Supplementary Materials: Robust Trajectory Prediction against Adversarial Attacks

Yulong Cao*[1,2], Danfei Xu[2,3], Xinshuo Weng[2], Z. Morley Mao[1], Anima Anandkumar[2,4], Chaowei Xiao[2,5], and Marco Pavone[2,6]

[1]University of Michigan
[2]NVIDIA
[3]Georgia Institute of Technology
[4]California Institute of Technology
[5]Arizona State University
[6]Stanford University

A Method and Implementations

A.1 Adversarial Attack on Trajectory Prediction

***Latent Attack* and *Context Attack*.** Noticed that, besides *Deterministic Attack* introduced in the main paper, there are also two other less intuitive attacks. Since the prediction $\hat{\mathbf{Y}}$ is dependent on posterior distribution $q_\phi(\mathbf{Z}|\mathbf{X},\mathbf{Y})$ and conditional variable $f(\mathbf{X})$, we can construct attacks based on that. *Latent attack* aims to increase the error of estimating $q_\phi(\mathbf{Z}|\mathbf{X},\mathbf{Y})$, which is formulated as:

$$\delta = \arg\max_\delta \mathrm{KL}(\, q_\phi(\mathbf{Z}|\mathbf{Y},\mathbf{X}) \parallel q_\phi(\mathbf{Z}|\mathbf{Y},\mathbf{X}+\delta)\,). \tag{S1}$$

Context attack aims to increase the error of encoding the conditional variable $f(Z)$, which is formulated as:

$$\delta = \arg\max_\delta \mathrm{d}(\, f(\mathbf{X}),\ f(\mathbf{X}+\delta)\,), \tag{S2}$$

where d is a distance function (e.g., L_2 norm). However, *latent attack* and *context attack* are effective due to two reasons. First, they are exploiting the vulnerability of a partial model. For example, *latent attack* only exploits the posterior estimation $q_\phi(\mathbf{Z}|\mathbf{X},\mathbf{Y})$ and *context attack* only exploits the conditional encoder $f(\mathbf{X})$. Second, these attacks aim for a different goal. For the *latent attack* and *context attack*, the objectives are set for finding adversarial perturbations that maximize the difference of generated posterior distribution/context given $\mathbf{X}$ and $\mathbf{X}+\delta$, due to lacking ground truth for intermediate latent variables. However, for the sample attack, the objective is directly set for maximizing the prediction errors from the ground truth (future trajectories), which is more effective.

Adversarial attack on consecutive frames. In order to fool a planner in a closed-loop manner to make consistent wrong decisions, we need to conduct adversarial attacks on consecutive frames. 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 frame. 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 sequence of decision made by AV planning module. Concretely, to generate the adversarial trajectories for L_p consecutive steps of predictions, we aggregate the adversarial losses over these frames. The objective for attacking a length of $H+L_p$ trajectory is:

$$\sum_{t\in[-L_p,\ldots 0]} \mathcal{L}_{\text{adv}}(\mathbf{X}_{\text{adv}}(t), \mathbf{Y}(t)), \tag{S3}$$

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

*Work done during an internship at NVIDIA

A.2 Adversarial Training on Generative Models

Challenges. One challenge that hinders the adversarial training process is the noisy conditional data distribution disturbing the training process. One hypothesis we mentioned in the main paper is, the context encoding can magnify the bounded perturbation δ on history trajectory $\mathbf{X}$ to an unbounded perturbation on the conditional variable $\mathbf{C} = f(\mathbf{X} + \delta)$, during the training process.

Lemma A.1 *For a neural network f which is not bounded on Lipschitz constant during the training procedure, given any constant η and an input $\mathbf{X}$, there exists a pair (δ, f), that satisfies*

$$\parallel f(\mathbf{X} + \delta) - f(\mathbf{X}) \parallel \geq \eta.$$

Lemma A.1 can be easily derived by the definition of Lipshitz constant. Lipshitz constant L is defined as

$$L := \sup \frac{\parallel f(\mathbf{X} + \delta) - f(\mathbf{X}) \parallel}{\parallel \delta \parallel}.$$

If L is not bounded, $\frac{\|f(\mathbf{X}+\delta)-f(\mathbf{X})\|}{\|\delta\|}$ is not bounded and so is $\parallel f(\mathbf{X} + \delta) - f(\mathbf{X}) \parallel$ given a bounded δ. This means that, a bounded perturbation δ can potentially be magnified to be noisier on encoded conditional variable $\mathbf{C} = f(\mathbf{X} + \delta)$.

