2025 International Conference on Intelligent Computing
July 26-29, Ningbo, China
https://www.ic-icc.cn/2025/index.php

Enhance Adversarial Attack against Trajectory Prediction Model by Considering the Characteristics of Trajectory Data and Model

Xin Guo[1],Yucheng Shi[2,✉ [0000-0002-8070-5363]], Zhenghan Gao[2], Guangyao Bai[2], Yufei Gao[1], Lei Shi[1] and Wenwen li[3]

[1] School of Cyber Science and Engineering, Zhengzhou University, China
[2] School of Computer and Artifcial Intelligence, Zhengzhou University, China
[3] China Mobile Online Services Company Limited, China
ieycshi@zzu.edu.cn

Abstract. Recent studies have revealed the vulnerability of trajectory prediction (TP) models to gradient-based adversarial attacks. However, existing gradient-based attacks overlook the characteristics of trajectory data and model, limiting their effectiveness in robustness evaluation. Therefore, we propose a gradient-based attack algorithm considering both rich physical information in data and common characteristics in model. For the data aspect, unlike adversarial attacks in the image, trajectory data carries more physical information than pixels. Considering this, our method introduces momentum in gradient-updates to reserve physical information in previous iterations to keep the generated adversarial trajectory realistic. And in order to search for more possible adversarial trajectories, it updates with different initial states and update with different step sizes. For the aspect of model, unlike convolutional neural networks (CNNs) used for image recognition, trajectory prediction models are typically based on recurrent neural networks (RNNs) which tend to focus more on specific points within the data rather than treating all inputs with equal importance. An attention loss function is designed to guide the attack to focus on that the model concern about. Experiments on three models and two datasets show that our attack algorithm increases mean displacement error(ADE) over 7.94% of the trajectory prediction error compared to previous state-of-the-art gradient-based attack. Our code is open source at Github : https://anonymous.4open.science/r/RMS-PGD.

Keywords: adversarial example, gradient-based attack, trajectory prediction, robustness evaluation

1 Introduction

Autonomous vehicles (AVs) are becoming popular among the public and changing the way we drive today. Trajectory prediction (TP) is a critical component of the prediction module, predicting future trajectories of surrounding vehicles based on history trajectories. As TP generates future results that affect driving behaviors, accurate trajectory predictions must be required for safe AVs driving. Recently, state-of-the-art TP models

This work is supported by the NSFC (under Grant62406293).

are mainly based on deep neural networks [7,13,14,18,28], but their prediction errors may increase sharply when facing adversarial attacks [5,26 ,28]. PGD [15] was initially proposed by Madry et. al to attack the field of image classification, causing the detected images to be misclassified. Due to its fast attack iteration and attack effectiveness, it was first transferred from image classification to trajectory prediction by Zhang et. al [5] to attack trajectory prediction models, resulting in significant errors. Subsequently, Yin et al. [26] also applied PGD and reconstruct the trajectory from scratch to generate smoother adversarial trajectories. The generated trajectories have smaller accelerations which are more realistic.

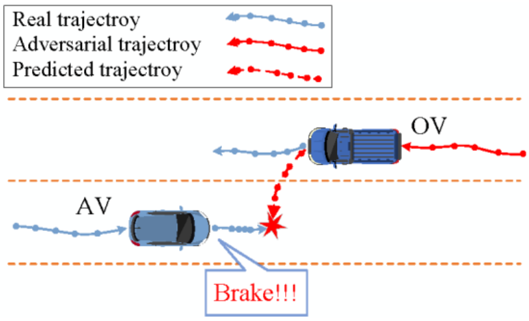


Figure 1. An attack scenario on trajectory prediction: fool the AV cause a brake.

However, existing gradient-based attacks, including PGD, have been directly transferred from image classification to trajectory prediction (TP) making it challenging to generate effective and realistic adversarial trajectories [4,5]. Yin et. al [26] smooth the adversarial trajectories to enhance authenticity and effective. However, the smoothing points may result in exceeding the tracking distance, making it impossible to achieve smoothness. Moreover, the application of trajectory reconstruction is prone to large perturbation value at some time step. This challenge arises due to the lack of consideration in data format and model structure. Specifically, trajectory data is sequential and contains more traffic semantics and physical information than image. Additionally, TP models often bases on recurrent neural networks(RNNs) to handle regression tasks. During predicting, models tend to rely heavily on the last points in sequence, often 'forget' earlier data, a phenomenon known as the long-term dependency issue [11].

Regarding the above challenges, we propose a gradient-based attack called RMS-PGD. In terms of data, we use multiple random initializations and adaptive learning rate to generate more possible adversarial trajectories. And we introduce momentum in gradient-updates to ensure that the physical information is kept between the generated adversarial trajectory and the original trajectory. In terms of models, the attention loss function guide the attack to consider more about the last points of trajectory which the TP models focus more on. Our goal is to generate adversarial trajectories that are both highly effective and realistic. We evaluate RMS-PGD on 6 different combinations of prediction models [13,14,18] and trajectory datasets [3,10]. Compared to the previous state-of-art gradient-based attacks [26,28], RMS-PGD has a 7.94% increase in average displacement error (ADE) and an average increase of 6.06% across four commonly used evaluation metrics of displacement error.

