ECTraj: Enhanced Consistency Training for Multi-Agent Trajectory Prediction

Alen Mrdovic[1], Qingze (Tony) Liu[1], Danrui Li[1], Mathew Schwartz[1], Kaidong Hu[1], Sejong Yoon[2], Mubbasir Kapadia[1], and Vladimir Pavlovic[1]

[1] Rutgers University, New Brunswick
[2] The College of New Jersey

Abstract. Diffusion models for multi-agent trajectory prediction are limited by iterative denoising, which causes inference latency that hinders their use in time-critical settings like autonomous driving. Fast-sampling variants using DDIM and informed initial noise distribution partially alleviate this issue, but they either fail to achieve true single-step generation or are constrained by the chosen noise distribution. Consistency Models (CMs) offer high-quality one-step generation by mapping noise directly to data, but are difficult to train from scratch. We propose ***EC-Traj***, an enhanced CM pipeline with improved training and conditional generation for trajectory prediction. Our framework extends the student-teacher consistency training scheme: the student produces standard outputs, while the teacher explicitly fuses its predictions with parts of the ground truth to give stronger supervision. We also exploit CMs' direct denoising for top-K multi-shot generation during training. Combining conditional generation with this enhanced consistency objective yields faster inference and improved prediction accuracy, establishing competitive new benchmarks on the large-scale Argoverse 2 dataset.[3]

Keywords: Trajectory prediction · Consistency models

1 Introduction

Trajectory prediction forecasts the short-term future motion of surrounding traffic participants from their observed history, environmental layout, and social interactions. It is vital for safety-critical applications such as robotic motion planning [16, 22, 24] and autonomous driving [17, 19], where predicted motions are required to be physically plausible, socially acceptable, and reflect multiple potential outcomes of participants' intents. Recent work using diffusion models for trajectory prediction [7, 9, 18, 32] has shown strong performance in the multi-modal setting and the flexibility to impose various sampling techniques to meet additional output constraints. However, high computational complexity hinders the application of diffusion models, where real-time inference is crucial during deployment in systems such as autonomous vehicles.

[3] Code: https://github.com/am3338/ECTraj

During inference, diffusion models gradually denoise a standard Gaussian sample toward the desired data point by following the dynamics specified by a stochastic differential equation (SDE), a process that typically demands hundreds of optimization steps. More efficient variants, such as DDIM [25], speed up sampling and reduce the number of required steps; however, they still need more than 10 denoising iterations. Beyond accelerated diffusion formulations, previous work [18, 32] also tried to replace the standard Gaussian with an informed initial distribution. These methods denoise from a lower-variance noise distribution whose means come from trajectory predictions of pre-trained or standalone modules. This warm start reduces the required diffusion steps and promotes a better exploration of multi-modal futures using additional prior knowledge [13]. However, such explicit usage makes the model heavily dependent on the pre-trained predictor, as denoising from an inaccurate prior hinders the model from learning the correct denoising path.

Consistency Models (CMs) [28] enable single-step sampling without relying on prior-informed sampling. CMs are a student-teacher based framework, either distilled from pre-trained diffusion models or trained from scratch, where the first approach achieves strong results in image generation. However, distillation requires robust pre-trained diffusion models, which are found scarce in trajectory prediction domain (shown later in this paper). The latter train-from-scratch method, on the other hand, requires carefully-tailored training guidance to ensure the quality.

We propose **ECTraj**, a conditional CM-based trajectory prediction framework. ECTraj is conditioned on both historical observations and a pre-trained prior, while performing denoising from a standard Gaussian distribution—thereby maximizing multi-modal exploration without being restricted by an informed initial distribution—and incorporates an enhanced consistency training pipeline. Additionally, the denoising component of ECTraj does not employ consistency distillation [28], and is instead trained from scratch [27, 28]. Our main contributions can be summarized as follows:

1. Instead of direct distillation or prior-informed sampling, we guide the training process by first using pretrained model components to encode the input into the latent space and then enhancing the student–teacher training with a ground-truth fusion to the teacher's output.

2. Tailoring for the multi-modal nature of the trajectory predictions, we exploit one-step denoising to sample multiple modes for student and teacher, compute a best-of-K loss for each network; this optimization is new for CMs and not directly applicable to noise-predicting diffusion models.

3. ECTraj outperforms the diffusion baseline OptTrajDiff on all standard trajectory benchmark metrics in the large-scale Argoverse 2 dataset, while using only 1/10 of the inference steps.

2 Related Work

2.1 Diffusion Models for Trajectory Prediction

Diffusion models [8, 29] have achieved impressive results in image generation. More recently, they have been shown to be a promising framework for trajectory prediction. MID [7] captures the uncertainty of agent's future movements by modeling the process as conditional diffusion, where the encoded historical context serves as a condition to generate future trajectories. LED [18] speeds up the sampling process by using DDIM denoising formulation and introducing a more informed prior, which is different from standard Gaussian noise. This in turn reduces the number of sampling steps needed to generate high-quality trajectories. TrajDiffuse [14] transforms the prediction task into planning-based trajectory inpainting and introduces map-based guidance module to ensure environmental compliance in model output. MotionDiffuser [9] applies diffusion models to the task of multi-agent trajectory prediction for autonomous driving, where they incorporate a permutation-invariant denoiser to enhance multi-agent prediction. OptTrajDiff [32] is a multi-agent prediction method that, similarly to LED, forms an informed prior as a via pretrained QCNet [37] and proposed estimated clean manifold guidance to boost the performance in controllable generation. However, all existing approaches still require more than 10 denoising steps, which limits the practicality of deploying the model in time-sensitive applications.

2.2 Consistency Models

To alleviate the computational overhead of iterative inference, various fast-sampling diffusion techniques have been developed [10, 21, 25, 28]. Prominent among these are Consistency Models (CMs) [28], which achieve single-step generation from noise via a student-teacher training paradigm. CMs are typically optimized using one of two strategies: Consistency Distillation (CD) or Consistency Training (CT). While CD initializes the model from a pretrained diffusion model, CT trains the model entirely from scratch. Although CD initially outperformed CT by a wide margin, more recent baselines [27] have systematically refined the design choices for CT, effectively bridging this performance gap. However, due to the lack of standardized pre-trained diffusion baselines for CD training, CMs have yet to be successfully adapted for the trajectory prediction domain. Inspired by recent extensions to the CM framework [3, 5, 6, 11], we propose ECTraj, a novel conditional CM-based trajectory prediction pipeline. ECTraj achieves state-of-the-art performance, while requiring only a tenth of the computational time compared to its diffusion counterparts.

