UniMotion: A Unified Motion Framework for Simulation, Prediction and Planning

Nan Song[1] **Junzhe Jiang**[1] **Jingyu Li**[1,2] **Xiatian Zhu**[3] **Li Zhang**[1,2*]

[1]School of Data Science, Fudan University

[2]Shanghai Innovation Institute [3]University of Surrey

https://github.com/LogosRoboticsGroup/UniMotion

Abstract

Motion simulation, prediction and planning are foundational tasks in autonomous driving, each essential for modeling and reasoning about dynamic traffic scenarios. While often addressed in isolation due to their differing objectives, such as generating diverse motion states or estimating optimal trajectories, these tasks inherently depend on shared capabilities: understanding multi-agent interactions, modeling motion behaviors, and reasoning over temporal and spatial dynamics. Despite this underlying commonality, existing approaches typically adopt specialized model designs, which hinders cross-task generalization and system scalability. More critically, this separation overlooks the potential mutual benefits among tasks. Motivated by these observations, we propose **UniMotion**, a unified motion framework that captures shared structures across motion tasks while accommodating their individual requirements. Built on a decoder-only Transformer architecture, UniMotion employs dedicated interaction modes and tailored training strategies to simultaneously support these motion tasks. This unified design not only enables joint optimization and representation sharing but also allows for targeted fine-tuning to specialize in individual tasks when needed. Extensive experiments on the Waymo Open Motion Dataset demonstrate that joint training leads to robust generalization and effective task integration. With further fine-tuning, UniMotion achieves state-of-the-art performance across a range of motion tasks, establishing it as a versatile and scalable solution for autonomous driving.

1 Introduction

Motion understanding is a cornerstone of autonomous driving, supporting essential capabilities such as interactive behavior modeling, motion analysis, and decision-making for the ego vehicle. Core motion tasks include motion simulation, trajectory prediction, and ego planning. Among them, trajectory prediction [1, 2, 3] and ego planning [4] are critical intermediaries between perception modules and control systems in the driving pipeline, while motion simulation [5] generates rich and diverse agent behaviors, serving both development and evaluation for motion models. The complexity of real-world driving, characterized by uncertain behaviors and densely interactive environments, makes these tasks challenging.

Accordingly, recent research has produced a wide range of task-specific models to optimize performance for its designated objective. Trajectory prediction and planning models have primarily focused on comprehensive representation learning [6, 7, 8, 9], along with a growing emphasis on precise waypoint estimation [10, 11, 12, 13]. In contrast, motion simulation has leaned heavily on

*Li Zhang (lizhangfd@fudan.edu.cn) is the corresponding author.

39th Conference on Neural Information Processing Systems (NeurIPS 2025).

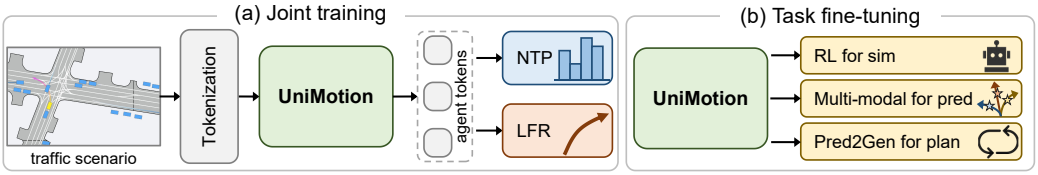


Figure 1: Overview of our UniMotion pipeline. (a) We train jointly our model with combined generative and forecasting supervision, resulting in a multi-task model that can simultaneously address all kinds of motion tasks in autonomous driving. (b) We further adopt dedicated fine-tuning strategies for each task, producing task-specific models to promote specialization.

generative modeling. Inspired by the success of Large Language Models (LLMs), recent GPT-style architectures [14, 15] have been applied to simulate agent behaviors. Furthermore, this paradigm has also been introduced into prediction [16, 17] and planning [13] tasks to achieve progressive and more accurate regression.

While these task-customized models have proven effective for different motion tasks, it is challenging to transfer models and learned knowledge across tasks. Considering that these tasks are essentially similar due to their reliance on interaction modeling and motion reasoning, our goal is to build a general model that encourages knowledge sharing among motion tasks and enables cross-task generalization. To achieve this aim, we revisit the formulation of motion tasks and propose a unified motion framework for autonomous driving. We begin by abstracting existing motion tasks into two fundamental categories: *diverse motion generation* and *long-range trajectory forecasting*, which serve as a conceptual foundation for unification. From this, we develop **UniMotion**, a unified framework based on a decoder-only Transformer, chosen for its simplicity, scalability, and strong generative capacity across tasks. By incorporating task-aware interaction modes and training strategies (see Figure 1(a)), UniMotion enables joint training across simulation, prediction and planning, fostering shared representations and efficient multi-task learning. To further enhance task proficiency, we also introduce dedicated fine-tuning strategies (see Figure 1(b)) that adapt the unified model to specific objectives, improving performance and practical deployment.

Our **contributions** are summarized as follows: **(i)** We abstract motion tasks into two core categories and propose UniMotion, a unified Transformer-based framework that jointly models simulation, prediction and planning to promote generality and cross-task knowledge sharing. **(ii)** We design effective fine-tuning strategies that specialize the jointly trained model for individual tasks, allowing the model to master task-specific expertise. **(iii)** Extensive experiments on the Waymo Open Motion Dataset (WOMD) show that UniMotion achieves competitive performance under joint training and sets new state-of-the-art results when fine-tuned for specific tasks.

2 Related work

2.1 Motion tasks in autonomous driving

Existing motion tasks in autonomous driving can be roughly categorized into three types according to their objectives: motion simulation, trajectory prediction, and ego planning. We provide a detailed discussion for each type in the following.

Motion simulation focuses on simulating diverse and complex motion patterns, augmenting training samples and offering off-board assessment for autonomous driving algorithms. To effectively measure simulation quality, Sim Agents [5] first proposes a benchmark with comprehensive metrics, expecting models to progressively evolve the motion states of all traffic agents. Motivated by the performance of LLMs, existing methods [14, 15] tokenize the agent trajectories and map elements, and employ a scalable decoder-only model to perform auto-regressive trajectory generation. Besides, CATK [18] utilizes closed-loop supervised finetuning with Closest Among Top-K rollouts, which mitigates covariate shift in open-loop behavior cloning and further improves these GPT-style methods. In contrast, UniMM [19] introduces a unified mixture model with both continuous mixture models and discrete GPT-style models to address simulation.

Trajectory prediction anticipates models to rapidly estimate the future trajectories of traffic participants, providing prior knowledge for subsequent decision-making. As a foundation task in autonomous driving, trajectory prediction has been widely explored [6, 7, 8, 11] with Transformer frameworks. Recent methods [20, 21, 22] uncover detailed temporal information and achieve more accurate prediction. Moreover, there are methods utilizing particular strategies for impressive performance improvements, such as model pre-training [23, 24] and trajectory post-refinement [10, 12].

Ego planning is responsible for giving final driving trajectories for ego vehicle according to the driving environment and upstream results. Although rule-based models [25, 26] still hold a crucial position, learning-based methods have emerged and exceeded. PlanTF [27] and PLUTO [28] effectively alleviate the limit of imitation-based planners through designed model architecture and training strategies. Besides, BeTopNet [9] further improves planning performance by explicitly representing behavioral topology among multi-agent future states.

2.2 GPT-style motion models

GPT-style LLMs have demonstrated strong capabilities in language tasks, which stimulates the emergence of similar architectures in autonomous driving [29, 30]. Given that both trajectories and text share a sequential structure, it is natural to introduce analogous modules and designs into motion models. In particular, the aforementioned simulation methods directly utilize decoder-only models with next-token prediction for simulation tasks, generating diverse multi-agent trajectory sets. There are also some prediction and planning methods adopting modified auto-regressive decoding. For example, MotionLM [17] and Trajeglish [16] introduce a causal decoder for iterative trajectory prediction, and similarly, CarPlanner [13] adopts consistent auto-regressive planning with reinforcement learning. In contrast, we unify all motion tasks under a GPT-style model, fully exploiting the capability of this framework.