Our main contributions are summarized as follows:

- We propose a gradient based attack on trajectory prediction models considering both the characteristics of trajectory data and model. From the data perspective, this method fully exploits the physical characteristics of trajectory data, including exploring a broader trajectory space and simulating the physical inertia during update iterations. On the model side, a novel attention-based loss function is introduced to guide the attack towards time frames emphasized by model.
- We compared our method with two state-of-the-art attack methods on various prediction models and trajectory datasets in six evaluations. Extensive experimental and visualization results demonstrate that our method is more effective and adheres to fundamental physical constraints with a small perturbation size.
- We explore defensive mechanisms against adversarial examples via data augmentation and trajectory smoothing.
- We conducted ablation experiments, demonstrating the necessity of three improvement methods of RMS-PGD and exploring the sensitivity of the main parameters.

2 Background and Related Work

Trajectory prediction (TP). TP is required by the prediction module of autonomous vehicle (AV) systems (e.g., Baidu Apollo [1], Tesla Autoware [22]). As the perception module of autonomous driving system, TP models are typically based on deep neural networks [13,14,18,27]. It takes the position coordinates of surrounding vehicles from the past few seconds along with additional data such as semantic maps to predict the future spatial coordinates of nearby vehicles. Different ways of manufacturing and utilizing additional data from trajectories can be used to classify different types of models. FQA [13] represents a group of traditional models based on a form of kinematics and statistics to predict trajectories, considering vehicle speed and acceleration. Trajectron++ [18] represents a method that utilizes graphs to process trajectory data. In this model, scenes are represented as spatiotemporal graphs, where nodes represent agents and edges denote their interactions. Local maps are processed by convolutional neural networks (CNNs), trajectories are encoded using Long Short-Term Memory (LSTM) networks, and multimodal solutions are addressed through Conditional Variational Autoencoders (CVAEs). Additionally, the trajectory decoder is based on Gated Recurrent Units (GRUs). GRIP++ [14] introduces a graph-based interactive perception trajectory prediction model. It employs a graph convolutional model composed of several convolutional layers and graph operations to model interactions between vehicles. The output of the graph convolutional model is fed into an LSTM encoder-decoder to predict multiple trajectories.

Although these models perform well in many normal situations and make accurate predictions, for safety reasons, we still need to focus on the robustness of trajectory prediction models, despite achieving high accuracy on different datasets. Especially under malicious adversarial attacks, whether these models can maintain their original prediction accuracy is an important issue.

Adversarial attacks against Trajectory prediction. Adversarial attacks were the first introduced by Szegedy et al. [17] to explore the stability of neural networks. Formally, an adversarial perturbation can be defined as the minimal perturbation R that alters the output f of a given classifier:

$$\begin{gathered} min \ \|R\|_q \\ s.t. \ \ f(X+R) \neq f(X), \end{gathered}$$

where X is the input image, and $\|\|_q$ represents q-norm, which means the Euclidean size when q is equals to 2.

Recent studies have demonstrated various types of attacks against TP models in AVs, including generating adversarial trajectories [4,5,9,17,21,23,29]. Previous studies have employed gradient-based attacks to generate adversarial examples targeting TP modles. In [5], the authors were the first to use PGD to attack TP models. However, the PGD method was directly transferred from the image field without considering the characteristics of the data and model. The generated adversarial trajectories did not cause significant prediction errors, nor did they generate relatively realistic trajectories. Yin et al. reconstruct the trajectory from scratch by fusing future trajectory trends and curvature constraints to generate trajectories with lower acceleration, but it may exceed the tracking distance in some situations [26]. And the generated perturbation may be relatively large at some time step, causing poor invisibility and authenticity because of the search of no hard constraints. The research we conducted is a white box attack, and the scenario described is shown in Figure 1 . It depicts a possible scenario where the other vehicle (OV) travels along a carefully crafted adversarial trajectory. The AV predicts that the OV will deviate and collide with itself, but in reality it did not. AV makes dangerous sudden braking or evasive maneuvers.

3 Method

3.1 Problem statement

Trajectory prediction models observe the state information and predict the future trajectories of surrounding vehicles for each time frame. First, the state of vehicle v at time frame t is denoted as s_t^v. We define the state sequence of vehicle from time frame t_1 to time frame t_2 as $s_{t_1:t_2}^v = \{s_{t_1}^v, \ldots, s_{t_2}^v\}$. At a specific time frame t, we assume there are N surrounding vehicles. The trajectory prediction algorithm collects history trajectory frames with a length of L_H and predicts future trajectory frames with a length of L_P. Thus, the history trajectories of all surrounding vehicles can be defined as $H_t = \{H_t^v = h_{t-L_H+1:t}^v | v \in [1, N]\}$, the predicted trajectories are defined as $P_t = \{P_t^v = p_{t+1:t+L_P}^v | v \in [1, N]\}$, and the actual trajectories of vehicles are denoted as $R_t = \{R_t^v = r_{t+1:t+L_P}^v | v \in [1, N]\}$ (h_t^v, p_t^v, r_t^v, and s_t^v represent the state of vehicle v at time frame t).

