


GTC-DAN: A graph-temporal convolutional model with dynamic adjacency for vehicle trajectory prediction

Hao CHEN¹ , Xuncheng WU¹, Ruoping ZHANG^{1*}, Wenfeng GUO², Yang CHEN³, Jiejie XU³, Weiwei ZHANG³, and Wangpengfei YU³

¹ School of Mechanical and Automotive Engineering, Shanghai University of Engineering Science, Shanghai, 201620, China

² School of Vehicle and Mobility, Tsinghua University, Beijing, 100084, China

³ Shanghai Smart Vehicle Cooperating Innovation Center Co., Ltd., Shanghai, 201805, China

Abstract. Autonomous driving is currently an issue of heated debate in automotive engineering. Accurate prediction of the future trajectory of self-driving cars can significantly reduce the occurrence of traffic accidents. However, predicting the future trajectories of vehicles is a challenging task since it is influenced by the interaction behaviours of neighbouring vehicles. This paper proposes a framework that allows for parameter sharing and cross-layer independence, based on a dynamic graph convolutional spatiotemporal network, to study the interactions between vehicles and the temporal dynamics in historical trajectories. By extracting dynamic adjacency matrices from different vehicle interaction features, the model can describe dynamic spatiotemporal relationships and facilitate addressing changes in traffic scenarios. Finally, the proposed model is experimentally compared with existing mainstream trajectory prediction methods using the NGSIM dataset. The results demonstrate that our trajectory prediction model achieved excellent performance in terms of model parameters and prediction accuracy. Compared to the four mainstream models, our model improved accuracy by 35.73%. In addition, we also analyze the relationship between model complexity and efficiency.

Keywords: autonomous driving; trajectory prediction; spatiotemporal modelling; dynamic interactions.

1. INTRODUCTION

1.1. Motivation

In recent years, both academia and industry have made significant efforts to develop verifiable and safe autonomous driving systems. Artificial intelligence (AI), in this context, refers to the capability of machines to perform tasks that would normally require human intelligence, such as visual perception, decision-making, and real-time response to dynamic environments. In autonomous driving, AI models are trained to observe and analyze traffic conditions [1], understand the behaviour of other road users, and adjust their speed and path accordingly. These models rely on large-scale data and computational power to simulate human-like decision-making processes.

In urban road scenarios, both human drivers and self-driving cars need to continuously analyze and adapt to the changing traffic environment. There are complex dynamic interactions between vehicles, often characterized by uncertainty and variability. Therefore, effectively modelling both the explicit and implicit spatiotemporal interactions between vehicles is essential for accurately predicting future vehicle movements [2], which is a key challenge that AI helps to address in autonomous driving.

Traditional vehicle trajectory prediction methods typically rely on handcrafted features or simplified assumptions. However, these methods often exhibit limitations in complex dynamic environments. For example, rule-based prediction methods (e.g., [3]) and statistical models (e.g., [4]) fail to fully capture the nonlinear and diverse interactions in traffic, leading to lower prediction accuracy. Recently, the widespread adoption of deep learning techniques has led to new advancements in vehicle trajectory prediction, utilizing methods such as recurrent neural networks (RNNs), attention mechanisms, and graph neural networks (GNNs). However, most existing deep learning-based methods focus primarily on either spatial or temporal dependencies, neglecting the intrinsic coupling between them [5, 6].

To address this gap, we propose a novel graph-temporal convolutional with dynamic adjacency network (GTC-DAN) that effectively captures both spatial and temporal dependencies in vehicle trajectory prediction. Our model integrates graph convolutional networks (GCNs) and temporal convolutional networks (TCNs) to jointly study the spatial interactions between vehicles and the temporal dynamics in historical trajectories [7]. Additionally, we introduce a dynamic adjacency matrix mechanism to adaptively adjust the spatial relationships between nodes, enabling our model to manage the dynamic changes in traffic scenarios.

*e-mail: zhangruoping@qq.com

Manuscript submitted 2024-08-05, revised 2024-10-29, initially accepted for publication 2024-11-01, published in March 2025.

1.2. Problem definition

In this study, the task we need to accomplish is to within a traffic flow of N vehicles, given each vehicle historical trajectory over the previous T seconds, study the historical trajectories of vehicles $V \in \{1, 2, 3, \dots, n\}$ and the interactions that occur between them in space, and predict the ego vehicle future positions Y_i^τ for the next τ seconds, where $\tau \in \{t+1, t+2, \dots, t+\text{pred}\}$ with the true trajectory denoted by Y^t and the predicted trajectory denoted by Y .

1.3. Related work

1.3.1. Based on the physical model method

Traditional prediction methods are based on physical models, with mainstream approaches using Kalman filtering prediction models and Monte Carlo methods. Barth *et al.* [8] achieved high accuracy in short-term vehicle prediction by using Kalman filtering with image data input. Carvalho *et al.* [9] used an interactive multiple-model Kalman filter (IMM-KF) combined with dedicated filters related to specific road directions at intersections. Danielsson *et al.* [10] employed Monte Carlo methods to predict potential hazards in a scene, enabling vehicles to choose more reasonable paths. Sepideh *et al.* [11] utilized Gaussian process regression to learn motion patterns from noisy historical trajectory data collected by static sensors. Yijing Wang [12] used Monte Carlo methods to output probabilistic occupancy grids for prediction targets, provided mapping from probabilistic occupancy to actual scenarios, and then used model predictive control to optimize the reference trajectory.

Physical methods typically consider the kinematic and dynamic constraints of vehicles, such as yaw angle, vehicle type, and acceleration, as well as environmental factors like road surface friction coefficients. Although these methods can achieve short-term motion prediction, they overlook prior and posterior knowledge of the driving scene, such as road structure, traffic rules, and the subjective intentions of drivers.

1.3.2. Based on deep learning methods

In recent years, the widespread application of neural networks has brought significant attention to deep learning-based prediction methods. Based on large datasets, deep learning models can consider both physical models and road environment structures, as well as train interactions between the target and surrounding participants.

As trajectory prediction is a specific type of sequential prediction, temporal prediction networks, represented by recurrent neural networks (RNNs), were initially employed by researchers. Deo *et al.* [13] proposed a manoeuvre-LSTM model that predicts vehicle trajectories by classifying driving manoeuvres, enabling the model to adjust its predictions based on the identified driving intents, such as lane changes and turns. In Alahi *et al.* [14], a social-LSTM model simulates the interaction between vehicles to account for the influence of surrounding vehicles on the trajectory of the vehicle. Deo *et al.* [15] designed a model framework that combines convolutional neural networks (CNNs) with LSTM to capture spatial and temporal features.

