AdvDO: Realistic Adversarial Attacks for Trajectory Prediction

Yulong Cao[1,2⋆], Chaowei Xiao[2], Anima Anandkumar[2,3], Danfei Xu[2], and Marco Pavone[2,4]

[1] University of Michigan, Ann Arbor
[2] NVIDIA
[3] California Institute of Technology
[4] Stanford University

Abstract. Trajectory prediction is essential for autonomous vehicles (AVs) to plan correct and safe driving behaviors. While many prior works aim to achieve higher prediction accuracy, few study the adversarial robustness of their methods. To bridge this gap, we propose to study the adversarial robustness of data-driven trajectory prediction systems. We devise an optimization-based adversarial attack framework that leverages a carefully-designed *differentiable dynamic model* to generate realistic adversarial trajectories. Empirically, we benchmark the adversarial robustness of state-of-the-art prediction models and show that our attack increases the prediction error for both general metrics and planning-aware metrics by more than 50% and 37%. We also show that our attack can lead an AV to drive off road or collide into other vehicles in simulation. Finally, we demonstrate how to mitigate the adversarial attacks using an adversarial training scheme[1].

Keywords: Adversarial Machine learning, Trajectory Prediction, Autonomous Driving

1 Introduction

Trajectory forecasting is an integral part of modern autonomous vehicle (AV) systems. It allows an AV system to anticipate the future behaviors of other nearby road users and plan its actions accordingly. Recent data-driven methods have shown remarkable performances on motion forecasting benchmarks [1–7]. At the same time, for a safety-critical system like an AV, it is as essential for its components to be high-performing as it is for them to be reliable and robust. But few existing work have considered the robustness of these trajectory prediction models, especially when they are subject to deliberate adversarial attacks.

A typical adversarial attack framework consists of a threat model, i.e., a function that generates "realistic" adversarial samples, adversarial optimization objectives, and ways to systematically determine the influence of the attacks.

⋆ This work was done during an internship at NVIDIA

1 Our project website is at https://robustav.github.io/RobustPred

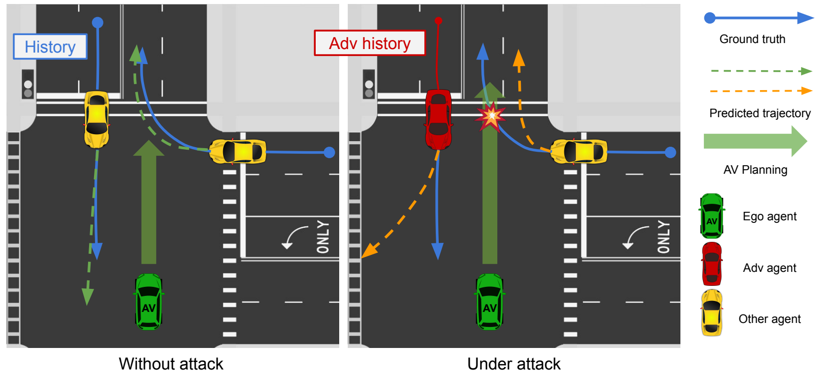


Fig. 1: An example of attack scenarios on trajectory prediction. By driving along the crafted adversarial history trajectory, the adverial agent misleads the prediction of the AV systems for both itself and the other agent. As a consequence, the AV planning based on the wrong prediction results in a collision.

However, a few key technical challenges remain in devising such a framework for attacking trajectory prediction models.

First, the threat model must synthesize adversarial trajectory samples that are 1) feasible subject to the physical constraints of the real vehicle (i.e. dynamically feasible), and 2) close to the nominal trajectories. The latter is especially important as a large alteration to the trajectory history conflates whether the change in future predictions is due to the vunerability of the prediction model or more fundamental changes to the meaning of the history. To this front, we propose an attack method that uses a carefully designed *differentiable dynamic model* to generate adversarial trajectories that are both effective and realistic. Furthermore, through a gradient-based optimization process, we can generate adversarial trajectories efficiently and customize the adversarial optimization objectives to create different safety-critical scenarios.

Second, not all trajectory prediction models react to attacks the same way. Features that are beneficial in benign settings may make a model more vulnerable to adversarial attacks. We consider two essential properties of modern prediction models: (1) motion property, which captures the influence of past agent states over future states; and (2) social property, which captures how the state of each agent affects others. Existing prediction models have proposed various architectures to explicitly models these properties either in silo [3] or jointly [4]. Specifically, we design an attack framework that accounts for the above properties. We show that our novel attack framework can exploit these design choices. As illustrated in Figure 1, by only manipulating the history trajectory of the adversarial agent, we are able to mislead the predicted future trajectory for the adversarial agent (i.e. incorrect prediction for left turning future trajectory of red car in Figure 1-right). Furthermore, we are able to mislead the prediction for *other* agent's behavior (i.e. turning right to turning left for the yellow car in Figure 1-right). During the evaluation, we could evaluate these two goals respectively. It helps us fine-grained diagnose vulnerability of different models.

Finally, existing prediction metrics such as average distance error (ADE) and final distance error (FDE) only measure errors of average cases and are thus too coarse for evaluating the effectiveness of adversarial attacks. They also ignore the influence of prediction errors in downstream planning and control pipelines in an AV stack. To this end, we incorporate various metrics with semantic meanings such as *off-road rates*, *miss rates* and *planning-aware metrics* [8] to systematically quantify the effectiveness of the attacks on prediction. We also conduct end-to-end attack on a prediction-planning pipeline by simulating the driving behavior of an AV in a close-loop manner. We demonstrate that the proposed attack can lead to both emergency brake and various of collisions of the AV.

We benchmark the adversarial robustness of state-of-the-art trajectory prediction models [4, 3] on the nuScenes dataset [9]. We show that our attack can increase prediction error by 50% and 37% on general metrics and planning-aware metrics, respectively. We also show that adversarial trajectories are realistic both quantitatively and qualitatively. Furthermore, we demonstrate that the proposed attack can lead to severe consequences in simulation. Finally, we explore the mitigation methods with adversarial training using the proposed adversarial dynamic optimization method (AdvDO). We find that the model trained with the dynamic optimization increase the adversarial robustness by 54%.