Analysis. In order to provide a quantitative analysis of the degeneration degree from conditional generative model to generative model (e.g., CVAE to VAE) with respect to the noise level, we propose a method to estimate the correlation between the degeneration and the noise level. Specifically, we trained a classifier with a 2-layer CNN achieving 99% accuracy on MNIST dataset. Then, given a conditional variable (the upper left quarter of an image of digit y), we generate images with the conditional generative models and use the classifier to calculate the confidence of the generated images labeled as digit y. We calculate the average confidence of 10 generated images on all 10,000 images in the MNIST test data set. The lower the score means the weaker the correlation between the generated images with the given conditions, or the stronger correlation between the degeneration and the noise level. We also provide visualization examples of generated images from models trained using data with different level of noises in Figure A and Figure B. We can see that, with the noise level increases, the generated images are less dependent on the conditional variable (i.e., not being the same digit). In the extreme case of high level noise, for example when $p = 0.9$, the model generates images solely depends on the random prior value it samples and generates the similar images for each row. We can also see that, the adversarial noises are more effective compared to the salt and pepper noise. With a small amount of noise (i.e., $\epsilon = 0.1$), it can degenerate the conditional generative model to a generative model (i.e., CVAE to VAE here).

A.3 Data Augmentation with Dynamic Model

For the data augmentation strategy $\mathbb{A}$, we use a kinematic bicycle model [1] as our dynamic model to generate realistic trajectories that can be driven in the real world. Representing the behavior of actors as kinematic bicycle model trajectories allows for physical feasibility and fine-grained behavior control. To generate realistic trajectories, we first parameterize the trajectory $\mathbf{S} = \{s^t\}_0^T$ as a sequence of kinematic bicycle model states $s^t = \{p^t, \kappa^t, a^t\}$, where p represents the position, κ represents the trajectory curvature, and a represents the acceleration. Then, trajectories can be generated by controlling the change of curvature $\dot{\kappa}_t$ and the acceleration a_t over time, and using the kinematic bicycle model to update corresponding other states for each timestamp. To generate diverse trajectories, we set the objectives as biasing the trajectories to a given direction (e.g. forward, backward, left and right), while not colliding with other agents. To that end, we optimize a carefully-designed objective function $\mathcal{L}_{\text{dyn}}$ over the control actions, i.e. $\dot{\kappa}_t$ and a_t for each agents. More specifically, the objective function consists of two components: $\mathcal{L}_{\text{dyn}} = \mathcal{L}_{\text{d}} + \gamma \mathcal{L}_{\text{col}}$, where $\mathcal{L}_{\text{d}}$ is the deviation objective loss, $\mathcal{L}_{\text{col}}$ is the collision regularization loss, and γ is a weight factor to balance the objectives. In each scene, we randomly pick a deviation objective loss $\mathcal{L}_{\text{d}}$ from the set {moving forward, backward, left, right} for each agent. More specifically, the deviation objective loss $\mathcal{L}_{\text{d}}$ is formulated as

$$\mathcal{L}_{\text{d}} = (\mathbf{X} - \mathbf{X}_{\text{aug}})\, \bar{\mathbf{d}},$$