Zyner *et al.* [16] collected real vehicle driving data and developed an LSTM network model to validate its predictive capability for single-lane roundabout behaviour in urban settings. Dai *et al.* [17] addressed the issue of low prediction accuracy in dense traffic by proposing a spatiotemporal LSTM prediction model. Xing *et al.* [18] utilized an LSTM encoder-decoder structure, assigning a decoder to each driving style to achieve personalized trajectory predictions. Furthermore, some researchers have explored the use of convolutional neural networks (CNNs) for better capturing spatial features, including interactions among various traffic participants. For instance, Cen *et al.* proposed using CNNs to extract spatiotemporal information from trajectory data, followed by gated recurrent units (GRUs) to extract temporal relationships in the trajectories [19]. Similarly, Semwal *et al.* [20] designed a model based on an LSTM and CNN encoder combination to generate trajectories classified by speed.

In many practical applications of trajectory prediction, numerous features are generated in non-Euclidean spaces, particularly the interactions between different traffic participants. Graph neural networks (GNNs) are adept at extracting structural and feature information from graphs, thereby constructing state-space scene graphs that incorporate these interactions. For instance, Sharma *et al.* [21] proposed a model that combines GNN embeddings with long short-term memory (LSTM) networks. This model utilizes GNNs to capture the spatiotemporal patterns in vehicle trajectory prediction, effectively managing spatiotemporal dynamics. Similarly, Wu *et al.* [22] introduced the graph-based interaction-aware multi-modality trajectory prediction framework. In this framework, vehicle movements are conceptualized as nodes in a time-varying graph, aiming to predict future vehicle trajectories by effectively capturing these interactions.

Cai *et al.* [23] proposed a hierarchical network called EA-Net, which is a trajectory prediction model that dynamically captures the interaction effects between vehicles using GNNs and an edge attention mechanism. Liang *et al.* [24] utilized vector maps to construct lane-to-lane graphs known as LaneGCN. This graph can extract structural features and topological relationships of high-precision maps. They used convolutional networks to capture vehicle historical trajectory information, then combined this information with the graph for joint processing to ultimately obtain the predicted trajectory. Abdelraouf *et al.* [25] designed a model framework that combines graph convolutional networks (GCNs) and long short-term memory (LSTM) networks. This model enhances vehicle trajectory prediction by incorporating personalized driving patterns, effectively capturing the spatiotemporal interactions between the target vehicle and surrounding traffic.

1.4. Contribution

The main contributions of this paper include:

1. Proposing a novel graph-temporal convolutional with dynamic adjacency network (GTC-DAN) that captures both temporal and spatial dependencies in vehicle trajectory prediction tasks. This model combines graph convolutional networks (GCNs) and temporal convolutional networks (TCNs), effectively learning features from both sequential and structured data.

- Introducing a dynamic adjacency matrix mechanism that dynamically adjusts the spatial topological relationships between nodes. This effectively captures the dynamic changes in node relationships in spatiotemporal data, enhancing the model applicability to complex scenarios.
- Employing an attention-enhanced GRU structure in the decoder, which can automatically extract more critical information for the prediction target, effectively capturing long-term dependencies and generating accurate future trajectory prediction sequences.

2. OVERALL STRUCTURE OF THE MODEL

In real traffic scenarios, there are complex interactions between traffic participants such as pedestrians and vehicles. The future movement trajectory of a vehicle depends not only on its historical trajectory (i.e., the influence of temporal factors) but also on the trajectories of surrounding vehicles (i.e., the influence of spatial factors). Therefore, a vehicle trajectory prediction model must consider dependencies in both temporal and spatial dimensions. Our proposed GTC-DAN model, as shown in Fig. 1, aims to effectively capture spatiotemporal dependencies in vehicle trajectory prediction tasks. This model integrates graph convolutional networks (GCN) [26] and temporal convolutional networks (TCN) [27] and introduces a novel dynamic adjacency matrix mechanism to dynamically adapt the spatial topological relationships between nodes. The encoder part of the model employs a spatiotemporal convolutional structure with shared and independent layers, achieving a good balance between feature extraction capability and computational efficiency.

Specifically, the GCN part of the model captures spatial interactions between traffic participants. Through the dynamic adjacency matrix mechanism, the model can adjust the connection weights between nodes in real time, adapting to dynamic changes in complex traffic scenarios. The TCN part focuses on temporal dynamics in historical trajectories, capturing both

short-term and long-term dependencies through convolutional operations. The shared layers of the encoder extract common features, while the independent layers optimize specific features for different vehicles.

Additionally, to further improve prediction accuracy, the decoder part of the model adopts an attention mechanism-enhanced gated recurrent unit (GRU) structure, which can automatically extract critical information for the prediction target, capture long-term dependencies, and generate precise future trajectory prediction sequences.

3. GTC-DAN MODEL

3.1. Model introduction

Each GTC-DAN layer includes shared and independent parts. The shared part shares parameters across layers, which not only reduces the number of model parameters but also extracts stable feature patterns. The independent part has separate parameters for each layer, capturing specific features at various levels, enabling the model to better understand and represent hierarchical details in the data.

3.1.1. Input representation

The input representation. Raw data is often unstructured or semi-structured, making it unsuitable for direct use. Thus, it is necessary to convert the raw data into a format conducive to efficient subsequent computation. Suppose that in the past T_0 time steps, we observed N vehicles in the scene. The information at the n -th layer is represented by an input tensor \mathbf{W}_n of size $(N \times T_0 \times C)$, where $C = 2$ represents the coordinates of the vehicles (x_i^n, y_i^n) . The input hidden state is processed by two modules in the independent part:

$$\mathbf{W}_n^S = f^{n,S}(\mathbf{W}_n), \quad \mathbf{W}_n^T = f^{n,T}(\mathbf{W}_n), \quad (1)$$