The objective of this work is to design a minor perturbation $\Delta_{t-L_H+1:t}^v = [\Delta_{t-L_H+1}^v, \ldots, \Delta_{t-1}^v, \Delta_t^v]$ to generate adversarial history trajectories $\sim h_{t-L_H+1:t}^v = [\sim h_{t-L_H+1}^v, \ldots, \sim h_{t-1}^v, \sim h_t^v]$ to fool TP model predicting trajectories that differ significantly from the true trajectory, where $\sim h_{t-L_H+1:t}^{v} = h_{t-L_H+1:t}^{v} + \mathit{\Delta}_{t-L_H+1:t}^{v} = [h_{t-L_H+1}^{v} + \mathit{\Delta}_{t-L_H+1}^{v}, \ldots, h_{t-1}^{v} + \mathit{\Delta}_{t-1}^{v}, h_t^{v} + \mathit{\Delta}_t^{v}]$. Trajectory prediction model D takes adversarial trajectories as input and outputs the predicted trajectories $\sim P_t^{v} = D(\sim h_{t-L_H+1:t}^{v})$. The goal is to maximize the displacement distance between $\sim P_t^{v} and R_t^{v}$.

To avoid being detected, perturbations added to trajectories must be controlled within a specific range. Imperceptible perturbations can be represented as follows:

$$\mathit{\Delta}_{t-L_H+1:t}^{v} = ||\sim h_{t-L_H+1:t}^{v} - h_{t-L_H+1:t}^{v}||_{\infty} < \varepsilon$$

where ε is the bound which is the half of the lane-width according to the dataset. Moreover, minor perturbations on trajectories may not always align with "normal driving trajectories", thus adversarial trajectories must exhibit the attribute of being "sufficiently natural" obeying physical laws. To meet this two condition, we uses the linear constraint. Iterate through all trajectories, calculating the mean (μ) and standard deviation (σ) of average velocity, horizontal and vertical accelerations, and the derivatives of horizontal and vertical accelerations. For each generated adversarial trajectory, these three metrics should fall within $\mu \pm 3\sigma$. If they fall outside this range, they are mapped back into it, which can be expressed using the following formula:

$$max\ \psi,$$
$$s.t. C(H^{v} + \psi\mathit{\Delta}), 0 <= \psi <= 1,$$

where C denotes linear constraints function. For perturbations $\mathit{\Delta}$ that exceed the constraints, we calculate the scaling factor ψ to transform the perturbations into $\psi\mathit{\Delta}$ to satisfying the constraints and ensure that the generated trajectory does not exceed physical laws.

3.2 Attack Model

We focus on white-box attack, aiming at simulating the model's vulnerabilities under the worst scenarios. The attacker has access to all parameters of the AV prediction model. As the attacker drives the OV close to the AV and prepares to attack, they need to select a future time frames and predict the trajectories of surrounding road vehicles for that period. It is an essential preparation to generate adversarial samples for OV. Then the attacker computes adversarial samples of the AV's trajectory for the chosen future time frames and precisely follows this trajectory (e.g., using software control). Zhang et. al [28] have demonstrated that the impact of attack is primarily determined by the trajectory of the OV itself, ensuring the effectiveness and feasibility of attack. The OV and AV are two randomly selected vehicle trajectories, which ensures that attacks can occur on any two vehicle trajectories and guarantees the generality of this attack method.

3.3 Generate Adversarial Examples by RMS-PGD

RMS-PGD utilizes multiple random initializations with adaptive learning rate to explore wider search space and selects the one that causes the greatest prediction error. It ensures that adversarial trajectories follow physical laws through momentum-updated gradients and guides the attack to concern model’s focus using an attention loss function. As is described in Figure 2 Firstly, random perturbations are added to the history trajectory, and the model generates prediction results. The loss value is computed using the loss function Equ. (2), and the perturbations are updated through an optimizer with adaptive learning rate. A portion of the perturbations is retained to account for the influence in the subsequent updates by the optimizer. Finally, the adversarial trajectory with the strongest attack is selected by initializing the perturbation in times.

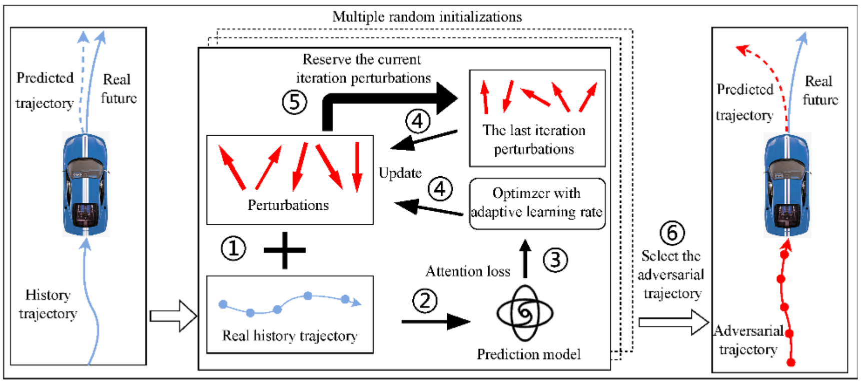


Figure 2. RMS-PGD attack methodology overview.

