS-DGCN (k = 1)
ETH	0.99 / 1.96	**0.25** / **0.24**	0.86 / 1.20	0.67 / 1.20
HOTEL	0.37 / 0.76	**0.09** / **0.13**	1.18 / 1.21	0.57 / 1.00
UNI	0.64 / 1.34	**0.21** / **0.38**	0.65 / 1.04	0.45 / 0.84
ZARA1	0.46 / 1.00	**0.16** / **0.27**	0.47 / 0.61	0.32 / 0.58
ZARA2	0.38 / 0.84	**0.12** / **0.20**	0.50 / 0.68	0.29 / 0.55
Avg.	0.57 / 1.18	**0.17** / **0.24**	0.71 / 0.95	0.46 / 0.83

TABLE V

PARAMETERS OF FIVE MODELS

Model	GRIP	GSTCN	NSP -SFM	STS-DGAT (ours)	STS-DGCN (ours)
Params(k)	496.3	48.9	1300.5	1.23	10.67

baselines in almost all cases in terms of ADE and FDE metrics. Moreover, STS-DGCN performs better than the best baseline T-GNN on average in ADE and FDE by 32.3% and 36.3%, respectively. The second group employs the classic leave-one-out evaluation approach [41], where the model is trained on four sub-datasets and evaluated on the remaining one. The experimental results are shown in Table IV. Note in this table that all models except NSP-FSM report the deterministic results for each agent. NSP-SFM is stochastic and learns the distribution per step, which allows it to generate 20 samples for each prediction step. Thus, NSP-SFM has a lower minimum than other models. Despite achieving the best performance on ADE/FDE, the physics model of NSP-FSM overly abstracts people into 2-D particles. Real-world situations involve significantly more complex agents whose behavior may be impacted by multiple variables, including their interactions with complex environments. Besides, this NSP-FSM requires generating 20 trajectories for each agent per step, which hinders its application in real-time scenarios, particularly those implemented on resource-limited embedded systems. To ensure a fair comparison, we only compare deterministic models AgentFormer and SEEM. Specifically, our proposed STS-DGCN outperforms AgentFormer by 19.3% and 29.7% on average in ADE and FDE. Compared with SEEM, STS-DGCN improves the average ADE and FDE by 35.2% and 12.6%. The experimental results further show that this method has good universality in the trajectory prediction task, which can not only effectively predict the vehicle trajectory prediction, but also accurately predict the pedestrian trajectory.

In Table V, we list the parameters of GRIP, GSTCN, NSP-SFM, STS-DGAT and STS-DGCN. The model size determines whether an algorithm can be deployed to an autonomous vehicle. One of the often used indicators for evaluating model size is parameters. A model with fewer parameters is more conducive to its deployment reasoning. As can be seen, GSTCN has one-tenth of the parameters of GRIP. Compared with GSTCN, the proposed STS-DGCN and STS-DGAT models have fewer parameters, which facilitates the deployment of the proposed models.

In addition, we qualitatively analyzed the predictive performance of STS-DGAT by visualizing some representative actual and predicted trajectories under three different driving scenarios, as shown in Fig. 6, it can observe that the rear vehicle appears to be influenced by the presence of the vehicle to its left as well as the vehicle ahead, resulting in a rightward lane change. This observation serves as empirical validation of the efficacy of our proposed model in capturing the intervehicle interactions. In Fig. 6(b) and (c), we further note that our model exhibits a high degree of success in accurately

Authorized licensed use limited to: UNIVERSITY OF NOTTINGHAM. Downloaded on May 16,2024 at 23:02:00 UTC from IEEE Xplore. Restrictions apply.

