Autonomous Intelligent Systems

ORIGINAL ARTICLE Open Access

CDJMP: an adaptive conditional diffusion model for multi-agent joint motion prediction in autonomous driving

Wenwen Zheng[1], Chao Huang[1*], Hao Zhang[1] and Huilin Yin[1]

Abstract

In autonomous driving trajectory prediction, it is important to generate multi-modal trajectories. As a generative method, diffusion model has been increasingly adopted in the field of trajectory prediction. In this paper, we propose CDJMP (Conditional Diffusion model-based Joint Motion Prediction), an adaptive conditional diffusion framework designed for multi-agent trajectory forecasting in autonomous driving. However, when conventional diffusion models generate multi-modal trajectories, particularly for multi-agent joint prediction, they often fail to balance the diversity of generated trajectories and the accuracy of prediction. At the same time, diffusion models also suffer from time-consuming inference. To address these issues, CDJMP utilizes a two-stage framework to generate diverse and highly accurate trajectories. In the first stage, a probabilistic initializer predicts proposal trajectories together with adaptive denoising steps. In the second stage, a group-aware conditional encoder captures dynamic multi-agent interactions and guides the diffusion process to produce coherent multimodal outcomes. Experiments on the INTERACTION and Argoverse datasets demonstrate that CDJMP achieves state-of-the-art performance, reducing minADE and minFDE by up to 9% and 10%, respectively, on the INTERACTION dataset. Ablation experiments demonstrate that CDJMP can effectively reduce inference time while maintaining prediction accuracy. These results highlight the potential of CDJMP as an efficient and accurate framework for real-time multi-agent trajectory prediction in autonomous driving.

Keywords: Motion prediction, Autonomous driving, Multi-modal trajectory prediction, Interaction modeling, Diffusion model

1 Introduction

Multi-agent joint motion prediction serves as a critical bridge between perception and decision-making modules in autonomous driving systems, ensuring safety and reliability (Liu et al. [1]). Despite significant progress, existing methods are difficult to effectively model complex agent interactions and generate reliable multi-modal trajectories (Kamenev et al. [2], Phan-Minh et al. [3], Casas et al. [4], Gao et al. [5], Cui et al. [6], Fang et al. [7], Ye et al. [8]).

Current approaches to modeling multi-agent interactions typically adopt three main technical paradigms: CNN-based methods (Cui et al. [9], Chai et al. [10]), Graph Networks (Gao et al. [5], Liang et al. [11], Gu et al. [12], Varadarajan et al. [13]) and transformer architectures (Li et al. [14], Liu et al. [15], Yuan et al. [16], Ngiam et al. [17], Zhou et al. [18], Nayakanti et al. [19], Shi et al. [20]). While these approaches have achieved remarkable success, they show inherent limitations. CNN-based methods can effectively capture spatial interactions, but fail to adequately

*Correspondence: csehuangchao@tongji.edu.cn
[1]College of Electronic and Information Engineering, Tongji University, Shanghai, 201804, China

model temporal dependencies. Although Transformer-based methods are capable of spatio-temporal joint modeling, such global correlation modeling struggles to capture local condition conflicts.

The diffusion model (Dhariwal and Nichol [21], Song et al. [22], Ho et al. [23]) and conditional diffusion model (Ho and Salimans [24]) provide an effective framework for learning the joint probability distribution of multi-agent trajectories through its iterative denoising process. By leveraging condition-guided learning, this approach captures complex spatio-temporal interaction patterns among multi-agents while maintaining trajectory diversity through its distinctive noise-addition and reduction mechanism. Nevertheless, two major challenges remain unresolved: (1) balancing trajectory diversity and predictive accuracy during multi-agent joint generation, and (2) improving the inference efficiency of diffusion-based predictors for real-time deployment. Existing diffusion-based predictors (Mao et al. [25]) rely on fixed denoising steps, which limit efficiency.

To overcome these limitations, we propose CDJMP, a Conditional Diffusion model based Joint Motion Prediction framework that integrates adaptive diffusion control and group-aware interaction modeling. CDJMP adopts a two-stage design: in the first stage, a probabilistic initializer predicts proposal trajectories and estimates adaptive diffusion steps; in the second stage, a group-aware conditional encoder captures dynamic agent interactions and guides the diffusion process via adaptive jump denoising. This design improves trajectory diversity while maintaining stable prediction accuracy.

The main contributions of this work are summarized as follows:

1. We introduce an adaptive conditional diffusion-based joint motion prediction model that jointly optimizes multi-modal diversity, interaction reasoning, and computational efficiency.
2. We propose uncertainty-guided adaptive diffusion-step module that dynamically adjusts the denoising depth, effectively reducing inference latency without sacrificing prediction accuracy.
3. We propose group-aware conditional encoder, a novel group-based attention mechanism is developed to encode multi-agent interactions, leveraging spatial proximity and dynamic visibility masks for robust social context modeling.
4. Extensive experiments on INTERACTION and Argoverse datasets show that CDJMP outperforms state-of-the-art methods, achieving up to 10% improvement in minFDE on INTERACTION dataset and significant reduction in inference time, highlighting both the accuracy and efficiency of our approach.

In summary, CDJMP provides a principled and efficient solution for joint motion prediction in complex multi-agent environments, paving the way for deeper integration between prediction and planning in autonomous driving systems.

2 Related work

2.1 Multi-agent interaction modeling

Multi-agent interaction plays a crucial role in accurate motion prediction, as future trajectories of road users are inherently interdependent. Early approaches mainly relied on vectorized representations, such as HiVT (Zhou et al. [18]), which decomposes the scene into hierarchical vector sets for local feature extraction and global interaction modeling. Although effective, these vectorized methods often struggle to capture explicit and fine-grained interaction dependencies, limiting their ability to model complex multi-agent coordination (Hagedorn et al. [26]).

To address this limitation, a growing body of work focuses on explicit interaction modeling through graph or attention mechanisms. DenseTNT (Gu et al. [12]) enhances target proposal reasoning by integrating local interaction cues, while HCAGCN-M (Lu et al. [27]) introduces a hierarchical cross-attention graph convolutional network that more effectively represents agent–agent relationships. PRIME (Song et al. [28]) performs probabilistic relational modeling to capture competitive and cooperative multi-agent behaviors. CCL (Mo et al. [29]) adopts curriculum contrastive learning to progressively learn robust interaction-aware features.

Transformer-based approaches such as QCNet (Zhou et al. [30]) and HDGT (Jia et al. [31]) further strengthen global interaction reasoning by leveraging attention-driven dynamic graph structures. Similarly, STG (Wang et al. [32]) and STCG (Liu et al. [33]) incorporate spatio-temporal graph representations to jointly capture spatial and temporal inter-agent dependencies, achieving strong performance in interaction-heavy scenarios. In addition, HyperMTP (Lu et al. [34]) introduces a hyper-network that adapts prediction parameters to scene-specific agent interactions, improving multi-modal prediction under complex traffic dynamics.

Despite these advancements, existing methods generally rely on fixed interaction graphs, static attention patterns, or manually predefined structures, which limits their ability to adaptively model diverse multi-agent relationships. Moreover, most methods lack a mechanism to couple interaction reasoning with trajectory uncertainty, constraining their robustness in highly multi-modal scenarios.

2.2 Motion prediction with diffusion models

Diffusion models such as DDPM (Ho et al. [23]) and DDIM (Song et al. [22]) have recently demonstrated strong capabilities in modeling the inherently multi-modal nature

of future trajectories (Luo [35]). Several trajectory forecasting methods leverage diffusion for generating diverse and realistic motion hypotheses. GOHOME (Gilles et al. [36]) uses scene-level heatmaps as diffusion conditions, lacking fine-grained modeling of dynamic interactions between agents. while ADM (Li et al. [37]) enhances diversity through action space diffusion, but does not explicitly model multi-agent interactions and relies on a fixed denoising step size.

Classical diffusion-based methods typically require numerous iterative denoising steps, leading to high latency that is unsuitable for real-time autonomous driving applications. LED (Mao et al. [25]) accelerates diffusion by learning an expressive initializer, enabling leapfrog-style denoising that reduces iterations. Yet, its refinement process is governed by fixed step sizes, which can result in either insufficient or excessive denoising for different candidate trajectories.

To jointly address the limitations in interaction reasoning and diffusion efficiency, we propose CDJMP, a conditional diffusion framework that explicitly integrates multi-agent interaction modeling. The method operates in two stages. First, a proposal module generates multiple future trajectories and an adaptive diffusion-step predictor determines the optimal number of refinement steps for each proposal—improving denoising efficiency and accuracy. Second, a group-aware conditional encoder explicitly models multi-agent interactions, enabling the diffusion process to incorporate structured relational cues. This design provides both accurate and diverse predictions while maintaining computational efficiency, outperforming prior interaction-centric and diffusion-based approaches on INTERACTION and Argoverse datasets.

3 Method

In this section, we first introduce the overall framework of CDJMP and model the agent state representation. Then, we introduce the main modules of CDJMP. It describes the design made in the group-aware conditional encoder, the adaptive diffusion-step module and present the theoretical justification of adaptive diffusion-step module. Finally, we introduce the loss function and the training flow of the denoising network.

3.1 Overall framework

The overall framework of the CDJMP method is illustrated in Fig. 1. The baseline diffusion model employs a U-Net (Ronneberger et al. [38]) architecture as the noise prediction network. During the initialization stage, instead of starting from random Gaussian noise, the process is initialized using the model's predicted trajectory, to which Gaussian noise is subsequently added to form the noisy input trajectory. The denoising timestep τ is then estimated by the proposed adaptive diffusion-step module. Subsequently, the group-aware conditional encoder computes the correlations between the ego agent and dynamic surrounding agents based on their historical trajectories. These joint interaction features are encoded as conditional inputs to the diffusion model, which guide the denoising process to produce coherent and multi-modal trajectory outcomes.