3 Method

The complete high-level architecture of our proposed model and its training scheme are depicted in Figure 1. We describe each component in detail below.

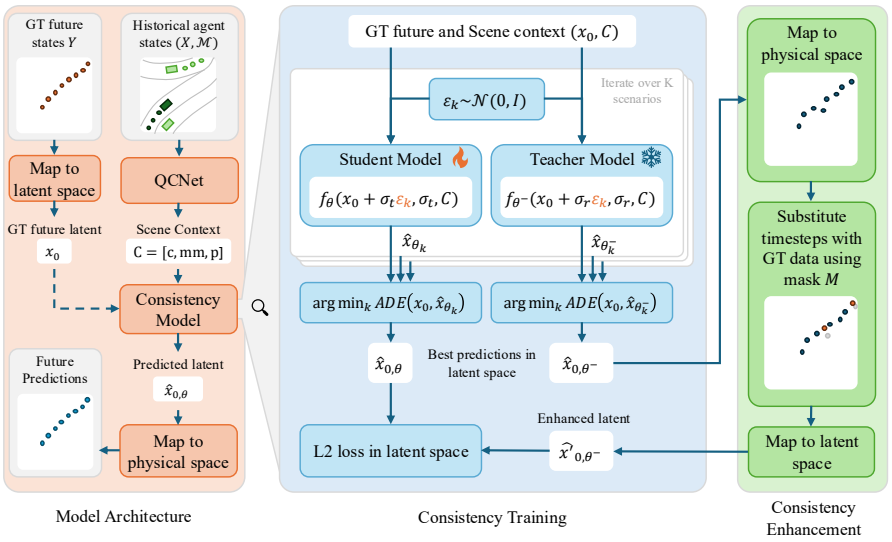

Fig. 1: Model Architecture and training scheme. (Left) The given historical trajectories and map information are encoded into a context latent vector. Then, the latent is processed by a consistency model and decoded to predicted future trajectories. (Middle) In the teacher-student training scheme of the consistency model, ECTraj samples K different Gaussian noises to produce K different future trajectories. Then the student model is updated using the distance between the best student latent prediction and the best teacher latent prediction. (Right) To enable better teacher guidance, the predicted teacher latent is enhanced by replacing some time steps with GT data in physical space.

3.1 Consistency Models Preliminaries

Consistency models (CMs) are built on the Probability Flow ODE (PF-ODE) in continuous-time diffusion models [30], which defines a smooth bijective trajectory that transforms the data distribution $p_0(x_0)$ into a tractable noise distribution $p_{\sigma_T}(x_{\sigma_T}) \sim \mathcal{N}(0, \epsilon)$. CMs impose a self-consistency constraint so that points x_{σ_t} and x_{σ_r} on the same trajectory map to the same initial data point x_0:

$$f(x_{\sigma_t}, \sigma_t) = f(x_{\sigma_r}, \sigma_r) = x_0,$$

enabling high-quality single-step generation from any intermediate state.

For training, the PF-ODE is discretized into time steps $t_1 < t_2 < \cdots < t_N = T$. The associated variances $\sigma_1 < \sigma_2 < \cdots < \sigma_N = T$ are obtained as a function of the time step, which is described in more detail in Section 3.3. CM then minimizes the distance between the denoised outputs of two adjacent steps:

$$\arg\min_\theta \mathbb{E}\big[w(\sigma_t)\, d\big(f_\theta(x_{\sigma_t}, \sigma_t), f_{\theta^-}(x_{\sigma_r}, \sigma_r)\big)\big],$$

where $t_1 \leq r < t \leq T$. Here $d(\cdot, \cdot)$ is a distance metric, f_θ is the neural network that estimates the consistency function, and f_{θ^-} is its exponential moving average (EMA).

3.2 Conditional CM for Multi-modal Trajectory Prediction

We propose a conditional CM framework for multi-agent trajectory prediction. A scene $(X, \mathcal{M})$ consists of agent states $X \in \mathbb{R}^{N_a \times T_h \times 2}$, where N_a is the number of agents, T_h the number of historical frames, and the last dimension the (x, y) coordinates, and a map $\mathcal{M} \in \mathbb{R}^{N_m \times D_p \times D_m}$, where N_m is the number of polylines, D_p the number of points per polyline, and D_m the number of attributes (e.g., position, orientation, magnitude). Given the encoded scene context $\mathbf{C}$, we learn a conditional consistency function $f_\theta(x_{\sigma_t}, \sigma_t, \mathbf{C})$ that directly maps a noisy prior to the future trajectory space to future predictions for $Y \in \mathbb{R}^{N_a \times T_f \times 2}$. To construct the conditional consistency function f_θ, we first encode the historical observations of the surrounding environment and social interactions into a context vector

$$c = \Phi(X, \mathcal{M}). \tag{1}$$

To account for the inherently multimodal nature of agent motion, prior work typically introduces auxiliary modeling mechanisms that construct priors—either as deterministic anchor trajectories or as learned prior distributions—to facilitate exploration over multiple plausible futures. However, owing to the characteristics of consistency models (CM), which perform denoising directly from Gaussian white noise, it is nontrivial to decode trajectories directly from such priors. Consequently, we instead supply the consistency function with marginal future predictions and their associated scores, obtained from a pre-trained model, as prior anchor trajectory contexts, denoted by mm and p during training time to direct the model with more efficient learning.

$$mm, p = \Theta(c) \tag{2}$$

After forming the combined context representation $\mathbf{C} = [c, mm, p]$, we apply three layers of composite attention blocks to the intermediate noisy input x_σ. Within each layer, we first perform cross-attention between x_σ and the historical encodings of the target agent. We then apply cross-attention between x_σ and the map encodings (e.g., polygonal representations), followed by cross-attention with the context encodings of the neighboring agents. Subsequently, we conduct self-attention among the future encodings of different agents to capture their joint interactions. Finally, we introduce an additional cross-attention operation between the marginal priors and the noisy input, thereby integrating prior information into the denoising process and obtained the denoising output.

$$\hat{x}_0 = f_\theta(x_\sigma, \sigma, \mathbf{C}) \tag{3}$$

We point out that the consistency function outputs only the (latent) trajectories. The function does not calculate the score for each joint prediction. Since

calculating this score can provide useful information about the confidence of each prediction, joint score calculation can be considered in future work.