2 Related works

Trajectory Prediction. Modern trajectory prediction models are usually deep neural networks that take state histories of agents as input and generate their plausible future trajectories. Accurately forecasting multiagent behaviors requires modeling two key properties: (1) motion property, which captures the influence of past agent states over future states; (2) social property, which captures how the state of each agent affects others. Most prior works model the two properties separately [2, 3, 10, 11, 7]. For example, a representative method Trajactron++ [3] summarizes temporal and inter-agent features using a time-sequence model and a graph network, respectively. But modeling these two properties in silo ignores dependencies across time and agents. A recent work Agentformer [4] introduced a joint model that allows an agent's state at one time to directly affect another agent's state at a future time via a transformer model.

At the same time, although these design choices for modeling motion and social properties may be beneficial in benign cases, they might affect a model's performance in unexpected ways when under adversarial attacks. Hence we select these two representative models [3, 4] for empirical evaluation.

Adversarial Traffic Scenarios Generation. Adversarial traffic scenario generation is to synthesize traffic scenarios that could potentially pose safety risks[12–16]. Most prior approaches fall into two categories. The first aims to capture traffic scenarios distributions from real driving logs using generative models and sample adversarial cases from the distribution. For example, STRIVE [16] learns a latent generative model of traffic scenarios and then searches for latent codes that map to risky cases, such as imminent collisions. However, these latent codes may not correspond to real traffic scenarios. As shown in the supplementary materials, the method generates scenarios that are unlikely in the real world (e.g. driving on the wrong side of the road). Note that this is a fundamental limitation of generative methods, because almost all existing datasets only include safe scenarios, and it is hard to generate cases that are rare or non-existent in the data.

Our method falls into the second category, which is to generate adversarial cases by perturbing real traffic scenarios. The challenge is to design a suitable threat model such that the altered scenarios remain realistic. AdvSim [17] plants adversarial agents that are optimized to jeopardize the ego vehicles by causing collisions, uncomfortable driving, etc. Although AdvSim enforces the dynamic feasibility of the synthesized trajectories, it uses black-box optimization which is slow and unreliable. Our work is most similar to a very recent work [18]. However, as we will show empirically, [18] fails to generate dynamically feasible adversarial trajectories. This is because its threat model simply uses dataset statistics (e.g. speed, acceleration, heading, etc.) as the dynamic parameters, which are too coarse to be used for generating realistic trajectories. For example, the maximum acceleration in the NuScenes dataset is over $20m/s^2$ where the maximum acceleration for a top-tier sports car is only around $10m/s^2$. In contrast, our method leverages a carefully-designed differentiable dynamic model to estimate *trajectory-wise* dynamic parameters. This allows our threat model to synthesize realistic and dynamically-feasible adversarial trajectories.

Adversarial Robustness. Deep learning models are shown to be generally vulnerable to adversarial attacks [19–30]. There is a large body of literature on improving their adversarial robustness [31–44]. In the AV context, many works examine on the adversarial robustness of the perception task [45], while analyzing the adversarial robustness of trajectory forecaster [18] is rarely explored. In this work, we focus on studying the adversarial robustness in the trajectory prediction task by considering its unique properties including motion and social interaction.

3 Problem Formulation and Challenges

In this section, we introduce the trajectory prediction task and then describe the threat model and assumptions for the attack and challenges.

Trajectory Prediction Formulation. In this work, we focus on the trajectory prediction task. The goal is to model the future trajectory distribution of N agents conditioned on their history states and other environment context such as maps. More specifically, a trajectory prediction model takes a sequence of observed state for each agent at a fixed time interval Δt, and outputs the predicted future trajectory for each agent. For observed time steps $t \leq 0$, we denote states of N agents at time step t as $\mathbf{X}^t = (x_1^t, \dots, x_i^t, \dots, x_N^t)$, where x_i^t is the state of agent i at time step t, which includes the position and the context information. We denote the history of all agents over H *observed* time steps as

$\mathbf{X} = \left(\mathbf{X}^{-H+1}, \ldots, \mathbf{X}^{0}\right)$. Similarly, we denote future trajectories of all N agents over T *future* time steps as $\mathbf{Y} = \left(\mathbf{Y}^{1}, \ldots, \mathbf{Y}^{T}\right)$, where $\mathbf{Y}^t = (y_1^t, \ldots, y_N^t)$ denotes the states of N agents at a future time step t ($t > 0$). We denote the ground truth and the predicted future trajectories as $\mathbf{Y}$ and $\hat{\mathbf{Y}}$, respectively. A trajectory prediction model $\mathcal{P}$ aims to minimize the difference between $\hat{\mathbf{Y}} = \mathcal{P}(\mathbf{X})$ and $\mathbf{Y}$. In an AV stack, trajectory prediction is executed repeatedly at a fixed time interval, usually the same as Δt. We denote L_p as the number of trajectory prediction being executed in several past consecutive time frames. Therefore, the histories at time frame ($-L_p < t \leq 0$) are $\mathbf{X}(t) = \left(\mathbf{X}^{-H-t+1}, \ldots, \mathbf{X}^{-t}\right)$, and similarly for $\mathbf{Y}$ and $\hat{\mathbf{Y}}$.

Adversarial Attack Formulation. In this work, we focus on the setting where an adversary vehicle (adv agent) attacks the prediction module of an ego vehicle by driving along an adversarial trajectory $\mathbf{X}_{\text{adv}}(\cdot)$. The trajectory prediction model predicts the future trajectories of both the adv agent and other agents. The attack goal is to mislead the predictions at each time step and subsequently make the AV plan execute unsafe driving behaviors. As illustrated in Figure 1, by driving along a carefully crafted adversarial (history) trajectory, the trajectory prediction model predicts wrong future trajectories for both the adv agent and the other agent. The mistakes can in term lead to severe consequences such as collisions. In this work, we focus on the white-box threat model, where the adversary has access to both model parameters, history trajectories and future trajectories of all agents, to explore what a powerful adversary can do based on the Kerckhoffs's principle [46] to better motivate defense methods.