In the following subsections, we detail the problem modeling for the multi-agent trajectory prediction task, the group-aware conditional encoder, the adaptive diffusion-step module, the theoretical justification of adaptive diffusion-step module, the denoising details of the diffusion model and the optimization objective.

3.2 Agent state representation

The past trajectories of an agent are represented as a sequence of T_{obs} states, the future trajectory of the agent is represented as a sequence of T_{fut} states.

The state of the agent $n \in [N]$ at time step $t \in [T_{\text{obs}}]$ is represented as $\mathbf{x}_t^n = \left[p_t^n, \phi_t^n, v_t^n\right]$, where p_t^n is the 2D absolute coordinates, ϕ_t^n is the heading angle, v_t^n is the 2D velocity.

For agent $n \in [N]$, the state of its dynamic surrounding agent $i \in \hat{N}(i \neq n)$ at time step $t \in [T_{obs}]$ is represented as $\mathbf{x}_t^{n,i} = \left[p_t^{n,i}, \phi_t^{n,i}, v_t^{n,i}\right]$, where $p_t^{n,i}$ is the 2D absolute coordinates of the surrounding agent i, $\phi_t^{n,i}$ is the heading angle of the surrounding agent i, and $v_t^{n,i}$ is the 2D velocity of the surrounding agent i.

For $t \in T_{obs}$ time steps, the ego agent is denoted as $\mathbf{X} = \{\mathbf{x}_{t\in T_{obs}}^n \mid n \in N\}$, global surrounding agent is denoted as $\mathbf{X}_N = \{\mathbf{x}_{t\in T_{obs}}^m \mid m \in N,\ m \neq n\}$, dynamic surrounding agent is denoted as $\mathbf{X}_{\hat{N}} = \{\mathbf{x}_{t\in T_{obs}}^{n,i} \mid n \in N, i \in \hat{N}\}$.

Figure 2 illustrates the concept of dynamic surrounding agents for a target agent. For each agent $n \in [N]$, its dynamic surrounding agents are indexed by $i \in \hat{N}(n)$. The set $\hat{N}(n)$ is dynamically determined according to both the overlapping time window and the Euclidean distance between the target agent and other agents in the scene.

3.3 Group-aware conditional encoder

In this section, we design the conditional input of the diffusion model, which can be summarized by Eq. (1), also called group-aware conditional encoder (group encoder):

$$\mathbf{C} = f_{\text{group}}(\mathbf{X}, \mathbf{X}_{\hat{N}}). \tag{1}$$

First, we define the input, the ego agent, $\mathbf{X} \in \mathbb{R}^{B\times N_e\times T\times d}$. The dynamic surrounding agents are defined as $\mathbf{X}_{\hat{N}} \in \mathbb{R}^{B\times N_e\times N_s\times T\times d}$. Group mask is defined as $\mathbf{M} \in \{0,1\}^{B\times N_e\times N_s\times T}$. The grouping masks can be generated based on spatial proximity, ensuring that only neighbors

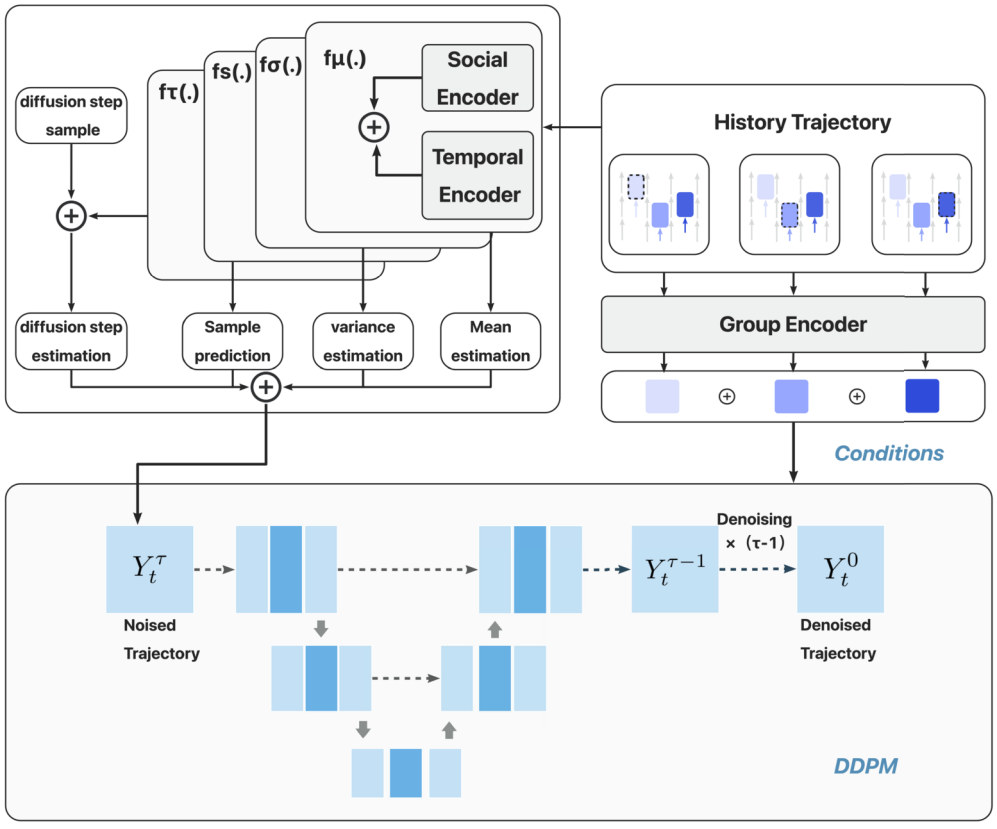


Figure 1 Overall framework of CDJMP

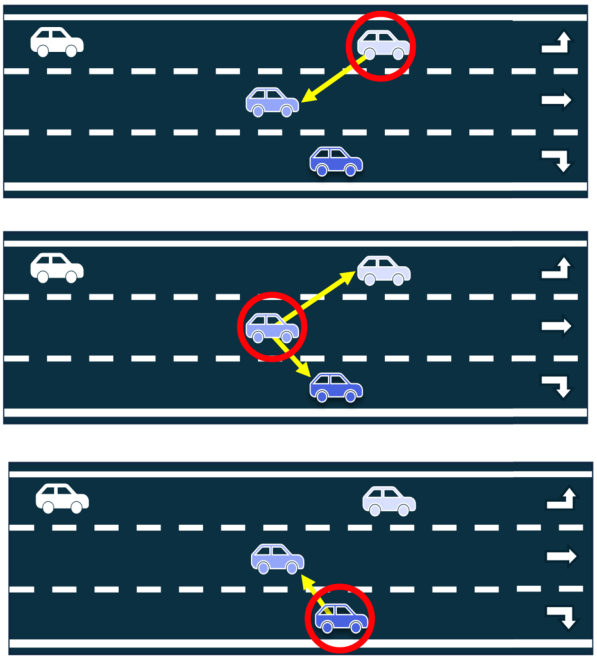


Figure 2 Illustration of dynamic surrounding agent. Each subfigure presents a different target agent (red-circled) and its dynamically selected surrounding agents (yellow arrows). The first, second, and third subfigures illustrate front-only, front-and-rear, and rear-only interaction cases, respectively

within a certain influence radius r_{th} are considered:

$$d_{ij} = \|\mathbf{X}_i - \mathbf{X}_{\hat{N}ij}\|_2,$$
$$m_{ij} = \begin{cases} 1, & \text{if } d_{ij} \le r_{th}, \\ 0, & \text{otherwise.} \end{cases} \tag{2}$$

For each timestamp, $\mathbf{X}_i \in \mathbb{R}^{B\times T\times d}$ is the current position of the i_{th} ego vehicle, and $\mathbf{X}_{\hat{N}ij} \in \mathbb{R}^{B\times T\times d}$ is the current position of the j_{th} neighbor. $m_{ij} \in \{0,1\}^{B\times T}$ represents the mask of the j_{th} dynamic surrounding agent of the i_{th} agent at all timestamps. Group mask $\mathbf{M} = [m_{ij}] \in \mathbb{R}^{N_e\times N_s}$.

The latent representation of the ego agent and dynamic surrounding agents are defined as follows,

$$\mathbf{h}_{\text{ego}} = \text{Repeat}(\mathbf{W}_{\text{ego}}, [B, N_e, 1, 1]), \quad \mathbf{W}_{\text{ego}} \in \mathbb{R}^{1\times d},$$
$$\mathbf{h}_{\text{sv}} = \text{Repeat}(\mathbf{W}_{\text{sv}}, [B, N_s, 1, 1]), \quad \mathbf{W}_{\text{sv}} \in \mathbb{R}^{1\times d_s}, \tag{3}$$

where $\mathbf{W}_{\text{ego}} \in \mathbb{R}^{1\times d}$ and $\mathbf{W}_{\text{sv}} \in \mathbb{R}^{1\times d_s}$ are learnable agent embeddings.