3 Methodology

3.1 Preliminary

Task definition. Motion understanding in autonomous driving focuses on three tasks: simulation, prediction and planning. All these tasks share the same traffic scenario input, including historical agent states $\mathcal{A}_h$ and a high-definition (HD) map $\mathcal{M}$ of driving scenes. **Motion simulation** is typically designed to support other motion tasks. Given $\mathcal{A}_h$ as initial motion states, it aims to iteratively and concurrently simulate multi-agent states $\{\mathcal{A}_s^i\}_{i=1}^K$, where K represents the number of generated rollouts. Simulation aims to exhibit high short-term generation diversity. In contrast, **trajectory prediction** provides distant future trajectories $\mathcal{A}_f$ of traffic agents over a span of time T_f. In real-world scenarios, we expect the predictors to generate the results directly and efficiently, rather than relying on segment-by-segment auto-regressive generation [16, 17]. Accordingly, this task evaluates the ability of models to perform accurate long-range predictions. **Ego planning** aims to determine feasible trajectories for the ego vehicle by referencing surrounding agents. It can be viewed as first predicting the behaviors of other traffic participants, followed by the progressive generation of the ego vehicle's trajectory. In light of the above observations, we summarize motion tasks into two fundamental components: *diverse motion generation* and *long-range trajectory forecasting*. As motivated earlier, we strive to integrate these tasks into a unified framework and jointly develop both capabilities in a single model, thereby enhancing overall motion understanding and fostering mutual complementarity between tasks.

Input representation. To ensure the generalization of representations across tasks, we standardize the input format via tokenization following [14, 15]. Concretely, the trajectories of all agents are segmented at fixed time intervals and normalized to form a trajectory set, which is then clustered to construct the agent token vocabulary $\mathcal{S}_a$. Similarly, map tokens $\mathcal{S}_m$ are generated by dividing original map polylines into fixed-length segments followed by clustering. The corresponding agent token embeddings $E_{sa} \in \mathbb{R}^{K_a \times C}$ and map token embeddings $E_{sm} \in \mathbb{R}^{K_m \times C}$ obtained by applying MLP modules to $\mathcal{S}_a$ and $\mathcal{S}_m$, respectively, where $K_{(\cdot)}$ and C are the number of tokens and the dim of feature embeddings. Subsequently, the agent states $\mathcal{A}$ and a HD map $\mathcal{M}$ are tokenized based on geometric similarity of segments, forming the input agent embeddings $E_a \in \mathbb{R}^{N_a \times T \times C}$ and map

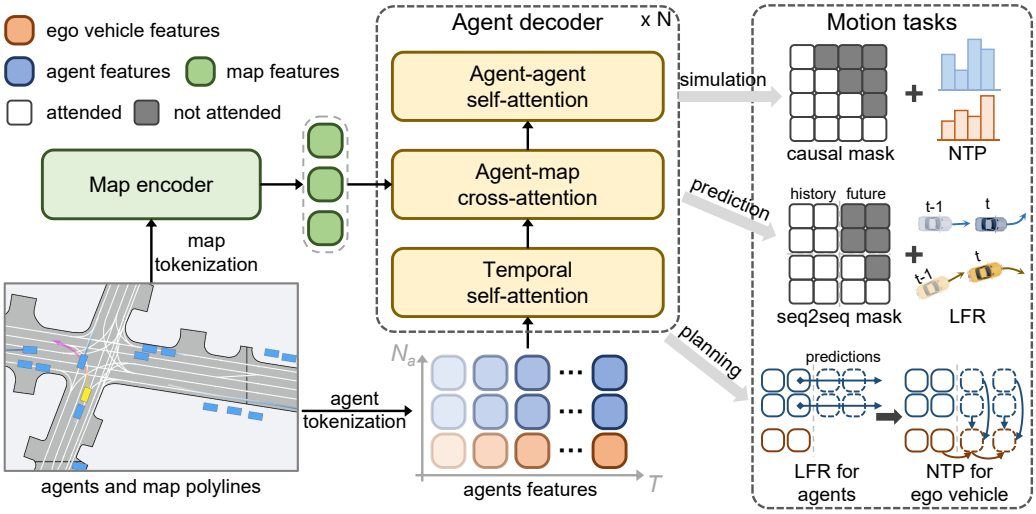


Figure 2: Overview of UniMotion architecture. It takes as input tokenized agent trajectories and map polylines, and adopts a decoder-only structure equipped with an additional map encoder to yield motion states. By employing task-specific attention masks and training objectives, UniMotion is empowered to flexibly and effectively address multiple motion tasks simultaneously.

embeddings $E_m \in \mathbb{R}^{N_m \times C}$, where N_a and N_m denote the number of agents and map segments, and T denotes the time frames.

3.2 UniMotion for multiple motion tasks

As depicted in Figure 2, the overall design of our ***UniMotion*** follows a decoder-only architecture for agent trajectory generation and prediction, with an additional map context encoder. For each specific motion task, we introduce dedicated interactive patterns and training objectives for specialization, but enable weight sharing to learn general representations.

3.2.1 Scene context encoding and motion decoding

To ensure the robustness and generalizability of the model, we employ the Transformer architecture with relative positional embedding following [8, 14, 15], which remains invariant under coordinate transformations. Specifically, we first stack several self-attention modules to encode map embeddings, yielding map features F_m. Then, agent embeddings E_a, along with F_m, are further delivered to a agent Transformer decoder constructed with factorized attentions, which are composed of temporal self-attention, agent-map cross-attention and agent-agent self-attention. This can be formulated as:

$$\begin{aligned} F_a &= \mathrm{MHSA}_t(\mathrm{Q}=F_a,\, \mathrm{K}=\mathrm{V}=F_a+\mathcal{R}_t(P_a)), \\ F_a &= \mathrm{MHCA}_{a-m}(\mathrm{Q}=F_a,\, \mathrm{K}=\mathrm{V}=F_m+\mathcal{R}_{a-m}(P_a,\, P_m)), \\ F_a &= \mathrm{MHSA}_{a-a}(\mathrm{Q}=F_a,\, \mathrm{K}=\mathrm{V}=F_a+\mathcal{R}_{a-a}(P_a)), \end{aligned} \tag{1}$$

where $\mathcal{R}$ is the relative positional embedding for spatial and temporal position information $P_{(\cdot)}$, and the initial F_a is directly derived from E_a. This decoder module enables the thorough interaction among agents and between agents and road elements, finally giving step-wise motion states.

3.2.2 Joint training across motion tasks

Following the designs of multi-task language models, we apply customized interactive attention masks for different motion tasks and introduce tailored training objectives to improve both step-wise generation and long-range prediction. At inference time, we adaptively leverage the model's capabilities according to the specific task.