Challenges. The challenges of devising effective adversarial attacks against prediction modules are two-fold: (1) **Generating realistic adversarial trajectory**. In AV systems, history trajectories are generated by upstream tracking pipelines and are usually sparsely queried due to computational constraints. On the other hand, dynamic parameters like accelerations and curvatures are high order derivatives of position and are usually estimated by numerical differentiation requiring calculating difference between positions *within a small-time interval*. Therefore, it is difficult to estimate correct dynamic parameters from such sparsely sampled positions in the history trajectory. Without the correct dynamic parameters, it is impossible to determine whether a trajectory is realistic or not, let alone generate new trajectories. (2) **Evaluating the implications of adversarial attacks**. Most existing evaluation metrics for trajectory prediction assume benign settings and are inadequate to demonstrate the implications for AV systems under attacks. For example, a large Average Distance Error (ADE) in prediction does not directly entail concrete consequences such as collision. Therefore, we need a new evaluation pipeline to systematically determine the consequences of adversarial attacks against prediction modules to further raise the awareness of general audiences on the risk that AV systems might face.

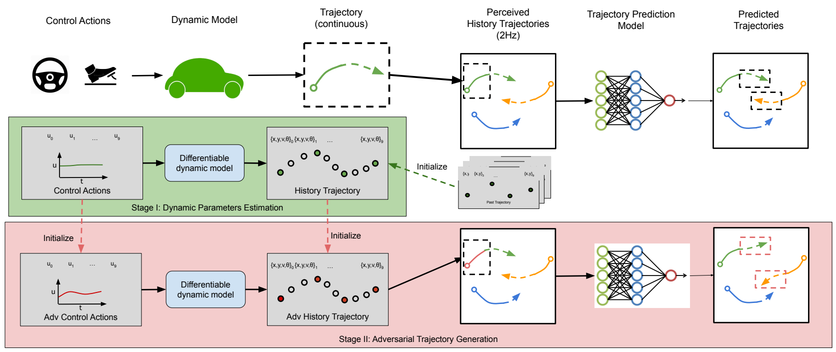


Fig. 2: Adversarial Dynamic Optimization (AdvDO) methodology overview

4 AdvDO: Adversarial Dynamic Optimization

To address the two challenges listed above, we propose **Adv**ersarial **D**ynamic **O**ptimization (AdvDO). As shown in Figure 2, given trajectory histories, AdvDO first estimates their dynamic parameters via a differentiable dynamic model. Then we use the estimated dynamic parameters to generate a realistic adversarial history trajectory given a benign trajectory by solving an adversarial optimization problem. Specifically, AdvDO consists of two stages: (1) dynamic parameters estimation, and (2) adversarial trajectory generation. In the first stage, we aim to estimate correct dynamic parameters by reconstructing a realistic dense trajectory from a sampled trajectory from the dataset. To reconstruct the dense trajectory, we leverage a differentiable dynamic model through optimization of control actions. When we get the estimated correct dynamic parameters of the trajectory, it could be used for the second stage. In the second stage, we aim to generate an adversarial trajectory that misleads future trajectory predictions given constraints. To achieve such goal, we carefully design the adversarial loss function with several regularization losses for the constraints. Then, we also extend the method to attacking consecutive predictions.

4.1 Dynamic Parameters Estimation

Differentiable dynamic model. A dynamic model computes the next state $s^{t+1} = \{p^{t+1}, \theta^{t+1}, v^{t+1}\}$ given current state $s^t = \{p^t, \theta^t, v^t\}$ and control actions $u^t = \{a^t, \kappa^t\}$. Here, p, θ, v, a, κ represent position, heading, speed, acceleration and curvature correspondingly. We adopt the kinematic bicycle model as the dynamic model which is commonly used [17]. We calculate the next state with a differential method, e.g., $v^{t+1} = v^t + a^t \cdot \Delta t$ where Δt denotes the time difference between two time steps. Given a sequence of control actions $u = (u^0, \ldots, u^t)$ and the initial state s^0, we denote the dynamic model as a differentiable function Φ such that it can calculate a sequence of future states $s = (s^0, \ldots, s^t) = \Phi(s^0, u; \Delta t)$. Noticed that the dynamic model also provides a reverse function Φ^{-1} that calculate a sequence of dynamic parameters $\{\theta, v, a, \kappa\} = \Phi^{-1}(p; \Delta t)$ given a trajectory $p = (p^0, \ldots, p^t)$. This discrete system can approximate the linear system in the real world when using a sufficiently small enough Δt. It can be also demonstrated that the dynamic model approximates better using a smaller Δt.

Optimization-based trajectory reconstruction. To accurately estimate the dynamic parameters $\{\theta, v, a, \kappa\}$ given a trajectory p, a small time difference Δt or a large sampling rates $f = 1/\Delta t$ is required. However, the sampling rate of the trajectory in the trajectory prediction task is decided by the AV stack, and is often small (e.g. 2Hz for nuScenes [9]) limited by the computation performance of the hardware. Therefore, directly estimating the dynamic parameters from the sampled trajectory is not accurate, making it difficult to determine whether the adversarial history $\mathbf{X}_{adv}$ generated by perturbing the history trajectory provided by the AV system is realistic or not. To resolve this challenge, we propose to reconstruct a densely trajectory first and then estimate a more accurate dynamic parameter from the reconstructed dense trajectory. To reconstruct a densely sampled history trajectory $\mathbf{D}_i = \left(\mathbf{D}_i^{-H\cdot f+1}, \ldots, \mathbf{D}_i^0\right)$ from a given history trajectory $\mathbf{X}_i$ with additional sampling rates f, we need to find a realistic trajectory $\mathbf{D}_i$ that passes through positions in $\mathbf{X}_i$. We try to find it through solving an optimization problem. In order to efficiently find a realistic trajectory, we wish to optimize over the control actions in stead of the positions in $\mathbf{D}_i$. To start with, we initialize $\mathbf{D}_i$ with a simple linear interpolation of $\mathbf{X}_i$, i.e. $\mathbf{D}_i^{-t\cdot f+j} = (1 - j/f) \cdot \mathbf{X}^{-t} + j/f \cdot \mathbf{X}^{-t+1}$. We then calculate the dynamic parameters for all steps $\{\theta, v, a, \kappa\} = \Phi^{-1}(\mathbf{D}_i; \Delta t)$. Now, we can represent the reconstructed densely sampled trajectory $\mathbf{D}_i$ with $\Phi(s^0, u; \Delta t)$, where $u = \{a, \kappa\}$. To further reconstruct a realistic trajectory, we optimize over the control actions u with a carefully designed reconstruction loss function $\mathcal{L}_{\text{recon}}$. The reconstruction loss function consists of two terms. We first include a MSE (Mean Square Error) loss to enforce the reconstructed trajectory passing through the given history trajectory $\mathbf{X}_i$. We also include l_{dyn}, a regularization loss based on a soft clipping function to bound the dynamic parameters in a predefined range based on vehicle dynamics [17]. To summarize, by solving the optimization problem of:

$$\min_u \mathcal{L}_{\text{recon}}(u; s^0, \Phi) = MSE(\mathbf{D}_i, \mathbf{X}_i) + l_{\text{dyn}}(\theta, v, a, \kappa)$$

,we reconstruct a densely sampled, dynamically feasible trajectory $\mathbf{D*}_i$ passing through the given history trajectory for the adversarial agent.

4.2 Adversarial Trajectory Generation

Attacking a single-step prediction. To generate realistic adversarial trajectories, we first initialize the dynamic parameters of the adversarial agent with estimation from the previous stage, noted as $\mathbf{D*}_{orig}$. Similarly to the optimization in the trajectory reconstruction process, we optimize the control actions u

to generate the optimal adversarial trajectories. Our adversarial optimization objective consists of four terms. The detailed formulation for each term is in the supplementary materials. The first term l_{obj} represents the attack goal. As motion and social properties are essential and unique for trajectory prediction models. Thus, our l_{obj} has accounted for them when designed. The second term l_{col} is a commonsense objective that encourages the generated trajectories to follow some commonsense traffic rules. In this work we only consider collision avoidance [11]. The third term l_{bh} is a regularization loss based on a soft clipping function, given a clipping range of $(-\epsilon, \epsilon)$. It bounds the adversarial trajectories to be close to the original history trajectory $\mathbf{X}_{\text{orig}}$. We also include l_{dyn} to bound the dynamic parameters. The full adversarial loss is defined as:

$$\mathcal{L}_{\text{adv}} = l_{\text{obj}}(\mathbf{Y}, \hat{\mathbf{Y}}) + \alpha \cdot \sum_{i} l_{\text{col}}(\mathbf{D}_{\text{adv}}, \mathbf{X}) + \beta \cdot l_{\text{bh}}(\mathbf{D}_{\text{adv}}, \mathbf{D^*}_{\text{orig}}) + \gamma l_{\text{dyn}}(\mathbf{D}_{adv})$$

,where α and β are weighting factors. We then use the projected gradient descent (PGD) method [33] to find the adversarial control actions u_{adv} bounded by constraints (u_{lb}, u_{ub}) attained from vehicle dynamics.

Attacking consecutive predictions. To attack L_p consecutive frames of predictions, we aim to generate the adversarial trajectory of length $H + L_p$ that uniformly misleads the prediction at each time frames. To achieve this goal, we can easily extend the formulation for attacking single-step predictions to attack a sequence of predictions, which is useful for attacking a sequential decision maker such as an AV planning module. Concretely, to generate the adversarial trajectories for L_p consecutive steps of predictions formulated in§ 3, we aggregate the adversarial losses over these frames. The objective for attacking a length of $H + L_p$ trajectory is:

$$\sum_{t \in [-L_p, \ldots 0]} \mathcal{L}_{\text{adv}}(\mathbf{X}(t), \mathbf{D}_{\text{adv}}(t), \mathbf{Y}(t))$$

, where $\mathbf{X}(t), \mathbf{D}_{\text{adv}}(t), \mathbf{Y}(t)$ are the corresponding $\mathbf{X}, \mathbf{D}_{\text{adv}}, \mathbf{Y}$ at time frame t.

5 Experiments

Our experiments seek to answer the following questions: (1) Are the current mainstream trajectory prediction systems robust against our attacks?;(2) Are our attacks more realistic compared to other methods?; (3) How do our attacks affect an AV prediction-planning system?; (4) Does features designed to model motion and/or social properties affect a model's adversarial robustness?; and (5) Could we mitigate our attack via adversarial training?

5.1 Experimental Setting

Models. We evaluate two state-of-the-art trajectory prediction models: AgentFormer and Trajectron++. As explained before, we select AgentFormer and

Trajectron++ for their representative features in modeling motion and social aspects in prediction. AgentFormer proposed a transformer-based social interaction model which allows an agent's state at one time to directly affect another agent's state at a future time. And Trajectron++ incorporates agent dynamics. Since semantic map is an optional information for these models, we prepare two versions for each model with map and without map.

Datasets. We follow the settings in [4, 3] and use nuScenes dataset [9], a large-scale motion prediction dataset focusing on urban driving settings. We select history trajectory length ($H = 4$) and future trajectory length ($T = 12$) following the official recommendation. We report results on all 150 validation scenes.

Baselines. We select the search-based attack proposed by Zhang et al. [18] as the baseline, named *search*. As we mentioned earlier in § 2, the original method made two mistakes: (1) incorrect estimated bound values for dynamic parameters and (2) incorrect choices of bounded dynamic parameters for generating realistic adversarial trajectories. We correct such mistakes by (1) using a set of real-world dynamic bound values [17]. and (2) bounding the curvature variable instead of heading derivatives since curvature is linear related to steering angle. We denote this attack method as *search**. For our methods, we evaluate two variations: (1) *Opt-init*, where the initial dynamics (i.e dynamics at ($t = -H$) time step) $\mathbf{D}_{adv}^{-H \cdot S+1}$ are fixed and (2) *Opt-end*, where the current dynamics ($t = 0$) $\mathbf{D}_{adv}^{0}$ are fixed. While *Opt-end* is not applicable for sequential attacks, we include *Opt-end* for understanding the attack with strict bounds, since the current position often plays an important role in trajectory prediction.