Then, the ego feature is processed in multi-head attention module (MHA) to generate the ego feature latent vector $\mathbf{h}'_{\text{ego}}$, and the surrounding agent feature is processed to generate the surrounding agent feature latent vector $\mathbf{h}'_{\text{sv}}$,

$$\mathbf{h}'_{\text{ego}} = \text{MHA}\left(Q = \mathbf{h}_{\text{ego}}, K = \mathbf{X}, V = \mathbf{X}\right),$$
$$\mathbf{h}'_{\text{sv}} = \text{MHA}\left(Q = \mathbf{h}_{\text{sv}}, K = \mathbf{X}_{\hat{N}}, V = \mathbf{X}_{\hat{N}}\right). \tag{4}$$

The ego agent features $\mathbf{h}'_{\text{ego}}$, the dynamic surrounding agents' features $\mathbf{h}'_{\text{sv}}$, and the group mask $\mathbf{M}$ are utilized to perform grouping cross-attention,

$$\mathbf{h}_{\text{out}} = \text{GCA}\left(\mathbf{h}'_{\text{ego}}, \mathbf{h}'_{\text{sv}}, \mathbf{M}\right). \tag{5}$$

Specifically, attention computation is performed in the Grouped Cross Attention module (GCA), for each ego-surrounding agent pair (i,j):

$$\mathbf{a}_{i,j} = \text{Softmax}\left(\frac{(\mathbf{h}_i^e\mathbf{W}_Q)(\mathbf{h}_j^s\mathbf{W}_K)^\top}{\sqrt{d}}\right),$$
$$\mathbf{h}_i^{\text{out}} = \frac{\sum_{j=1}^{N_s} m_{ij}\cdot\mathbf{a}_{i,j}}{\sum_{j=1}^{N_s} m_{ij}+\epsilon}\cdot(\mathbf{h}_j^s\mathbf{W}_V). \tag{6}$$

Here, $\mathbf{W}_Q$, $\mathbf{W}_K$, $\mathbf{W}_V$ represent learnable parameters, $\mathbf{h}_i^e$ represents the latent variable of the i_{th} ego agent in $\mathbf{h}'_{\text{ego}}$ and $\mathbf{h}_j^s$ the latent variable of the j_{th} dynamic surrounding agent in $\mathbf{h}'_{\text{sv}}$. d represents the dimension of the input vector. Finally, the interaction feature $\mathbf{h}_i^{\text{out}}$ related to all dynamic surrounding agents on the i_{th} agent was obtained.

Finally, we aggregate the interaction features to obtain group-aware conditional embedding $\mathbf{C}$,

$$\mathbf{C} = \text{Stack}(\mathbf{h}_1^{\text{out}}, \ldots, \mathbf{h}_{N_e}^{\text{out}}) \in \mathbb{R}^{B\times N_e\times d}. \tag{7}$$

The function of Group Encoder is to extract the features of ego agent and dynamic surrounding agents through self-attention and control the interaction between agents using group masks. The final output result is input into the next step as a conditional latent vector.

3.4 Adaptive diffusion-step module

Diffusion-based motion predictors typically rely on fixed reverse diffusion steps when generating trajectories. Although simple, this approach does not account for varying interaction complexity among agents. To address this limitation, we propose an adaptive diffusion-step module, which dynamically determines the reverse diffusion depth by modeling uncertainty over the denoising process, enabling efficient and adaptive trajectory refinement.

Inspired by Mao et al. [25], we redesign the model initialization module to generate proposal trajectories and diffusion step estimates in the first stage of CDJMP. The distribution $P(\hat{\mathbf{Y}}_k)$ is learned as the initial input for the diffusion model's decoding process. $P(\hat{\mathbf{Y}}_k)$ is decomposed into three components: mean, variance and sample prediction. Then, the estimation of the diffusion step size is calculated according to the latent features of $P(\hat{\mathbf{Y}}_k)$.

These are modeled by four trainable modules: $f_\mu(\cdot), f_\sigma(\cdot), f_\tau(\cdot)$, and $f_{\hat{\mathbf{S}}}(\cdot)$. $f_\mu(\cdot)$ and $f_\sigma(\cdot)$ are the mean estimation network and the standard deviation estimation network respectively, while $f_\tau(\cdot)$ is the adaptive diffusion step size prediction network, and $f_{\hat{\mathbf{S}}}(\cdot)$ is the sample prediction module. All of them are implemented based on MLP. Here, μ_θ and σ_θ represent the mean deviation and standard deviation of $P(\hat{\mathbf{Y}}_k)$, respectively, while $\hat{\mathbf{s}}_{\theta,k}$ denotes the normalized position of the k^{th} sample. $\hat{\mathbf{Y}}_k$ represents the initial trajectory output of the k^{th} sample.

$$\mu_\theta = f_\mu(\mathbf{X}, \mathbf{X}_N) \in \mathbb{R}^{B\times N_e\times T_{fut}\times 2},$$
$$\sigma_\theta = f_\sigma(\mathbf{X}, \mathbf{X}_N) \in \mathbb{R}^{B\times N_e},$$
$$\tau_\theta = f_\tau(\mathbf{X}, \mathbf{X}_N, \tau_{\text{list}}) \in \mathbb{R}^{B\times N_e},$$
$$\hat{\mathbf{S}}_\theta = [\hat{\mathbf{s}}_{\theta,1}, \ldots, \hat{\mathbf{s}}_{\theta,K}]$$
$$= f_{\hat{\mathbf{S}}}(\mathbf{X}, \mathbf{X}_N, \sigma_\theta) \in \mathbb{R}^{B\times N_e\times T_{fut}\times 2\times K},$$
$$\hat{\mathbf{Y}}_k = \mu_\theta + \sigma_\theta\cdot\hat{\mathbf{s}}_{\theta,k} \in \mathbb{R}^{B\times N_e\times T_{fut}\times 2}. \tag{8}$$

The ego agent $\mathbf{X}$ and the global surrounding agent $\mathbf{X}_N$ are encoded by social embedding to obtain the social context representation $\mathbf{e}_{\text{social}}$. Here, $\mathbf{W}_Q$, $\mathbf{W}_K$, $\mathbf{W}_V$ represents learnable parameters. The standard deviation σ_θ is

then derived using a multilayer perceptron (MLP) decoder $f_{\text{decode}}(\cdot)$:

$$\begin{aligned}\mathbf{e}_{\text{social}} &= \text{Softmax}\left(\frac{(\mathbf{X}\mathbf{W}_Q)(\mathbf{X}_N\mathbf{W}_K)^\top}{\sqrt{d}}\right)\cdot(\mathbf{X}_N\mathbf{W}_V),\\ \sigma_\theta &= f_{\text{decode}}(\mathbf{e}_{\text{social}}).\end{aligned} \tag{9}$$

Temporal embeddings are obtained by feature encoder $F_{\text{CONV1D}}(\cdot)$, which maps the original coordinates into high-dimensional features. These features are then fed into a Gated Recurrent Unit $f_{\text{GRU}}(\cdot)$ to capture temporal dependencies, producing embedding $\mathbf{e}_{\text{temp}}$. Finally, the mean estimate μ_θ is computed through MLP f_{fusion}:

$$\begin{aligned}\mathbf{e}_{\text{temp}} &= f_{\text{GRU}}\left(F_{\text{CONV1D}}(\mathbf{X})\right),\\ \mu_\theta &= f_{\text{fusion}}\left([\mathbf{e}_{\text{social}}, \mathbf{e}_{\text{temp}}]\right).\end{aligned} \tag{10}$$

The standard deviation estimate σ_θ is further encoded to obtain its embedding $\mathbf{e}_\sigma$. This embedding is then fused with the social and temporal embeddings ($\mathbf{e}_{\text{social}}$ and $\mathbf{e}_{\text{temp}}$) through MLP to generate the final predicted output $\hat{\mathbf{S}}_\theta$:

$$\begin{aligned}\mathbf{e}_\sigma &= f_{\text{encode}}(\sigma_\theta),\\ \hat{\mathbf{S}}_\theta &= f_{\text{fusion}}([\mathbf{e}_{\text{social}}, \mathbf{e}_{\text{temp}}, \mathbf{e}_\sigma]).\end{aligned} \tag{11}$$

Let $\tau_{list} = \{\tau_1, \tau_2, \ldots, \tau_m\}$ denote a set of candidate reverse diffusion start steps. τ_{list} is embedded by an MLP encoder $f_{\text{encode}}(\cdot)$, producing embedding e^τ_ω:

$$\mathbf{e}^\tau_w = f_{\text{encode}}(\tau_{\text{list}}). \tag{12}$$

To determine whether further diffusion refinement is needed, we reinterpret the diffusion step τ as a latent variable representing the required denoising depth conditioned on the scene complexity. Specifically, the trajectory proposal $\hat{\mathbf{S}}_\theta$ is encoded into a latent representation through $f_{\text{encode}}(\cdot)$, which captures the structural properties and uncertainty of the predicted motion. This representation is fused with the diffusion-step embedding:

$$\mathbf{p}_\theta = f_{\text{fusion}}\left(\left[f_{\text{encode}}(\hat{\mathbf{S}}_\theta), \mathbf{e}^\tau_w\right]\right). \tag{13}$$

The output $\mathbf{p}_\theta$ is interpreted as the logits of a categorical posterior distribution over the candidate diffusion-step set τ_{list}. The corresponding posterior probability is defined as:

$$q_\phi(\tau|\mathbf{X}, \mathbf{Y}) = \text{Softmax}(\mathbf{p}_\theta), \tag{14}$$

where $q_\phi(\tau|\mathbf{X}, \mathbf{Y})$ denotes the probability of selecting diffusion step τ conditioned on the observed scene $\mathbf{X}$ and target trajectory $\mathbf{Y}$.

During inference, the adaptive diffusion step τ_θ is selected by:

$$\tau_\theta = \tau_{\text{list}}\left[\arg\max(\mathbf{p}_\theta)\right]. \tag{15}$$

Since no ground-truth diffusion-step annotation is available, the adaptive-step module is optimized in an unsupervised variational manner. The training objective is defined as:

$$\begin{aligned}\mathcal{L}_{\text{distance}} = &\sum_{\tau\in\tau_{\text{list}}} q_\phi(\tau|\mathbf{X}, \mathbf{Y})\|\mathbf{Y} - \hat{\mathbf{Y}}^\tau_k\|^2_2\\ &- \alpha H\left(q_\phi(\tau|\mathbf{X}, \mathbf{Y})\right),\end{aligned} \tag{16}$$

where

$$H\left(q_\phi(\tau|\mathbf{X}, \mathbf{Y})\right) = -\sum_\tau q_\phi(\tau|\mathbf{X}, \mathbf{Y})\log q_\phi(\tau|\mathbf{X}, \mathbf{Y}) \tag{17}$$

is the Shannon entropy of the learned diffusion-step posterior, and α controls the entropy regularization strength. $\hat{\mathbf{Y}}^\tau_k$ represents the initialized trajectory output corresponding to τ denoising step.