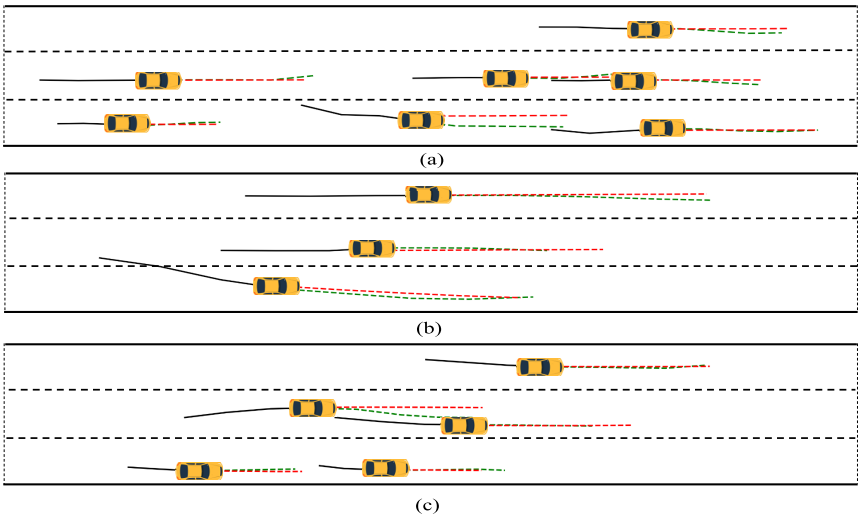

Fig. 6. Visualization outcomes of predicted trajectories in (a) mild, (b) moderate, and (c) congested driving environment. The historical and future ground truth trajectories, along with the future predicted trajectories of the vehicles, are depicted by the black-solid lines, green dashed lines, and red dotted lines, respectively.

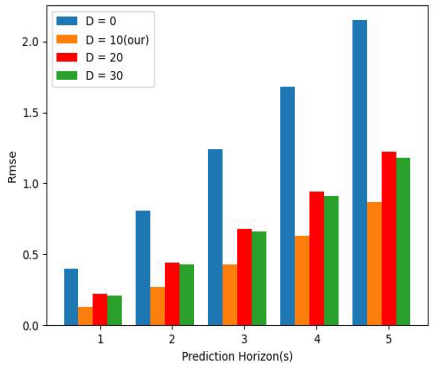

Fig. 7. Comparison among various D on highD dataset.

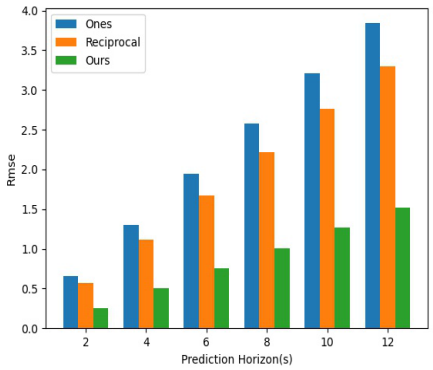

Fig. 8. RMSE results under different weighted adjacency matrices.

predicting the future trajectories of all vehicles, indicating that the proposed model can effectively capture the subtle and complex interactions among multiple vehicles. In a nutshell, these visualized results prove that our method could better capture the interactions of vehicles and accurately predict the future trajectories of vehicles under different driving scenarios, such as lane-changing driving behavior, etc.

D. Sensitivity Analysis

1) Influence of Different Neighbor Distance Thresholds: In this section, we have chosen partial data for the experiments to verify the effect of different neighbor distance thresholds D on the prediction error. In this experiment, we set D to the number from $\{0, 10, 20, 30\}$ and compare the results on highD dataset. Note that $D = 0$ indicates that interactions with nearby vehicles are not considered. The findings of our experiments are depicted in Fig. 7. The RMSE decreases as D increases (from 0 to 10), indicating that incorporating interactions with surrounding vehicles can effectively reduce the model prediction error. However, the prediction error gradually increases when the D surpasses 10. The reason may be that the large probability only influences the trajectory of the vehicle that the neighbor vehicles are close enough. Therefore, other cars considered far away did not improve the prediction effect. Based on this observation, $D = 10$ by default.