Input compression. In the trajectory space, the flattened input has dimension $N_h \times 2$. Following [9] and [32], we accelerate the decoding by linearly compressing the trajectories into 10-dimensional latent vectors. The trajectory-level input X is mapped to the latent input x as:

$$x = XU, \tag{4}$$

and mapped back by applying:

$$X = Vx. \tag{5}$$

The matrices U and V are learnable; details are given in the Appendix.

3.3 ECTraj - Enhanced Consistency training

Consistency formulation. During training, we first sample the *student* index t from $\{1, \ldots, N\}$, where N is the number of discretization steps of the ODE trajectory and determines the noise added to the clean input. Following [27], N is chosen adaptively. In ECTraj, we start with $N = 10$ and, over E training epochs, double the value of N every $\frac{E}{3}$ epochs, so training ends with $N = 40$. The index t is sampled from a discrete log-normal distribution over indices, as in [27]; further details are in the Appendix. Given t, we then obtain the *teacher* index r as follows:

$$r = \lfloor t(1 - \frac{n(t')}{q^{\lceil \frac{e}{d} \rceil}}) \rfloor. \tag{6}$$

This parameterization is applied in line with ECT [6]. In Equation (6), $t' = \frac{t}{N}$, scaling t to $t' \in [0, 1]$, and $n(t') = 1 + \frac{k}{1+e^{bt'}}$; where $k = 8$ and $b = 1$. Additionally, e denotes the current epoch and d denotes a threshold epoch that is aligned with the discretization curriculum. More precisely, $d = E//3$ (where $//$ denotes integer division). In cases where $n(t') > q^{\lceil \frac{e}{d} \rceil}$, r is clamped to 0 to avoid negative indexing. The choice of q varies between domains and $q = 4$ was the best choice for our use case. This parameterization leads to a hybrid diffusion-consistency training procedure. In the beginning, the model essentially optimizes for reconstruction, since $r = 0$ in the initial stages of training due to clamping. As training progresses, r approaches t, transitioning the training process to standard consistency training [28] as training nears its end. Equation (6) describes the index retrieval. The noise to be added to the input is a function of the sampled index. For some index t, the variance is calculated in line with [10]:

$$\sigma_t = (\sigma_{min}^{\frac{1}{\rho}} + \frac{t-1}{N-1}(\sigma_{max}^{\frac{1}{\rho}} - \sigma_{min}^{\frac{1}{\rho}}))^{\rho}, \tag{7}$$

where $\sigma_{min} = 0.002$, $\sigma_{max} = 1.0$[4] and $\rho = 7$. For the purposes of discrete log-normal sampling, we also left-pad $\sigma_0 = 0.0$. The explanation for prepending this

[4] This is different from the standard training for image generation where $\sigma_{max} = 80$. This value proved to be too large for our use case.

value is deferred to the Appendix. The student and teacher samples are obtained as:

$$x_{\sigma_t} = x_0 + \sigma_t \epsilon, \tag{8}$$

$$x_{\sigma_r} = x_0 + \sigma_r \epsilon, \tag{9}$$

where $\epsilon \sim \mathcal{N}(0, \mathbf{I})$, with $\mathbf{I}$ being the identity matrix. We point out that x_0 are compressed trajectory vectors and that optimization is done in the latent space.

Multi-shot generation at training time. Our model produces high-quality trajectories in a single forward pass, which allows us to define a top-K consistency loss. This type of loss is not practical for diffusion-based methods such as OptTrajDiff [32], which operate in the noise space, making it non-trivial to identify the best trajectory. We demonstrate this empirically in Section 4.3. Specifically, we sample K random noise vectors and add them to both the student and the teacher:

$$x_{\sigma_t,k} = x_0 + \sigma_t \epsilon_k, \tag{10}$$

$$x_{\sigma_r,k} = x_0 + \sigma_r \epsilon_k, \tag{11}$$

for all $k \in \{1, 2, ..., K\}$. $\epsilon_k \sim \mathcal{N}(0, I)$ represents K different random noise vectors. Next, we separately denoise the trajectories from each group (K student and K teacher modes). After denoising, we choose the best trajectories from both groups based on ADE:

$$k_\theta = \arg\min_k \text{ADE}(f_\theta(x_{\sigma_t,k}, \sigma_t, \mathbf{C}), x_0), \tag{12}$$

$$k_{\theta^-} = \arg\min_k \text{ADE}(f_{\theta^-}(x_{\sigma_r,k}, \sigma_r, \mathbf{C}), x_0). \tag{13}$$

θ^- is obtained using the exponential moving average (EMA) of θ. In line with [10], the denoiser f_θ is parameterized as follows:

$$f_\theta(x, \sigma, \mathbf{C}) = c_{skip}(\sigma)x + c_{out}(\sigma)F_\theta(x, \sigma, \mathbf{C}), \tag{14}$$

where F_θ denotes the trainable part of the denoiser (usually a neural network). The teacher parameterization is done similarly. $c_{skip}(\cdot)$ and $c_{out}(\cdot)$ denote differentiable functions for which $c_{skip}(\sigma_{min}) = 1$ and $c_{out}(\sigma_{min}) = 0$. The exact choices of these functions will be provided in the Appendix. For brevity, we will use the following two notations:

$$\hat{x}_{0,\theta} = f_\theta(x_{\sigma_t,k_\theta}, \sigma_t, \mathbf{C}_t), \tag{15}$$

$$\hat{x}_{0,\theta^-} = f_\theta(x_{\sigma_r,k_{\theta^-}}, \sigma_r, \mathbf{C}_r). \tag{16}$$

Enhanced consistency objective. Standard CMs directly compare $\hat{x}_{0,\theta}$ and $\hat{x}_{0,\theta^-}$. In ECTraj, we refine the teacher output $\hat{x}_{0,\theta^-}$ by fusing it with parts of the ground truth to provide a stronger supervision for the student. Let X_f be

the ground-truth future in trajectory space, and $\hat{X}_{0,\theta^-}$ the predicted trajectory-space output derived from $\hat{x}_{0,\theta^-}$ as described in Section 3.2. The modified teacher output is then defined as:

$$\hat{X}'_{0,\theta^-} = (1 - M) \odot \hat{X}_{0,\theta^-} + M \odot X_f. \tag{17}$$