Finally, the initialized trajectory output $\hat{\mathbf{Y}}_k$ from the model initialization module, along with the predicted denoising step τ_θ, serve as inputs to the joint conditional diffusion model for the subsequent denoising process.

3.5 Theoretical justification of adaptive diffusion-step module

To provide a probabilistic foundation for the adaptive diffusion-step module, we reinterpret the diffusion depth τ as a latent random variable representing the reverse denoising depth conditioned on scene complexity. Instead of using a fixed step size, we model the predictive process as a marginalization over τ, yielding the following joint distribution:

$$p_\theta(\mathbf{Y}|\mathbf{X}) = \sum_{\tau\in\tau_{\text{list}}} p_\theta(\mathbf{Y}|\mathbf{X}, \tau)\, p_\theta(\tau|\mathbf{X}), \tag{18}$$

where $\mathbf{Y}$ denotes the target future trajectory, $\mathbf{X}$ is the observed history and environmental condition, and τ indicates the latent diffusion step determining the denoising depth. Here, $p_\theta(\mathbf{Y}|\mathbf{X}, \tau)$ represents the conditional likelihood of the trajectory given a specific diffusion depth, and $p_\theta(\tau|\mathbf{X})$ denotes the prior distribution of diffusion steps given the input context.

Since the true posterior $p_\theta(\tau|\mathbf{X}, \mathbf{Y})$ is intractable, we introduce a variational approximation $q_\phi(\tau|\mathbf{X}, \mathbf{Y})$ parameterized by the adaptive-step predictor in Eq. (16). Maximizing the marginal likelihood $\log p_\theta(\mathbf{Y}|\mathbf{X})$ corresponds to maxi-

mizing the following evidence lower bound (ELBO):

$$\log p_\theta(\mathbf{Y}|\mathbf{X}) \geq \mathbb{E}_{q_\phi(\tau|\mathbf{X},\mathbf{Y})}\Big[\log p_\theta(\mathbf{Y}|\mathbf{X},\tau)\Big] - \mathrm{KL}\Big(q_\phi(\tau|\mathbf{X},\mathbf{Y}) \,\|\, p_\theta(\tau|\mathbf{X})\Big). \tag{19}$$

The first term in Eq. (19) encourages accurate reconstruction of $\mathbf{Y}$ under the predicted diffusion depth τ, while the second term regularizes the posterior distribution of τ through the Kullback-Leibler divergence. By assuming a uniform prior $p_\theta(\tau|\mathbf{X})$ and assuming $p_\theta(\mathbf{Y}|\mathbf{X},\tau) = \mathcal{N}(\mathbf{Y};\hat{\mathbf{Y}}_k^\tau, I)$, the ELBO in Eq. (19) can be approximated as

$$\mathcal{L}_{\mathrm{ELBO}} = -\sum_{\tau\in\tau_{\mathrm{list}}} q_\phi(\tau|\mathbf{X},\mathbf{Y})\|\mathbf{Y}-\hat{\mathbf{Y}}_k^\tau\|_2^2 + \alpha H\big(q_\phi(\tau|\mathbf{X},\mathbf{Y})\big), \tag{20}$$

$$\mathcal{L}_{\mathrm{distance}} = -\mathcal{L}_{\mathrm{ELBO}}, \tag{21}$$

where $H(\cdot)$ denotes the Shannon entropy, and α is a balancing coefficient. According to Eq. (20), we can determine the loss function Eq. (16), that we need to minimize.

The entropy term $H\big(q_\phi(\tau|\mathbf{X},\mathbf{Y})\big)$ explicitly quantifies the model's uncertainty regarding the appropriate denoising depth.

In summary, Eq. (20) provides a rigorous interpretation in Eq. (16), showing that the adaptive diffusion-step module can be viewed as a variational approximation of a latent-step diffusion process. This formulation grounds the adaptive mechanism in a principled probabilistic framework, balancing accuracy and efficiency during multi-agent trajectory generation. The detailed derivation of Eq. (20) can be found in the Appendix.

3.6 Denoising network

Given the initialized trajectory $\hat{\mathbf{Y}}_k$, the adaptive diffusion-step τ_θ, and the group-aware conditional embedding $\mathbf{C}$, we perform the denoising process to progressively refine the predicted trajectory. Specifically, a conditional noise estimation network f_ϵ is employed to predict the residual noise at each reverse diffusion timestep, enabling iterative refinement from a coarse initialization to a physically consistent and interaction-aware future trajectory.

At diffusion step γ, the conditional denoising process can be formulated as:

$$\begin{aligned}\boldsymbol{\epsilon}_\theta^\gamma &= f_\epsilon(\hat{\mathbf{Y}}_k^{\gamma+1}, \mathbf{C}, \gamma+1),\\ \hat{\mathbf{Y}}_k^\gamma &= \frac{1}{\sqrt{\alpha_\gamma}}\left(\hat{\mathbf{Y}}_k^{\gamma+1} - \frac{1-\alpha_\gamma}{\sqrt{1-\bar{\alpha}_\gamma}}\boldsymbol{\epsilon}_\theta^\gamma\right) + \sqrt{1-\alpha_\gamma}\mathbf{z},\end{aligned} \tag{22}$$

where $\boldsymbol{\epsilon}_\theta^\gamma$ is noise estimation, α_γ and $\bar{\alpha}_\gamma = \prod_{i=1}^{\gamma}\alpha_i$ denote predefined diffusion hyperparameters, and the stochastic perturbation $\mathbf{z}\sim\mathcal{N}(\mathbf{z};0,\mathbf{I})$ ensures sample diversity. The predicted noise $\boldsymbol{\epsilon}_\theta^\gamma$ adjusts the denoising direction conditioned on the group-aware interaction embedding $\mathbf{C}$, enabling the diffusion model to adaptively refine trajectories according to dynamic surrounding agent behaviors.

Algorithm 1 Trajectory Prediction Algorithm

Require: Ego agent history trajectory $\mathbf{X}$, global surrounding agent trajectory $\mathbf{X}_N$, dynamic surrounding agent trajectory $\mathbf{X}_{\hat{N}}$

Ensure: Predicted trajectory $\mathbf{Y}^0$

$\mu_\theta \leftarrow f_\mu(\mathbf{X},\mathbf{X}_N) \in \mathbb{R}^{B\times N_e\times T_{fut}\times 2}$

$\sigma_\theta \leftarrow f_\sigma(\mathbf{X},\mathbf{X}_N) \in \mathbb{R}^{B\times N_e}$

$\hat{\mathbf{S}}_\theta \leftarrow [\hat{\mathbf{s}}_{\theta,1},\ldots,\hat{\mathbf{s}}_{\theta,K}] = f_{\hat{\mathbf{S}}}(\mathbf{X},\mathbf{X}_N,\sigma_\theta) \in \mathbb{R}^{B\times N_e T_{fut}\times 2\times K}$

$\tau_\theta \leftarrow f_\tau(\mathbf{X},\mathbf{X}_N,\tau_{\mathrm{list}}) \in \mathbb{R}^{B\times N_e}$

$\hat{\mathbf{Y}}_k \leftarrow \mu_\theta + \sigma_\theta\cdot\hat{\mathbf{s}}_{\theta,k} \in \mathbb{R}^{B\times N_e\times T_{fut}\times 2}$

for $\gamma = \tau - 1$ **down to** 0 **do**

  $\mathbf{C} \leftarrow f_{\mathrm{group}}(\mathbf{X},\mathbf{X}_{\hat{N}})$

  $\boldsymbol{\epsilon}_\theta^\gamma \leftarrow f_\epsilon(\hat{\mathbf{Y}}_k^{\gamma+1},\mathbf{C},\gamma+1)$

  $\hat{\mathbf{Y}}_k^\gamma \leftarrow \frac{1}{\sqrt{\alpha_\gamma}}\left(\hat{\mathbf{Y}}_k^{\gamma+1} - \frac{1-\alpha_\gamma}{\sqrt{1-\bar{\alpha}_\gamma}}\boldsymbol{\epsilon}_\theta^\gamma\right) + \sqrt{1-\alpha_\gamma}\mathbf{z}$

end for

return $\mathbf{Y}^0$

The full denoising procedure is summarized in Algorithm 1. After initializing the future trajectory using learned mean μ_θ and uncertainty σ_θ, the model iteratively performs conditional noise removal from timestep τ_θ down to 0. At each iteration, a group-aware conditional encoder extracts dynamic surrounding agent context features, which are fused into the noise estimation module to refine the trajectory hypothesis. This iterative refinement produces the final trajectory prediction $\mathbf{Y}^0$.

3.7 Loss function

We take a two-stage approach to training the diffusion model. In the first stage, the denoising network is pretrained with a fixed initial noise distribution. In the second stage, different distributions $P(\hat{\mathbf{Y}})$ are used to fine-tune the conditional encoding and noise prediction network.