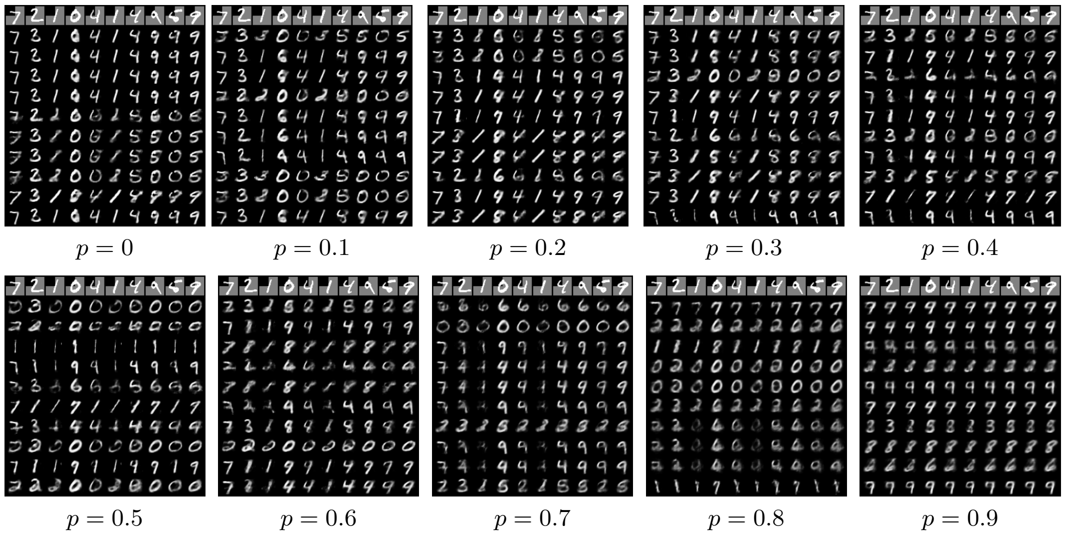

Figure A: Visual examples of images generated from models trained with different levels of salt and pepper noises.

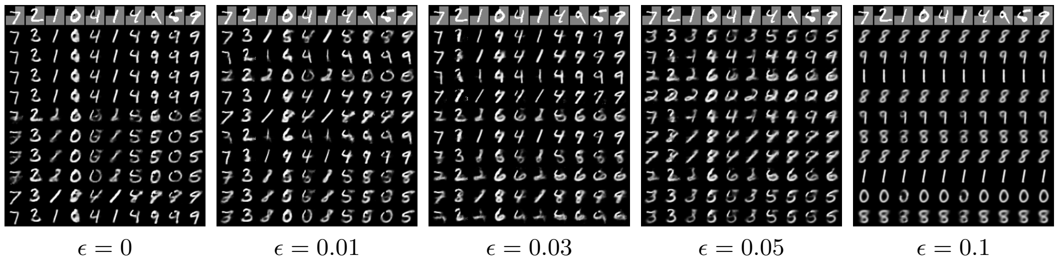

Figure B: Visual examples of images generated from models trained with different levels of adversarial noises.

where $\mathbf{X}_{\text{aug}}$ represents the generated trajectories by perturbing the trajectories in the dataset and $\bar{\mathbf{d}}$ represents the unit vectors for the target deviation directions in the set of $\{$moving forward, backward, left, right$\}$. And the collision regularization loss $\mathcal{L}_{\text{col}}$ is formulated as

$$\mathcal{L}_{\text{col}}(\mathbf{X}_{\text{aug}}, \mathbf{X}) = \frac{1}{n-1} \sum_{i \neq \text{aug}}^{n-1} \frac{1}{\|\mathbf{X}_{\text{aug}} - \mathbf{X}_i\| + 1} \,,$$

We also clip the maximum deviation of the positions so that the trajectories are constrained to be in the lane.

A.4 MPC-based Planner

Planner. In this work, to demonstrate the explicit consequences of the adversarial trajectory, we implement a simple yet effective planner that uses conformal lattice [2] for sampling paths and model predictive control (MPC) [3] for motion planning. We call this planner *MPC-based planner*.

Planning strategy. In this work, we consider a closed-loop planning strategy. Though for the closed-loop planning we have to replay the ground truth trajectories of other agents, we do notice reduced collisions and driving off-road consequences compared to open-loop planning and consider the closed-loop planning fashion meaningful.

B Experiment and Results

B.1 More details on Experimental Setup

Models. Since the adversarial training process is computationally heavy, we use a lightweight version of the AgentFormer in the analysis and ablation studies, namely mini-AgentFormer. In mini-AgentFormer, we (1) remove the map context and (2) reduce the transformer layer from two layers to a single layer. We report the final results for all three models: AgentFormer (AF), mini-AgentFormer (mini-AF) and Social-GAN.