Motion simulation. The motion simulation task closely resembles language generation, which aims to iteratively select suitable tokens from motion vocabulary. Accordingly, we adopt a causal attention mask and employ Next-Token Prediction (NTP) as the learning strategy, modeling the generation of

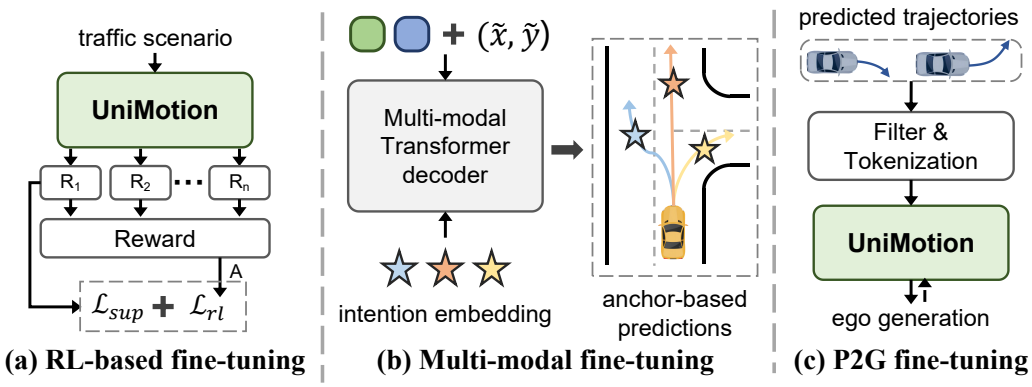


Figure 3: Illustration of task-specific fine-tuning strategies. We adopt (a) **RL-based fine-tuning** to improve simulation likelihood while maintaining closed-loop consistency. (b) **Multi-modal fine-tuning** is introduced to refine the multi-modal behavior of predictions. With reference to the planning inference, we utilize (c) **Pred2Gen fine-tuning** to address distribution mismatch.

the next agent token $\mathcal{A}_t$ at step t from the conditional probability $P(\mathcal{A}_t \mid \mathcal{A}_{<t})$. During inference, motion processes for all agents are simultaneously simulated through auto-regressive generation.

Trajectory prediction. We regard trajectory prediction as a sequence-to-sequence task, which maps historical segments to future trajectories. Tokens within the historical segments are allowed to attend only to each other, while future tokens can attend both to preceding tokens within the future segment and to all tokens in the historical segments. Moreover, we propose Long-range Future Regression (LFR) to meet the strict requirements of prediction task on accuracy. Specifically, we replace token classification with normalized trajectory regression, and it predicts a complete long-range future trajectory instead of short segments with token-level length, in which case future trajectories $\mathcal{A}_f$ can be derived from:

$$\mathcal{A}_f = \mathrm{MLP}(F_a),\ \mathcal{A}_f \in \mathbb{R}^{N_a \times T \times T_f \times 2}. \tag{2}$$

Notably, all these tokens are utilized for dense supervision in the training phase, while we only need to perform single-pass reasoning at current step for inference.

Ego planning. As mentioned in Section 3.1, we consider planning task to be the combination of generation and prediction. Hence, the training objectives for simulation and prediction naturally benefit the planning task, without requiring any additional training strategies. After training, we adopt a two-stage inference approach for ego trajectory generation. Concretely, we first jointly predict future trajectories for all traffic participants, and then tokenize the future parts of these predictions to form the future tokens. Given the predicted future tokens of surrounding agents, we then progressively generate the planning trajectory of ego vehicle.

3.2.3 Task-specific fine-tuning strategies

Despite the capability of solving multiple tasks simultaneously, models are typically expected to exhibit higher specialization for individual tasks in real-world applications. To this end, we introduce task-specific fine-tuning strategies to further enhance the specialization of UniMotion. Overall, we prefer not to introduce additional parameters during fine-tuning stage. According to this principle, we utilize reinforcement learning-based fine-tuning for simulation and two-stage pred2gen finetuning for planning. Additionally, trajectory prediction requires to produce multi-modal outputs, which will significantly increase the computational burden under dense supervision. To alleviate this, we employ a multi-modal decoder to support the prediction process as a compromise.

In line with the purpose of Reinforcement Learning from Human Feedback (RLHF) in LLMs, we enhance the model to generate trajectories that align with human driving behavior and traffic rules, which cannot be constrained through direct supervision alone. Hence, we introduce RL-based fine-tuning for simulation. As shown in Figure 3(a), we first generate n rollouts of driving scenes as a group following GRPO [31], with activating the gradient for only one rollout. Subsequently,

rewards R are computed using the same algorithms as the metrics in Sim Agents [5], which consist of kinematic reward and collision reward for simplicity. We formulate this as:

$$R = \exp(\frac{1}{T}\sum_{i=1}^{T}\log q^{\text{gt}}) + \mathbf{1}\{\text{not collision}\}, \tag{3}$$

where the kinematic reward is defined as the exponentiated log-likelihood of ground truth samples under the distribution induced by corresponding generated trajectories, whereas the collision reward is an indicator function. Then, the advantage A is derived from the normalized rewards within the group of each agent. Notably, we only supervise one rollout to reduce computational overhead. More details are provided in Section 3.3 and supplementary materials.

To improve multi-modality of predicted trajectories, we introduce a lightweight transformer decoder similar to [11, 32] as illustrated in Figure 3(b). At this stage, we only focus on the agents of interest that are utilized for evaluation, and perform prediction within perspective local system. To this end, the coordinate-invariant features are transformed into the local agent features F_a^* and map features F_m^* through adding the local normalized position content encoded by MLP layer. For each interested agent, the estimated multi-modal trajectories $a_f^* \in \mathbb{R}^{N_{mo}\times T_f\times 2}$ for N_{mo} modes can be derived by:

$$a_f^* = \text{Transformer}(\text{Q} = F_{mo},\ \text{K} = \text{V} = \{F_a^*, F_m^*\}), \tag{4}$$

where F_{mo} denotes the mode features encoded from intention points, and F_a^* and F_m^* are with respect to current interested agent. Benefiting from distinct intention points, our model can provide more diverse trajectory choices after fine-tuning.

During planning inference, the predicted trajectory tokens for surrounding agents will participate in and significantly affect ego generation, inconsistent with the independent training process. To alleviate potential distribution mismatch, we further fine-tune ego generation using the predicted results. As shown in Figure 3(c), the incorrectly predicted trajectories are first filtered out and replaced by the ground truth. Then, we tokenize all predictions and feed them into our network for ego generation, which is fine-tuned with end-to-end supervision.

3.3 Model training

During the joint training process, we mainly adopt NTP classification loss $\mathcal{L}_{\text{ntp}}$ between next tokens $\mathcal{A}_s$ and corresponding ground truth $\hat{\mathcal{A}}_s$ for generation supervision, and LFR regression loss $\mathcal{L}_{\text{lfr}}$ between $\mathcal{A}_f$ and ground truth $\hat{\mathcal{A}}_f$ for trajectory prediction supervision. We combine these two individual losses with equal weights to form the overall loss $\mathcal{L}$, formulated as follows:

$$\mathcal{L} = \mathcal{L}_{\text{ntp}}(\mathcal{A}_s, \hat{\mathcal{A}}_s) + \mathcal{L}_{\text{lfr}}(\mathcal{A}_f, \hat{\mathcal{A}}_f), \tag{5}$$

where we employ the cross-entropy loss and the smooth-L1 loss for $\mathcal{L}_{\text{ntp}}$ and $\mathcal{L}_{\text{lfr}}$, respectively.