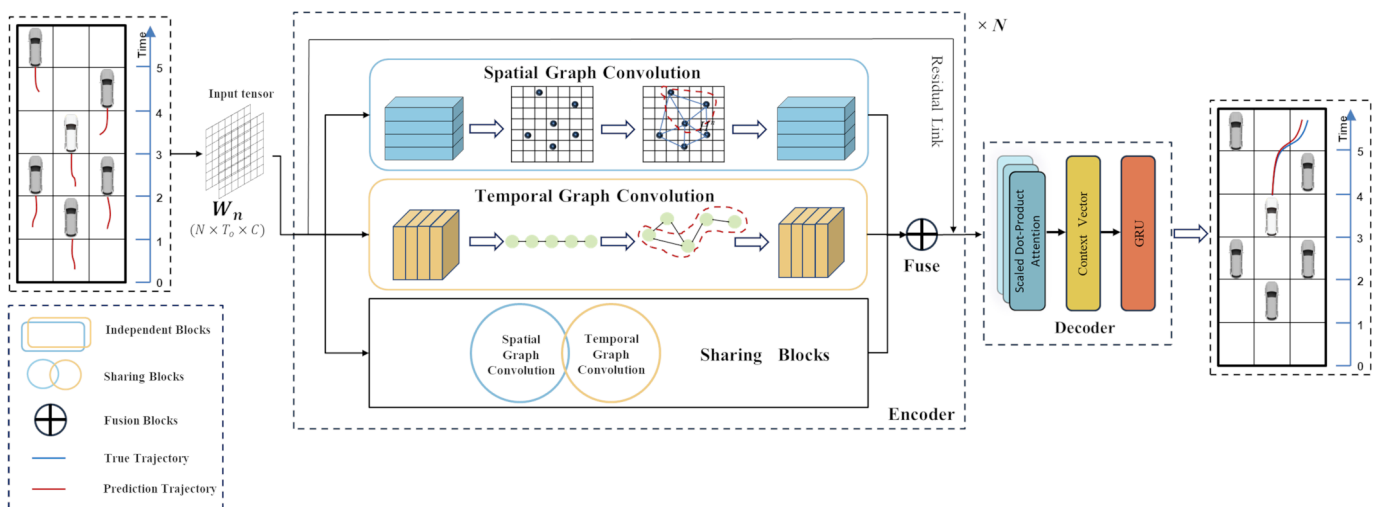


Fig. 1. GTC-DAN Model. The overall structure of the model consists of an encoder, a dynamic adjacency matrix module, a multi-head attention module, and a future trajectory decoder. The encoder is composed of identical layers, each containing shared and independent parts. The spatial convolution module and temporal convolution module process the spatial and temporal hidden state vectors, respectively

where $f^{n,S}$ and $f^{n,T}$ are the spatial and temporal modules of the n -th layer independent part, respectively. The input hidden state also passes through the shared part:

$$\mathbf{W}_n^{S'} = f^S(\mathbf{W}_n), \quad \mathbf{W}_n^{T'} = f^T(\mathbf{W}_n), \quad (2)$$

where f^S and f^T are the spatial and temporal modules of the shared part, respectively, with the same parameters shared across different layers. Finally, the hidden states from the four modules in this layer are fused:

$$\mathbf{W}'_n = \mathbf{P}^{n,f} * [\mathbf{W}_n^S; \mathbf{W}_n^T; \mathbf{W}_n^{S'}; \mathbf{W}_n^{T'}] + \mathbf{b}^{n,f}, \quad (3)$$

where \mathbf{W}'_n is the fused information of the input \mathbf{W}_n of the n -th layer after nonlinear transformation. Then, the residual connection and normalization are applied to obtain the input for the next layer:

$$\mathbf{W}_{n+1} = \text{Batch norm}(\mathbf{W}_n + \text{ReLU}(\mathbf{W}'_n)). \quad (4)$$

3.2. Dynamic adjacency matrix module

In traditional graph convolutional networks (GCNs), the adjacency matrix A is usually static and unchanging [28]. However, in dynamic spatiotemporal data scenarios, the strength of relationships between nodes may change dynamically over time. A fixed static adjacency matrix cannot accurately capture these dynamic relationship changes. To effectively capture the influence of temporal changes on the strength of spatial topological relationships between nodes, we introduce a dynamic adjacency matrix module, as shown in Fig. 2. This module dynamically adjusts the spatial relationship strength between nodes based on time, updating the adjacency matrix to better capture the dynamic evolution characteristics in spatiotemporal data, as shown.

The construction process of the dynamic adjacency matrix $A_{\phi(t)} \in \mathbb{R}^{N \times N}$ is as follows:

$$A_{\phi(t),ij} = \frac{S_{ij}}{\sum_k S_{ik}}. \quad (5)$$

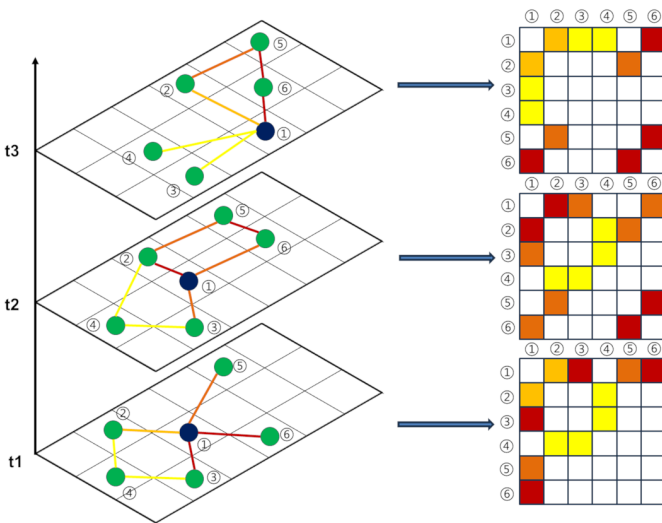


Fig. 2. Dynamic adjacency matrix construction for lane graphs

First, the input feature matrix \mathbf{W}_n is mapped to a new feature space through a linear transformation, resulting in a new feature matrix \mathbf{W}'_n . Based on the new feature matrix \mathbf{W}'_n , the similarity matrix S is calculated. Each element S_{ij} of the similarity matrix is computed using the following formula:

$$S_{ij} = \exp\left(-\frac{\|\mathbf{W}'_n(i,:) - \mathbf{W}'_n(j,:)\|^2}{2\sigma^2}\right), \quad (6)$$

where $\|\mathbf{W}'_n(i,:) - \mathbf{W}'_n(j,:)\|^2$ represents the Euclidean distance between the feature vectors of the i -th and j -th nodes, and σ is a scale parameter that controls the similarity calculation.

The similarity matrix S captures the relationship strengths between nodes based on their features. By normalizing this matrix, we obtain the dynamic adjacency matrix $A_{\phi(t)}$, which reflects the adjusted spatial relationships between nodes over time:

$$A_{\phi(t),ij} = \frac{S_{ij}}{\sum_k S_{ik}}. \quad (7)$$

Normalization ensures that the sum of the similarities for each node is 1, making $A_{\phi(t)}$ a proper adjacency matrix for the graph convolutional operations. This dynamic adjustment allows the model to better capture the evolving spatial relationships in the traffic scene, thereby improving the accuracy of trajectory predictions.

By dynamically updating the adjacency matrix, our model can adapt to the changing interactions between vehicles, reflecting the real-time evolution of traffic scenarios. This enhances the model ability to predict future trajectories accurately by incorporating both temporal and spatial dependencies effectively.

3.3. Trajectory encoder

3.3.1. Spatial convolution module

To better represent the dynamic spatial relationships between vehicles, we first construct a spatial state scene graph at time t to describe the current traffic scenario $G = [V_t E_t]$:

$$V_t = \text{big}\{v_t^i \mid \forall i \in \{1, \dots, N\}\}.$$

This represents the set of all vehicles at time t where v_t^m is the feature vector of the m -th vehicle at time step t , including its position, velocity, and heading angle. E_t represents the set of all edges: $E_t = \{e_t^{ij} \mid \forall ij \in \{1, \dots, N\}\}$. This represents the potential relationship or spatial interaction influence of vehicle i on vehicle j .

Traditional graph convolutions, while capable of extracting local spatial information to manage the relationships between nodes in space, do not account for features that evolve. To dynamically model the strength of interactions between two nodes, the spatial convolution in the GTC-DAN model replaces the normal adjacency matrix with a dynamic adjacency matrix, thereby more accurately capturing and reflecting the complex spatial dependencies in trajectory data.