Evaluating impacts to downstream planners. To demonstrate the impacts to downstream planners, we generate adversarial examples for consecutive frames on traffic scenarios in nuScenes dataset. With the MPC-based planner plugged in, we can demonstrate the consequences of the adversarial attacks on trajectory prediction models. We use the prediction results of AgentFormer trained with different methods due to its best performance among the three models. As we mentioned in the main paper, we show 10 cases where the AV collides with other vehicles under attack. We visualize 3 scenarios in the demo video of the supplementary material.

Hyperparameter choices. To select the hyperparameter β, we conduct adversarial training with different β for controlling the regularization. The results are shown in Table A. We find that $\beta = 0.1$ achieves a good trade-off between robustness and clean performance. Therefore, we use $\beta = 0.1$ for the experiments in the rest of the paper.

β	0.01	0.1	0.5	1	10
ADE	2.19	2.29	2.37	2.39	2.57
Robust ADE	3.91	3.76	3.80	3.78	3.79

Table A: Ablation study on different regularization loss weights.

To select the PGD attack step for evaluation, we conduct ablation experiments to show the convergence of different PGD steps. As shown in Figure C, the attack converges at 20 steps. Thus, we select the 20-step PDG attack for the experiments in this paper

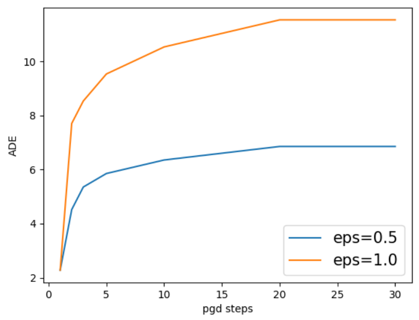

Figure C: PGD step convergence for attack convergence with *Deterministic Attack*. Attack converges around 20 steps.

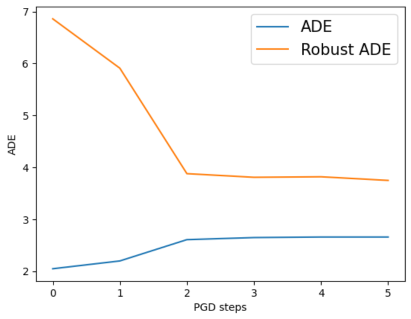

Figure D: PGD step sizes ablation study. We find that except for 1 step PGD adversarial training, adversarial training with all the other step sizes achieves similar results.

To select the PGD attack steps of adversarial training, we conduct experiments on adversarially training the model with different PGD steps. Since we are using PGD attack with adaptive step sizes [4], attacks with any PGD steps are able to fully utilize the attack capability controlled by ϵ. In Figure D, we show that except for the 1-step PGD attack, all other steps show the similar robustness and clean performance.

Evaluation with existing attack [5]. We also evaluate the robust trained model with the existing *search* attack [5]. The results are shown in the Table B. We show that the existing attack increased less prediction error (e.g., 78% less for $\epsilon = 1.0$ on AgentFormer trained with clean data) due to the additional constraints of the attack. We demonstrate that the proposed *RobustTraj* achieves both best robustness and least clean performance degradation, compared to the baselines.

Table B: Evaluation results of the proposed methods and existing methods on the *search* attack proposed by Zhang et al. [5]. mini-AF, AF and SGAN represent mini-AgentFormer, AgentFormer, and Social-GAN respectively. *DA* represents data augmentation with adversarial examples.