Here, M denotes a binary mask that determines which parts of the ground truth are replacing the original output, and $\odot$ the Hadamard product. $\hat{X}_{0,\theta^-}$ is then mapped back into the latent space representation $\hat{x}'_{0,\theta^-}$, also as explained in Section 3.2. We note the similarity between Equation (17) and the future mixup in TimeDiff [23], which can also be viewed as a variant of teacher forcing [34]. The key difference lies in masking: TimeDiff uses random masking, while we deterministically replace two waypoints with ground truth—the midpoint and endpoint. Mixing these GT waypoints into the teacher output exposes the student network to higher quality supervision, as this simple modification refines the teacher output and provides the student with several ground truth anchors, thereby aiding learning. The ECTraj objective is:

$$\mathcal{L}_{ECTraj} = w(\sigma_t) d(\hat{x}_{0,\theta}, \hat{x}'_{0,\theta^-}), \tag{18}$$

where $w(\sigma_t) = \frac{1}{\sigma_t - \sigma_r}$ and d is a distance metric (in our case, the L_2 norm). Algorithm 1 shows the ECTraj training algorithm. The teacher weights are updated as the EMA of the student weights. Since teacher weights are not backpropagated, we apply the stop-gradient operator (sg) to them.

3.4 Inference

We design our inference procedure based on the configuration recommended in [27]. For multi-step sampling, the intermediate time indices are defined as

$$\tau_n = \frac{n}{N} \tau_N, \tag{19}$$

where $n \in \{1, 2, ..., N\}$. Here, τ_N is the largest index and corresponds to the maximum variance. ECTraj performs single-step denoising starting from standard Gaussian noise. In contrast to earlier diffusion-based methods [18, 32], ECTraj does not rely on an informed or specialized initial distribution. Although multi-step inference is feasible, we empirically observed that single-step sampling yields the best performance (multi-step result in supplementary). We hypothesize that this is because multi-step inference excessively corrupts the samples through repeated noise injection, in line with the observations reported in [28]. Algorithm 2 shows the inference algorithm.

4 Experiments

Dataset and metrics. We conduct our experiments on the Argoverse 2 Motion Forecasting Dataset [35], a widely used large-scale benchmark for trajectory prediction. Each scenario provides 50 past time steps (5 seconds) and 60

Algorithm 1 ECTraj Training

Input: Historical data $(X_h, \mathcal{M})$, ground-truth future data X_f, linear mapping U and its inverse V, QCNet encoder/decoder (Φ, Θ), max epochs E, discrete lognormal params (μ, Σ), modes K, EMA parameter α, binary mask M
Init: Randomly initialize θ and θ^-
for $e = 1, \ldots, E$ **do**
  for $\big((x_h, m), x_f\big)$ in $\big((X_h, \mathcal{M}), X_f\big)$ **do**
    $\mathbf{c} = \Phi(x_h, m)$
    $(\mathbf{mm}, \mathbf{p}) = \Theta(\mathbf{c})$
    $N = 10 \cdot 2^{\lfloor \frac{3e}{E} \rfloor}$
    $t \sim \text{DiscreteLogNormal}(\mu, \Sigma, N)$
    $r = \lfloor t(1 - \frac{n(t')}{q^{\lceil \frac{e}{d} \rceil}}) \rfloor$
    $x_0 = x_f U$
    for $k = 1, \ldots, K$ **do**
      $\varepsilon_k \sim \mathcal{N}(0, I)$
      $x_{\sigma_t,k} = x_0 + \sigma_t \, \varepsilon_k$
      $x_{\sigma_r,k} = x_0 + \sigma_r \, \varepsilon_k$
    end for
    $\mathbf{C} = (\mathbf{c}, \mathbf{mm}, \mathbf{p})$
    $k_\theta = \arg\min_k \text{ADE}\big(f_\theta(x_{\sigma_t,k}, \sigma_t, \mathbf{C}), x_0\big)$
    $k_\theta^- = \arg\min_k \text{ADE}\big(f_\theta^-(x_{\sigma_r,k}, \sigma_r, \mathbf{C}), x_0\big)$
    $\hat{x}_{0,\theta} = f_\theta(x_{\sigma_t,k_\theta}, \sigma_t, \mathbf{C})$
    $\hat{x}_{0,\theta^-} = f_{\theta^-}(x_{\sigma_r,k_{\theta^-}}, \sigma_r, \mathbf{C})$
    $\hat{X}_{0,\theta^-} = V x_{0,\theta^-}$
    $\hat{X}'_{0,\theta^-} = (1 - M) \odot \hat{X}_{0,\theta^-} + M \odot X_f$
    $\hat{x}'_{0,\theta^-} = \hat{X}'_{0,\theta^-} U$
    $\mathcal{L}_{ECTraj} = w(\sigma_t) d(\hat{x}_{0,\theta}, \hat{x}'_{0,\theta^-})$
    $\theta^- = sg((1 - \alpha)\,\theta + \alpha\,\theta^-)$ ▷ EMA update for teacher; sg = stop-gradient
  end for
end for
return $\mathcal{L}_{ms}$

future time steps (6 seconds). We adopt the standard metrics from the Argoverse 2 evaluation. Specifically, we incorporate the following accuracy metrics for joint forecasts: avg min ADE_6/FDE_6 and avg min ADE_1/FDE_6. In the following text and all tables reporting these metrics, we omit "avg min" to preserve space. ADE_K and FDE_K with $K = 1$ and $K = 6$ to assess the upper bound of the model prediction performance. We further report Brier-FDE_6 (abbreviated as b-FDE_6), which augments the FDE with a confidence penalty term of $(1 - p^2)$, where p is the predicted probability of the best-of-K trajectory. In addition, we measure the miss rate (MR_K), defined as the fraction of scenarios in which none of the K predicted trajectories has the FDE less than or equal to **2.0m**, and the collision rate (CR_K), defined as the fraction of scenarios in which at least two agents collide, using a collision distance threshold of **1.0m**.