The noise estimation loss is used in the first stage to calculate the L_2 loss between the noise and the predicted noise. Here, $\gamma\sim\mathcal{U}\{1,2,\ldots,\Gamma\}$, $\boldsymbol{\epsilon}\sim\mathcal{N}(\boldsymbol{\epsilon};0,\mathbf{I})$, $\mathbf{Y}^{\gamma+1} = \sqrt{\bar{\alpha}_\gamma}\mathbf{Y}^0 + \sqrt{1-\bar{\alpha}_\gamma}\boldsymbol{\epsilon}$.

$$\mathcal{L}_{\mathrm{NE}} = \|\boldsymbol{\epsilon} - f_\epsilon(\mathbf{Y}^{\gamma+1}, f_{\mathrm{group}}(\mathbf{X},\mathbf{X}_{\hat{N}}),\gamma+1)\|_2^2. \tag{23}$$

In the second stage, as demonstrated in Eq. (16), the L2 loss of the predicted trajectory and the real future trajectory is used as the distance loss function, and the influence of the probability of the predicted time step on the overall loss is considered.

In addition, the uncertainty loss function is also used in the second stage, incorporating the uncertainty induced by

the scaled variance σ_θ into the prediction refinement.

$$\mathcal{L}_{\text{uncertainty}} = \frac{\sum_k \|\mathbf{Y} - \hat{\mathbf{Y}}_k\|_2^2}{\sigma_\theta^2} + \log \sigma_\theta^2. \tag{24}$$

Finally, the second stage loss function is obtained, where λ is the hyperparameter that balances the loss functions $\mathcal{L}_{\text{distance}}$ and $\mathcal{L}_{\text{uncertainty}}$:

$$\mathcal{L}_{\text{total}} = \lambda \mathcal{L}_{\text{distance}} + \mathcal{L}_{\text{uncertainty}}. \tag{25}$$

4 Experiments

In this section, we first introduce the experimental setup, including the dataset, evaluation metrics, hyperparameter settings and hardware settings. Second, we conduct a detailed quantitative analysis, including the comparative analysis with other methods and ablation studies. Finally, we conduct a qualitative analysis and show the visualization results of CDJMP.

4.1 Experiments setup

4.1.1 Dataset

We use INTERACTION dataset (Zhan et al. [39]) which is an adversarial, and cooperative motion dataset in an interactive driving scenario. A variety of driving scenarios from multiple countries and regions are included in the dataset. The dataset contains a large number of highly complex behaviors, such as negotiation and aggressive decision making. Through this interactive driving scene data, it helps us to better study the interactive behavior of multi-agent. The time horizon of input data is 1 s with 10 Hz and that of output data is 3 s with 10 Hz. There are a total of 39,600 tracks and traffic participants are all vehicles. This dataset contains 425,192 and 104,627 normalized trajectory sequences for training and test sets, respectively.

We also use Argoverse dataset (Chang et al. [40]), which is jointly released by Argo AI, Carnegie Mellon University, and the Georgia Institute of Technology. The Argoverse dataset includes sensor measurements, vehicle and pedestrian trajectories, and high-definition maps. The trajectory data of vehicles and pedestrians are collected using high-precision GPS and LiDAR sensors, while the map data are generated from GPS and camera observations. Among its subsets, the Argoverse Motion Forecasting dataset offers a curated collection of 324,557 driving scenarios, each lasting 5 seconds and sampled at 10 Hz in a bird's-eye-view representation of object centroids, with the core task revolving around the prediction of future 3-second trajectories based on 2 seconds of historical observational data. This dataset is split into training and test sets, which have 205,942 and 78,143 standardized trajectory sequences respectively.

4.1.2 Evaluation metrics

We evaluate model performance using three metrics: minimum Average Displacement Error (minADE), minimum Final Displacement Error (minFDE), and Missing Rate (MR). These metrics systematically assess the optimal predicted trajectory among $K = 6$ hypothetical trajectories for a single target agent, comparing them against the ground truth. They are commonly used in previous studies.

The minADE metric computes the minimum Euclidean distance between the predicted trajectory and the true trajectory and measures the prediction performance within the Euclidean space.

$$\text{minADE} = \min_{z=1,\dots,Z} \frac{1}{\tau} \sum_{t=1}^{\tau} \|\mathbf{c}_t^z - \mathbf{c}_t^*\|_2, \tag{26}$$

where Z is the total number of modes and $\mathbf{c}_t$ represents the ground truth coordinates at time t.

The minFDE metric calculates the minimum Euclidean distance between the final position of the predicted trajectory and the true trajectory to evaluate the prediction performance in terms of the goal or intention:

$$\text{minFDE} = \min_{z=1,\dots,Z} \|\mathbf{c}_T^z - \mathbf{c}_T^*\|_2. \tag{27}$$

The MR metric calculates the proportion of samples whose minFDE exceeds 2 meters, measuring the performance of coarse-grained target or intention prediction:

$$\text{MR} = \begin{cases} 0, & \exists z \in \{1,\dots,Z\}, \|\mathbf{c}_T^z - \mathbf{c}_T^*\|_2 \le 2, \\ 1, & \text{otherwise.} \end{cases} \tag{28}$$

4.1.3 Experimental settings

CDJMP is implemented by PyTorch. Following the setup in Mao et al. [25], we set the diffusion step size to $\Gamma = 100$ for the diffusion model pre-train. In the CDJMP initializer, the social encoder and the temporal domain encoder are likewise built. The hidden layer size of the encoder for the diffusion step list τ_{list} in the diffusion step estimation module is 32. In the denoising module, group encoder, hidden layer size is 256 and the number of heads is 4.

In our experiment, referring to Limeros et al. [41], for the INTERACTION dataset and Argoverse dataset, the influence radius r_th was set to 20 meters. The temporal overlap window is aligned with the observation horizon (1 s for INTERACTION, 2 s for Argoverse), ensuring that only temporally co-existing agents are considered as dynamic neighbors.

To train the CDJMP diffusion model, we train the denoising module for 100 epochs with an initial learning rate of 1×10^{-2} and decay to 0.5 every 16 epochs. With a frozen denoising module, we then train the CDJMP initializer for 100 epochs with an initial learning rate of 1×10^{-3}, decaying by 0.5 every 8 epochs using Adam optimizer. We

set weight parameter $\lambda = 50$ to emphasize the distance loss. All experiments were conducted on a workstation equipped with an Intel i9-14900K (6 GHz) CPU, 128 GB RAM, and an NVIDIA RTX 4090 GPU.

Inference time was measured as the average forward latency per sample on a single RTX 4090 GPU with a batch size of one, using FP16 precision and excluding data loading.

4.1.4 Fairness of comparison

To ensure fair comparison, all baseline methods are evaluated under same settings across the following dimensions.

Data Splits. All methods use the official training/validation/test splits provided by each dataset without any additional modification.

Observation and Prediction Horizons. On INTERACTION, all methods observe 1 second of history and predict 3 seconds into the future at 10 Hz. On Argoverse, the observation horizon is 2 seconds and the prediction horizon is 3 seconds, also at 10 Hz. All baselines are evaluated under these identical temporal configurations.

Number of Prediction Modes. All methods produce $K = 6$ predicted trajectories, consistent with the standard minADE_6 / minFDE_6 / MR_6 evaluation protocol.

Map Information. The proposed CDJMP does not rely on HD map inputs, and is therefore compared against map-free baselines such as LED and HyperMTP under equivalent conditions.

Inference Settings. All inference experiments are conducted on a single NVIDIA RTX 4090 GPU with a batch size of 1 and FP16 precision. Data loading time is excluded from all reported inference latency measurements to ensure a fair and consistent comparison across methods.

4.2 Comparison with state of the art

Table 1 and Table 2 present comprehensive comparisons between the proposed CDJMP model and several state-of-the-art trajectory prediction approaches on the INTERACTION and Argoverse datasets. The compared baselines include graph-based models (e.g., HCAGCN-M (Lu et al. [27]), HDGT (Jia et al. [31])), query-centric transformers (e.g., QCNet (Zhou et al. [30])), joint modeling frameworks (e.g., HyperMTP (Lu et al. [34])) and diffusion-based methods (e.g., LED (Mao et al. [25]), ADM (Li et al. [37])). For a fair evaluation, all methods are tested under the same prediction horizon and input configuration.

4.2.1 Results on the INTERACTION dataset

CDJMP achieves the best performance across all evaluation metrics, with a minimum Average Displacement Error (minADE) of 0.2695, a minimum Final Displacement Error (minFDE) of **0.8434**, and a Miss Rate (MR) of 0.0695. Compared with the previous best-performing method HyperMTP (Lu et al. [34]), CDJMP reduces minADE by 9% and minFDE by 10%, while decreasing MR by a remarkable 65%. These results indicate that the proposed method captures the overall trajectory shape more precisely and predicts the final destinations with higher accuracy. Moreover, the extremely low MR suggests that CDJMP rarely generates incorrect trajectory hypotheses, demonstrating its capability in handling interactive driving scenarios.