Metrics. We evaluate the attack with four metrics in the nuscenes prediction challenges: ADE/FDE, Miss Rates (MR), Off Road Rates (ORR) [9] and their correspondence with planning-awareness version: PI-ADE/PI-FDE, PI-MR, PI-ORR [8] where metric values are weighted by the sensitivity to AV planning. In addition, to compare which attack method generates the most realistic adversarial trajectories, we calculate the violation rates (VR) of the curvature bound, where VR is the ratio of the number of adversarial trajectories violating dynamics constraints over the total number of generated adversarial trajectories.

Implementation details. For the trajectory reconstruction, we use the Adam optimizer and set the step number of optimization to 5. For the PGD-based attack, we set the step number to 30 for both AdvDO and baselines. We empirically choose $\beta = 0.1$ and $\alpha = 0.3$ for best results.

5.2 Main Results

Trajectory prediction under attacks. First, we compare the effectiveness of the attack methods on prediction performances. As shown in Table 1, our proposed attack (*Opt-init*) causes the highest prediction errors across all model variants and metrics. *Opt-init* overperforms *Opt-end* by a large margin, which shows that the dynamics of the current frame play an important role in trajectory prediction systems. Note that *search* proposed by Zhang *et al.* has a significant violation rates (VR) over 10%. It further validates our previous claim that *search* generates unrealistic trajectories.

Table 1: Attack evaluation results on general metrics.

Model	Attack	ADE	FDE	MR	ORR	Violations
Agentformer w/ map	None	1.83	3.81	28.2%	4.7%	0%
	search	2.34	4.78	34.3%	6.6%	10%
	*search**	1.88	3.89	29.2%	4.8%	0%
	Opt-end	2.23	4.54	34.5%	6.3%	0%
	Opt-init	**3.39**	**5.75**	**44.0%**	**10.4%**	0%
Agentformer w/o map	None	2.20	4.82	35.0%	7.3%	0%
	search	2.66	5.53	40.3%	8.9%	9%
	*search**	2.20	4.94	35.1%	7.4%	0%
	Opt-end	2.54	5.54	39.3%	8.8%	0%
	Opt-init	**3.81**	**6.01**	**49.8%**	**13.3%**	0%
Trajectron++ w/ map	None	1.88	4.10	35.1%	7.9%	0%
	search	2.53	5.03	44.4%	9.4%	12%
	*search**	1.93	4.26	36.3%	8.3%	0%
	Opt-end	2.48	5.57	47.5%	11.3%	0%
	Opt-init	**3.20**	**8.56**	**57.2%**	**15.9%**	0%
Trajectron++ w/o map	None	2.10	5.00	41.1%	9.6%	0%
	search	2.76	8.02	50.5%	16.1%	14%
	*search**	2.17	5.25	42.2%	10.0%	0%
	Opt-end	2.49	7.54	49.5%	14.2%	0%
	Opt-init	**3.58**	**9.36**	**76.8%**	**17.8%**	0%

To further demonstrate the impact of the attacks on downstream pipelines like planning, here we report prediction performance using planning-aware metrics proposed by Ivanovic *et al.* [8]. As described above, these metrics consider how the predictions accuracy of surrounding agents behaviors impact the ego's ability to plan its future motion. Specifically, the metrics are computed from the partial derivative of the planning cost over the predictions to estimate the sensitivity of the ego vehicle's further planning. Furthermore, by aggregating weighted prediction metrics (e.g., ADE, FDE, MR, ORR) with such sensitivity measurement, we could report planning awareness metrics including (PI-ADE/FDE, PI-MR, PI-ORR) quantitatively. As shown in Table 2, results are consistent with the previous results.

Attack fidelity analysis. Here, we aim to demonstrate the fidelity of the generated adversarial trajectories qualitatively and quantitatively. We show our analysis on AgentFormer with map as a case study. In Figure 3, we visualize the adversarial trajectories generated by *search* and *Opt-end* methods. We demonstrate that our

Table 3: Quantitative comparison of generated adversarial trajectories

Method	*search*	*Opt-end*	*Opt-init*
ΔSensitivity	2.33	1.12	1.34

Table 2: Attack evaluation results on planning-aware metrics.

Model	Attack	PI-ADE	PI-FDE	PI-MR	PI-ORR	VR
Agentformer w/ map	None	1.38	2.76	20.5%	22.8%	0%
	search	1.62	3.32	25.7%	25.2%	13%
	*search**	1.39	2.79	21.4%	23.0%	0%
	Opt-end	1.57	3.11	23.7%	24.8%	0%
	Opt-init	**2.05**	**3.81**	**32.9%**	**29.0%**	0%
Agentformer w/o map	None	1.46	3.76	26.8%	30.3%	0%
	search	1.63	4.12	28.9%	34.2%	11%
	*search**	1.49	3.74	27.5%	31.1%	0%
	Opt-end	1.63	4.11	28.2%	39.3%	0%
	Opt-init	**2.24**	**5.91**	**34.3%**	**41.3%**	0%
Trajectron++ w/ map	None	1.42	2.81	26.5%	25.6%	0%
	search	1.68	3.38	29.2%	28.3%	14%
	*search**	1.43	2.83	26.7%	27.7%	0%
	Opt-end	1.65	3.14	27.2%	28.1%	0%
	Opt-init	**2.11**	**3.85**	**37.8%**	**32.7%**	0%
Trajectron++ w/o map	None	1.76	3.20	30.9%	44.0%	0%
	search	2.02	3.96	35.0%	49.6%	19%
	*search**	1.77	3.25	31.0%	46.8%	0%
	Opt-end	1.95	3.55	31.6%	46.3%	0%
	Opt-init	**2.46**	**4.26**	**41.2%**	**53.7%**	0%

method (*Opt-end*) can generate effective attack without changing the semantic meaning of the driving behaviors. In contrast, *search* either generates unrealistic trajectories or changes the driving behaviors dramatically. For example, the middle row shows that the adversarial trajectory generated by *search* takes a near 90-degree sharp turn within a small distance range, which is dynamically feasible, whereas by our method (right image in the first row) generates smooth and realistic adversarial trajectories. More examples of generated adversarial trajectories can be found in Appendix.