During the fine-tuning stage, we employ different supervision strategies for each motion task. For simulation fine-tuning, we follow [33] to simplify the GRPO but still preserve the consistency constraint through closed-loop supervision on the gradient-activated rollout $\mathcal{A}_r \in \mathbb{R}^{N_a\times T}$. Besides, in light of efficiency concerns, we only use $\mathcal{A}_r$ rather than all rollouts to perform a single policy update per iteration. The simulation fine-tuning loss is:

$$\mathcal{L}_{\text{ft-sim}} = \mathcal{L}_{\text{ce}}(\mathcal{A}_r, \hat{\mathcal{A}}_r) + w_{\text{s}}\mathcal{L}_{\text{p}}(\mathcal{A}_r, A), \tag{6}$$

where $\mathcal{L}_{\text{ce}}$ is the cross-entropy loss between $\mathcal{A}_r$ and ground truth $\hat{\mathcal{A}}_r$, $\mathcal{L}_{\text{p}}$ is the policy utilized in GRPO with taking $\mathcal{A}_r$ and advantage A as inputs, and w_s is a balancing factor. The prediction loss consists of Gaussian NLL loss $\mathcal{L}_{\text{nll}}$ to supervise multi-modal trajectories a_f^* and cross-entropy loss for mode classification. Meanwhile, the model also learns to regress the local future trajectories $\mathcal{A}_f^*$ for other agents as auxiliary supervision. The loss can be formulated as:

$$\mathcal{L}_{\text{ft-pred}} = w_{\text{pr}}\mathcal{L}_{\text{lfr}}(\mathcal{A}_f^*, \hat{\mathcal{A}}_f^*) + \mathcal{L}_{\text{nll}}(a_f^*, \hat{a}_f^*) + \mathcal{L}_{\text{ce}}. \tag{7}$$

In addition, we simply supervise the generation a_{ego} for ego vehicle and the predictions $\mathcal{A}_f^{\text{surr}}$ for surrounding agents in the planning fine-tuning stage as:

$$\mathcal{L}_{\text{ft-plan}} = w_{\text{pl}}\mathcal{L}_{\text{lfr}}(\mathcal{A}_f^{\text{surr}}, \hat{\mathcal{A}}_f^{\text{surr}}) + \mathcal{L}_{\text{ntp}}(a_{\text{ego}}, \hat{a}_{\text{ego}}). \tag{8}$$

Table 1: Performance comparison on 2025 Waymo Open Sim Agents Challenge.

Method	Realism Meta Metric ↑	Kinematic Metrics ↑	Interactive Metrics ↑	Map-based Metrics ↑	minADE ↓
UniTAM	0.7698	0.4869	0.7876	0.9085	1.3940
UniTFormer	0.7776	0.4892	0.7997	0.9140	1.3592
LLM2AD	0.7779	0.4846	0.8048	0.9109	**1.2827**
UniMM [34]	0.7829	0.4914	0.8089	0.9161	1.2949
CATK [18]	0.7846	0.4931	**0.8106**	0.9177	1.3065
UniMotion	**0.7851**	**0.4943**	0.8105	**0.9187**	1.3036

Table 2: Performance comparison on WOMD Prediction Leaderboard. For each metric, the best and second best results are highlighted in **bold** and underline, respectively.

Method	minADE ↓	minFDE ↓	MR ↓	mAP ↑	Soft mAP ↑
HDGT [35]	0.5933	1.2055	0.1854	0.3577	0.3709
MTR [11]	0.6050	1.2207	0.1351	0.4129	0.4216
MTR++ [36]	0.5906	1.1939	0.1298	0.4329	0.4414
MGTR [37]	0.5918	1.2135	0.1298	0.4505	0.4599
EDA [32]	**0.5718**	1.1702	0.1169	0.4487	0.4596
ControlMTR [38]	0.5897	1.1916	0.1282	0.4414	0.4572
RMP-YOLO [39]	0.5737	1.1697	**0.1160**	0.4523	**0.4673**
UniMotion	**0.5718**	**1.1643**	0.1162	**0.4534**	0.4642

4 Experiments

4.1 Experimental settings

Datasets and metrics. To evaluate the performance of our method, we conduct extensive experiments on the Waymo Open Motion Dataset, comprising 486,995 training scenarios, 44,097 validation scenarios and 44,920 testing scenarios. Each scenario has an observation window of 9.1 seconds, consisting of 91 frames sampled at 10 Hz. Given 11-frame historical agent states and a HD map, the motion models are tasked with generating or predicting the future states of 80 frames. Additionally, prediction and planning tasks only perform one-step inference at the current time, while simulation task requires 32 simulations per scenario.

We employ respective benchmark metrics to assess our model. For simulation, the core metric is the Realism Meta Metric, which is utilized to measure the similarity between the simulations and the real-world distribution. It is calculated by the weighted sum of Kinematic Metrics for motion features, Interactive Metrics to capture the interaction among agents and Map-based Metrics to test the agent behavior on the map. For prediction, 6 predicted trajectories for each interested agent are expected for evaluation, with metrics including minADE (minimum Average Displacement Error), minFDE (minimum Final Displacement Error), Miss Rate, mAP and Soft MAP. In addition, due to the absence of an official benchmark for planning, we adopt the metrics proposed by [40] as measures.

Implementation details. We tokenize the trajectories and map elements following [15, 18], with 2,048 tokens for agents and 1,024 tokens for map. During the training phase, We train our models using the AdamW [41] optimizer with a total batch size of 48 on 8 NVIDIA A6000 GPUs for 30 epochs, with the weight decay set to 0.01. We initialize the learning rate of 5×10^{-4} , which is then decayed following a cosine annealing schedule. Regarding the fine-tuning stage, we employ the same learning rate for prediction, while a lower learning rate of 5×10^{-5} for simulation and planning. Beside, we also fine-tune these tasks with the same epochs.

Each agent token covers 0.5 seconds, and each map token covers about 0.5 meters. After tokenization, the original 91-step trajectories in Waymo are compressed into 18 frames. For the training phase, the training time for full data is approximately 3.5 days, while using 20% of the data takes around 1 day in our experimental settings. In addition, we set the balance factor w_{s}, w_{pr} and w_{pl} to 0.1, 0.1 and 0.5, respectively.

Table 3: Performance comparison for Waymo planning.

Method	Collision ↓	Red light ↓	Off route ↓	Planning error ↓		
				@1s	@3s	@5s
IL	5.469	2.772	4.816	0.175	1.416	4.194
IL+Prediction	3.930	1.670	4.542	0.127	0.892	2.901
Sep. Plan+Pred.	1.813	1.327	0.527	0.238	1.251	3.466
DIPP [40]	1.802	**1.235**	0.506	0.227	1.187	3.335
UniMotion	**1.565**	1.309	**0.477**	**0.083**	**0.591**	**2.246**

4.2 Comparison with state of the art

We compare the performance of our UniMotion with top-ranked competitors on Sim Agents [5], WOMD Prediction and Waymo Planning [40] benchmarks. The leaderboards for Sim Agents and WOMD Prediction are presented in Table 1 and Table 2, respectively. For simulation, our method achieve the best score of 78.51 on Realism Meta Metric. Benefiting from the kinematic reward of fine-tuning, the model is encouraged to generate trajectories with higher motion similarity to real-world situations, driving better Kinematic Metrics than other methods. However, the minADE metric exhibits relatively poorer performance, which can be attributed to the tokenization-based design's stronger emphasis on generation diversity at the cost of slightly reduced prediction accuracy. The leaderboard of the prediction task demonstrate that our method achieves promising results across almost all metrics, except for Soft mAP. Regarding the planning task, UniMotion has far outperformed most of previous approaches as depicted in Table 3, showing that our method stands distinctly ahead of others in terms of planning error. Moreover, to ensure generality and consistency, we supervise model training solely using logged trajectories. This mechanism leads to suboptimal performance in reacting to red lights, which can be mitigated by incorporating traffic rule constraints, as in [40].

4.3 General ablation study

We conduct ablation studies on the WOMD validation split to examine the effectiveness of each component in UniMotion. For efficiency, we use only 20% of the training data for ablation studies, while adopting the default experimental settings following Section 4.1 for all other aspects. We focus on the core modules here, while leaving task-specific ablation to the supplementary materials.

Effects of NTP and LFR. As shown in Table 4, we simply assess the effectiveness of NTP and LFR in our network on simulation and prediction tasks. The first and second rows demonstrate the results of training UniMotion using only NTP or LFR, respectively. Despite the absence of dedicated supervision for another one of tasks, we report all relevant metrics to compare with joint training and showcase the model's ability to handle multiple tasks. It can be observed that the model benefits from both NTP and LFR in generation and prediction, while it performs poorly on another task. In the third row, we implement a joint training strategy with these two types of supervision, simultaneously endowing the model with diverse generative and long-range predictive capabilities. Besides, we find that joint training further improves prediction accuracy compared to using LFR alone, which we attribute to the diverse generation directions serving as targets to guide the prediction process.