Model	Method	ADE		Robust ADE		FDE		Robust FDE	
		0.5	1.0	0.5	1.0	0.5	1.0	0.5	1.0
mini-AF	Clean	2.05	2.05	3.24	4.61	4.41	4.41	6.19	8.02
	Naïve AT	2.75	2.78	3.38	4.46	5.92	5.89	6.97	8.17
	DA + Train-time Smoothing	2.41	2.39	3.28	4.04	5.05	5.09	6.36	8.33
	Detection + Test Smoothing	2.31	2.28	3.26	4.41	4.96	4.91	6.41	8.27
	RobustTraj	2.14	2.11	2.50	2.51	4.36	4.35	5.07	5.11
AF	Clean	1.86	1.86	2.62	3.34	3.89	3.89	5.22	6.72
	Naïve AT	2.52	2.56	2.86	3.52	5.18	5.32	5.68	6.75
	DA + Train-time Smoothing	2.17	2.13	2.72	3.44	4.59	4.51	5.38	6.17
	Detection + Test Smoothing	2.08	2.03	2.58	3.29	4.43	4.26	5.49	6.22
	RobustTraj	1.91	1.95	2.14	2.21	4.02	4.01	4.31	4.31
SocialGAN	Clean	4.80	4.80	6.45	8.08	5.52	5.52	7.78	11.12
	Naïve AT	6.43	6.55	6.99	8.66	7.60	7.53	8.98	10.54
	DA + Train-time Smoothing	5.63	5.61	6.42	8.01	6.44	6.41	8.34	9.88
	Detection + Test Smoothing	5.35	5.37	6.34	8.72	6.12	6.07	7.77	10.21
	RobustTraj	4.95	5.07	5.01	5.49	5.72	5.73	6.68	6.40

B.2 Main Results

We evaluate our methods and existing methods with four metrics in Table C. We observe that the results are consistent where the proposed *RobustTraj* achieves the best results compared to the baselines and existing methods [5].

In the demo video, we visualized scenarios where adversarial attacks on trajectory prediction models lead to collisions on both model trained on clean data and model trained with an existing defense [5], while model trained with *RobustTraj* is able to avoid the collisions.

Table C: Additional evaluation results of the proposed methods and existing methods. mini-AF, AF and SGAN represent mini-AgentFormer, AgentFormer, and Social-GAN respectively. *DA* represents data augmentation with adversarial examples.