2) Influence of Different Numbers of Attention Heads and Hidden Units: In general, choosing the appropriate hyperparameters can improve the performance of our STS-DGAT model. Thus, to confirm the best hyperparameters of the proposed model, a method called DOE is adopted here, which is similar to [53]. We first test the effect of different values of K and N_g on the model performance, see Tables VI and VII. The results reveal that the RMSE decreases as attention heads increase. The RMSE will no longer decrease, when K transcends a specific value. It shows that considering the influence of attention heads can reduce prediction errors. When the number of attention heads is big enough, it will not further reduce the error. In addition, the model yields the minimum prediction error (average RMSE equal to 0.47 for highD and ADE/FDE equal to 0.64/1.08 for ETH dataset) when the number of attention heads and hidden units is $16 + 6$, $16 + 32$, and $32 + 8$. Considering the complexity of the model, we choose $K = 16$ and $N_{\text{hidden}} = 16$ as our default settings in this experiment. Apart from the above parameters, all experiments have the same hyperparameters for training and testing.

Authorized licensed use limited to: UNIVERSITY OF NOTTINGHAM. Downloaded on May 16,2024 at 23:02:00 UTC from IEEE Xplore. Restrictions apply.

TABLE VI
COMPARISON OF RMSE FOR THE MODELS WITH DISTINCT NUMBERS OF ATTENTION HEADS AND HIDDEN UNITS

Dataset	Prediction Horizon(s)	Number of K and N_g														
		2+8	2+16	2+32	4+8	4+16	4+32	8+8	8+16	8+32	16+8	16+16	16+32	32+8	32+16	32+32
highD	1	0.63	1.53	0.61	0.38	0.22	0.38	0.18	0.17	0.29	0.16	0.13	0.16	0.22	0.13	0.16
	2	1.27	3.05	1.21	0.76	0.44	0.76	0.36	0.36	0.59	0.33	0.27	0.33	0.44	0.28	0.32
	3	1.92	4.59	1.83	1.16	0.68	1.17	0.57	0.58	0.90	0.53	0.43	0.52	0.69	0.44	0.51
	4	2.59	6.13	2.45	1.56	0.93	1.58	0.79	0.82	1.22	0.75	0.63	0.73	0.95	0.64	0.71
	5	3.27	7.67	3.08	1.98	1.21	2.01	1.04	1.09	1.55	1.00	0.87	0.97	1.24	0.88	0.95
	avg.	1.94	4.59	1.83	1.17	0.69	1.18	0.59	0.61	0.91	0.55	**0.47**	0.54	0.71	0.48	0.53

TABLE VII
COMPARISON OF ADE AND FDE FOR THE MODELS WITH DISTINCT NUMBERS OF ATTENTION HEADS AND HIDDEN UNITS

Metric	Number of K and N_g														
	2+8	2+16	2+32	4+8	4+16	4+32	8+8	8+16	8+32	16+8	16+16	16+32	32+8	32+16	32+32
ADE	1.18	0.88	0.77	0.75	0.74	0.80	0.70	0.67	0.64	0.64	**0.61**	0.80	0.66	0.66	0.66
FDE	2.04	1.49	1.33	1.28	1.26	1.38	1.22	1.17	1.13	1.13	**1.08**	140	1.18	1.15	1.15

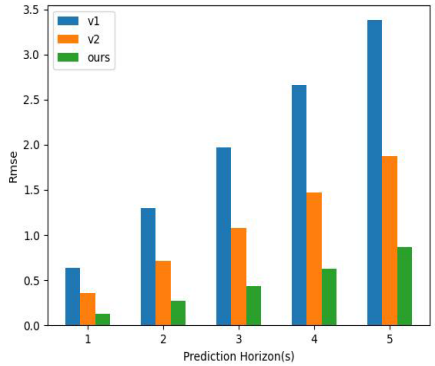

Fig. 9. Comparison of different modules on highD dataset.