Multiple random initializations with adaptive learning rate. A direct way to improve the effectiveness of attacks is to search and update adversarial trajectory in a larger search space. To achieve this goal, we need to use the physical information present in trajectory data. We consider that the initialization of perturbations added to the history trajectory inherently represents a certain physical process. For instance, if the perturbation increases the distance between sampled points of a vehicle's history trajectory, it implies a higher speed along that segment of the trajectory. The perturbations added on the trajectory represent the vehicle's speed and direction. In order to utilize the physical information of trajectory data and find a broader search space, we employ multiple random initializations (M times) to replace the single random initialization of PGD. And we design an adaptive learning rate (lr), which was initialized with a larger value at the beginning of the attack and half it when the prediction error no longer increases after a certain number of iterations to avoid getting stuck in local optimal solutions when updating perturbations.

Momentum-updated gradients. To ensure the authenticity of the adversarial trajectory, a certain proportion of the adversarial perturbation generated in the previous iteration is retained in each iteration of the attack. This approach reflects the physical inertia inherent in real-world vehicle dynamics, where changes in driving speed and direction are influenced by the previous state. Since the computation of perturbations relies on gradients, simulating such inertia requires preserving a portion of the gradient from the previous iteration and combining it with the current gradient through weighted summation. It can be represented by the equation:

$$g_{i+1} = \theta g_i + (1-\theta)\nabla_H \mathcal{L}(D(\sim H_i), R),$$

$$\sim H_{i+1} = \prod_{B_\varepsilon}(\sim H_i + lr \cdot norm(g_{i+1})),$$

where g denotes the gradient; i denotes the number of iterations; θ signifies the proportion of retained history gradients termed as history weight. It is noticed that the introduction of momentum in this paper is not aimed at enhancing the transferability of the attack, as commonly observed in image-based attacks. Instead, it serves to simulate physical inertia, ensuring that the generated perturbations are not excessively large and that the trajectories remain realistic.

An attention loss function. The loss function quantifies the difference between the model's original predicted trajectory and the adversarial predicted trajectory. Our objective is to maximize this difference. Additionally, since perturbations are calculated based on the gradient of the input relative to the loss function, the loss function inherently reflects the focal points of the attack. In classification tasks, the focus of attack aligns with model's focus, typically using cross-entropy loss to emphasize correct or incorrect result. Similarly, in trajectory prediction, the attack should also pay more attention to what the trajectory model focuses on. As trajectories data consists of sequential format, the TP models usually incorporate sequence modules and often focus more on the last trajectory time steps and 'forget' the earlier data in sequence. For instance, the Trajectron++ [18] model comprises a LSTM [19] and GRU [8] network, primarily leveraging the last trajectory time steps during prediction. The loss function serve as a crucial objective for attack and guide attacks what to focus on in regression task such as TP. Specifically, ADE (average displacement error) and FDE (final displacement error) [6] are two most commonly used metrics for trajectory prediction models. The ADE focuses more on the overall error performance, while FDE emphasizes the performance of last time frame. A naive method is to combine both factors using a parameter to balance their weights, allowing the attack to focus more on later time frames:

$$\mathcal{L}_{ADE}(P^v, R^v) = \frac{1}{L_P}\sum_{i=1}^{L_P} \|p_i^v - r_i^v\|_q,$$
$$\mathcal{L}_{FDE}(P^v, R^v) = \|p_n^v - r_n^v\|_q,$$
$$\mathcal{L} = \mathcal{L}_{ADE}(P, R) + \beta \cdot \mathcal{L}_{FDE}(P, R), \tag{1}$$

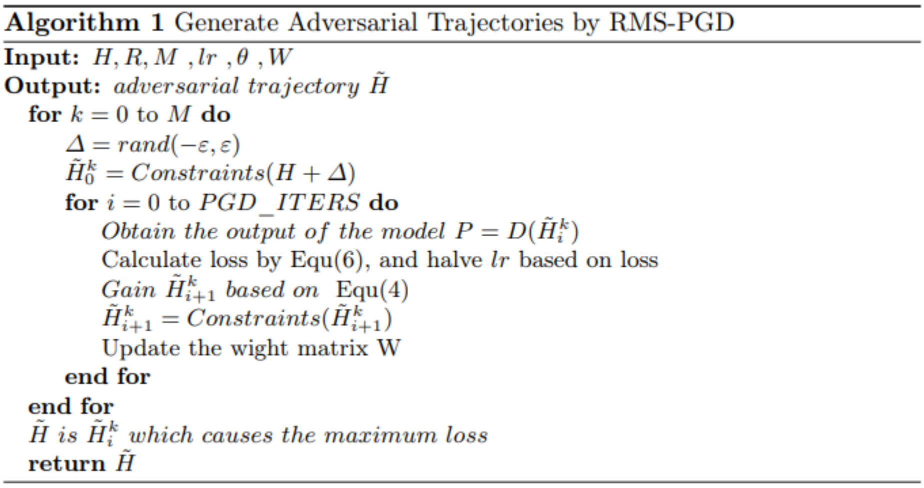
where $0 < \beta$ is a tunable parameter referred to the balance parameter, utilized to adjust the proportion of FDE in combined loss function.