Algorithm 2 Inference

Input: Historical data $(X_h, \mathcal{M})$, noisy prior x_N, QCNet encoder/decoder Φ/Θ, learned consistency model θ, number of inference steps N, intermediate sampling timestep indices $\tau_0 < \tau_1 < \tau_2 < ... < \tau_{N-1}$, such that $\sigma_{\tau_0} = 0$, $\sigma_{\tau_1} = \sigma_{min}$, $\sigma_{\tau_{N-1}} = \sigma_{max}$:
for (x_h, m) in (X_h, M) **do**
  $x_{next} = x_N$
  $\mathbf{c} = \Phi(x_h, m)$
  $(\mathbf{mm}, \mathbf{p}) = \Theta(\mathbf{c})$
  $\epsilon \sim \mathcal{N}(0, I)$
  $\mathbf{C} = (\mathbf{c}, \mathbf{mm}, \mathbf{p})$
  for i in N-1..1 **do**
    $x_{0,\tau_i} = f_\theta(x_{next}, \sigma_{\tau_i}, \mathbf{C})$
    $x_{next} = x_{0,\tau_i} + \sigma_{\tau_{i-1}}\epsilon$
  end for
end for
return x_{next}

For $K = 6$, we start with K independently sampled noise vectors. We do not apply any post-processing, as our goal is to evaluate how well ECTraj performs as a standalone consistency-based trajectory prediction model. This is different from OptTrajDiff, which generates $K = 128$ predictions that are used to refine QCNet predictions, using a clustering algorithm to generate candidate joint trajectories. More details are provided in [32]. Another popular multi-agent trajectory prediction dataset is the **Waymo Open Motion Dataset** [2]. Results on the Waymo dataset are provided in the Appendix.

Baselines. We compared our method with several recent state-of-the-art baselines whose results on Argoverse 2 are publicly available. Specifically, we compare ECTraj to OptTrajDiff [32] as its closest counterpart, which is the only joint diffusion-based model directly evaluated on Argoverse 2. We list the benchmark results in Table 1.

Training details. Our model was trained for 60 epochs, using four RTX A6000 GPUs. During training, we used the batch size of 16, and AdamW [15] optimizer with an initial learning rate of 0.002 and a weight decay of 0.0001. For multi-shot trajectory generation, we generated $K = 6$ future joint trajectories for each scene in the training set, since metric evaluation is commonly done using this number of modes.

4.1 Quantitative Result: Argoverse 2

Our main experiment consists of comparing ECTraj with several state-of-the-art baselines. The results are shown in Table 1. ECTraj achieves a state-of-the-art result for FDE_6, and is on par with QCNeXt for all metrics. ECTraj achieves

Table 1: Standard Argoverse 2 metric evaluation

Method	Year	Venue	ADE_6	FDE_6	ADE_1	FDE_1	$b\text{-}FDE_6$	MR_6	CR_6
FJMP [20]	2023	CVPR	0.81	1.89	1.52	4.00	2.59	0.19	**0.01**
GNet [4]	2023	RA-L	0.69	1.46	1.23	3.05	2.12	0.19	**0.01**
QCNeXt [39]	2023	CVPRW	**0.50**	1.02	**0.94**	**2.29**	**1.65**	**0.13**	**0.01**
Forecast-MAE [1]	2023	ICCV	0.69	1.55	1.30	3.33	2.24	0.19	**0.01**
OptTrajDiff [32]	2024	ECCV	0.60	1.31	1.08	2.71	1.95	0.17	**0.01**
DeMo [36]	2024	NeurIPS	0.58	1.24	1.12	2.78	1.93	0.16	**0.01**
RealMotion [26]	2024	NeurIPS	0.62	1.32	1.14	2.87	2.01	0.18	**0.01**
FutureNet-LOF [31]	2025	ICRA	0.58	1.25	0.96	<u>2.34</u>	<u>1.68</u>	0.18	0.02
DONUT [12]	2025	ICCV	0.55	1.13	1.07	2.62	1.79	0.15	**0.01**
ECTraj (ours)			**0.50**	**0.96**	**0.94**	2.37	<u>1.68</u>	**0.13**	**0.01**

Table 2: Inference speed of ECTraj

Method	NFE	GFLOPs
OptTrajDiff [33]	10	~ 1311
ECTraj	1	~ 132

Table 3: Single-scenario inference speed statistics for Argoverse 2 in milliseconds.

Method	Denoising only	QCNet	QCNet + denoising
QCNet [38]	N/A	24.84	N/A
OptTrajDiff [33]	29.71	24.84	54.55
ECTraj	2.97	24.84	27.81

better performance than OptTrajDiff across all metrics, showcasing the effectiveness of our CM formulation. Although ECTraj is only the third in terms of FDE_1, the runner-up method in terms of this metric (FutureNet-LOF) achieves noticeably weaker accuracy metrics and miss rate for $K = 6$. We also demonstrate additional examples of ECTraj that exhibit better diversity in Section 4.2.

Inference speed. One major motivation for replacing the diffusion denoiser with a consistency one lies in its inference speed. We compare our method with OptTrajDiff [32] in this aspect and show that our method achieves up to 10 times faster inference compared to OptTrajDiff in Table 2. OptTrajDiff uses 10 inference steps, while ECTraj uses only one, which we report in the NFE (**N**umber of **F**unction **E**valuations) column. Since both OptTrajDiff and ECTraj incorporate QCNet into their pipeline, it is useful to evaluate how much computational overhead the denoising component of either method adds. In Table 3, we show the average inference speed per scenario on the Argoverse 2 dataset. We demonstrate that ECTraj adds much less overhead ($\approx 12\%$) compared to OptTrajDiff ($\approx 120\%$). To further demonstrate the ten-fold speedup achieved by ECTraj com-

pared to OptTrajDiff, we also report Giga-floating-point operations (GFLOPs) as a more objective measure compared to raw GPU time. Combined with the superior prediction performance, ECTraj demonstrates the strong potential of using CM as an alternative backbone for the task of trajectory prediction.