Table 4: Ablation on joint training with NTP and LFR. "Kin."/"Inter.": Kinematic/Interactive metrics.

NTP	LFR	Simulation				Prediction		
		Kin.	Inter.	Map	minADE	minADE	minFDE	mAP
✓		0.4884	0.7961	**0.9148**	1.4074	0.7697	1.5547	0.2629
	✓	0.4401	0.7742	0.8949	1.2965	0.6668	1.3613	0.2935
✓	✓	**0.4892**	**0.7968**	0.9144	**1.4011**	**0.6508**	**1.3296**	**0.3147**

Effects of task-specific fine-tuning. After training a preliminary model with a joint strategy, we further tailor it with designs specific to the target task. As demonstrated in Table 5, fine-tuning methods can intuitively bring consistent improvements across all motion tasks. For simulation, all metrics exhibit varying degrees of gains after fine-tuning, especially in the reward items. The 3rd and

4th rows of prediction task show that the mAP metric has been remarkably enhanced by introducing an extra anchor-based decoder, which can cover more potential future trajectories. In addition, noticeable improvements are also observed on the minADE and MR metrics, achieving the expected goals of fine-tuning. Regrading the planning task, the 5th row shows limited performance due to the distribution mismatch between predicted tokens and original trajectory tokens. After fine-tuning, the model is adapted to the two-stage process, and there are considerable improvements for all metrics.

Table 5: Ablation on task-specific fine-tuning for simulation, prediction and planning. Notably, we selected a set of metrics with substantial variation to facilitate comparison. Additionally, we fine-tune planning model with the data following [40] instead of 20% data.

Task	ft.	Metrics			
Simulation		Kinematic ↑	Interactive ↑	Map-based ↑	minADE ↓
	✗	0.4892	0.7968	0.9144	1.4011
	✓	**0.4939**	**0.8032**	**0.9159**	**1.3904**
Prediction		minADE ↓	minFDE ↓	MR ↓	mAP ↑
	✗	0.6508	**1.3296**	0.1626	0.3147
	✓	**0.6413**	1.3368	**0.1493**	**0.3856**
Planning		Collision ↓	Err @1s ↓	Err @3s ↓	Err @5s ↓
	✗	1.7347	0.1326	0.7943	2.7809
	✓	**1.5650**	**0.0834**	**0.5908**	**2.2455**

4.4 Task-specific ablation study

In this section, we adopt the same experimental settings and conduct task-specific ablation studies on the simulation and prediction tasks to investigate the effects of different module designs.

Effects of RL-based fine-tuning for simulation. We present the effects of different constraints in Table 6. As shown in the 2nd and 3rd rows, closed-loop supervised fine-tuning leads to notable improvements, while relying solely on RL-based supervision, like DAPO [33], fails to produce satisfactory results. We attribute this to two primary factors: insufficient reward signals and limited model capacity.

Table 6: Ablation on RL-based fine-tuning . "con.": consistency constraint. "Kin."/"Inter.": Kinematic/Interactive metrics.

con.	rl	Kin.	Inter.	Map	minADE
		0.4892	0.7968	0.9144	1.4011
✓		0.4921	0.8019	0.9156	1.3933
	✓	0.4913	0.7974	0.9120	1.3967
✓	✓	**0.4939**	**0.8032**	**0.9159**	**1.3904**

Effects of fine-tuning for prediction. We present the effects of different fine-tuning strategies in Table 7. The comparison between the 2nd and 3rd rows demonstrates that although the end-to-end fine-tuning strategy (without intention anchors) can significantly reduces prediction errors, it severely compromises prediction diversity and confidence, leading to degraded mAP performance. Hence, we retain the intention anchors as part of the fine-tuning process.

Table 7: Ablation on prediction fine-tuning.

Strategy	minADE	minFDE	MR	mAP
w/o ft.	0.6508	1.3296	0.1626	0.3147
ft. w/o intention	**0.6174**	**1.2639**	**0.1313**	0.3389
ft. w/ intention	0.6413	1.3368	0.1493	**0.3856**

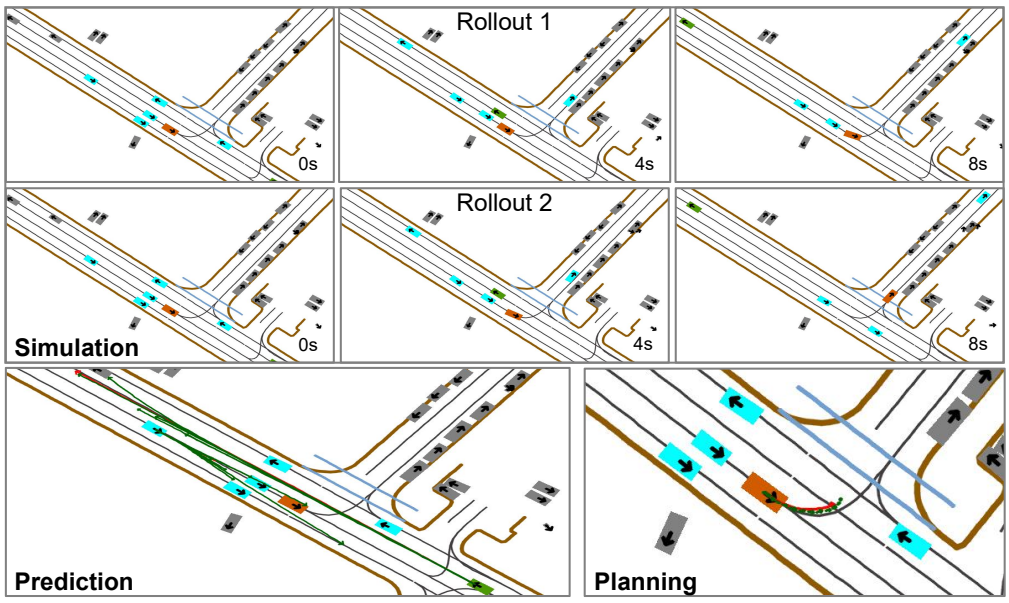


Figure 4: Qualitative results on WOMD validation split. The top and bottom rows present two rollouts of simulation and the results of prediction and planning, respectively. The red and green arrowed curves are ground truth and predicted trajectories.

4.5 Qualitative results

In Figure 4, we present qualitative results of our network on three motion tasks. The first two panels illustrate the diverse simulation results of all traffic agents at 0s, 4s, and 8s, respectively. The bottom-left figure depicts the multi-modal predictions for interested agents,while the bottom-right figure presents the ego trajectory generation process in a token-by-token manner. These qualitative results demonstrate our network's capability in handling multiple motion tasks.

5 Conclusion

In this work, we abstract mainstream motion tasks into two core types and aim to address all motion tasks within a unified framework, thereby promoting knowledge sharing and enhancing model generalization. Achieving this, we present UniMotion, an integrated model designed particularly to support joint training for these tasks. The critical components of our framework are distinct interactive modes and training strategies. Besides, we further perform task-specific fine-tuning for each task, fulfilling the requirement for expert-level models in real-world applications. Our extensive experiments on several task benchmarks comprehensively demonstrate that UniMotion is capable of unifying motion tasks, and the fine-tuned versions surpass the current state-of-the-art performance in the respective tasks.

Limitations and future work. Although we employ a decode-only framework to ensure flexibility, the coupled spatial and temporal interaction and relative position embedding rely on detailed positional information of traffic elements, constraining the model from leveraging more LLM-related technologies. Moreover, after applying tokenization and conducting joint multi-task training, it is worthwhile to explore cross-dataset learning, which could lead to substantial improvements in joint training performance. Limited by computational resources and training time, we have not explored this technique in the present work.

Acknowledgments

This work was supported in part by National Natural Science Foundation of China (Grant No. 62376060).