3) Influence of Different Weighted Adjacency Matrices: In this section, we choose to use ETH in the model STS-DGCN to verify the effect of different weighted adjacency matrices on the prediction error. To reduce the complexity of the model, we downsampled every two frames, namely {2, 4, 6, 8, 10, 12}. In Section IV, we have defined (4) in the adjacency matrix. This definition assigns exponential weights to all surrounding vehicles. Intuitively, closer vehicles are assigned larger weights, while the weights decay exponentially for more distant vehicles to reduce their impact. Therefore, to verify the effectiveness of this definition for our model, we compare our method with two other forms. One is to set all weights to 1 directly, and the other is to use the reciprocal of the distance between two vehicles to mark the weights.

The comparison results are shown in Fig. 8. It can be seen that our method obtains the lowest prediction errors (green bars in Fig. 8) for all prediction horizons, which reveals that marking the appropriately weighted adjacency matrix helps improve the prediction performance of the model. In addition, we also note that the prediction error is the largest when setting all edge weights to 1. The reason behind this is that the adjacency matrix has the same edge weight of 1 between any two vehicles, ignoring the fact that different neighborhood vehicles have different influences.

E. Ablation Study

1) Influence of Different Modules: To validate the effectiveness of the relevant components in the proposed model, we compare the complete STS-DGAT with the following three variants of the model.

1) The first variant (**v1**) removes the GAT module from our STS-DGAT model, resulting in the omission of the spatial correlation information of cars.
2) The second variant (**v2**) uses two components separately to capture the spatial and temporal dependencies, where GAT is used to model the topological relationships between vehicles throughout the scenario, and then GRU are used to capture the temporal dependencies. As a result, they only concentrate on capturing the spatial relations and temporal dependencies independently rather than cooperatively extracting the spatial–temporal features.

The results are reported in Fig. 9, where we can observe that removing any module from STS-DGAT will increase RMSE, illustrating the validity of each module. It is also easy to find that the **v1** network has the largest RMSE value since it hardly considers vehicle interactivity. In addition, the RMSE of STS-DGAT is lower than **v2**, indicating that the combined extraction of spatiotemporal features can effectively reduce model prediction errors. In summary, our model yields a marginally superior predictive performance compared to the two variants of the model.

VI. CONCLUSION, LIMITATIONS, AND FUTURE WORK

In this article, a novel vehicle trajectory prediction model called STS-DGNN is proposed, which manages to construct the driving scene as dynamic graphs. Specifically, we employ GNN on the 3-D dynamic graph tensor and use GRU in a synchronous manner to update the GNN parameters. The spatial–temporal features are then passed to the CNN and a MLP to generate future trajectories. Thanks to the above design, the proposed model obtains higher-level spatial–temporal features than the previous models, which

Authorized licensed use limited to: UNIVERSITY OF NOTTINGHAM. Downloaded on May 16,2024 at 23:02:00 UTC from IEEE Xplore. Restrictions apply.

further improves the prediction accuracy of the model. In particular, to the extent of our knowledge, this research represents the first-ever attempt to jointly extract the spatial–temporal features among cars. The performance of the STS-DGNN model was assessed on the highD, EWAP and UCY datasets and compared with other existing models. The outcomes illustrate that the proposed model has the best prediction performance. Simultaneously, several ablation studies are conducted to ascertain the efficacy of each module incorporated within our proposed model. Moreover, we also implement qualitative analyses of the influence of adjacent vehicles on trajectory prediction under different traffic scenes. The visualization results evidently depict that the STS-DGNN model has the ability to comprehend intricate and subtle interactions among multiple vehicles and can produce reasonable predictions concerning these interactions.

Currently, STS-DGNN can only produce deterministic predictions, whereas the motion of the vehicles is intrinsically multimodal. In future work, we would like to add other road users and their interactions, such as bicycles, pedestrians, buses, etc., to our model to improve prediction accuracy. In addition, we will explore methods to enhance the generalizability of the proposed model across different driving environments.

References

[1] O. Sharma, N. C. Sahoo, and N. B. Puhan, "Recent advances in motion and behavior planning techniques for software architecture of autonomous vehicles: A state-of-the-art survey," *Eng. Appl. Artif. Intell.*, vol. 101, May 2021, Art. no. 104211.