Table 1 Comparative experiments on the INTERACTION dataset

Method	INTERACTION dataset		
	minADE ↓	minFDE ↓	MR ↓
DenseTNT (Gu et al. [12])	0.4284	1.1355	0.2369
HCAGCN-M (Lu et al. [27])	0.4093	1.0827	0.2252
PRIME (Song et al. [28])	0.3875	1.0524	0.2108
CCL (Mo et al. [29])	0.3894	1.0823	0.2209
QCNet (Zhou et al. [30])	0.3527	1.0482	0.2128
HDGT (Jia et al. [31])	0.3047	0.9594	0.2084
STG (Wang et al. [32])	0.3672	1.0629	0.2212
STCG (Liu et al. [33])	0.3723	1.1004	0.2257
HyperMTP (Lu et al. [34])	0.2958	0.9372	0.1958
CDJMP	**0.2695**	**0.8434**	**0.0695**

Table 2 Comparative experiments on the Argoverse dataset

Method	Argoverse dataset		
	minADE ↓	minFDE ↓	MR ↓
GOHOME (Gilles et al. [36])	0.9425	1.4503	**0.1048**
DenseTNT (Gu et al. [12])	0.8817	1.2815	0.1258
HiVT (Zhou et al. [18])	0.7994	1.2321	0.1369
LED (Mao et al. [25])	0.8187	1.2619	0.1874
ADM (Li et al. [37])	0.7916	1.2191	0.1409
CDJMP	**0.6413**	**1.0288**	0.1456

4.2.2 Results on the Argoverse dataset

A similar performance trend can be observed on the Argoverse benchmark. CDJMP attains a minADE of 0.6413, a minFDE of 1.0288, and an MR of 0.1456, outperforming all baseline methods. The results reveal that CDJMP effectively learns the global trajectory structure and accurately predicts the final destinations even under complex urban scenes. However, despite achieving the best overall results, there remains room for improvement in multi-modal consistency. This is mainly because the Argoverse dataset features more intricate map structures, while the current version of CDJMP does not explicitly incorporate map priors. Since MR is more sensitive to model stability and precision in challenging corner cases, the relatively higher MR on Argoverse reflects the model's sensitivity to map-induced uncertainty.

Table 3 Ablation experiments on the INTERACTION and Argoverse datasets

B	M1	M2	INTERACTION dataset			Argoverse dataset		
			ADE(1s)	ADE(2s)	ADE(3s)	ADE(1s)	ADE(2s)	ADE(3s)
✓			0.1186	0.2314	0.4263	0.2313	0.3971	0.5238
✓	✓		0.0917	0.2012	0.3886	0.2151	0.3609	0.4636
✓	✓	✓	0.0549	0.1257	0.2695	0.2082	0.3491	0.4432
B	M1	M2	INTERACTION dataset			Argoverse dataset		
			FDE(1s)	FDE(2s)	FDE(3s)	FDE(1s)	FDE(2s)	FDE(3s)
✓			0.1474	0.3702	1.0816	0.2671	0.3812	1.5560
✓	✓		0.1327	0.3436	1.0359	0.2424	0.3591	1.4495
✓	✓	✓	0.0797	0.2491	0.8434	0.2328	0.3494	1.3200

Table 4 Comparison of inference times and performance metrics

B	M1	M2	τ	Time	minADE	minFDE
✓			$\tau = 30$	$374.78 \pm 8.32ms$	0.4756	1.5836
✓			$\tau = 10$	$94.64 \pm 2.17ms$	0.4488	1.3609
✓			$\tau = 5$	$28.16 \pm 0.91ms$	0.5038	1.4514
✓	✓	✓	$\tau = \tau_\theta$	$29.60 \pm 1.43ms$	0.4432	1.3200

Overall, the superiority of CDJMP across both datasets validates the effectiveness of the proposed diffusion-based joint motion prediction framework. By integrating group-aware conditional encoding and adaptive diffusion-step module, CDJMP achieves robust and accurate multi-modal trajectory forecasting across diverse traffic scenarios.

4.3 Ablation studies

The ablation experiments presented in Table 3 evaluate the contribution of each proposed module in our framework. ADE(1s), ADE(2s), ADE(3s) correspond to cumulative errors at 1, 2, and 3 seconds. The baseline model (B) is a diffusion-based trajectory predictor following the LED formulation (Mao et al. [25]). We progressively integrate the *group-aware conditional encoder* (M1) and the *adaptive diffusion-step module* (M2) to assess their individual and combined effects on trajectory prediction performance.

It can be observed that introducing the group-aware conditional encoder (M1) substantially improves both ADE and FDE across all prediction horizons. On the INTERACTION dataset, ADE(3s) decreases from 0.4263 to 0.3886, corresponding to an approximate 9% improvement, while on the Argoverse dataset, ADE(3s) is reduced by about 11% compared with the baseline. These improvements indicate that the proposed encoder effectively captures inter-agent dependencies and enhances contextual representation learning.

Further incorporating the adaptive diffusion-step module (M2) leads to additional and consistent gains. On the INTERACTION dataset, FDE(3s) decreases by around 22% (from 1.0816 to 0.8434), and on the Argoverse dataset, ADE(3s) further decreases by approximately 15%. This demonstrates that the adaptive diffusion-step mechanism refines the denoising process by dynamically adjusting the diffusion time step according to motion uncertainty, thus mitigating cumulative prediction errors over longer horizons.

Overall, the progressive performance improvements from B $\rightarrow$ B + M1 $\rightarrow$ B + M1 + M2 verify the effectiveness and complementarity of the two modules. The group-aware encoder strengthens interaction modeling, while the adaptive diffusion-step module enhances the temporal consistency of denoising, jointly contributing to more accurate and stable trajectory predictions on both benchmarks.

Table 4 compares the inference efficiency of different model variants on the Argoverse dataset. For each model variant, we randomly sampled 1000 samples from the Argoverse test set and reported the mean and standard deviation.

It can be observed that the inference time of the model without the adaptive diffusion-step module (B + M1) increases significantly as the number of denoising steps τ grows. In contrast, the model equipped with the adaptive diffusion step module (B + M1 + M2) maintains the best prediction accuracy while achieving substantially low inference latency. Additionally, its standard deviation of inference time is small, indicating good inference stability.

These results demonstrate that the proposed adaptive diffusion-step module effectively balances accuracy and computational efficiency by dynamically adjusting the number of denoising steps based on motion uncertainty.

Moreover, the overall ablation results confirm that both the group-aware conditional encoder and the adaptive diffusion-step module contribute to the improved accuracy and efficiency of CDJMP in multi-agent trajectory prediction tasks.

4.4 Inference efficiency

Autonomous driving systems typically require prediction latency within 100 ms. CDJMP achieves a single-

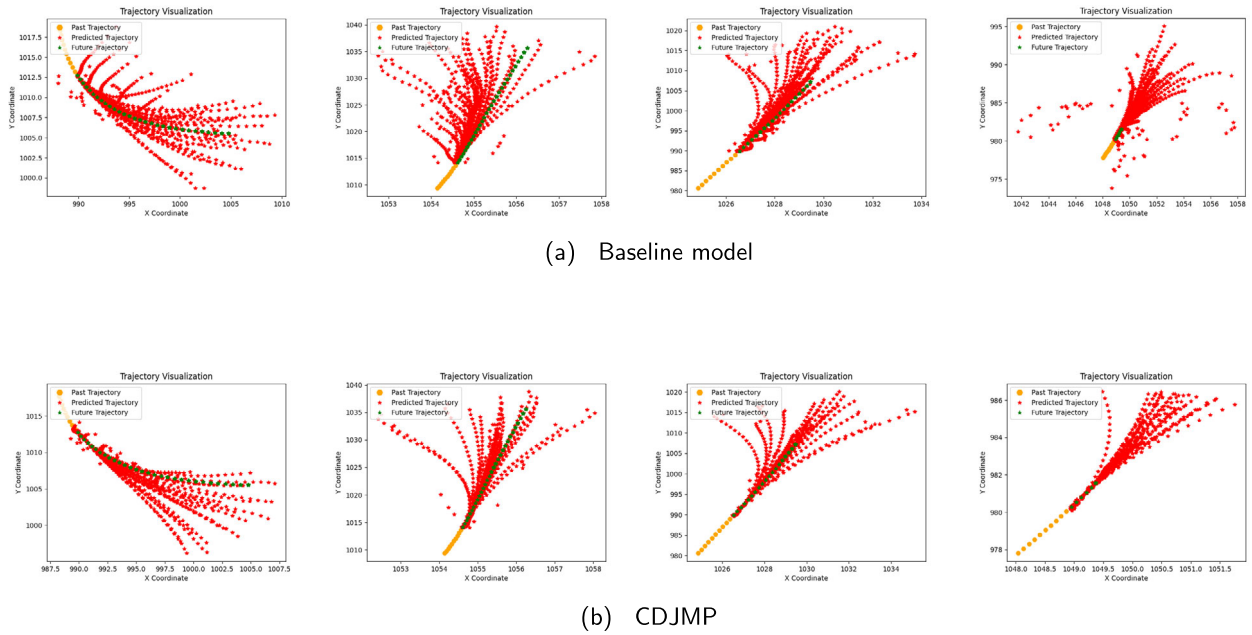


Figure 3 Trajectory prediction results for single agent on the INTERACTION dataset

sample inference time of 29.60 ms, well within this budget, whereas the fixed-step baseline ($\tau = 30$) reaches 374.78 ms, violating real-time constraints. For multi-agent scenarios with 10 agents, our parallel batch inference processes all agents simultaneously via GPU parallelism, yielding approximately 31.8 ms on typical INTERACTION scenes, satisfying the 100 ms requirement. We acknowledge that scalability under extremely dense scenarios with more than 50 agents remains an open challenge for future work.

4.5 Qualitative results

Figure 3 presents the qualitative results of single-agent trajectory prediction on the INTERACTION dataset. Figure 3(a) illustrates the prediction results of the baseline model, while Fig. 3(b) shows those of our proposed method. It can be observed that our method achieves noticeably higher prediction accuracy and generates more coherent trajectory patterns compared with the baseline, demonstrating its superior capability in modeling continuous motion dynamics.