First, the dynamic adjacency matrix $A_{\phi(t)} \in \mathbb{R}^{N \times N}$ is generated based on the temporal and spatial features of the vehicles

and can perform spatial convolution operations. Below, we introduce the independent and shared parts of the spatial convolution module.

The spatial convolution of each layer has independent parameters to extract spatial features at various levels:

$$f^{n,S}(\mathbf{W}_n)_{:,i} = \sum_{j=1}^N \mathbf{A}_{\phi(t),i,j}^{n,S} \mathbf{W}_n^{:,j}, \quad (8)$$

where $f^{n,S}(\mathbf{W}_n)_{:,i}$ represents the spatial convolution features of the dynamic graph independent part at the n -th layer; $\mathbf{W}_n \in \mathbb{R}^{N \times d}$ is the input feature matrix at the n -th layer, where N is the number of nodes and d is the feature dimension. $\mathbf{A}_{\phi(t)}^{n,S} \in \mathbb{R}^{N \times N}$ is the dynamic adjacency matrix at the n -th layer, representing the relationships between nodes at time step $\phi(t)$ and $\mathbf{W}_n^{:,j}$ represents the feature vector of neighbour node j . The purpose of the spatial convolution module is to aggregate the feature vector information of all neighbour nodes to node i based on $\mathbf{A}_{\phi(t)}$.

In the shared part of the spatial convolution module at the n -th layer, it is defined as follows:

$$f^S(\mathbf{W}_n) = \mathbf{W}_n \times_2 \mathbf{A}_{\phi(t)}^S, \quad (9)$$

where \mathbf{A}^S represents the dynamic adjacency matrix of the shared part, sharing the same parameter set across different layers.

3.3.2. Temporal convolutional module

To capture the temporal dependencies of historical vehicle data, we use a temporal convolution module. This module mainly extracts temporal domain information between vehicles through convolution operations. Similar to the spatial convolution module mentioned earlier, the temporal graph convolution in the independent part of the n -th layer can be defined as:

$$f^{n,T}(\mathbf{W}_n)_i = \sum_{j=1}^D \mathbf{A}_{\phi(t),i,j}^{n,T} \mathbf{W}_n^{:,j}, \quad (10)$$

where $f^{n,T}(\mathbf{W}_n)_{:,i}$ represents the temporal convolution features of the dynamic graph in the independent part of the n -th layer. In the shared part of the n -th layer, the temporal graph convolution module is defined as:

$$f^T(\mathbf{W}_n) = \mathbf{W}_n \times_1 \mathbf{A}_{\phi(t)}^T, \quad (11)$$

where \mathbf{A}^T represents the dynamic adjacency matrix shared by other layers, sharing the same parameter set across different layers. Through this approach, the temporal convolution module can dynamically capture the complex temporal dependencies between vehicles, thereby improving the temporal correlation and accuracy of trajectory predictions.

3.4. Future trajectory joint decoder

After extracting the spatiotemporal features of trajectory data, the vehicle trajectory prediction task can be transformed into a

typical sequence generation problem. Given the computational efficiency and superior performance of the gated recurrent unit (GRU) in sequence modelling tasks, this paper adopts an attention mechanism-based GRU [29] decoder structure to mine critical information from the extracted feature data and generate future trajectory predictions.

Specifically, for the trajectory prediction sequence of the i -th vehicle $\mathbf{Y}_i = y_i^1 y_i^2 \dots y_i^{T'}$, where T' represents the prediction length, and $y_i^t \in \mathbb{R}^{C'}$ is a C' dimensional vector that typically includes features such as position coordinates and velocity, the hidden state \mathbf{h}_i^t of the decoder at time step t is updated by the following formula:

$$\mathbf{h}_i^t = \text{DecoderGRU}(\mathbf{y}_i^{t-1}, \mathbf{h}_i^{t-1}, \mathbf{c}_i), \quad (12)$$

where DecoderGRU denotes the GRU unit of the decoder, \mathbf{y}_i^{t-1} is the predicted output from the previous time step, \mathbf{h}_i^{t-1} is the previous hidden state, and \mathbf{c}_i is the context vector obtained from the encoder, which remains constant for all time steps. The context vector \mathbf{c}_i is calculated through a multi-head attention mechanism from the hidden states of the last layer of the encoder $\mathbf{H}_i^{\text{enc}} \in \mathbb{R}^{N \times d}$, where N is the number of encoder layers and d is the hidden state dimension:

$$\mathbf{c}_i = \text{MultiHeadAttn}(\mathbf{H}_i^{\text{enc}}, \mathbf{H}_i^{\text{enc}}, \mathbf{H}_i^{\text{enc}}). \quad (13)$$

MultiHeadAttn denotes the multi-head attention mechanism [23]. By computing attention coefficients, the model can dynamically adjust its focus on different historical information, thereby better capturing complex spatiotemporal dependencies. As shown in Fig. 3, the weight a_{ij} represents the attention coefficient of the decoder at the j -th position of the encoder:

$$\alpha_{ij} = \frac{\exp(e_{ij})}{\sum_{k=1}^L \exp(e_{ik})}, \quad (14)$$

$$e_{ij} = \mathbf{v}^T \tanh(\mathbf{W}_1 \mathbf{h}_i^t + \mathbf{W}_2 \mathbf{h}_n^{\text{enc}}),$$

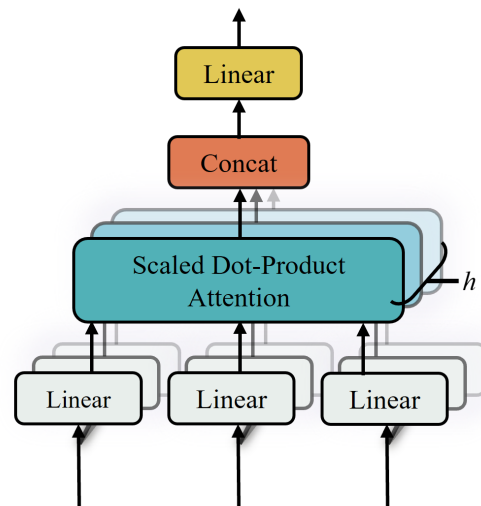


Fig. 3. Multi-head attention mechanism

where $\mathbf{h}_n^{\text{enc}}$ is the hidden state of the n -th layer of the encoder, and \mathbf{v} , \mathbf{W}_1 , \mathbf{W}_2 are learnable parameters. Given the hidden state \mathbf{h}_i^t of the decoder, the future trajectory output is:

$$\hat{\mathbf{y}}_i^t = \text{OutLayer}(\mathbf{h}_i^t, \mathbf{c}_i) = \mathbf{W}_o [\mathbf{h}_i^t; \mathbf{c}_i] + \mathbf{b}_o, \quad (15)$$

where OutLayer is a fully connected output layer, and \mathbf{W}_o and \mathbf{b}_o are the weight and bias, respectively.