[2] Y. Wang, Z. Liu, Z. Zuo, Z. Li, L. Wang, and X. Luo, "Trajectory planning and safety assessment of autonomous vehicles based on motion prediction and model predictive control," *IEEE Trans. Veh. Technol.*, vol. 68, no. 9, pp. 8546–8556, Sep. 2019.

[3] M. Schreier, V. Willert, and J. Adamy, "An integrated approach to maneuver-based trajectory prediction and criticality assessment in arbitrary road environments," *IEEE Trans. Intell. Transp. Syst.*, vol. 17, no. 10, pp. 2751–2766, Oct. 2016.

[4] 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.

[5] H. Gao, Y. Qin, C. Hu, Y. Liu, and K. Li, "An interacting multiple model for trajectory prediction of intelligent vehicles in typical road traffic scenario," *IEEE Trans. Neural Netw. Learn. Syst.*, early access, Dec. 31, 2021, doi: 10.1109/TNNLS.2021.3136866.

[6] L. Lin, W. Li, H. Bi, and L. Qin, "Vehicle trajectory prediction using LSTMs with spatial–temporal attention mechanisms," *IEEE Intell. Transp. Syst. Mag.*, vol. 14, no. 2, pp. 197–208, Mar. 2022.

[7] K. Zhang, L. Zhao, C. Dong, L. Wu, and L. Zheng, "AI-TP: Attention-based interaction-aware trajectory prediction for autonomous driving," *IEEE Trans. Intell. Vehicles*, vol. 8, no. 1, pp. 73–83, Jan. 2023.

[8] J. Pan et al., "Lane-attention: Predicting vehicles' moving trajectories by learning their attention over lanes," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Oct. 2020, pp. 7949–7956.

[9] H. Kim, D. Kim, G. Kim, J. Cho, and K. Huh, "Multi-head attention based probabilistic vehicle trajectory prediction," in *Proc. IEEE Intell. Vehicles Symp. (IV)*, Oct. 2020, pp. 1720–1725.

[10] C. Song, Y. Lin, S. Guo, and H. Wan, "Spatial–temporal synchronous graph convolutional networks: A new framework for spatial–temporal network data forecasting," in *Proc. AAAI Conf. Artif. Intell.*, 2020, vol. 34, no. 1, pp. 914–921.

[11] A. Mohamed, K. Qian, M. Elhoseiny, and C. Claudel, "Social-STGCNN: A social spatio-temporal graph convolutional neural network for human trajectory prediction," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 14424–14432.

[12] A. Pareja, "Evolvegcn: Evolving graph convolutional networks for dynamic graphs," in *Proc. AAAI Conf. Artif. Intell.*, 2020, vol. 34, no. 4, pp. 5363–5370.

[13] J. Li, H. Ma, Z. Zhang, J. Li, and M. Tomizuka, "Spatio-temporal graph dual-attention network for multi-agent prediction and tracking," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 8, pp. 10556–10569, Aug. 2022.

[14] X. Chen, H. Zhang, F. Zhao, Y. Hu, C. Tan, and J. Yang, "Intention-aware vehicle trajectory prediction based on spatial–temporal dynamic attention network for Internet of Vehicles," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 10, pp. 19471–19483, Oct. 2022.

[15] L. Ye, Z. Wang, X. Chen, J. Wang, K. Wu, and K. Lu, "GSAN: Graph self-attention network for learning spatial–temporal interaction representation in autonomous driving," *IEEE Internet Things J.*, vol. 9, no. 12, pp. 9190–9204, Jun. 2022.

[16] X. Mo, Y. Xing, and C. Lv, "Graph and recurrent neural network-based vehicle trajectory prediction for highway driving," in *Proc. IEEE Int. Intell. Transp. Syst. Conf. (ITSC)*, Sep. 2021, pp. 1934–1939.

[17] R. Chandra et al., "Forecasting trajectory and behavior of road-agents using spectral clustering in graph-LSTMs," *IEEE Robot. Autom. Lett.*, vol. 5, no. 3, pp. 4882–4890, Jul. 2020.