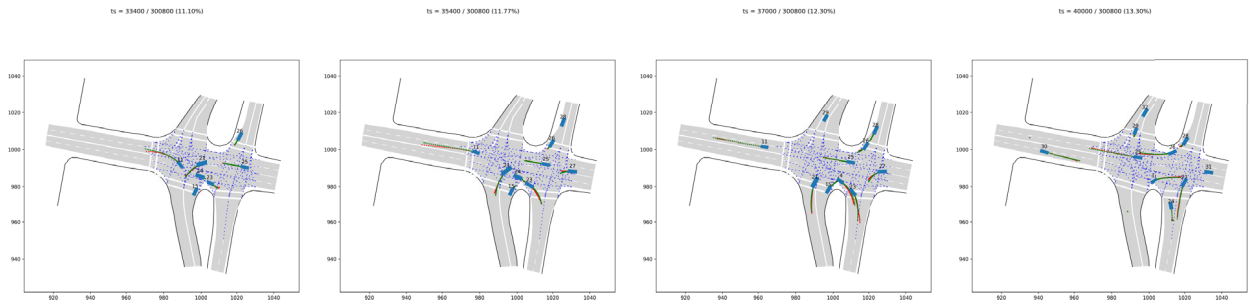
Figure 4 further visualizes the multi-agent trajectory prediction results in three representative driving scenarios of the INTERACTION dataset, namely the *interaction*, *merge*, and *roundabout* scenes. In the visualization, the green lines denote the ground-truth trajectories, while the red lines represent the predicted trajectories selected using the winner-takes-all strategy. The results indicate that our method can accurately capture the diverse motion behaviors of multiple agents and maintain strong robustness across different traffic scenarios.

5 Conclusion

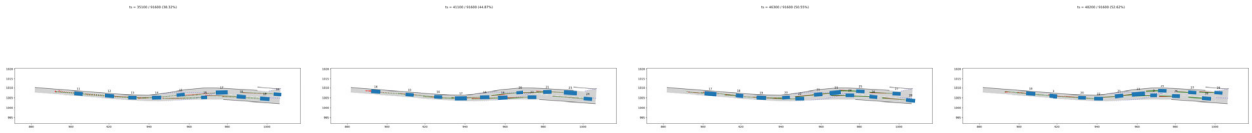
In this paper, we have presented CDJMP, a novel conditional diffusion model based joint motion prediction framework for multi-agent trajectory prediction that explicitly incorporates joint interaction modeling during the denoising process. By integrating group-aware social encoding with a probabilistic trajectory initializer, CDJMP is capable of capturing both individual agent dynamics and complex inter-agent interactions. Extensive experiments on the INTERACTION and Argoverse datasets demonstrate that our method achieves state-of-the-art performance on key metrics, including minADE, minFDE, and MR, significantly outperforming prior approaches such as HyperMTP and LED, especially in highly interactive driving scenarios.

The proposed framework also introduces an adaptive diffusion-step module, which adaptively selects denoising steps based on predicted step probabilities, enhancing the robustness and reliability of trajectory predictions. This method not only enhance the prediction accuracy but also accelerates the denoising efficiency of the diffusion model, which are critical for safe autonomous driving applications.

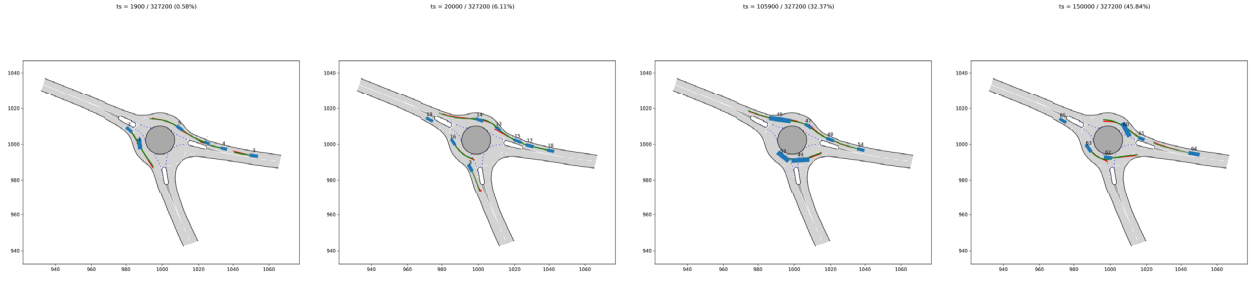
For future work, we plan to explore the extension of CDJMP to heterogeneous traffic scenarios, including mixed



(a) Interaction scenario



(b) Merge scenario



(c) Roundabout scenario

Figure 4 Trajectory prediction results for multi-agent on the INTERACTION dataset. Green lines represent ground-truth trajectories; red lines indicate CDJMP predictions

human-driven and autonomous vehicles, and to investigate online adaptation strategies that can further improve predictive performance in dynamic and previously unseen environments. Moreover, integrating high-definition map information and traffic rules into the diffusion-based framework is expected to enhance scene understanding and trajectory plausibility.

Overall, CDJMP provides a principled and effective solution for probabilistic multi-agent trajectory prediction, advancing the state-of-the-art in intelligent vehicle perception and decision-making.

Appendix

A.1 Variational derivation of the adaptive diffusion-step objective

A.1.1 Latent variable formulation

We reinterpret the diffusion step τ as a latent random variable representing the denoising depth. The marginal likelihood can be written as:

$$\begin{aligned} p_\theta(\mathbf{Y}|\mathbf{X}) &= \sum_{\tau \in \tau_{list}} p_\theta(\mathbf{Y}, \tau|\mathbf{X}) \\ &= \sum_{\tau} p_\theta(\mathbf{Y}|\mathbf{X}, \tau) p_\theta(\tau|\mathbf{X}). \end{aligned} \tag{29}$$

Taking logarithm:

$$\log p_\theta(\mathbf{Y}|\mathbf{X}) = \log \sum_\tau p_\theta(\mathbf{Y}|\mathbf{X},\tau)p_\theta(\tau|\mathbf{X}). \tag{30}$$

A.1.2 Variational approximation
Since the posterior $p_\theta(\tau|\mathbf{X},\mathbf{Y})$ is intractable, we introduce a variational distribution:

$$q_\phi(\tau|\mathbf{X},\mathbf{Y}). \tag{31}$$

Multiply and divide inside the log:

$$\log p_\theta(\mathbf{Y}|\mathbf{X}) = \log \sum_\tau q_\phi(\tau|\mathbf{X},\mathbf{Y}) \frac{p_\theta(\mathbf{Y}|\mathbf{X},\tau)p_\theta(\tau|\mathbf{X})}{q_\phi(\tau|\mathbf{X},\mathbf{Y})}. \tag{32}$$

Rewrite as expectation:

$$= \log \mathbb{E}_{q_\phi(\tau|\mathbf{X},\mathbf{Y})} \left[\frac{p_\theta(\mathbf{Y}|\mathbf{X},\tau)p_\theta(\tau|\mathbf{X})}{q_\phi(\tau|\mathbf{X},\mathbf{Y})} \right]. \tag{33}$$

A.1.3 Jensen inequality
Applying Jensen's inequality:

$$\log p_\theta(\mathbf{Y}|\mathbf{X}) \geq \mathbb{E}_{q_\phi} \left[\log \frac{p_\theta(\mathbf{Y}|\mathbf{X},\tau)p_\theta(\tau|\mathbf{X})}{q_\phi(\tau|\mathbf{X},\mathbf{Y})} \right]. \tag{34}$$

Expanding:

$$\begin{aligned} &= \mathbb{E}_{q_\phi} \left[\log p_\theta(\mathbf{Y}|\mathbf{X},\tau) \right] + \mathbb{E}_{q_\phi} \left[\log p_\theta(\tau|\mathbf{X}) \right] \\ &\quad - \mathbb{E}_{q_\phi} \left[\log q_\phi(\tau|\mathbf{X},\mathbf{Y}) \right]. \end{aligned} \tag{35}$$

Rearranging:

$$= \mathbb{E}_{q_\phi} \left[\log p_\theta(\mathbf{Y}|\mathbf{X},\tau) \right] - KL\left(q_\phi(\tau|\mathbf{X},\mathbf{Y}) \parallel p_\theta(\tau|\mathbf{X}) \right). \tag{36}$$

This gives the standard ELBO:

$$\begin{aligned} \mathcal{L}_{ELBO} &= \mathbb{E}_{q_\phi(\tau|\mathbf{X},\mathbf{Y})}[\log p_\theta(\mathbf{Y}|\mathbf{X},\tau)] \\ &\quad - KL(q_\phi(\tau|\mathbf{X},\mathbf{Y}) \| p_\theta(\tau|\mathbf{X})). \end{aligned} \tag{37}$$

A.1.4 Uniform prior simplification
Assuming a uniform prior:

$$p_\theta(\tau|\mathbf{X}) = \frac{1}{|\tau_{list}|}, \tag{38}$$

the KL term becomes:

$$KL(q||p) = \sum_\tau q(\tau) \log \frac{q(\tau)}{p(\tau)} \tag{39}$$

$$= \sum_\tau q(\tau) \log q(\tau) - \sum_\tau q(\tau) \log p(\tau) \tag{40}$$

$$= -H(q) + \log |\tau_{list}|. \tag{41}$$

Thus:

$$KL(q||p) = -H(q) + \text{const}. \tag{42}$$

Substituting into ELBO:

$$\mathcal{L}_{ELBO} = \mathbb{E}_{q_\phi}[\log p_\theta(\mathbf{Y}|\mathbf{X},\tau)] + H(q_\phi) + \text{const}. \tag{43}$$

A.1.5 Gaussian likelihood assumption
Assume:

$$p_\theta(\mathbf{Y}|\mathbf{X},\tau) = \mathcal{N}(\mathbf{Y}; \hat{\mathbf{Y}}_k^\tau, I), \tag{44}$$

then:

$$\log p_\theta(\mathbf{Y}|\mathbf{X},\tau) = -\|\mathbf{Y} - \hat{\mathbf{Y}}_k^\tau\|_2^2 + \text{const}. \tag{45}$$

Therefore:

$$\mathbb{E}_{q_\phi}[\log p_\theta(\mathbf{Y}|\mathbf{X},\tau)] = -\sum_\tau q_\phi(\tau)\|\mathbf{Y} - \hat{\mathbf{Y}}_k^\tau\|_2^2 + \text{const}. \tag{46}$$

A.1.6 Final objective
Dropping constants, we obtain:

$$\mathcal{L}_{\text{ELBO}} = -\sum_\tau q_\phi(\tau|\mathbf{X},\mathbf{Y})\|\mathbf{Y} - \hat{\mathbf{Y}}_k^\tau\|_2^2 + \alpha H(q_\phi(\tau|\mathbf{X},\mathbf{Y})). \tag{47}$$

This matches the entropy-weighted distance loss used in Eq. (20) in the main paper.