4. EXPERIMENTS AND ANALYSIS

4.1. Experimental setup

The experiments were conducted on an Ubuntu 20.04.6 LTS system, with training performed on an Nvidia GTX 3060Ti GPU. All models were implemented using PyTorch. The stochastic gradient descent (SGD) algorithm was adopted as the optimizer. The model was trained for 300 epochs with a batch size of 128. The initial learning rate was set to 0.01, and a decay of 0.002 was applied after 150 epochs.

For evaluation, we used the next-generation simulation (NGSIM) dataset, which contains high-resolution vehicle trajectory data from real-world highway scenarios. This dataset captures various traffic behaviours, such as lane changes and vehicle-following patterns, providing a robust benchmark for trajectory prediction models.

4.2. Experimental results comparison and analysis

4.2.1. Model prediction – accuracy comparison

We conducted comparative experiments between the proposed model and existing trajectory prediction models. The models were evaluated using the root mean square error (RMSE) to measure prediction accuracy, which calculates the square root of the average squared differences between the predicted and actual trajectories. Manoeuvre-LSTM (M-LSTM): M-LSTM is an encoder-decoder-based model where the encoder encodes the trajectories of the target and surrounding vehicles. The encoded vector and manoeuvre code are input into the decoder, which decodes them to generate multimodal trajectory predictions.

Social-LSTM (S-LSTM): This model addresses the problem by connecting LSTMs corresponding to adjacent sequences through a new architecture. It introduces a “social” pooling layer, which can automatically learn typical interactions occurring between trajectories that overlap in time.

EA-Net: This model uses graph neural networks (GNNs) to extract the interaction relationships between vehicles and a recurrent neural network (RNN) decoder to predict the future trajectories of target vehicles.

CS-LSTM: This model combines convolutional neural networks (CNNs) and long short-term memory networks (LSTMs). It extracts features from trajectory data through convolutional layers and then uses LSTM layers to capture dynamic dependencies in the time series.

Table 1 and Fig. 4 list the comparative experiment results of the proposed model and the aforementioned trajectory prediction models.

Table 1

Model prediction accuracy comparison

Datasets	Methods	Prediction horizon				
		1 s	2 s	3 s	4 s	5 s
NGSIM	M-LSTM	0.58	1.26	2.12	3.24	4.56
	S-LSTM	0.84	1.49	2.31	3.32	4.57
	EA-NET	0.42	0.88	1.43	2.15	3.07
	CS-LSTM	0.61	1.27	2.09	3.10	4.37
	GTC-DAN (Our)	0.41	0.75	1.25	1.88	2.51

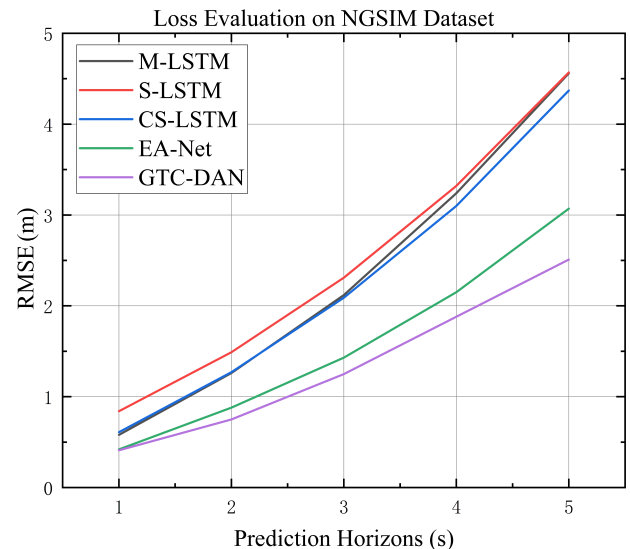


Fig. 4. Model accuracy line chart

Figure 4 visualizes the data from Table 1 which compares model accuracy within a five-second prediction horizon. From both the table and the figure, it is evident that the proposed trajectory prediction model shows significant advantages over the listed models through experimental comparison. Specifically, compared to the EA-Net model, which also uses graph neural networks, our model achieves approximately a 15% improvement in overall prediction performance on the NGSIM dataset. Compared to the CS-LSTM model, which uses convolutional pooling to establish interaction relationships, our model shows a 40% improvement in prediction accuracy. These results indicate that our model has higher accuracy and robustness in handling complex traffic scenarios and vehicle interactions. Furthermore, compared to current existing research, our model shows significant improvements at 3 s, 4 s, and 5 s prediction intervals. This demonstrates that our model has reliable performance in long-term prediction, better capturing the future movement trends of vehicles, especially in scenarios that require consideration of long-term dependencies.

4.2.2. Model time analysis

In the previous model comparison experiments, we unified the input historical duration to 3 seconds. However, the variation in the time domain can also impact prediction accuracy. To in-

investigate the influence of different historical time domains on prediction accuracy, we tested the models using three historical time domains: 1 second, 3 seconds, and 5 seconds. As shown in Fig. 5, for the same prediction duration, the longer the historical trajectory relationship between the target vehicle and surrounding vehicles utilized by this model, the smaller the root mean square error (RMSE). Under the same historical trajectory duration, the longer the prediction time, the larger the error. This is because driving behaviour is uncontrollable, and the longer the prediction time, the greater the error.

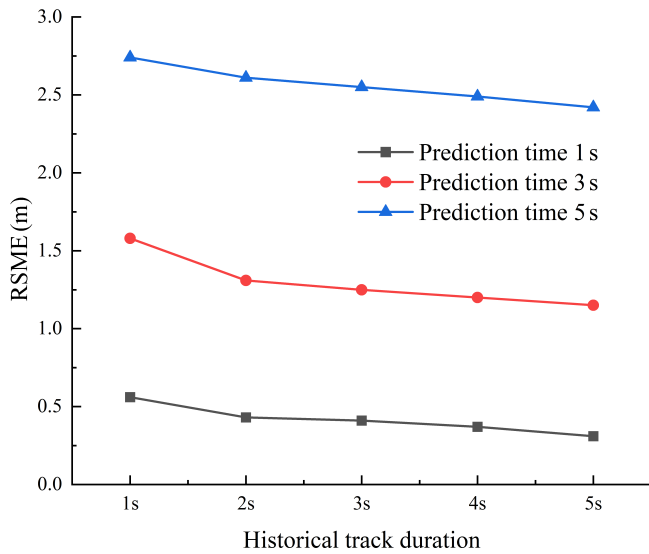


Fig. 5. xperimental results on the influence of different historical time domains on prediction accuracy

4.2.3. Model complexity analysis

In practical applications, although model performance is important, overall model efficiency is even more crucial. To explore the efficiency of this model, we compared the model size of GTC-DAN with some representative baselines. Table 2 shows the total number of parameters for each model. For the NGSIM dataset, Fig. 6 presents a scatter plot of model size versus accuracy. We can observe that GTC-DAN strikes a balance between spatial complexity and performance.