To further quantify the attack fidelity, we propose to use the sensitivity metric in [8] to measure the degree of behavior alteration caused by the adversarial attacks. The metric is to measure the influence of an agent's behavior over other agents' future trajectories. We calculate the difference of aggregated sensitivity of non-adv agents between the benign and adversarial settings. Detailed formulation is in Appendix. We demonstrate that our proposed attacks (*Opt-init*, *Opt-end*) cause smaller sensitivity changes. This corroborates our qualitative analysis that our method generates more realistic attacks at the behavior level.

Case studies with planners. To explicitly demonstrate the consequences of our attacks to the AV stack, we evaluate the adversarial robustness of a prediction-planning pipeline in an end-to-end manner. We select a subset of validation scenes and evaluate two planning algorithms, rule-based [16] and MPC-

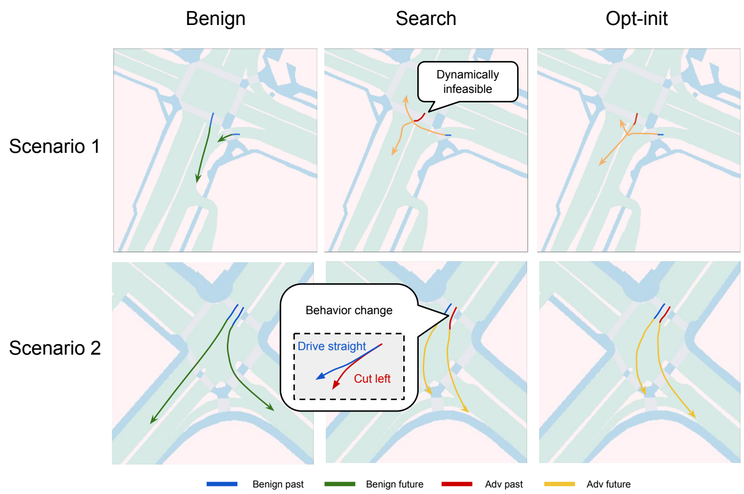
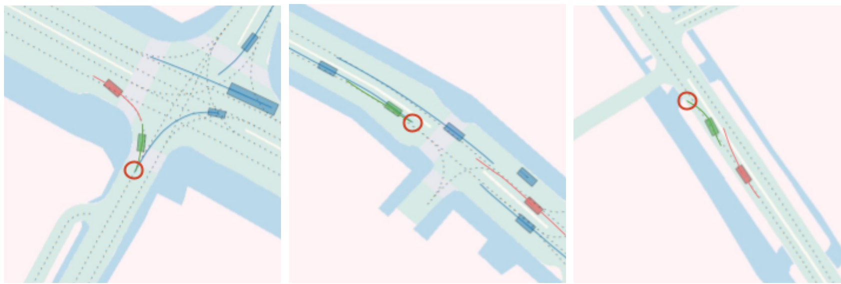


Fig. 3: Qualitative comparison of generated adversarial trajectories. We demonstrate that the proposed AdvDO generates adversarial trajectories both realist and effective whereas the search-stats could either generate dynamically infeasible trajectories (sharp turn on the first row) or changing the behavior dramatically (behavior change from driving straight to swerving left on the second row).

Table 4: Planning results

Planner	Open-loop		Closed-loop	
	Rule-based	MPC	Rule-based	MPC
Collisions	26/150	10/150	12/150	7/150
Off road	–	43/150	–	23/150

based [47], in in two rollout settings, open-loop and closed-loop. Detailed description for the planners can be found in Appendix. In the open-loop setting, an ego vehicle generates and follows a 6-second plan without replanning. The closed-loop setting is to replan every 0.5 seconds. We replay the other actors' trajectories in both cases. For the closed-loop scenario, we conduct the sequential attack using $L_p = 6$. As demonstrated in Table 4, our attacks causes the ego to collide with other vehicles and/or leave drivable regions. We visualize a few representative cases in Figure 4. Figure 4(a) shows the attack leads to a side collision. Figure 4(b) shows the attack misleads the prediction and forces the AV to stop and leads to a rear-end collision. Note that no attack can lead the rule-based planner to leave drivable regions because it is designed to keep the ego vehicle in the middle of the lane. At the same time, we observed that attacking the rule-based planner results in more collisions since it cannot dodge head-on collisions.



(a) Side collision (b) Rear-end collision (c) Driving off-road

Fig. 4: Visualized results for planner evaluation. Ego vehicle in green, adv agent in red and other agents in blue. The red cycle represents the collision or driving off-road consequence.

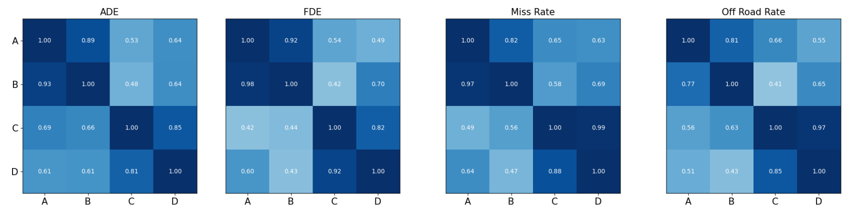
Motion and social modeling. As mentioned in § 2, trajectory prediction model aims to learn (1) the motion dynamics of each agent and (2) social interactions between agents. Here we conduct more in-depth attack analysis with respect to these two properties. For the motion property, we introduce a *Motion* metric that measures the changes of predicted future trajectory of the adversarial agent as a result of the attack. For the social property, we hope to evaluate the influence of the attack on the predictions of non-adv agents. Thus, we use a metric named *Interaction* to measure the average prediction changes among all non-adv agents. As shown in Table 5, the motion property is more prone to attack than the interaction property. This is because perturbing the adv agent's history directly impacts its future, while non-adv agents are affected only through the interaction model. We observed that our attack leads to larger *Motion* error for AgentFormer than for Trajectron++. A possible explanation is that AgentFormer enables direct interactions between past and future trajectories across all agents, making it more vunerable to attacks.