However, the parameter β is difficult to determine, and it does not implement the idea of "the attack focusing on what the model focuses on". The main drawback of Equ.(1) is that it gives a fixed of attention to specific time frame once parameter β is fixed, primarily focusing on the last time frame. This limits the attack's ability to focus on an appropriate number of time frames, as this focus should vary according to the changes in different trajectories rather than being fixed. To address this issue, we allow the attack to autonomously focus on the time step itself. We introduce an attention mechanism where the attack assigns weights related to each agent's time step. The equation for the attention-based loss is as follows:

$$\max_{R,W} Tr(W^T \cdot \|\mathcal{P}_t^v - \mathcal{R}_t^v\|_q) - \lambda_r \|R\|_q + \lambda_\omega \|W\|_q,$$
$$s.t. \sum_t w_t = 1,\, w_t \geq 0, \tag{2}$$

Here, $W \in \mathbb{R}^{L_P \times 1}$ represents the attention weight matrix, where w_t denotes the attention weight for agent v at time step t in prediction. The loss function is defined as the trace(Tr) of the product of $||P_t^v - R_t^v||$ and the transpose of W, plus a regularization on the perturbation with a balancing coefficient λ_r which encourages finding a small perturbation. Also, we discourage uniformity of weights by subtracting the Frobenius norm of W multiplied by a scalar λ_ω. This trace operation Tr actually sums the values of all elements in the matrix. The matrix W is initialized with a uniform distribution, updated with its elements summed to 1, achieved through a single softmax layer [2]. W is updated iteratively, assigning higher weights to targets with a greater possibility of collision. The W and R are jointly optimized for each input sample.

We simplify the definition of history trajectory, true trajectory and predict trajectory of attack vehicle v as H, R and P. $Constraints$ is the Equ.(2) ensuring the speed and acceleration are in a reasonable range of dataset. RMS-PGD can be described as Algorithm1.



Algorithm 1 Generate Adversarial Trajectories by RMS-PGD

```
Input: $H, R, M$ , $lr$ , $\theta$ , $W$
Output: $adversarial\ trajectory\ \tilde{H}$
  for $k = 0$ to $M$ do
    $\Delta = rand(-\varepsilon, \varepsilon)$
    $\tilde{H}_0^k = Constraints(H + \Delta)$
    for $i = 0$ to $PGD\_ITERS$ do
      $Obtain\ the\ output\ of\ the\ model\ P = D(\tilde{H}_i^k)$
      Calculate loss by Equ(6), and halve $lr$ based on loss
      $Gain\ \tilde{H}_{i+1}^k\ based\ on$ Equ(4)
      $\tilde{H}_{i+1}^k = Constraints(\tilde{H}_{i+1}^k)$
      Update the wight matrix W
    end for
  end for
  $\tilde{H}\ is\ \tilde{H}_i^k\ which\ causes\ the\ maximum\ loss$
  return $\tilde{H}$
```

4 Experiments

4.1 Experiment settings

Datasets. We use Apolloscape [1] and nuScenes [3]. They both contains diverse urban scene data, including various weather, lighting, seasonal and traffic conditions. According to the official recommendations, for the nuScenes, we select history trajectory length ($L_H = 4$) and future trajectory length ($L_P = 12$). For the Apolloscape dataset, we select history trajectory length ($L_H = 6$) and future trajectory length ($L_P = 6$). We randomly selected 100 scenes from each dataset as test cases.

Trajectory prediction models. We use three state-of-the-art trajectory prediction models FQA[13], GRIP++[14], and Trajectron++ [18]. The FQA is based on physics modeling to predict trajectories, including considerations of vehicle speed and acceleration. The GRIP++ model utilizes graph models to model the relationship between interaction behavior and trajectory prediction. By constructing interaction graphs, it captures interaction information between different behavioral entities. The Trajectron++ model features handling multi-agent interaction, considering uncertainty and multi-modal prediction, and integrating kinematics and social rules. In addition, for Trajectron++, we select the predicted trajectory with the highest probability as the final result.

Baselines. We select the search-based attack proposed by Zhang et. al [28] and the speed-adaptive attack [26] as two baselines, henceforth referred to as search and sa-attack. The sa-attack causes higher errors by searching without constrains, but it cause larger perturbations. To make fair comparisons, the parameter values shared among the three methods remain consistent, and detailed values will be provided below.

Parameter settings. We employ the SGD optimizer with a multi-random initialization count of N set to 10. The PGD iteration count $iter$ is set to 100, initial learning rate lr is set to 0.01, and a maximum counter Max_Count for halving the learning rate is set to 10. The history weight θ for momentum update gradients is 0.2. The scalar θ and λ_ω for the attention loss function is set to 0.1 and 0.1. We set the threshold of dangerous offset distance ε at 1.8 meters, which corresponds to half the width of a lane. If a vehicle's deviation exceeds half a lane width, it is considered sufficiently hazardous.