Table 2

Comparison of parameter count, training time, and testing time across different models

Model	M-LSTM	CS-LSTM	S-LSTM	EA-NET	GTC-DAN (Our)
Total parameters (byte)	67 342	97 821	159 628	185 264	128 456
Training time (s)	26	52	72	71	56
TestF time (s)	22	48	70	67	53

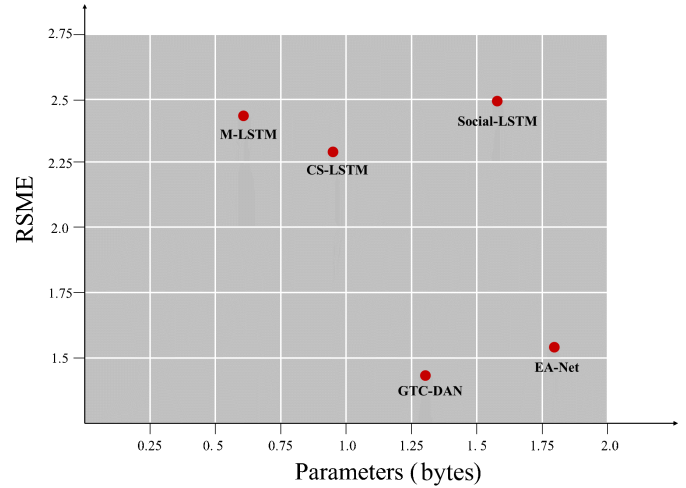


Fig. 6. Total parameters of model vs RMSE

4.2.4. Model ablation experiment

To further validate the effectiveness of each module in the proposed model, we conducted an ablation study on the NGSIM dataset. We compared the performance by separately removing the independent module without utilizing all layers of the encoder, the shared module without utilizing all layers of the encoder, replacing it with a fixed adjacency matrix, and removing the multi-head attention mechanism module.

The results are shown in Table 3 and Fig. 7.

Table 3

Model ablation study results

Method	RMSE				
	$T = 1$ s	$T = 2$ s	$T = 3$ s	$T = 4$ s	$T = 5$ s
No sharing blocks	0.63	1.16	1.53	1.95	2.54
No independent blocks	0.59	1.12	1.49	1.83	2.32
Fixed adjacency matrix	0.68	1.15	1.78	2.12	2.66
No attention blocks	0.79	1.21	1.90	2.21	2.75
GTC-DAN (Our)	0.41	0.75	1.25	1.88	2.51

Figure 7 shows the ablation study comparing different model variants. GTC-DAN is the proposed model in this paper. “No Sharing blocks” is the model without the shared module. “No Independent blocks” is the model without the independent module. “Fixed adjacency matrix” uses a fixed adjacency matrix. “No Attention blocks” is the model where the decoder does not use the attention mechanism.

As shown in the figure, with the increase in prediction time, capturing the spatial and temporal dynamic relationships becomes increasingly important. Therefore, the impact of the dynamic adjacency matrix and independent module gradually increases, while the impact of the shared module and multi-head attention mechanism is relatively small in the short term but also becomes important over a longer time. Compared to the models without independent blocks and shared blocks, GTC-DAN

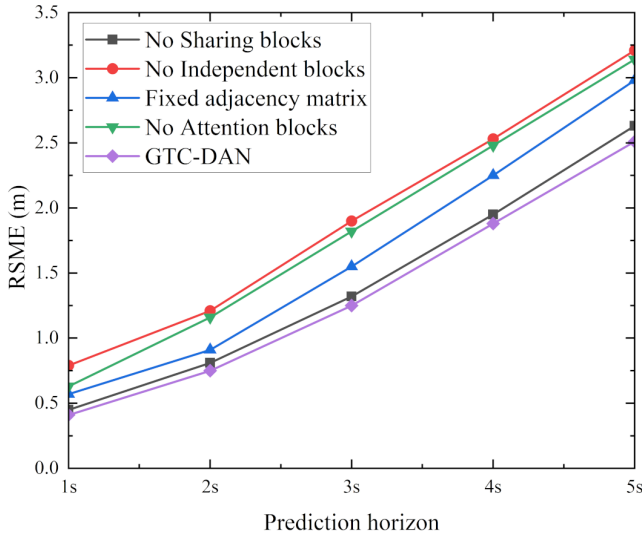


Fig. 7. Comparison of the impact of different modules on model prediction accuracy

demonstrates better performance, indicating the importance of hierarchical parameter learning at different layers. Additionally, the results suggest that the hierarchical parameter mechanism outperforms the shared mechanism.

4.2.5. Attention weight analysis and visualization

Attention weights can reflect the importance of one element to other elements [30]. To further analyze the performance of the proposed model, we visualized the attention distribution of the last layer, as shown in Fig. 8. Each sub-figure represents an attention head, with the horizontal axis representing features (including position, velocity, and acceleration), and the vertical axis representing time steps from the past to the present. Darker

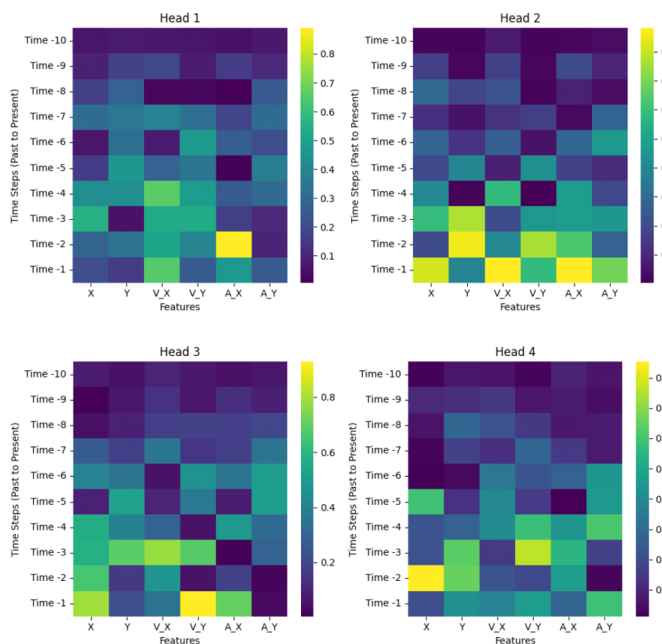


Fig. 8. Weight distribution of multi-head attention mechanism

colours in the figure indicate higher attention weights. It can be observed that different heads exhibit varying degrees of attention to distinctive features. From the overall trend, the closer to the current time step, the higher the attention weight; the farther away from the current time step, the lower the attention weight. The results indicate that future trajectories depend on the driving trajectories at the current moment and a period in the past.

Next, we will highlight and analyze the learned spatial attention weight distribution of the model through visualization. The spatial attention mechanism enables the model to adaptively capture the dynamic spatial dependencies between different vehicle nodes, which is key to accurately predicting future vehicle trajectories.

Figures 9, 10, and 11 illustrate a typical traffic scenario before, during, and after a vehicle lane change. The real-time complex interactive behaviours between vehicles lead to dynamic changes in their spatial relationships. Distinct colours represent different vehicles in the figures.