References

[1] Scott Ettinger, Shuyang Cheng, Benjamin Caine, Chenxi Liu, Hang Zhao, Sabeek Pradhan, Yuning Chai, Ben Sapp, Charles R Qi, Yin Zhou, et al. Large scale interactive motion forecasting for autonomous driving: The waymo open motion dataset. In *ICCV*, 2021. 1, 13

[2] Ming-Fang Chang, John Lambert, Patsorn Sangkloy, Jagjeet Singh, Slawomir Bak, Andrew Hartnett, De Wang, Peter Carr, Simon Lucey, Deva Ramanan, et al. Argoverse: 3d tracking and forecasting with rich maps. In *CVPR*, 2019. 1

[3] Benjamin Wilson, William Qi, Tanmay Agarwal, John Lambert, Jagjeet Singh, Siddhesh Khandelwal, Bowen Pan, Ratnesh Kumar, Andrew Hartnett, Jhony Kaesemodel Pontes, Deva Ramanan, Peter Carr, and James Hays. Argoverse 2: Next generation datasets for self-driving perception and forecasting. In *NeurIPS*, 2021. 1

[4] Holger Caesar, Juraj Kabzan, Kok Seang Tan, Whye Kit Fong, Eric Wolff, Alex Lang, Luke Fletcher, Oscar Beijbom, and Sammy Omari. nuplan: A closed-loop ml-based planning benchmark for autonomous vehicles. *arXiv preprint*, 2021. 1

[5] Nico Montali, John Lambert, Paul Mougin, Alex Kuefler, Nicholas Rhinehart, Michelle Li, Cole Gulino, Tristan Emrich, Zoey Yang, Shimon Whiteson, et al. The waymo open sim agents challenge. In *NeurIPS*, 2023. 1, 2, 6, 8, 13

[6] Jiyang Gao, Chen Sun, Hang Zhao, Yi Shen, Dragomir Anguelov, Congcong Li, and Cordelia Schmid. Vectornet: Encoding hd maps and agent dynamics from vectorized representation. In *CVPR*, 2020. 1, 3

[7] Zikang Zhou, Luyao Ye, Jianping Wang, Kui Wu, and Kejie Lu. Hivt: Hierarchical vector transformer for multi-agent motion prediction. In *CVPR*, 2022. 1, 3

[8] Zikang Zhou, Jianping Wang, Yung-Hui Li, and Yu-Kai Huang. Query-centric trajectory prediction. In *CVPR*, 2023. 1, 3, 4

[9] Haochen Liu, Li Chen, Yu Qiao, Chen Lv, and Hongyang Li. Reasoning multi-agent behavioral topology for interactive autonomous driving. *NeurIPS*, 2024. 1, 3

[10] Sehwan Choi, Jungho Kim, Junyong Yun, and Jun Won Choi. R-pred: Two-stage motion prediction via tube-query attention-based trajectory refinement. In *ICCV*, 2023. 1, 3

[11] Shaoshuai Shi, Li Jiang, Dengxin Dai, and Bernt Schiele. Motion transformer with global intention localization and local movement refinement. In *NeurIPS*, 2022. 1, 3, 6, 7

[12] Yang Zhou, Hao Shao, Letian Wang, Steven L Waslander, Hongsheng Li, and Yu Liu. Smartrefine: An scenario-adaptive refinement framework for efficient motion prediction. In *CVPR*, 2024. 1, 3

[13] Dongkun Zhang, Jiaming Liang, Ke Guo, Sha Lu, Qi Wang, Rong Xiong, Zhenwei Miao, and Yue Wang. Carplanner: Consistent auto-regressive trajectory planning for large-scale reinforcement learning in autonomous driving. *CVPR*, 2025. 1, 2, 3

[14] Zikang Zhou, HU Haibo, Xinhong Chen, Jianping Wang, Nan Guan, Kui Wu, Yung-Hui Li, Yu-Kai Huang, and Chun Jason Xue. Behaviorgpt: Smart agent simulation for autonomous driving with next-patch prediction. *NeurIPS*, 2024. 2, 3, 4

[15] Wei Wu, Xiaoxin Feng, Ziyan Gao, and Yuheng Kan. Smart: Scalable multi-agent real-time motion generation via next-token prediction. *NeurIPS*, 2024. 2, 3, 4, 7

[16] Jonah Philion, Xue Bin Peng, and Sanja Fidler. Trajeglish: Traffic modeling as next-token prediction. *ICLR*, 2024. 2, 3

[17] Ari Seff, Brian Cera, Dian Chen, Mason Ng, Aurick Zhou, Nigamaa Nayakanti, Khaled S Refaat, Rami Al-Rfou, and Benjamin Sapp. Motionlm: Multi-agent motion forecasting as language modeling. In *ICCV*, 2023. 2, 3

[18] Zhejun Zhang, Peter Karkus, Maximilian Igl, Wenhao Ding, Yuxiao Chen, Boris Ivanovic, and Marco Pavone. Closed-loop supervised fine-tuning of tokenized traffic models. *CVPR*, 2025. 2, 7

[19] Longzhong Lin, Xuewu Lin, Kechun Xu, Haojian Lu, Lichao Huang, Rong Xiong, and Yue Wang. Revisit mixture models for multi-agent simulation: Experimental study within a unified framework. *arXiv preprint*, 2025. 2

[20] Xiaolong Tang, Meina Kan, Shiguang Shan, Zhilong Ji, Jinfeng Bai, and Xilin Chen. Hpnet: Dynamic trajectory forecasting with historical prediction attention. In *CVPR*, 2024. 3

[21] Nan Song, Bozhou Zhang, Xiatian Zhu, and Li Zhang. Motion forecasting in continuous driving. *NeurIPS*, 2024. 3

[22] Bozhou Zhang, Nan Song, and Li Zhang. Decoupling motion forecasting into directional intentions and dynamic states. *NeurIPS*, 2024. 3

[23] Jie Cheng, Xiaodong Mei, and Ming Liu. Forecast-mae: Self-supervised pre-training for motion forecasting with masked autoencoders. In *ICCV*, 2023. 3

[24] Zhiqian Lan, Yuxuan Jiang, Yao Mu, Chen Chen, and Shengbo Eben Li. Sept: Towards efficient scene representation learning for motion prediction. In *ICLR*, 2024. 3

[25] Martin Treiber, Ansgar Hennecke, and Dirk Helbing. Congested traffic states in empirical observations and microscopic simulations. *Physical review E*, 2000. 3

[26] Daniel Dauner, Marcel Hallgarten, Andreas Geiger, and Kashyap Chitta. Parting with misconceptions about learning-based vehicle motion planning. In *CoRL*, 2023. 3

[27] Jie Cheng, Yingbing Chen, Xiaodong Mei, Bowen Yang, Bo Li, and Ming Liu. Rethinking imitation-based planners for autonomous driving. In *ICRA*, 2024. 3

[28] Jie Cheng, Yingbing Chen, and Qifeng Chen. Pluto: Pushing the limit of imitation learning-based planning for autonomous driving. *arXiv preprint*, 2024. 3

[29] Yuntao Chen, Yuqi Wang, and Zhaoxiang Zhang. Drivinggpt: Unifying driving world modeling and planning with multi-modal autoregressive transformers. *arXiv preprint*, 2024. 3

[30] Xiaosong Jia, Junqi You, Zhiyuan Zhang, and Junchi Yan. Drivetransformer: Unified transformer for scalable end-to-end autonomous driving. *ICLR*, 2025. 3

[31] Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, YK Li, Y Wu, et al. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint*, 2024. 5

[32] Longzhong Lin, Xuewu Lin, Tianwei Lin, Lichao Huang, Rong Xiong, and Yue Wang. Eda: Evolving and distinct anchors for multimodal motion prediction. In *AAAI*, 2024. 6, 7