Model	Method	ADE		Robust ADE		FDE		Robust FDE	
		0.5	1.0	0.5	1.0	0.5	1.0	0.5	1.0
mini-AF	Clean	2.05	2.05	6.86	11.53	4.41	4.41	13.08	20.15
	Naïve AT	2.75	2.78	5.44	9.20	5.92	5.89	10.13	15.78
	DA	2.31	2.32	5.54	9.32	5.01	4.92	10.09	15.77
	Train-time Smoothing	3.14	3.07	5.67	9.31	6.77	6.61	10.51	17.48
	Test-time Smoothing	2.97	3.07	4.96	8.50	6.49	6.31	9.25	14.13
	DA + Train-time Smoothing	2.41	2.39	5.48	9.00	5.05	5.09	10.23	16.87
	Detection + Test Smoothing	2.31	2.28	5.91	9.85	4.96	4.91	11.49	17.57
	RobustTraj	**2.14**	**2.11**	**3.69**	**3.82**	**4.36**	**4.35**	**7.10**	**7.59**
AF	Clean	1.86	1.86	5.09	8.57	3.89	3.89	9.42	14.41
	Naïve AT	2.52	2.56	3.81	6.81	5.18	5.32	7.11	10.76
	DA	2.10	2.08	4.35	7.22	4.33	4.38	8.08	12.15
	Train-time Smoothing	2.11	2.13	4.19	6.79	4.40	4.46	8.01	11.13
	Test-time Smoothing	2.40	2.41	4.43	7.44	5.02	4.99	8.23	12.47
	DA + Train-time Smoothing	2.17	2.13	4.14	6.62	4.59	4.51	7.85	11.00
	Detection + Test Smoothing	2.08	2.03	4.45	7.59	4.43	4.26	8.01	12.74
	RobustTraj	**1.91**	**1.95**	**2.73**	**2.86**	**4.02**	**4.01**	**5.22**	**5.48**
SGAN	Clean	4.80	4.80	10.52	20.15	5.52	5.52	15.60	24.79
	Naïve AT	6.43	6.55	8.34	14.63	7.60	7.53	13.71	17.93
	DA	5.41	5.40	8.85	17.25	6.16	6.21	13.33	20.83
	Train-time Smoothing	5.50	5.47	8.74	16.51	6.27	6.31	14.03	19.48
	Test-time Smoothing	6.16	6.17	9.05	17.42	7.14	7.07	13.52	21.81
	DA + Train-time Smoothing	5.63	5.61	8.60	16.14	6.44	6.41	13.82	19.08
	Detection + Test Smoothing	5.35	5.37	9.28	17.39	6.12	6.07	13.36	21.59
	RobustTraj	**4.95**	**5.07**	**5.20**	**6.94**	**5.72**	**5.73**	**8.97**	**8.89**
Model	Method	MR		Robust MR		ORR		Robust ORR	
		0.5	1.0	0.5	1.0	0.5	1.0	0.5	1.0
mini-AF	Clean	0.33	0.33	0.77	0.93	0.08	0.08	0.28	0.45
	Naïve AT	0.45	0.45	0.60	0.70	0.11	0.10	0.22	0.34
	DA	0.37	0.38	0.61	0.72	0.09	0.09	0.22	0.35
	Train-time Smoothing	0.50	0.50	0.66	0.79	0.12	0.12	0.22	0.38
	Test-time Smoothing	0.48	0.50	0.56	0.65	0.11	0.11	0.20	0.32
	DA + Train-time Smoothing	0.39	0.40	0.65	0.77	0.09	0.09	0.22	0.37
	Detection + Test-time Smoothing	0.37	0.38	0.69	0.81	0.09	0.09	0.25	0.41
	RobustTraj	**0.34**	**0.36**	**0.54**	**0.54**	**0.08**	**0.08**	**0.10**	**0.11**
AF	Clean	0.29	0.29	0.66	0.88	0.04	0.04	0.16	0.30
	Naïve AT	0.39	0.38	0.51	0.69	0.06	0.06	0.13	0.22
	DA	0.32	0.32	0.56	0.74	0.05	0.05	0.14	0.26
	Train-time Smoothing	0.33	0.33	0.56	0.71	0.05	0.05	0.13	0.25
	Test-time Smoothing	0.37	0.37	0.58	0.77	0.05	0.05	0.14	0.26
	DA + Train-time Smoothing	0.33	0.33	0.54	0.70	0.05	0.05	0.13	0.24
	Detection + Test-time Smoothing	0.32	0.33	0.59	0.76	0.04	0.05	0.14	0.27
	RobustTraj	**0.29**	**0.31**	**0.46**	**0.51**	**0.04**	**0.04**	**0.05**	**0.07**
SocialGAN	Clean	0.40	0.40	0.85	0.99	0.14	0.14	0.52	0.60
	Naïve AT	0.53	0.53	0.63	0.77	0.19	0.19	0.39	0.44
	DA	0.44	0.44	0.72	0.85	0.16	0.16	0.44	0.51
	Train-time Smoothing	0.45	0.45	0.67	0.82	0.16	0.16	0.42	0.50
	Test-time Smoothing	0.51	0.51	0.74	0.85	0.19	0.19	0.45	0.52
	DA + Train-time Smoothing	0.47	0.46	0.66	0.81	0.17	0.17	0.41	0.49
	Detection + Test-time Smoothing	0.45	0.44	0.74	0.89	0.16	0.16	0.46	0.53
	RobustTraj	**0.41**	**0.42**	**0.60**	**0.62**	**0.15**	**0.15**	**0.24**	**0.29**

References

[1] P. Polack, F. Altché, B. d'Andréa Novel, and A. de La Fortelle. The kinematic bicycle model: A consistent model for planning feasible trajectories for autonomous vehicles? In *2017 IEEE Intelligent Vehicles Symposium (IV)*, pages 812–818, 2017. doi:10.1109/IVS.2017.7995816.

[2] M. McNaughton, C. Urmson, J. M. Dolan, and J.-W. Lee. Motion planning for autonomous driving with a conformal spatiotemporal lattice. In *2011 IEEE International Conference on Robotics and Automation*, pages 4889–4895. IEEE, 2011.

[3] E. F. Camacho and C. B. Alba. *Model predictive control*. Springer science & business media, 2013.

[4] F. Croce and M. Hein. Reliable evaluation of adversarial robustness with an ensemble of diverse parameter-free attacks. In *International Conference on Machine Learning*, pages 2206–2216. PMLR, 2020.

[5] Q. Zhang, S. Hu, J. Sun, Q. A. Chen, and Z. M. Mao. On adversarial robustness of trajectory prediction for autonomous vehicles. *arXiv preprint arXiv:2201.05057*, 2022.