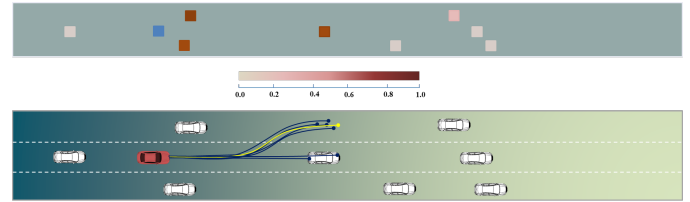


Fig. 9. Before lane change; trajectory prediction of the ego vehicle in pre-lane change driving scenario

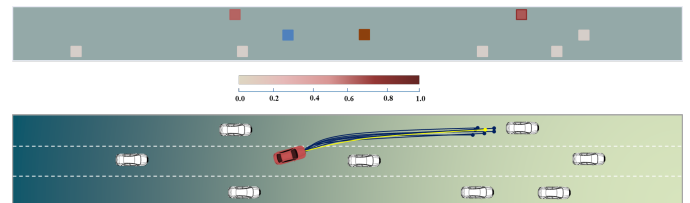


Fig. 10. During lane change; trajectory prediction of the ego vehicle in lane change driving scenario

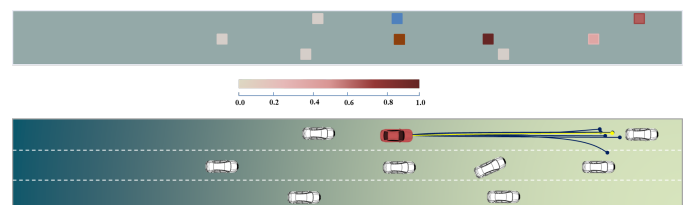


Fig. 11. After lane change; trajectory prediction of the ego vehicle in post-lane change driving scenario

The red vehicle is the ego vehicle. The bottom part of Fig. 9 visualizes the real trajectories of vehicles in this scenario, where we can observe distinct vehicle behaviours such as lane changing and following. In this figure, the yellow curve represents the real trajectories, while the deep blue lines indicate multiple predicted trajectories generated by the proposed model, highlighting the various potential future movements based on

different contextual factors. The small figures at the top of Fig. 9 show the distribution of spatial attention weights assigned by the model to each pair of vehicle nodes at different time steps. Deeper colours indicate larger learned spatial attention weights, representing stronger spatial dependencies between nodes.

From the visualization results, we can see that the spatial attention weight distribution exhibits noticeable dynamic changes. Taking the blue vehicle about to change lanes as an example, we can observe that the model automatically increases the attention weights between it and nearby vehicles in the new lane, as their future trajectories will have significant spatial dependencies. Meanwhile, the attention weights between the blue vehicle and vehicles in its future lane also increase accordingly.

This dynamic adjustment behaviour fully demonstrates our model capability to autonomously learn and capture the dynamic spatial topological relationships between nodes in complex interactive scenarios. By effectively modelling the dynamically changing spatial dependencies, the model can more accurately predict future vehicle motion trajectories.

5. CONCLUSIONS AND FUTURE WORK

In this paper, we proposed a novel graph-temporal convolutional with dynamic adjacency network (GTC-DAN) that effectively captures spatio-temporal dependencies to achieve accurate vehicle trajectory prediction. Our model seamlessly integrates graph convolutional networks (GCNs) and temporal convolutional networks (TCNs) to jointly learn spatial interactions between vehicles and temporal dynamics from historical trajectories. Furthermore, we introduced a dynamic adjacency matrix mechanism that can adaptively adjust the spatial relationships between nodes, enabling our model to manage dynamic changes in complex traffic scenarios.

The encoder part of our model adopts a unique architecture with shared and independent components, striking a good balance between feature extraction capability and computational efficiency. Extensive experiments on real-world datasets show that our proposed GTC-DAN model outperforms existing popular methods for vehicle trajectory prediction. The visualization of spatial attention weights further validates our model ability to dynamically capture the constantly changing spatial dependencies.

The GTC-DAN model paves a new way for spatio-temporal modelling and has tremendous application potential in autonomous driving and related fields. Likely future directions include extending our model to manage more complex scenarios with heterogeneous traffic agents, incorporating high-level semantic information, and exploring more efficient real-time deployment architectures.

ACKNOWLEDGEMENTS

This work was supported by the Postdoctoral Fellowship Program of CPSF under Grant GZB20240351.

REFERENCES

- [1] M. Soori, B. Arezoo, and R. Dastres, "Artificial intelligence, machine learning and deep learning in advanced robotics, a review," *Cogn. Robot.*, vol. 3, pp. 54–70, 2023, doi: [10.1016/j.cogr.2023.04.001](https://doi.org/10.1016/j.cogr.2023.04.001).
- [2] Y. Bezin and B.A. Pålsson, "Multibody simulation benchmark for dynamic vehicle-track interaction in switches and crossings: modelling description and simulation tasks," *Veh. Syst. Dyn.*, vol. 61, no. 3, pp. 644–659, Mar. 2023, doi: [10.1080/00423114.2021.1942079](https://doi.org/10.1080/00423114.2021.1942079).
- [3] C. Tang, J. Chen, and M. Tomizuka, "Adaptive Probabilistic Vehicle Trajectory Prediction Through Physically Feasible Bayesian Recurrent Neural Network," in *2019 International Conference on Robotics and Automation (ICRA)*, Montreal, Canada: IEEE, May 2019, pp. 3846–3852. doi: [10.1109/ICRA.2019.8794130](https://doi.org/10.1109/ICRA.2019.8794130).
- [4] M. Deng, S. Li, X. Jiang, and X. Li, "Vehicle Trajectory Prediction Method Based on 'Current' Statistical Model and Cubature Kalman Filter," *Electronics*, vol. 12, no. 11, p. 2464, May 2023, doi: [10.3390/electronics12112464](https://doi.org/10.3390/electronics12112464).
- [5] M.M. Taye, "Understanding of Machine Learning with Deep Learning: Architectures, Workflow, Applications and Future Directions," *Computers*, vol. 12, no. 5, p. 91, Apr. 2023, doi: [10.3390/computers12050091](https://doi.org/10.3390/computers12050091).
- [6] U.A. Bhatti, H. Tang, G. Wu, S. Marjan, and A. Hussain, "Deep Learning with Graph Convolutional Networks: An Overview and Latest Applications in Computational Intelligence," *Int. J. Intell. Syst.*, vol. 2023, pp. 1–28, Feb. 2023, doi: [10.1155/2023/8342104](https://doi.org/10.1155/2023/8342104).
- [7] H.V. Dudukcu, M. Taskiran, Z.G. Cam Taskiran, and T. Yildirim, "Temporal Convolutional Networks with RNN approach for chaotic time series prediction," *Appl. Soft Comput.*, vol. 133, p. 109945, Jan. 2023, doi: [10.1016/j.asoc.2022.109945](https://doi.org/10.1016/j.asoc.2022.109945).
- [8] A. Barth and U. Franke, "Where will the oncoming vehicle be the next second?," in *2008 IEEE Intelligent Vehicles Symposium*, Eindhoven, Netherlands: IEEE, Jun. 2008, pp. 1068–1073. doi: [10.1109/IVS.2008.4621210](https://doi.org/10.1109/IVS.2008.4621210).
- [9] A. Carvalho, Y. Gao, S. Lefevre, and F. Borrelli, "Stochastic Predictive Control of Autonomous Vehicles in Uncertain Environments" in *12th International Symposium on Advanced Vehicle Control*, 2014.
- [10] S. Danielsson, L. Petersson, and A. Eidehall, "Monte Carlo based Threat Assessment: Analysis and Improvements," in *2007 IEEE Intelligent Vehicles Symposium*, Istanbul, Turkey: IEEE, Jun. 2007, pp. 233–238. doi: [10.1109/IVS.2007.4290120](https://doi.org/10.1109/IVS.2007.4290120).
- [11] S.A. Goli, B.H. Far, and A.O. Fapojuwo, "Vehicle Trajectory Prediction with Gaussian Process Regression in Connected Vehicle Environment★," in *2018 IEEE Intelligent Vehicles Symposium (IV)*, Changshu: IEEE, Jun. 2018, pp. 550–555. doi: [10.1109/IVS.2018.8500614](https://doi.org/10.1109/IVS.2018.8500614).
- [12] 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, doi: [10.1109/TVT.2019.2930684](https://doi.org/10.1109/TVT.2019.2930684).
- [13] N. Deo and M.M. Trivedi, "Multi-Modal Trajectory Prediction of Surrounding Vehicles with Maneuver based LSTMs," in *2018 IEEE Intelligent Vehicles Symposium (IV)*, Jun. 2018, pp. 1179–1184. doi: [10.1109/IVS.2018.8500493](https://doi.org/10.1109/IVS.2018.8500493).