4.2 Qualitative examples

Accuracy, diversity and compliance analysis. In Figure 2, we present visual results for the following four methods: *QCNeXt* [39], *OptTrajDiff* [32], *ECTraj (one-shot loss)* and *ECTraj (six-shot loss)*. In these scenarios, ECTraj excels at various aspects of trajectory prediction:

- ***Scenario 1 - diversity and environmental compliance***; although all three models are able to model diverse movements, only ECTraj is able to predict an alternate mode that goes straight ahead without violating environmental constraints. Additionally, six-shot ECTraj is able to generate trajectories that better align with the ground truth compared to the one-shot variant.
- ***Scenario 2 - accuracy***; although all three methods overshoot the ground truth goal point, ECTraj does so to the smallest extent. In this scenario, accuracy-wise, there is no significant difference between one-shot and six-shot ECTraj. This may be due to the simple setting of this scenario, which does not allow for many socially compliant types of intent (i.e. the only feasible intents are going straight ahead or stopping)
- ***Scenario 3 - complex movements***; although all three methods recognize the intent of a sharp left turn, both QCNeXt and OptTrajDiff predict some modes which pass a non-traversable area, which does not happen in ECTraj. This is even more apparent in the six-shot setting.

In summary, the qualitative comparisons clearly establish the proposed ECTraj along with the multi-shot loss as the superior approach. Although training with one-shot loss provides a competent baseline for simpler scenarios, it is the six-shot training formulation that unlocks the full robust capability of our method. By improving alignment with ground-truth trajectories and better adherence to environmental constraints during complex maneuvers, ECTraj demonstrate enhanced generative quality with reduced computation latency.

A failure mode: U-turns. In addition to scenarios where ECTraj outperforms the baselines such as OptTrajDiff and QCNet, we also demonstrate a typical failure mode of ECTraj, which are U-turns. In Figure 3, we show that this failure mode is not unique to ECTraj and is present in OptTrajDiff and QCNet as well. Additionally, even in this failure scenario, the modes predicted by ECTraj appear to align better with the ground truth mode. We suspect that the main reason for this is the incorporation of the teacher fusion mechanism. This mechanism helps the model determine more easily (although still not entirely successfully) the type of movement it needs to predict, as the agent was stationary prior to making the U-turn, which hinders the inference process.

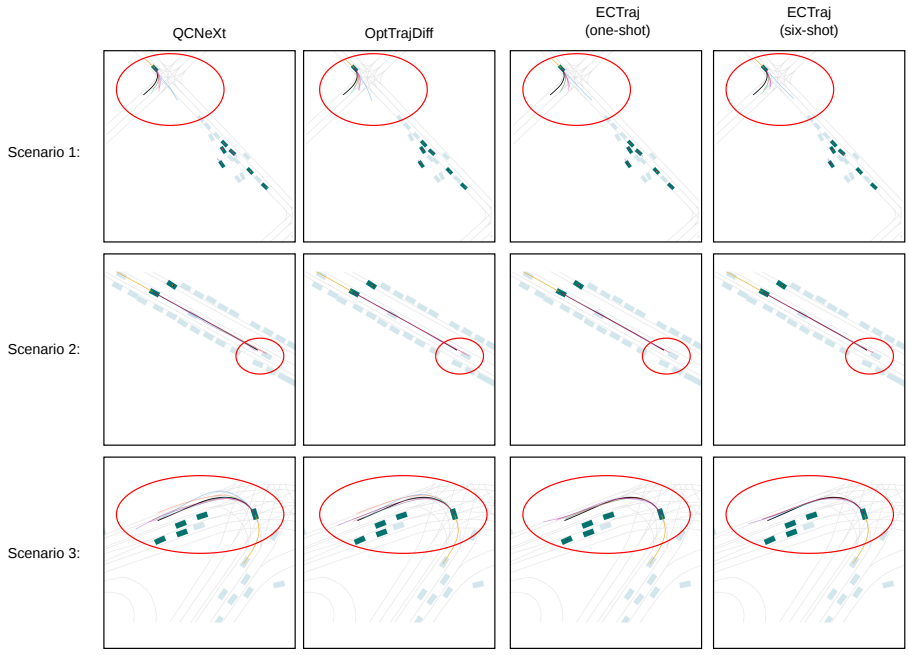

Fig. 2: Qualitative examples. Historical trajectories are denoted in **orange**, and ground truth future is denoted in **black**. Each of the 6 predicted modes is coded in different colors for easier interpretability. Regions of interest in each scenarios are encircled in **red**, also for easier interpretability.

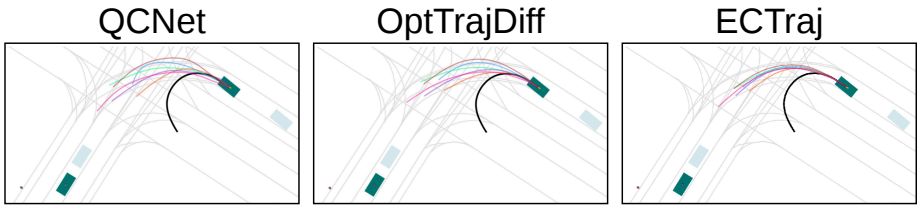

Fig. 3: A typical failure mode. Ground truth past shown in orange, ground truth future in **black**, and the six predicted modes in different colors to differentiate them from each other.

4.3 Ablation study

In this section, we analyze the impact of enhanced consistency training designs, top-K training in Diffusion formulation, and CM scheduling choices.

Enhanced consistency objective. With the results shown in Table 4, all of our training components contribute to the performance of the model. As indicated in row 1 of Table 4, one-shot training shows reduced performance, which

Table 4: Ablation study on the individual components of ECTraj. The ablated components denote the following: (1) *one-shot training* - replacing six-shot mode generation at training with one-shot, (2) *standard CM objective* - removing fusion of ground truth with teacher, (3) *reconstruction objective* - changing teacher output with ground truth (full ground truth fusion).

Ablated component	ADE_6	FDE_6	ADE_1	FDE_1	$b\text{-}FDE_6$	MR_6	CR_6
One-shot training	0.53	1.03	0.98	2.39	1.76	0.14	0.007
Standard CM objective	0.52	0.99	0.96	2.38	1.70	0.14	0.007
Reconstruction objective	0.57	1.10	1.08	2.53	1.87	0.17	0.009
ECTraj	**0.50**	**0.96**	**0.94**	**2.37**	**1.68**	**0.13**	**0.006**

aligns with the observation from qualitative analysis. We also ablate the teacher output enhancement in two ways.

- ***Standard CM objective***: No fusion of ground truth with teacher outputs.
- ***Reconstruction objective***: We replace every single teacher output point with the ground truth, which reduces the method to the standard direct denoising objective in some variants of diffusion models [10].