Metrics. (1) Average Displacement Error(ADE) / Final Displacement Error(FDE): the average / final displacement error between the model's predictions and the ground-truth which is a commonly utilized metric for assessing the performance of trajectory prediction models. This metric is expressed in meters. (2) Horizontal Displacement Error(HDE) / Vertical Displacement Error(VDE): these metrics are used to evaluate the difference between two trajectories in the horizontal and vertical directions. The HDE can reflect the vehicle's lane departure condition, while the VDE can indicate the extent of the vehicle's forward or backward movement. They can be described as follows:

$$\mathcal{G} = -\frac{1}{L_P}\sum_{\alpha=t+1}^{t+L_P}(p_\alpha^v - r_\alpha^v)^T \cdot F(r_{\alpha+1}^v, r_\alpha^v),$$

where t denotes the time frame index, F is a function generating specific directions. For example, the vertical component can be approximated as $r_{\alpha+1}^v - r_\alpha^v$. (3) Off-Road Rate(ORR): The probability of lane deviation, which is can be determined by horizontal displacement error and lane width. (4) The maximum size of perturbation(P-max): the maximum size of perturbation sizes at each time step. For P-max, smaller perturbations are less likely to be detected, so we hope to gain the small perturbation. The greater the prediction error caused in the targeted model by the attack, the stronger the attack’s effectiveness. And the smaller perturbations introduced by the attack are less likely to be detected, thereby enhancing the trajectory’s authenticity.

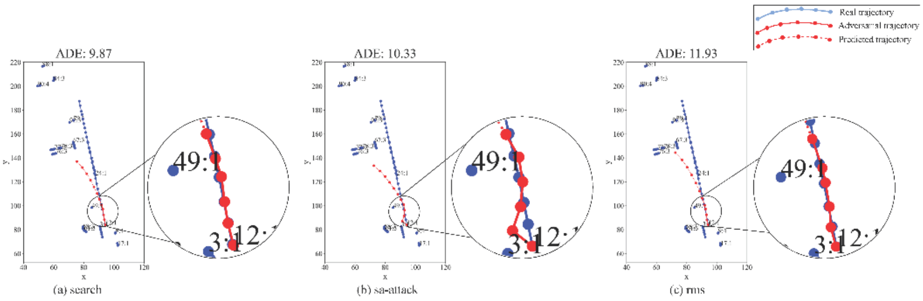
4.2 Main results

Attack effectiveness. Firstly, for each combination of model and dataset, we analyzed the effectiveness of the rms. Table 1 presents the average prediction error before and after attack. Compared to normal predictions, rms increases the average displacement error (ADE) and final displacement error (FDE) by 144.47% and 134.69%.

Furthermore, we utilized metrics such as off-road rate(ORR) and the maximum size of perturbation(P-max) to further describe the model's ability to predict the true trajectory. Our experiments demonstrate that the rms is effective across different datasets and models. The results also indicate that Trajectron++ achieves higher prediction accuracy than the other models, attributable to the integration of heterogeneous data. However, Trajectron++ exhibits higher sensitivity to adversarial attacks compared to Grip++. As shown in Table 1, the rms has a more significant impact on Trajectron++ than on Grip++.

Table 1. The average prediction performance before and after different attacks..

Model & Dataset	Attack	ADE	FDE	HDE	VDE	ORR(%)	P-max
FQA & Apol-loscape	origin	2.37	3.82	0.048	0.387	0%	0
	search	9.05	14.16	2.54	7.30	86%	1.00
	sa-at-tack	9.56	14.62	2.67	7.39	87%	2.36
	rms	10.10	14.85	2.96	7.88	92%	1.00
FQA & nuScenes	origin	5.82	10.00	0.299	0.814	0%	0
	search	7.65	14.14	1.15	3.25	48%	0.33
	sa-at-tack	7.82	14.19	1.20	3.23	52%	0.88
	rms	7.98	14.88	1.20	3.23	57%	0.30
GRIP++ & Apol-loscape	origin	1.97	3.18	0.013	0.016	0%	0
	search	6.75	10.65	2.29	5.01	78%	0.95
	sa-at-tack	6.99	10.23	2.33	5.18	83%	2.36
	rms	7.65	11.10	2.44	5.14	86%	0.82
GRIP++ & nuScenes	origin	5.46	10.30	0.233	1.04	0%	0
	search	8.03	15.02	1.35	3.62	62%	0.32
	sa-at-tack	8.13	15.02	1.55	3.58	67%	1.03
	rms	8.94	15.58	1.67	4.12	76%	0.28
Trajectron++ & Apol-loscape	origin	3.79	5.94	0.15	0.35	0%	0
	search	8.81	13.95	2.46	6.68	82%	0.52
	sa-at-tack	10.11	15.80	2.60	6.70	85%	2.36
	rms	10.39	16.75	2.60	6.93	87%	0.52
Trajectron++ & nuScenes	origin	8.75	17.05	0.33	1.89	0%	0
	search	10.55	19.66	1.05	4.14	57%	0.33
	sa-at-tack	10.32	19.70	1.25	4.07	58%	0.73
	rms	11.58	19.98	1.39	4.21	60%	0.46



(a) search (b) sa-attack (c) rms

Figure 3. The examples under three different attacks with combination of Trajectron++&Apolloscape. In the figure, x and y represent the scene's coordinate axes, measured in meters. Each point on the figure corresponds to the two-dimensional coordinates of a vehicle.