[18] L. Zhao, Y. Liu, A. Y. Al-Dubai, A. Y. Zomaya, G. Min, and A. Hawbani, "A novel generation-adversarial-network-based vehicle trajectory prediction method for intelligent vehicular networks," *IEEE Internet Things J.*, vol. 8, no. 3, pp. 2066–2077, Feb. 2021.

[19] S. Kim, D. Kum, and J. W. Choi, "RECUP Net: Recursive prediction network for surrounding vehicle trajectory prediction with future trajectory feedback," in *Proc. IEEE 23rd Int. Conf. Intell. Transp. Syst. (ITSC)*, Sep. 2020, pp. 1–6.

[20] Z. Zhong, Y. Luo, and W. Liang, "STGM: Vehicle trajectory prediction based on generative model for spatial–temporal features," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 10, pp. 18785–18793, Oct. 2022.

[21] Y. Xing, C. Lv, and D. Cao, "Personalized vehicle trajectory prediction based on joint time-series modeling for connected vehicles," *IEEE Trans. Veh. Technol.*, vol. 69, no. 2, pp. 1341–1352, Feb. 2020.

[22] 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.

[23] X. Liu, Y. Wang, Z. Zhou, K. Nam, C. Wei, and C. Yin, "Trajectory prediction of preceding target vehicles based on lane crossing and final points generation model considering driving styles," *IEEE Trans. Veh. Technol.*, vol. 70, no. 9, pp. 8720–8730, Sep. 2021.

[24] X. Mo, Z. Huang, Y. Xing, and C. Lv, "Multi-agent trajectory prediction with heterogeneous edge-enhanced graph attention network," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 7, pp. 9554–9567, Jul. 2022.

[25] Y. Sun, W. Zuo, and M. Liu, "See the future: A semantic segmentation network predicting ego-vehicle trajectory with a single monocular camera," *IEEE Robot. Autom. Lett.*, vol. 5, no. 2, pp. 3066–3073, Apr. 2020.

[26] Y. Huang, J. Du, Z. Yang, Z. Zhou, L. Zhang, and H. Chen, "A survey on trajectory-prediction methods for autonomous driving," *IEEE Trans. Intell. Vehicles*, vol. 7, no. 3, pp. 652–674, Sep. 2022.

[27] V. Lefkopoulos, M. Menner, A. Domahidi, and M. N. Zeilinger, "Interaction-aware motion prediction for autonomous driving: A multiple model Kalman filtering scheme," *IEEE Robot. Autom. Lett.*, vol. 6, no. 1, pp. 80–87, Jan. 2021.

[28] 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.

[29] 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. Vehicles Symp. (IV)*, Jun. 2018, pp. 1672–1678.

[30] A. Zyner, S. Worrall, and E. Nebot, "A recurrent neural network solution for predicting driver intention at unsignalized intersections," *IEEE Robot. Autom. Lett.*, vol. 3, no. 3, pp. 1759–1764, Jul. 2018.

[31] 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. 1468–1476.

[32] T. Zhao et al., "Multi-agent tensor fusion for contextual trajectory prediction," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 12126–12134.

[33] C. Fei, X. He, and X. Ji, "Multi-modal vehicle trajectory prediction based on mutual information," *IET Intell. Transp. Syst.*, vol. 14, no. 3, pp. 148–153, Mar. 2020.

Authorized licensed use limited to: UNIVERSITY OF NOTTINGHAM. Downloaded on May 16,2024 at 23:02:00 UTC from IEEE Xplore. Restrictions apply.

[34] J. Yan et al., "Trajectory prediction for intelligent vehicles using spatial-attention mechanism," *IET Intell. Transp. Syst.*, vol. 14, no. 13, pp. 1855–1863, 2020.

[35] K. Messaoud, I. Yahiaoui, A. Verroust-Blondet, and F. Nashashibi, "Attention based vehicle trajectory prediction," *IEEE Trans. Intell. Vehicles*, vol. 6, no. 1, pp. 175–185, Mar. 2021.