Both ablations yield suboptimal performance across all metrics. This is intuitive for the following reasons: with the *standard CM objective*, although this objective is the easiest for the student model to learn, it can be unreliable because the teacher model itself may produce inaccurate outputs. With the *reconstruction objective*, the teacher's output is, in principle, as accurate as possible, but it becomes difficult for the student to learn from—especially when trained with a small number of steps, as in ECTraj. In this case, the student model would require many more iterations to converge. Consequently, the ECTraj training pipeline strikes a careful balance between the quality of supervision and the ease of optimization.

One-shot vs. multi-shot generation at training time. As demonstrated in the ablation study above, introducing multiple diverse modes facilitates faster learning, as the model can learn dedicated denoising trajectories for distinct types of motion. In diffusion-based methods such as OptTrajDiff, the best-of-K strategy is not directly applicable: because diffusion models are trained to predict noise, it is not evident how to define or select the "best" noisy mode. Consequently, we investigate two alternative formulations to analyze the impact of multi-shot loss training on the denoising process.

- ***Variant 1***: The best noisy mode is selected as the mode that is the closest to the ground truth **data**.
- ***Variant 2***: The best noisy mode is selected as the mode that is the closest to the ground truth **noise**.

We studied the performance of these two variants in Table 5. Although ***Variant 1*** does outperform OptTrajDiff in terms of FDE_6 and is on par with it in terms

Table 5: Best-of-K approach used in ECTraj tested on OptTrajDiff

Variant	ADE_6	FDE_6	ADE_1	FDE_1	$b\text{-}FDE_6$	MR_6	CR_6
Variant 1	0.64	1.17	1.44	3.46	2.02	0.17	**0.01**
Variant 2	0.99	1.42	2.38	5.10	2.30	0.21	0.03
OptTrajDiff (one-shot)	0.60	1.31	1.08	2.71	1.95	0.17	**0.01**
ECTraj	**0.50**	**0.96**	**0.94**	**2.37**	**1.68**	**0.13**	**0.01**

Table 6: ICT vs ECT scheduling

Scheduling Type	ADE_6	FDE_6	ADE_1	FDE_1	$b\text{-}FDE_6$	MR_6	CR_6
ICT	0.52	1.02	0.95	2.38	1.77	0.15	**0.006**
ECT (**ECTraj**)	**0.50**	**0.96**	**0.94**	**2.37**	**1.68**	**0.13**	**0.006**

of MR_6, both variants result in substantially worse metric values compared to ECTraj.

Consistency scheduling type. In ECTraj, we use timestep scheduling in line with ECT [6]. In image generation, this method was shown to surpass previous baselines such as [28] and [27]. This is also the case in trajectory prediction, as we found that our ECT scheduling achieves a small improvement compared to prior methods (for example, ICT scheduling) [27] in Table 6.

5 Conclusion and Future Work

In this paper, we presented **ECTraj**, a multi-agent trajectory prediction model that utilizes consistency models. We depart from the standard notion of consistency models, and allow fuse parts of the teacher output with the ground truth, where we use the midpoint and the endpoint of the ground truth future to promote accuracy. On top of that, we incorporate a best-of-K training process, which is not feasible in diffusion-based baselines, which optimizes for noise prediction (such as OptTrajDiff). Both of these components help ECTraj achieve state-of-the-art results in almost every Argoverse 2 leaderboard metric, which is particularly noticeable with FDE_6. One limitation of ECTraj along with previous diffusion counterparts is its reliance on marginal QCNet priors. We demonstrate the necessity of these priors in the Appendix, showing that the removal of these priors results in worse accuracy metrics on Argoverse 2. That said, in future work, we will consider an end-to-end approach that either does not depend on these priors or uses self-obtained priors.

References

1. Cheng, J., Mei, X., Liu, M.: Forecast-mae: Self-supervised pre-training for motion forecasting with masked autoencoders. In: Proceedings of the IEEE/CVF Interna-

tional Conference on Computer Vision (ICCV). pp. 8679–8689 (2023)
2. Ettinger, S., Cheng, S., Caine, B., Liu, C., Zhao, H., Pradhan, S., Chai, Y., Sapp, B., Qi, C.R., Zhou, Y., Yang, Z., Chouard, A., Sun, P., Ngiam, J., Vasudevan, V., McCauley, A., Shlens, J., Anguelov, D.: Large scale interactive motion forecasting for autonomous driving: The waymo open motion dataset. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV). pp. 9710–9719 (October 2021)
3. Frans, K., Hafner, D., Levine, S., Abbeel, P.: One step diffusion via shortcut models. In: The Thirteenth International Conference on Learning Representations (ICLR) (2025)
4. Gao, X., Jia, X., Li, Y., Xiong, H.: Dynamic scenario representation learning for motion forecasting with heterogeneous graph convolutional recurrent networks. IEEE Robotics and Automation Letters **8**(5), 2946–2953 (2023)
5. Geng, Z., Deng, M., Bai, X., Kolter, J.Z., He, K.: Mean flows for one-step generative modeling. In: The Thirty-ninth Annual Conference on Neural Information Processing Systems (NeurIPS) (2025)
6. Geng, Z., Pokle, A., Luo, W., Lin, J., Kolter, J.Z.: Consistency models made easy. In: The Thirteenth International Conference on Learning Representations (ICLR) (2025)
7. Gu, T., Chen, G., Li, J., Lin, C., Rao, Y., Zhou, J., Lu, J.: Stochastic trajectory prediction via motion indeterminacy diffusion. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 17113–17122 (2022)
8. Ho, J., Jain, A., Abbeel, P.: Denoising diffusion probabilistic models. Advances in Neural Information Processing Systems (NeurIPS) **33**, 6840–6851 (2020)
9. Jiang, C., Cornman, A., Park, C., Sapp, B., Zhou, Y., Anguelov, D., et al.: MotionDiffuser: Controllable multi-agent motion prediction using diffusion. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 9644–9653 (2023)
10. Karras, T., Aittala, M., Aila, T., Laine, S.: Elucidating the design space of diffusion-based generative models. Advances in Neural Information Processing Systems (NeurIPS) **35**, 26565–26577 (2022)
11. Kim, D., Lai, C.H., Liao, W.H., Murata, N., Takida, Y., Uesaka, T., He, Y., Mitsufuji, Y., Ermon, S.: Consistency trajectory models: Learning probability flow ode trajectory of diffusion. In: The Twelfth International Conference on Learning Representations (ICLR) (2024)
12. Knoche, M., de Geus, D., Leibe, B.: Donut: A decoder-only model for trajectory prediction. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV). pp. 28903–28912 (2025)
13. Li, C., Zhang, R., Huang, S., Zhong, X., Jiang, H.: Agma: Adaptive gaussian mixture anchors for prior-guided multimodal human trajectory forecasting. arXiv preprint arXiv:2602.04204 (2026)
14. Liu, Q.T., Li, D., Sohn, S.S., Yoon, S., Kapadia, M., Pavlovic, V.: TrajDiffuse: A conditional diffusion model for environment-aware trajectory prediction. In: International Conference on Pattern Recognition (ICPR). pp. 382–397. Springer (2024)
15. Loshchilov, I., Hutter, F.: Decoupled weight decay regularization. In: International Conference on Learning Representations (ICLR) (2018)
16. Luo, J., Zhang, C., Si, W., Jiang, Y., Yang, C., Zeng, C.: A physical human–robot interaction framework for trajectory adaptation based on human motion prediction and adaptive impedance control. IEEE Transactions on Automation Science and Engineering **22**, 5072–5083 (2024)