Table 5: Ablation results for Motion and Interaction metrics

Model	Scenarios	ADE	FDE	MR	ORR	Model	ADE	FDE	MR	ORR
AgentFormer	*Motion*	8.12	12.35	57.3%	18.6%	Trajectron++	8.75	13.27	59.6%	16.6%
	Interaction	2.03	4.21	30.3%	5.1%		1.98	4.68	43.0%	8.71%

Transferability analysis. Here we evaluate whether the adversarial examples generated by considering one model can be transferred to attack another model. We report *transfer rate* (more details in the appendix). Results are shown in Figure 5. Cell (i, j) shows the normalized transfer rate value of adversarial examples generated against model j and evaluate on model i. We demonstrate that the generated adversarial trajectories are highly transferable (transfer rates $\geq 77\%$)

when sharing the same backbone network. In addition, the generated adversarial trajectories can transfer among different backbones as well. These results show the feasibility for black-box attacks against unseen models in the real-world.



(a) ADE (b) FDE (c) Miss Rate (d) Off Road Rate

Fig. 5: Transferability heatmap. A: AgentFormer w/ map; B: & AgentFormer w/o map; C: Trajectron++ w/ map ; D: Trajectron++ & w/o map

Mitigation. To mitigate the consequences of the attacks, we use the standard mitigation method, adversarial training [33], which has been shown as the most effective defense. As shown in Table C in the Appendix, we find that the adversarial trained model using the *search* attack is much worse than the adversarial trained model using our *Opt-init* attack. This can be due to unrealistic adversarial trajectories generated by the *search* attack lead to the mode failure since the performance of it on clean data are worse than the model under strong attacks. This also emphasizes that generating realistic trajectory is essential to success of improving adversarial robustness.

6 Conclusion

In this paper, we study the adversarial robustness of trajectory prediction systems. We present an attack framework to generate *realistic* adversarial trajectories via a carefully-designed differentiable dynamic model. We have shown that prediction models are generally vulnerable and certain model designs (e.g, modeling motion and social properties simultaneously) beneficial in benign settings may make a model more vulnerable to adversarial attacks. In addition, both motion (predicted future trajectory of adversarial agent) and social (predicted future trajectory of other agents) properties could be exploited by only manipulating the adversarial agent's history trajectories. We also show that prediction errors influence the downstream planning and control pipeline, leading to severe consequences such as collision. We hope our study can shed light on the importance of evaluating worst-case performance under adversarial examples and raise awareness on the types of security risks that AV systems might face, so forth encourages robust trajectory prediction algorithms.

References

1. Alexandre Alahi, Kratarth Goel, Vignesh Ramanathan, Alexandre Robicquet, Li Fei-Fei, and Silvio Savarese. Social lstm: Human trajectory prediction in crowded spaces. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 961–971, 2016.
2. Boris Ivanovic and Marco Pavone. The trajectron: Probabilistic multi-agent trajectory modeling with dynamic spatiotemporal graphs. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2375–2384, 2019.
3. Tim Salzmann, Boris Ivanovic, Punarjay Chakravarty, and Marco Pavone. Trajectron++: Dynamically-feasible trajectory forecasting with heterogeneous data. In *European Conference on Computer Vision*, pages 683–700. Springer, 2020.
4. Ye Yuan, Xinshuo Weng, Yanglan Ou, and Kris Kitani. Agentformer: Agent-aware transformers for socio-temporal multi-agent forecasting. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021.
5. Nicholas Rhinehart, Kris M Kitani, and Paul Vernaza. R2p2: A reparameterized pushforward policy for diverse, precise generative path forecasting. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 772–788, 2018.
6. Nicholas Rhinehart, Rowan McAllister, Kris Kitani, and Sergey Levine. Precog: Prediction conditioned on goals in visual multi-agent settings. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2821–2830, 2019.
7. Vineet Kosaraju, Amir Sadeghian, Roberto Martín-Martín, Ian Reid, Hamid Rezatofighi, and Silvio Savarese. Social-bigat: Multimodal trajectory forecasting using bicycle-gan and graph attention networks. *Advances in Neural Information Processing Systems*, 32, 2019.
8. Boris Ivanovic and Marco Pavone. Injecting planning-awareness into prediction and detection evaluation. *CoRR*, abs/2110.03270, 2021.
9. Holger Caesar, Varun Bankiti, Alex H Lang, Sourabh Vora, Venice Erin Liong, Qiang Xu, Anush Krishnan, Yu Pan, Giancarlo Baldan, and Oscar Beijbom. nuscenes: A multimodal dataset for autonomous driving. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11621–11631, 2020.
10. Nachiket Deo, Eric Wolff, and Oscar Beijbom. Multimodal trajectory prediction conditioned on lane-graph traversals. In *5th Annual Conference on Robot Learning*, 2021.
11. Simon Suo, Sebastian Regalado, Sergio Casas, and Raquel Urtasun. Trafficsim: Learning to simulate realistic multi-agent behaviors. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10400–10409, 2021.
12. Wenhao Ding, Baiming Chen, Minjun Xu, and Ding Zhao. Learning to collide: An adaptive safety-critical scenarios generating method. In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2243–2250. IEEE, 2020.
13. Mark Koren and Mykel J Kochenderfer. Efficient autonomy validation in simulation with adaptive stress testing. In *2019 IEEE Intelligent Transportation Systems Conference (ITSC)*, pages 4178–4183. IEEE, 2019.
14. Wenhao Ding, Baiming Chen, Bo Li, Kim Ji Eun, and Ding Zhao. Multimodal safety-critical scenarios generation for decision-making algorithms evaluation. *IEEE Robotics and Automation Letters*, 6(2):1551–1558, 2021.