[36] K. Messaoud, I. Yahiaoui, A. Verroust-Blondet, and F. Nashashibi, "Non-local social pooling for vehicle trajectory prediction," in *Proc. IEEE Intell. Vehicles Symp. (IV)*, Jun. 2019, pp. 975–980.

[37] Y. Cai et al., "Environment-attention network for vehicle trajectory prediction," *IEEE Trans. Veh. Technol.*, vol. 70, no. 11, pp. 11216–11227, Nov. 2021.

[38] Z. Ding, Z. Yao, and H. Zhao, "RA-GAT: Repulsion and attraction graph attention for trajectory prediction," in *Proc. IEEE Int. Intell. Transp. Syst. Conf. (ITSC)*, Sep. 2021, pp. 734–741.

[39] H. Guo, Q. Meng, D. Cao, H. Chen, J. Liu, and B. Shang, "Vehicle trajectory prediction method coupled with ego vehicle motion trend under dual attention mechanism," *IEEE Trans. Instrum. Meas.*, vol. 71, pp. 1–16, 2022.

[40] Y. Yuan, X. Weng, Y. Ou, and K. Kitani, "AgentFormer: Agent-aware transformers for socio-temporal multi-agent forecasting," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2021, pp. 9813–9823.

[41] J. Yue, D. Manocha, and H. Wang, "Human trajectory prediction via neural social physics," in *Proc. Eur. Conf. Comput. Vis.* Tel Aviv-Yafo, Israel: Springer, Oct. 2022, pp. 376–394.

[42] D. Wang, H. Liu, N. Wang, Y. Wang, H. Wang, and S. McLoone, "SEEM: A sequence entropy energy-based model for pedestrian trajectory all-then-one prediction," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 45, no. 1, pp. 1070–1086, Jan. 2023.

[43] X. Li, X. Ying, and M. C. Chuah, "GRIP: Graph-based interaction-aware trajectory prediction," in *Proc. IEEE Intell. Transp. Syst. Conf. (ITSC)*, Oct. 2019, pp. 3960–3966.

[44] L. Zhao et al., "T-GCN: A temporal graph convolutional network for traffic prediction," *IEEE Trans. Intell. Transp. Syst.*, vol. 21, no. 9, pp. 3848–3858, Sep. 2020.

[45] Y. Wang, S. Zhao, R. Zhang, X. Cheng, and L. Yang, "Multi-vehicle collaborative learning for trajectory prediction with spatio-temporal tensor fusion," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 1, pp. 236–248, Jan. 2022.

[46] Y. Wu et al., "HSTA: A hierarchical spatio-temporal attention model for trajectory prediction," *IEEE Trans. Veh. Technol.*, vol. 70, no. 11, pp. 11295–11307, Nov. 2021.

[47] Z. Sheng, Y. Xu, S. Xue, and D. Li, "Graph-based spatial–temporal convolutional network for vehicle trajectory prediction in autonomous driving," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 10, pp. 17654–17665, Oct. 2022.

[48] Y. Lu, W. Wang, X. Hu, P. Xu, S. Zhou, and M. Cai, "Vehicle trajectory prediction in connected environments via heterogeneous context-aware graph convolutional networks," *IEEE Trans. Intell. Transp. Syst.*, vol. 24, no. 8, pp. 8452–8464, Aug. 2023.

[49] N. Nikhil and B. T. Morris, "Convolutional neural network for trajectory prediction," in *Proc. Eur. Conf. Comput. Vis. (ECCV) Workshops*, 2018, pp. 1–11.

[50] Y. Xu, L. Wang, Y. Wang, and Y. Fu, "Adaptive trajectory prediction via transferable GNN," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2022, pp. 6510–6521.

[51] J. An, W. Liu, Q. Liu, L. Guo, P. Ren, and T. Li, "DGInet: Dynamic graph and interaction-aware convolutional network for vehicle trajectory prediction," *Neural Netw.*, vol. 151, pp. 336–348, Jul. 2022.