[33] Qiying Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, Yu Yue, Tiantian Fan, Gaohong Liu, Lingjun Liu, Xin Liu, et al. Dapo: An open-source llm reinforcement learning system at scale. *arXiv preprint*, 2025. 6, 9

[34] Longzhong Lin, Xuewu Lin, Kechun Xu, Haojian Lu, Lichao Huang, Rong Xiong, and Yue Wang. Revisit mixture models for multi-agent simulation: Experimental study within a unified framework. In *arXiv preprint*, 2025. 7

[35] Xiaosong Jia, Penghao Wu, Li Chen, Yu Liu, Hongyang Li, and Junchi Yan. Hdgt: Heterogeneous driving graph transformer for multi-agent trajectory prediction via scene encoding. *IEEE TPAMI*, 2023. 7

[36] Shaoshuai Shi, Li Jiang, Dengxin Dai, and Bernt Schiele. Mtr++: Multi-agent motion prediction with symmetric scene modeling and guided intention querying. *IEEE TPAMI*, 2024. 7

[37] Yiqian Gan, Hao Xiao, Yizhe Zhao, Ethan Zhang, Zhe Huang, Xin Ye, and Lingting Ge. Multi-granular transformer for motion prediction with lidar. In *ICRA*, 2024. 7

[38] Jiawei Sun, Chengran Yuan, Shuo Sun, Shanze Wang, Yuhang Han, Shuailei Ma, Zefan Huang, Anthony Wong, Keng Peng Tee, and Marcelo H Ang. Controlmtr: Control-guided motion transformer with scene-compliant intention points for feasible motion prediction. In *IEEE ITSC*, 2024. 7

[39] Jiawei Sun, Jiahui Li, Tingchen Liu, Chengran Yuan, Shuo Sun, Zefan Huang, Anthony Wong, Keng Peng Tee, and Marcelo H Ang Jr. Rmp-yolo: A robust motion predictor for partially observable scenarios even if you only look once. *arXiv preprint*, 2024. 7

[40] Zhiyu Huang, Haochen Liu, Jingda Wu, and Chen Lv. Differentiable integrated motion prediction and planning with learnable cost function for autonomous driving. *TNNLS*, 2023. 7, 8, 9

[41] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *ICLR*, 2018. 7

A Details for simulation fine-tuning

After obtaining the generated rollout $\mathcal{A}_r$ and advantage A, we update policy gradient using simplified objective to ensure stable training of lightweight models, the gradient of which for token t can be estimated as:

$$\nabla_\theta \log \mathcal{F}_\theta(\mathcal{A}_r[t]) \cdot A_{(\mathcal{A}_r)}[t], \tag{9}$$

where $\mathcal{F}_\theta$ denotes our trainable models. Besides, considering the relatively uniform rewards and the large number of iterations, we perform policy update for only one generation per iteration to promote efficient training.

B Discussions

Why we employ decoder-only Transformers as the backbone? We opt for a decoder-only Transformer structure because it is better suited for multi-step generative tasks like simulation, where input and output lengths can vary. Conversely, the encoder-decoder structure, while excellent for sequence-to-sequence prediction tasks, cannot generalize as well across the diverse demands of our multi-task framework. Adopting a decoder-only architecture during joint training ensures stronger generalization capabilities across all our tasks.

Why the joint training in UniMotion works well? As detailed in Section 3.1, our primary objectives are NTP and LFR. Intuitively, this joint training paradigm fosters the learning of shared, rich intermediate representations across multiple tasks. This enables tokens within the model to attend simultaneously to both their immediate surroundings and more distant scene contexts, leading to a more holistic understanding of the environment.

Further, despite their apparent differences, the two task objectives possess crucial commonalities. Specifically, while LFR provides direct supervision on long-range logged trajectories, the initial motion direction embedded within these trajectories concurrently serves as the maximization objective for NTP. This underlying consistency and synergistic relationship between LFR and NTP are fundamental to ensuring stable joint training and preventing mode collapse, allowing the model to learn effectively from both detailed local cues and broad long-range dependencies.

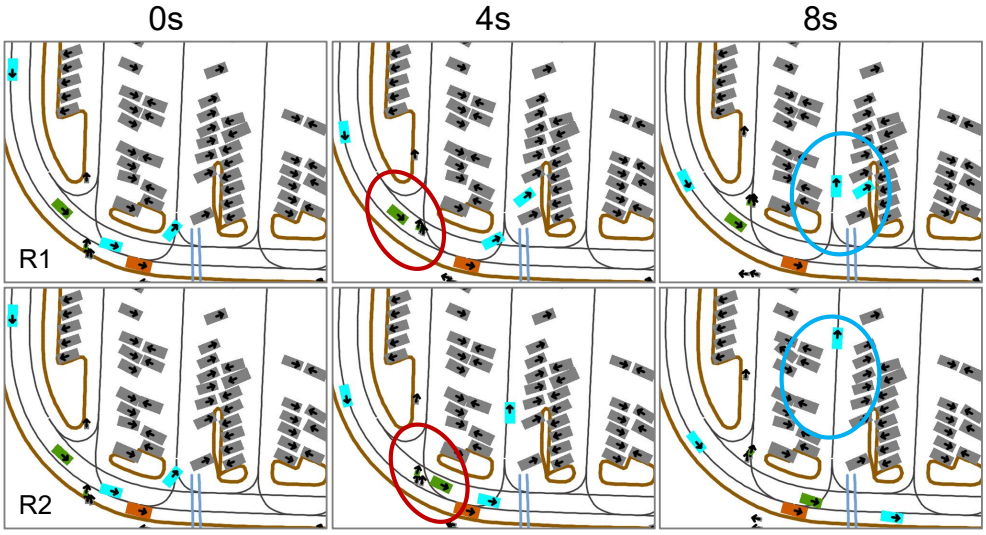
C More qualitative results

We provide more qualitative results of our framework in Figure 5 and Figure 6 on the validation set of Waymo Open Motion Dataset [1, 5].

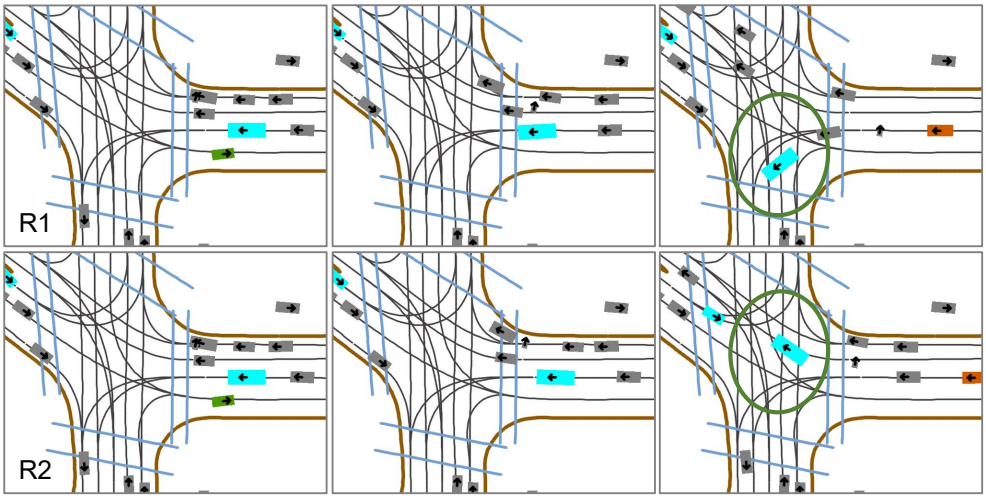
D Failure cases

Although our UniMotion achieves strong performance across multiple motion tasks, it still encounters certain failure cases. We provide qualitative results and analysis to offer a more comprehensive understanding, which we hope will support future efforts toward developing more robust and capable algorithms, as shown in Figure 7 and Figure 8.