Acknowledgements
The authors are grateful for the efforts of our colleagues at the Lab. Of Intelligent Vehicle and Cooperative control of Multi-agent, Tongji University.

Author contributions
Wenwen Zheng conceived the main idea, conducted the experiments, and wrote the original manuscript. Chao Huang supervised the research and contributed to manuscript revision and improvement. Hao Zhang provided financial support and project administration. Huilin Yin offered technical guidance and support for the experiments. All authors reviewed and approved the final manuscript.

Funding information
This work was supported by the National Natural Science Foundation of China (NSFC) Essential project under Grant No. 62433014; the National Natural Science Foundation of China (NSFC) under Grant Nos. T2541013 and 62373282.

Data availability
Not applicable.

Code availability
https://github.com/zheng-wenwen/CDJMP

Declarations

Competing interests
There are no conflicts of interest for this paper.

Received: 1 April 2026 Revised: 1 June 2026 Accepted: 8 August 2026
Published online: 01 September 2026

References

1. J. Liu, X. Mao, Y. Fang, et al., A survey on deep-learning approaches for vehicle trajectory prediction in autonomous driving, in *2021 IEEE International Conference on Robotics and Biomimetics (ROBIO)* (IEEE, 2021), pp. 978–985
2. A. Kamenev, L. Wang, O.B. Bohan, et al., Predictionnet: real-time joint probabilistic traffic prediction for planning, control, and simulation, in *2022 International Conference on Robotics and Automation (ICRA)* (IEEE, 2022), pp. 8936–8942
3. T. Phan-Minh, E.C. Grigore, F.A. Boulton, et al., Covernet: multimodal behavior prediction using trajectory sets, in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (2020), pp. 14074–14083
4. S. Casas, C. Gulino, R. Liao, et al., Spagnn: spatially-aware graph neural networks for relational behavior forecasting from sensor data, in *2020 IEEE International Conference on Robotics and Automation (ICRA)* (IEEE, 2020), pp. 9491–9497
5. J. Gao, C. Sun, H. Zhao, et al., Vectornet: encoding hd maps and agent dynamics from vectorized representation, in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (2020), pp. 11525–11533
6. A. Cui, S. Casas, K. Wong, et al., GoRela: go relative for viewpoint-invariant motion forecasting. arXiv preprint (2022). arXiv:2211.02545
7. L. Fang, Q. Jiang, J. Shi, et al., Tpnet: trajectory proposal network for motion prediction, in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (2020), pp. 6797–6806
8. M. Ye, J. Xu, X. Xu, et al., Dcms: motion forecasting with dual consistency and multi-pseudo-target supervision. arXiv preprint (2022). arXiv:2204.05859
9. H. Cui, V. Radosavljevic, F.C. Chou, et al., Multimodal trajectory predictions for autonomous driving using deep convolutional networks, in *2019 International Conference on Robotics and Automation (ICRA)* (IEEE, 2019), pp. 2090–2096
10. Y. Chai, B. Sapp, M. Bansal, et al., Multipath: multiple probabilistic anchor trajectory hypotheses for behavior prediction. arXiv preprint (2019). arXiv:1910.05449
11. M. Liang, B. Yang, R. Hu, et al., Learning lane graph representations for motion forecasting, in *European Conference on Computer Vision* (Springer, Berlin, 2020), pp. 541–556
12. J. Gu, C. Sun, H. Zhao, Densetnt: end-to-end trajectory prediction from dense goal sets, in *Proceedings of the IEEE/CVF International Conference on Computer Vision* (2021), pp. 15303–15312
13. B. Varadarajan, A. Hefny, A. Srivastava, et al., Multipath++: efficient information fusion and trajectory aggregation for behavior prediction, in *2022 International Conference on Robotics and Automation (ICRA)* (IEEE, 2022), pp. 7814–7821
14. L.L. Li, B. Yang, M. Liang, et al., End-to-end contextual perception and prediction with interaction transformer, in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (IEEE, 2020), pp. 5784–5791
15. Y. Liu, J. Zhang, L. Fang, et al., Multimodal motion prediction with stacked transformers, in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (2021), pp. 7577–7586
16. Y. Yuan, X. Weng, Y. Ou, et al., Agentformer: agent-aware transformers for socio-temporal multi-agent forecasting, in *Proceedings of the IEEE/CVF International Conference on Computer Vision* (2021), pp. 9813–9823
17. J. Ngiam, B. Caine, V. Vasudevan, et al., Scene transformer: a unified architecture for predicting multiple agent trajectories. arXiv preprint (2021). arXiv:2106.08417
18. Z. Zhou, L. Ye, J. Wang, et al., Hivt: hierarchical vector transformer for multi-agent motion prediction, in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (2022), pp. 8823–8833
19. N. Nayakanti, R. Al-Rfou, A. Zhou, et al., Wayformer: motion forecasting via simple & efficient attention networks. arXiv preprint (2022). arXiv:2207.05844
20. S. Shi, L. Jiang, D. Dai, et al., Motion transformer with global intention localization and local movement refinement. Adv. Neural Inf. Process. Syst. **35**, 6531–6543 (2022)
21. P. Dhariwal, A. Nichol, Diffusion models beat gans on image synthesis. Adv. Neural Inf. Process. Syst. **34**, 8780–8794 (2021)
22. J. Song, C. Meng, S. Ermon, Denoising diffusion implicit models. arXiv preprint (2020). arXiv:2010.02502
23. J. Ho, A. Jain, P. Abbeel, Denoising diffusion probabilistic models. Adv. Neural Inf. Process. Syst. **33**, 6840–6851 (2020)
24. J. Ho, T. Salimans, Classifier-free diffusion guidance. arXiv preprint (2022). arXiv:2207.12598
25. W. Mao, C. Xu, Q. Zhu, et al., Leapfrog diffusion model for stochastic trajectory prediction, in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (2023), pp. 5517–5526
26. S. Hagedorn, M. Hallgarten, M. Stoll, et al., The integration of prediction and planning in deep learning automated driving systems: a review. IEEE Trans. Intell. Veh. **10**, 3626–3643 (2024)
27. Y. Lu, W. Wang, X. Hu, et al., Vehicle trajectory prediction in connected environments via heterogeneous context-aware graph convolutional networks. IEEE Trans. Intell. Transp. Syst. **24**(8), 8452–8464 (2022)
28. H. Song, D. Luan, W. Ding, et al., Learning to predict vehicle trajectories with model-based planning, in *Conference on Robot Learning* (PMLR, 2022), pp. 1035–1045
29. X. Mo, H. Liu, Z. Huang, et al., Map-adaptive multimodal trajectory prediction via intention-aware unimodal trajectory predictors. IEEE Trans. Intell. Transp. Syst. **25**(6), 5651–5663 (2023)
30. Z. Zhou, J. Wang, Y.H. Li, et al., Query-centric trajectory prediction, in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (2023), pp. 17863–17873
31. X. Jia, P. Wu, L. Chen, et al., Hdgt: heterogeneous driving graph transformer for multi-agent trajectory prediction via scene encoding. IEEE Trans. Pattern Anal. Mach. Intell. **45**(11), 13860–13875 (2023)
32. X. Wang, J. Liu, H. Lin, et al., A multi-modal spatial–temporal model for accurate motion forecasting with visual fusion. Inf. Fusion **102**, 102046 (2024)
33. J. Liu, H. Lin, X. Wang, et al., Reliable trajectory prediction in scene fusion based on spatio-temporal structure causal model. Inf. Fusion **107**, 102309 (2024)
34. Y. Lu, W. Wang, R. Bai, et al., Hyper-relational interaction modeling in multi-modal trajectory prediction for intelligent connected vehicles in smart cites. Inf. Fusion **114**, 102682 (2025)
35. C. Luo, Understanding diffusion models: a unified perspective. arXiv preprint (2022). arXiv:2208.11970
36. T. Gilles, S. Sabatini, D. Tsishkou, et al., Gohome: graph-oriented heatmap output for future motion estimation, in *2022 International Conference on Robotics and Automation (ICRA)* (IEEE, 2022), pp. 9107–9114
37. J. Li, T. Shen, Z. Gu, et al., Adm: accelerated diffusion model via estimated priors for robust motion prediction under uncertainties, in *2024 IEEE 27th International Conference on Intelligent Transportation Systems (ITSC)* (IEEE, 2024), pp. 2221–2227
38. O. Ronneberger, P. Fischer, T. Brox, U-net: convolutional networks for biomedical image segmentation, in *International Conference on Medical Image Computing and Computer-Assisted Intervention* (Springer, Cham, 2015), pp. 234–241
39. W. Zhan, L. Sun, D. Wang, et al., Interaction dataset: an international, adversarial and cooperative motion dataset in interactive driving scenarios with semantic maps. arXiv preprint (2019). arXiv:1910.03088
40. M.F. Chang, J. Lambert, P. Sangkloy, et al., Argoverse: 3d tracking and forecasting with rich maps, in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (2019), pp. 8748–8757
41. S.C. Limeros, S. Majchrowska, J. Johnander, et al., Towards explainable motion prediction using heterogeneous graph representations. Transp. Res., Part C, Emerg. Technol. **157**, 104405 (2023)

Publisher's note