[52] V. Katariya, M. Baharani, N. Morris, O. Shoghli, and H. Tabkhi, "DeepTrack: Lightweight deep learning for vehicle trajectory prediction in highways," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 10, pp. 18927–18936, Oct. 2022.

[53] R. Chai, A. Tsourdos, A. Savvaris, S. Chai, Y. Xia, and C. L. P. Chen, "Multiobjective overtaking maneuver planning for autonomous ground vehicles," *IEEE Trans. Cybern.*, vol. 51, no. 8, pp. 4035–4049, Aug. 2021.

[54] J. Sun, Q. Jiang, and C. Lu, "Recursive social behavior graph for trajectory prediction," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 660–669.

[55] Y. Xu, D. Ren, M. Li, Y. Chen, M. Fan, and H. Xia, "Tra2Tra: Trajectory-to-trajectory prediction with a global social spatial–temporal attentive neural network," *IEEE Robot. Autom. Lett.*, vol. 6, no. 2, pp. 1574–1581, Apr. 2021.

[56] L. Shi et al., "SGCN: Sparse graph convolution network for pedestrian trajectory prediction," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2021, pp. 8994–9003.

Feng-Jie Li received the B.S. degree in computer science and technology from the College of Computer Science and Engineering, Guangxi Normal University, Guilin, China, in 2018, and the M.S. degree in computer science and technology from the College of Computer and Electronics Information, Guangxi University, Nanning, China, in 2021. She is currently pursuing the Ph.D. degree with the College of Computer and Data Science, Fuzhou University, Fuzhou, China.

Her current research interests include autonomous driving, machine learning, and deep learning.

Chun-Yang Zhang received the B.S. degree in mathematics from Beijing Normal University, Zhuhai, China, in 2010, and the M.S. degree in mathematics and the Ph.D. degree in computer science from the University of Macau, Macau, in 2012 and 2015, respectively.

He is currently working as an Associate Professor with the College of Computer and Data Science, Fuzhou University, Fuzhou, China. His research interests include machine learning, computer vision, computational intelligence, and network analysis.

C. L. Philip Chen (Fellow, IEEE) received the M.S. degree in electrical and computer engineering from the University of Michigan, Ann Arbor, ML, USA, in 1985, and the Ph.D. degree in electrical and computer engineering from Purdue University, West Lafayette, IN, in 1988.

He is currently the Chair Professor and the Dean of the College of Computer Science and Engineering, South China University of Technology, Guangzhou, China. His current research interests include systems, cybernetics, and computational intelligence.

Dr. Chen is a fellow of American Association for the Advancement of Science (AAAS), International Association for Pattern Recognition (IAPR), Chinese Association of Automation (CAA), and The Hong Kong Institution of Engineers (HKIE); and a member of Academia Europaea (AE), the European Academy of Sciences and Arts (EASA), and the International Academy of Systems and Cybernetics Science (IASCYS). He has received the IEEE Norbert Wiener Award in 2018 for his contribution in systems and cybernetics, and machine learning. He was a recipient of the 2016 Outstanding Electrical and Computer Engineers Award from his alma mater, Purdue University, after his graduation. Being a Program Evaluator of the Accreditation Board of Engineering and Technology Education (ABET), USA, for computer engineering, electrical engineering, and software engineering programs, he successfully architects the University of Macau's Engineering and Computer Science Programs receiving accreditations from Washington/Seoul Accord through HKIE, of which is considered as his utmost contribution in engineering/computer science education for Macau as the former Dean of the Faculty of Science and Technology. He was the Chair of the TC 9.1 Economic and Business Systems of International Federation of Automatic Control from 2015 to 2017. He was the IEEE Systems, Man, and Cybernetics Society President from 2012 to 2013. He is the Vice President of CAA. He was a Highly Cited Researcher in computer science by Clarivate Analytics in 2018.

Authorized licensed use limited to: UNIVERSITY OF NOTTINGHAM. Downloaded on May 16,2024 at 23:02:00 UTC from IEEE Xplore. Restrictions apply.