In simulation task, there is still a possibility of collisions in complex scenario as illustrated in Figure 7 (a). Besides, we have also observed some unrealistic simulations in intersection scenarios as shown in Figure 7 (b), where agents remain stationary despite the green light. We provide the failure cases of prediction task in Figure 8 (a), which demonstrate that it is challenging to predict complex motion patterns, especially sudden turns. Similar issues are also observed in planning, particularly when dealing with sudden U-turns. Additionally, the model might yield reasonable trajectories that fail to reflect the driver's subjective intention.

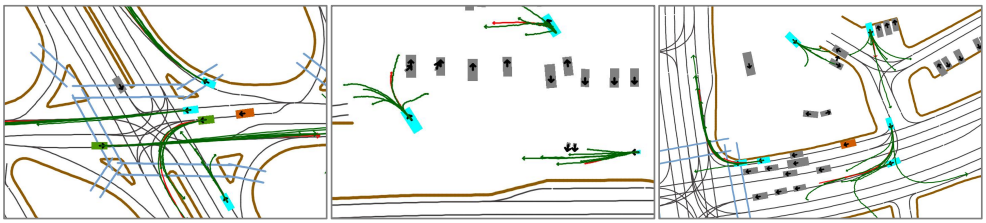


(a) Parking lot scene

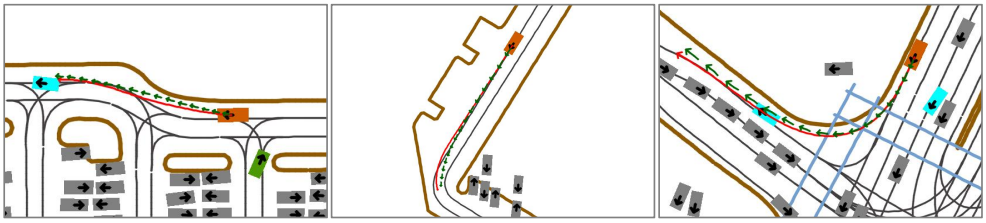


(b) Intersection scene

Figure 5: Qualitative results on the Sim Agents. We present several complex scenarios in autonomous driving, each accompanied by two rollouts. The circled areas highlight the key differences between the two rollouts.

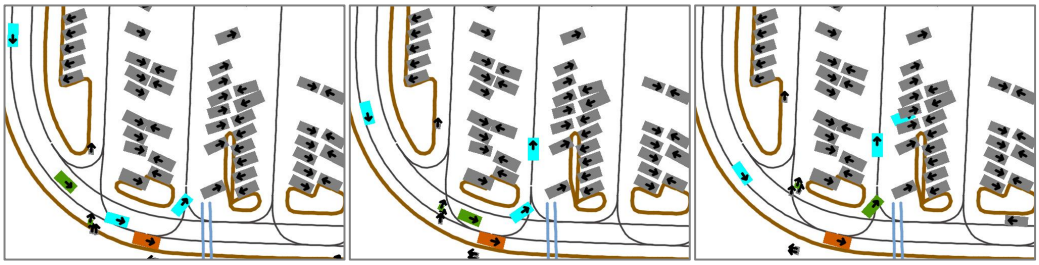


(a) Prediction

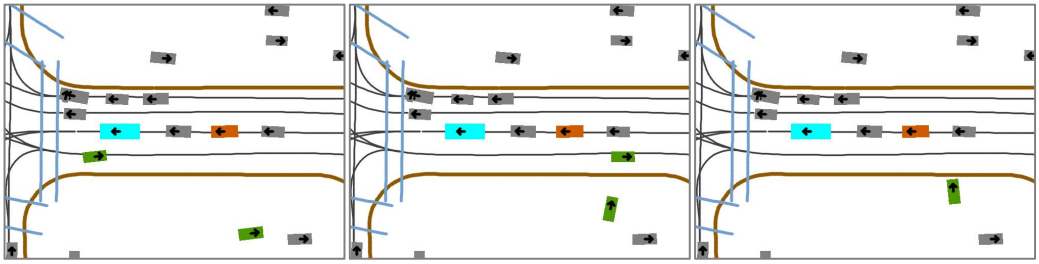


(b) Planning

Figure 6: Qualitative results on the WOMD for prediction and planning task.

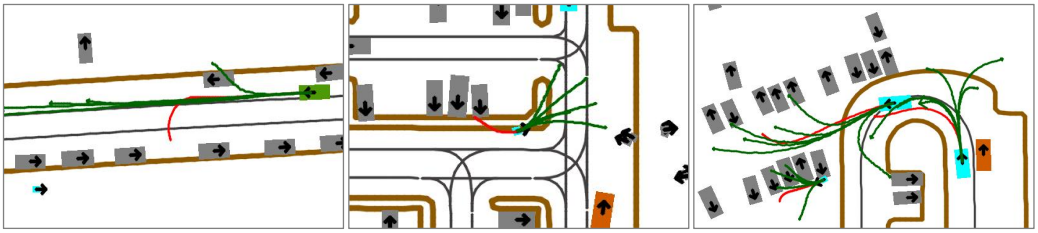


(a) Collision

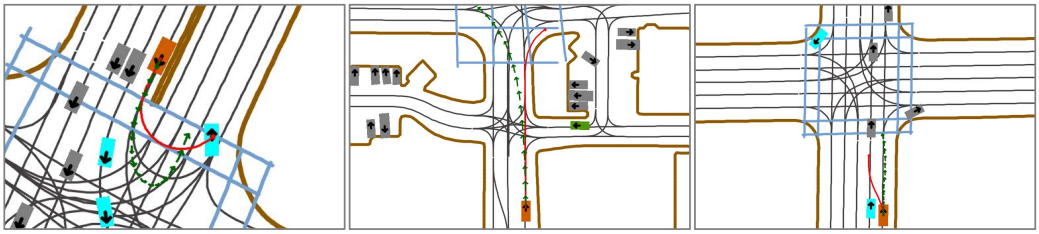


(b) Non-red-light stop

Figure 7: Failure cases in simulation. In the first row, the model causes a collision in a crowded parking lot. In the second row, it incorrectly incorrectly stop without a red light at an intersection.



(a) Prediction



(b) Planning

Figure 8: Failure cases in prediction and planning. The model fails to accurately predict the trajectories of vehicles and pedestrians. In terms of planning, it struggles with scenarios where the targets involve drivers' subjective intentions.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: Please see the experiment section.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: Please see the conclusion section.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [NA]

Justification: This paper does not include theoretical results.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: Please see the experiment section.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [No]

Justification: The details are provided in the main paper and the appendix.

Guidelines:

- The answer NA means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (https://nips.cc/public/guides/CodeSubmissionPolicy) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (https://nips.cc/public/guides/CodeSubmissionPolicy) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: Please see the experiment section and appendix.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [No]

   Justification: [NA]

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).
   - It should be clear whether the error bar is the standard deviation or the standard error of the mean.

- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: Please see the experiment section.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics https://neurips.cc/public/EthicsGuidelines?

   Answer: [Yes]

   Justification: [NA]

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: Our main idea is specifically designed for applications in autonomous driving.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
    - The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to

generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.

- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA]

    Justification: This paper poses no such risks.

    Guidelines:

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: Please see the experiment section.

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
    - If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

Answer: [NA]

Justification: This paper does not release new assets.

Guidelines:

- The answer NA means that the paper does not release new assets.
- Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
- The paper should discuss whether and how consent was obtained from people whose asset is used.
- At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

Answer: [NA]

Justification: This paper does not involve crowdsourcing nor research with human subjects.

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

Answer: [NA]

Justification: This paper does not involve crowdsourcing nor research with human subjects.

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorousness, or originality of the research, declaration is not required.

Answer: [NA]

Justification: [NA]

Guidelines:

- The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy (https://neurips.cc/Conferences/2025/LLM) for what should or should not be described.