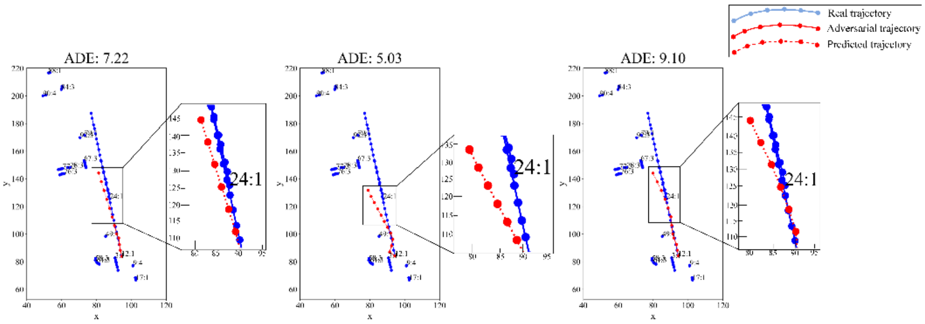
Moreover, we compared the performance of rms against search and sa-attack across these combinations. As shown in Table 1, rms outperforms both search and sa-attack in terms of ADE and FDE metrics, demonstrating stronger attack effectiveness. Furthermore, we used ORR and P-max as metrics to evaluate the authenticity of the generated adversarial trajectories. The results show that rms achieves higher ORR and lower P-max in most combinations, indicating that the adversarial trajectories generated by rms are more likely to cause hazardous deviations and maintain small perturbations at every time frame, which makes them harder to be detected. It is worth noting that sa-attack, which employs trajectory reconstruction, generates larger perturbations among all time steps, causing the highest values across all combinations.

In Figure 3, we visualize the adversarial trajectories generated by the three attack methods under the combination of Trajectron++&Apolloscape. Within the scenarios, our approach generates more effective attacks through small deceptive perturbations, resulting in greater prediction errors. However, the adversarial trajectories generated by sa-attack exhibit larger perturbations at the second history time step, making it easier to be detected. Such trajectories are prone to being flagged as anomalies, which limits their effectiveness in the presence of defensive mechanisms.

Performance after applying defensive mechanisms. We aimed to evaluate the performance of the attack methods when defense mechanisms are applied by the models. Previous works have shown that adversarial training may exhibit poor robust generalization on unseen attacks targeting trajectory prediction [4,11,16,24], so we use the general defense mechanisms: trajectory smoothing and data augmentation. We assume that the attacker has complete knowledge of the mitigation methods and applies the same defense mechanism during white-box attacks for each prediction. Our convolution-based trajectory smoothing approach is differentiable, allowing gradients to directly account for the mitigation's effects. Even if the smoothing method were replaced with a non-differentiable one, the attacker could approximate the gradients using a differentiable function, provided they have sufficient knowledge of the smoothing algorithm. Using three models and the Apolloscape dataset, we tested after the mitigation strategies applied, and the results are illustrated in Table 2. We only compare search and rms in the table, as sa-attack demonstrates a significant performance drop after applying the defense mechanisms as showed in Figure 4. Compared to search, rms continues to cause higher prediction errors. Notably, after the application of defense measures, the maximum perturbation caused by the attacks does not show a significant change.

Table 2. The average prediction performance of three attacks after applying different defensive mechanisms on Apolloscape.

Model	Defensive Mechanisms	ADE	FDE	ORR	P-max
		search/rms	search/rms	search/rms	search/rms
FQA	smooth	6.44/6.64	8.87/10.03	70%/72%	1.00/0.99
	augment	9.30/9.35	14.28/14.36	81%/82%	1.00/1.00
	smooth & augment	6.50/6.54	9.88/9.95	70%/70%	1.00/1.00
Grip++	smooth	7.20/7.31	11.18/11.31	80%/83%	0.95/0.82
	augment	7.25/7.37	11.42/11.58	70%/70%	0.95/0.82
	smooth & augment	7.16/7.24	11.05/11.23	74%/79%	0.95/0.82
Trajectron++	smooth	8.70/9.15	13.10/13.74	76%/79%	0.52/0.52
	augment	8.95/9.35	13.10/13.74	70%/70%	0.52/0.52
	smooth & augment	6.14/6.38	9.27/9.70	70%/70%	0.52/0.52



(a) search (b) sa-attack (c) rms

Figure 4. The average prediction performance of three attacks after applying data augmentation under combination of Trajectron++&Apolloscape.

The effects of data augmentation and trajectory smoothing vary in different models. Results show that compared to no attack, the rms can still cause prediction errors exceeding 206%/197% on ADE/FDE, with horizontal and vertical deviations reaching 2.3/4.9m. The data augmentation generates more complex trajectories alleviating overfitting issues, and trajectory smoothing reduces reliance on the final history time frame. We applied data augmentation during the train-time and trajectory smoothing during the test-time. Data augmentation and trajectory smoothing are effective for the Trajectron++, with an average reduction of 6.26% and 6.76% in prediction error across its six evaluation metrics. Trajectory smoothing is effective for the FQA, reducing prediction error by 29.56%. Figure 4 shows the performance of three methods in apolloscape dataset after applying data augmentation, rms causes the largest error.