17. Madjid, N.A., Ahmad, A., Mebrahtu, M., Babaa, Y., Nasser, A., Malik, S., Hassan, B., Werghi, N., Dias, J., Khonji, M.: Trajectory prediction for autonomous driving: Progress, limitations, and future directions. Information Fusion p. 103588 (2025)
18. Mao, W., Xu, C., Zhu, Q., Chen, S., Wang, Y.: Leapfrog diffusion model for stochastic trajectory prediction. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 5517–5526 (2023)
19. Raina, S., Challagundla, J., Singh, M.: Trajectory prediction in autonomous driving: A comprehensive review of deep learning models and future direction. In: 2025 IEEE 15th Annual Computing and Communication Workshop and Conference (CCWC). pp. 00753–00759. IEEE (2025)
20. Rowe, L., Ethier, M., Dykhne, E.H., Czarnecki, K.: FJMP: Factorized joint multi-agent motion prediction over learned directed acyclic interaction graphs. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 13745–13755 (2023)
21. Salimans, T., Ho, J.: Progressive distillation for fast sampling of diffusion models. In: International Conference on Learning Representations (ICLR) (2022)
22. Samavi, S., Lem, A., Sato, F., Chen, S., Gu, Q., Yano, K., Schoellig, A.P., Shkurti, F.: Sicnav-diffusion: Safe and interactive crowd navigation with diffusion trajectory predictions. IEEE Robotics and Automation Letters (2025)
23. Shen, L., Kwok, J.: Non-autoregressive conditional diffusion models for time series prediction. In: International Conference on Machine Learning (ICML). pp. 31016–31029. PMLR (2023)
24. Singha, A., Nanwani, L., Jain, S., Singamaneni, P.T., Singh, A.K., Krishna, K.M., et al.: Crowd-fm: Learned optimal selection of conditional flow matching-generated trajectories for crowd navigation. arXiv preprint arXiv:2602.06698 (2026)
25. Song, J., Meng, C., Ermon, S.: Denoising diffusion implicit models. In: International Conference on Learning Representations (ICLR) (2021)
26. Song, N., Zhang, B., Zhu, X., Zhang, L.: Motion forecasting in continuous driving. Advances in Neural Information Processing Systems (NeurIPS) **37**, 78147–78168 (2024)
27. Song, Y., Dhariwal, P.: Improved techniques for training consistency models. In: The Twelfth International Conference on Learning Representations (ICLR) (2024)
28. Song, Y., Dhariwal, P., Chen, M., Sutskever, I.: Consistency models. In: Proceedings of the 40th International Conference on Machine Learning (ICML). pp. 32211–32252 (2023)
29. Song, Y., Ermon, S.: Generative modeling by estimating gradients of the data distribution. Advances in Neural Information Processing Systems (NeurIPS) **32** (2019)
30. Song, Y., Sohl-Dickstein, J., Kingma, D.P., Kumar, A., Ermon, S., Poole, B.: Score-based generative modeling through stochastic differential equations. In: International Conference on Learning Representations (ICLR) (2021)
31. Wang, M., Ren, X., Jin, R., Li, M., Zhang, X., Yu, C., Wang, M., Yang, W.: FutureNet-LOF: Joint trajectory prediction and lane occupancy field prediction with future context encoding. In: 2025 IEEE International Conference on Robotics and Automation (ICRA). pp. 8841–8848. IEEE (2025)
32. Wang, Y., Tang, C., Sun, L., Rossi, S., Xie, Y., Peng, C., Hannagan, T., Sabatini, S., Poerio, N., Tomizuka, M., et al.: Optimizing diffusion models for joint trajectory prediction and controllable generation. In: European Conference on Computer Vision (ECCV). pp. 324–341. Springer (2024)

33. Wang, Y., Tang, C., Sun, L., Rossi, S., Xie, Y., Peng, C., Hannagan, T., Sabatini, S., Poerio, N., Tomizuka, M., et al.: Optimizing diffusion models for joint trajectory prediction and controllable generation. In: ECCV (2024)
34. Williams, R.J., Zipser, D.: A learning algorithm for continually running fully recurrent neural networks. Neural Computation **1**(2), 270–280 (1989)
35. Wilson, B., Qi, W., Agarwal, T., Lambert, J., Singh, J., Khandelwal, S., Pan, B., Kumar, R., Hartnett, A., Pontes, J.K., et al.: Argoverse 2: Next generation datasets for self-driving perception and forecasting. In: Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2) (2021)
36. Zhang, B., Song, N., Zhang, L.: Decoupling motion forecasting into directional intentions and dynamic states. Advances in Neural Information Processing Systems (NeurIPS) **37**, 106582–106606 (2024)
37. Zhou, Z., Wang, J., Li, Y.H., Huang, Y.K.: Query-centric trajectory prediction. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 17863–17873 (2023)
38. Zhou, Z., Wang, J., Li, Y.H., Huang, Y.K.: Query-centric trajectory prediction. In: CVPR (2023)
39. Zhou, Z., Wen, Z., Wang, J., Li, Y.H., Huang, Y.K.: QCNeXt: A next-generation framework for joint multi-agent trajectory prediction. arXiv preprint arXiv:2306.10508 (2023)