15. Yasasa Abeysirigoonawardena, Florian Shkurti, and Gregory Dudek. Generating adversarial driving scenarios in high-fidelity simulators. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 8271–8277. IEEE, 2019.
16. Davis Rempe, Jonah Philion, Leonidas J. Guibas, Sanja Fidler, and Or Litany. Generating useful accident-prone driving scenarios via a learned traffic prior. In *arXiv:2112.05077*, 2021.
17. Jingkang Wang, Ava Pun, James Tu, Sivabalan Manivasagam, Abbas Sadat, Sergio Casas, Mengye Ren, and Raquel Urtasun. Advsim: Generating safety-critical scenarios for self-driving vehicles. *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021.
18. Qingzhao Zhang, Shengtuo Hu, Jiachen Sun, Qi Alfred Chen, and Z. Morley Mao. On adversarial robustness of trajectory prediction for autonomous vehicles. *CoRR*, abs/2201.05057, 2022.
19. Nicholas Carlini, Anish Athalye, Nicolas Papernot, Wieland Brendel, Jonas Rauber, Dimitris Tsipras, Ian Goodfellow, Aleksander Madry, and Alexey Kurakin. On evaluating adversarial robustness. *arXiv preprint arXiv:1902.06705*, 2019.
20. Ambra Demontis, Marco Melis, Maura Pintor, Matthew Jagielski, Battista Biggio, Alina Oprea, Cristina Nita-Rotaru, and Fabio Roli. Why do adversarial attacks transfer? explaining transferability of evasion and poisoning attacks. In *28th USENIX Security Symposium (USENIX Security 19)*, pages 321–338, Santa Clara, CA, August 2019. USENIX Association.
21. Nicholas Carlini and David Wagner. Towards evaluating the robustness of neural networks. In *2017 ieee symposium on security and privacy (sp)*, pages 39–57. IEEE, 2017.
22. Chaowei Xiao, Bo Li, Jun-Yan Zhu, Warren He, Mingyan Liu, and Dawn Song. Generating adversarial examples with adversarial networks. *arXiv preprint arXiv:1801.02610*, 2018.
23. Chenglin Yang, Adam Kortylewski, Cihang Xie, Yinzhi Cao, and Alan Yuille. Patchattack: A black-box texture-based attack with reinforcement learning. In *European Conference on Computer Vision*, pages 681–698. Springer, 2020.
24. Cihang Xie, Jianyu Wang, Zhishuai Zhang, Yuyin Zhou, Lingxi Xie, and Alan Yuille. Adversarial examples for semantic segmentation and object detection. In *International Conference on Computer Vision*. IEEE, 2017.
25. Lifeng Huang, Chengying Gao, Yuyin Zhou, Cihang Xie, Alan Yuille, Changqing Zou, and Ning Liu. Universal physical camouflage attacks on object detectors, 2019.
26. Lifeng Huang, Chengying Gao, Yuyin Zhou, Cihang Xie, Alan L Yuille, Changqing Zou, and Ning Liu. Universal physical camouflage attacks on object detectors. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 720–729, 2020.
27. Chong Xiang, Charles R Qi, and Bo Li. Generating 3d adversarial point clouds. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9136–9144, 2019.
28. Yuxin Wen, Jiehong Lin, Ke Chen, and Kui Jia. Geometry-aware generation of adversarial and cooperative point clouds. 2019.
29. Abdullah Hamdi, Sara Rojas, Ali Thabet, and Bernard Ghanem. Advpc: Transferable adversarial perturbations on 3d point clouds. In *European Conference on Computer Vision*, pages 241–257. Springer, 2020.

30. Chaowei Xiao, Dawei Yang, Bo Li, Jia Deng, and Mingyan Liu. Meshadv: Adversarial meshes for visual recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6898–6907, 2019.
31. Adam Noack, Isaac Ahern, Dejing Dou, and Boyang Li. An empirical study on the relation between network interpretability and adversarial robustness. *SN Computer Science*, 2(1):1–13, 2021.
32. Anindya Sarkar, Anirban Sarkar, Sowrya Gali, and Vineeth N Balasubramanian. Adversarial robustness without adversarial training: A teacher-guided curriculum learning approach. *Advances in Neural Information Processing Systems*, 34, 2021.
33. Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. In *International Conference on Learning Representations*, 2018.
34. Yuzhe Yang, Guo Zhang, Dina Katabi, and Zhi Xu. Me-net: Towards effective adversarial robustness with matrix estimation. *arXiv preprint arXiv:1905.11971*, 2019.
35. Weilin Xu, David Evans, and Yanjun Qi. Feature squeezing: Detecting adversarial examples in deep neural networks. *arXiv preprint arXiv:1704.01155*, 2017.
36. Mitali Bafna, Jack Murtagh, and Nikhil Vyas. Thwarting adversarial examples: An l_0-robustsparse fourier transform. *arXiv preprint arXiv:1812.05013*, 2018.
37. Nicolas Papernot, Patrick McDaniel, Xi Wu, Somesh Jha, and Ananthram Swami. Distillation as a defense to adversarial perturbations against deep neural networks. In *2016 IEEE symposium on security and privacy (SP)*, pages 582–597. IEEE, 2016.
38. Dongyu Meng and Hao Chen. Magnet: a two-pronged defense against adversarial examples. In *Proceedings of the 2017 ACM SIGSAC conference on computer and communications security*, pages 135–147, 2017.
39. Huan Zhang, Hongge Chen, Chaowei Xiao, Sven Gowal, Robert Stanforth, Bo Li, Duane Boning, and Cho-Jui Hsieh. Towards stable and efficient training of verifiably robust neural networks. *arXiv preprint arXiv:1906.06316*, 2019.
40. Huan Zhang, Hongge Chen, Chaowei Xiao, Bo Li, Duane S Boning, and Cho-Jui Hsieh. Robust deep reinforcement learning against adversarial perturbations on observations. 2020.
41. Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. *arXiv preprint arXiv:1706.06083*, 2017.
42. Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014.
43. Eric Wong, Leslie Rice, and J Zico Kolter. Fast is better than free: Revisiting adversarial training. *arXiv preprint arXiv:2001.03994*, 2020.
44. Ali Shafahi, Mahyar Najibi, Amin Ghiasi, Zheng Xu, John Dickerson, Christoph Studer, Larry S Davis, Gavin Taylor, and Tom Goldstein. Adversarial training for free! *arXiv preprint arXiv:1904.12843*, 2019.
45. Cihang Xie, Jianyu Wang, Zhishuai Zhang, Yuyin Zhou, Lingxi Xie, and Alan Yuille. Adversarial Examples for Semantic Segmentation and Object Detection. In *IEEE International Conference on Computer Vision (ICCV)*, 2017.
46. Claude E Shannon. Communication theory of secrecy systems. *Bell Labs Technical Journal*, 28(4):656–715, 1949.
47. Eduardo F Camacho and Carlos Bordons Alba. *Model predictive control*. Springer science & business media, 2013.