- [14] A. Alahi, K. Goel, V. Ramanathan, A. Robicquet, L. Fei-Fei, and S. Savarese, "Social LSTM: Human Trajectory Prediction in Crowded Spaces," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Las Vegas, NV, USA: IEEE, Jun. 2016, pp. 961–971. doi: [10.1109/CVPR.2016.110](https://doi.org/10.1109/CVPR.2016.110).
- [15] N. Deo and M.M. Trivedi, "Convolutional Social Pooling for Vehicle Trajectory Prediction," in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, Salt Lake City, USA: IEEE, Jun. 2018, pp. 1549–15498. doi: [10.1109/CVPRW.2018.00196](https://doi.org/10.1109/CVPRW.2018.00196).
- [16] 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, doi: [10.1109/LRA.2018.2805314](https://doi.org/10.1109/LRA.2018.2805314).
- [17] S. Dai, L. Li, and Z. Li, "Modeling Vehicle Interactions via Modified LSTM Models for Trajectory Prediction," *IEEE Access*, vol. 7, pp. 38287–38296, 2019, doi: [10.1109/ACCESS.2019.2907000](https://doi.org/10.1109/ACCESS.2019.2907000).
- [18] 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, doi: [10.1109/TVT.2019.2960110](https://doi.org/10.1109/TVT.2019.2960110).
- [19] B. Yu, H. Yin, and Z. Zhu, "Spatio-Temporal Graph Convolutional Networks: A Deep Learning Framework for Traffic Forecasting," in *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence*, Jul. 2018, pp. 3634–3640. doi: [10.24963/ijcai.2018/505](https://doi.org/10.24963/ijcai.2018/505).
- [20] V.B. Semwal, R. Jain, P. Maheshwari, and S. Khatwani, "Gait reference trajectory generation at different walking speeds using LSTM and CNN," *Multimed. Tools Appl.*, vol. 82, no. 21, pp. 33401–33419, Sep. 2023, doi: [10.1007/s11042-023-14733-2](https://doi.org/10.1007/s11042-023-14733-2).
- [21] S. Sharma, A. Singh, G. Sistu, M. Halton, and C. Eising, "Optimizing Ego Vehicle Trajectory Prediction: The Graph Enhancement Approach," *arXiv: arXiv:2312.13104*. [Online]. Available: <http://arxiv.org/abs/2312.13104>
- [22] K. Wu, Y. Zhou, H. Shi, X. Li, and B. Ran, "Graph-Based Interaction-Aware Multimodal 2D Vehicle Trajectory Prediction Using Diffusion Graph Convolutional Networks," *IEEE Trans. Intell. Veh.*, vol. 9, no. 2, pp. 3630–3643, Feb. 2024, doi: [10.1109/TIV.2023.3341071](https://doi.org/10.1109/TIV.2023.3341071).
- [23] Y. Cai *et al.*, "Environment-Attention Network for Vehicle Trajectory Prediction," *IEEE Trans. Veh. Technol.*, vol. 70, no. 11, pp. 11216–11227, Nov. 2021, doi: [10.1109/TVT.2021.3111227](https://doi.org/10.1109/TVT.2021.3111227).
- [24] M. Liang *et al.*, "Learning Lane Graph Representations for Motion Forecasting," *arXiv: arXiv:2007.13732*. [Online]. Available: <http://arxiv.org/abs/2007.13732>
- [25] A. Abdelraouf, R. Gupta, and K. Han, "Interaction-Aware Personalized Vehicle Trajectory Prediction Using Temporal Graph Neural Networks," in *2023 IEEE 26th International Conference on Intelligent Transportation Systems (ITSC)*, Bilbao, Spain: IEEE, Sep. 2023, pp. 2070–2077. doi: [10.1109/ITSC57777.2023.10421928](https://doi.org/10.1109/ITSC57777.2023.10421928).
- [26] L. Shi *et al.*, "SGCN: Sparse Graph Convolution Network for Pedestrian Trajectory Prediction," in *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Nashville, USA: IEEE, Jun. 2021, pp. 8990–8999. doi: [10.1109/CVPR46437.2021.00888](https://doi.org/10.1109/CVPR46437.2021.00888).
- [27] 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, doi: [10.1109/TITS.2022.3155749](https://doi.org/10.1109/TITS.2022.3155749).
- [28] T.N. Kipf and M. Welling, "Semi-Supervised Classification with Graph Convolutional Networks," *arXiv: arXiv:1609.02907*. [Online]. Available: <http://arxiv.org/abs/1609.02907>
- [29] Z. Hao, X. Huang, K. Wang, M. Cui, and Y. Tian, "Attention-Based GRU for Driver Intention Recognition and Vehicle Trajectory Prediction," in *2020 4th CAA International Conference on Vehicular Control and Intelligence (CVCI)*, Hangzhou, China: IEEE, Dec. 2020, pp. 86–91. doi: [10.1109/CVCI51460.2020.9338510](https://doi.org/10.1109/CVCI51460.2020.9338510).
- [30] S. Xing, P. Fan, X. Ma, and Y. Wang, "Research on robot path planning by integrating state-based decision-making A* algorithm and inertial dynamic window approach," *Intell. Serv. Robot.*, vol. 17, pp. 901–914, Jun. 2024, doi: [10.1007/s11370-024-00547-0](https://doi.org/10.1007/s11370-024-00547-0).