Ablation studies. We conducted ablation experiments to explore the contributions of three enhancement methods in rms. We remove one improvement method relative to rms each experiment. As shown in Figure 5, any combination of two improvement methods cause larger errors than the search, and lacking any improvement method makes it difficult to achieve the performance of rms. From the impact of each improvement method, the average error reduction without using attention loss is 5.68%, which is the most significant error reduction. It demonstrates that guiding the attack to focus on the time frames prioritized by model is effective.

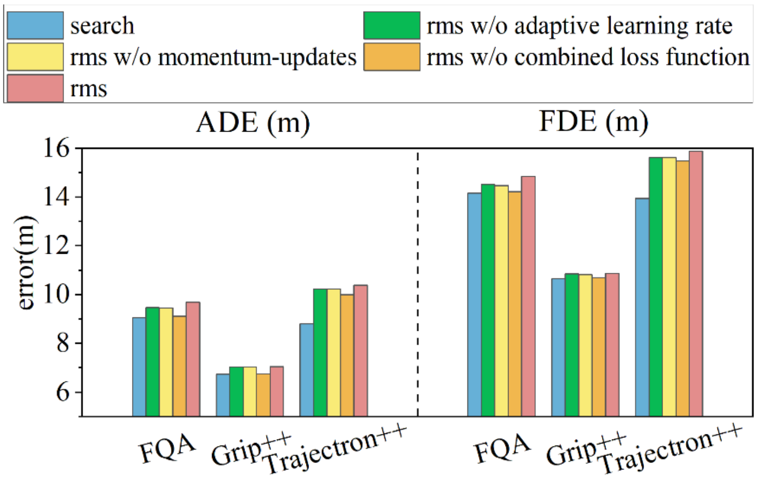


Figure 5. Prediction error (ADE and FDE) caused by removing different methods on the Apolloscape dataset.

We specifically analyzed the impact of multiple random initializations. Increasing the number of initializations expands the search space, causing greater displacement prediction errors, but it still need a increased cost of computation. Experiments demonstrate that 10 random initializations result in a 6.06% higher displacement error compared to a single initialization, while 20 initializations yield an additional 2.55% increase compared to 10. In practical attack scenarios, employing multiple random initializations may hinder timely generation of adversarial trajectories. However, during robustness evaluation phases where time constraints are less stringent, multiple random initializations can be used to maximize model prediction errors. This can achieve providing an accurate assessment of models robustness of the worst-case performance.

5 Conclusion

This paper proposes three improvements upon PGD tailored to the characteristics of trajectory data and prediction models for the malicious interference in vehicle trajectory prediction tasks. We employ multiple random initializations with adaptive learning rate to explore a broader search space. And we incorporate momentum in gradient updates to ensure that consecutive iterations produce trajectories conforming to certain physical rules. Last but most importantly, we propose an attention-based loss function to guide the attack focus on the time frames emphasized by the model during prediction. Experiment results validate the effectiveness and authenticity of our method.

Acknowledgments. A third level heading in 9-point font size at the end of the paper is used for general acknowledgments, for example: This study was funded by X (grant number Y).

References

1. Baidu: Baidu-apollo (2024), https://www.apollo.auto/autonomous-driving
2. Bridle J S. Probabilistic interpretation of feedforward classification network outputs, with relationships to statistical pattern recognition[M]//Neurocomputing: Algorithms, architectures and applications. Berlin, Heidelberg: Springer Berlin Heidelberg, 1990: 227-236.
3. Caesar H, Bankiti V, Lang A H, et al. nuscenes: A multimodal dataset for autonomous driving[C]//Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. 2020: 11621-11631.
4. Cao Y, Xiao C, Anandkumar A, et al. Advdo: Realistic adversarial attacks for trajectory prediction[C]//European Conference on Computer Vision. Cham: Springer Nature Switzerland, 2022: 36-52.
5. Cao Y, Xu D, Weng X, et al. Robust trajectory prediction against adversarial attacks[C]//Conference on robot learning. PMLR, 2023: 128-137.
6. Chandra R, Bhattacharya U, Roncal C, et al. Robusttp: End-to-end trajectory prediction for heterogeneous road-agents in dense traffic with noisy sensor inputs[C]//Proceedings of the 3rd ACM Computer Science in Cars Symposium. 2019: 1-9.
7. Chen J, Wang W, Chen J, et al. Dynamic vehicle graph interaction for trajectory prediction based on video signals[C]//ICASSP 2023-2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). IEEE, 2023: 1-5.
8. Cho K, Van Merriënboer B, Gulcehre C, et al. Learning phrase representations using RNN encoder-decoder for statistical machine translation[J]. arXiv preprint arXiv:1406.1078, 2014.
9. Gibson M, Babazadeh D, Tomlin C, et al. Hacking predictors means hacking cars: Using sensitivity analysis to identify trajectory prediction vulnerabilities for autonomous driving security[C]//6th Annual Learning for Dynamics & Control Conference. PMLR, 2024: 745-757.
10. Huang X, Cheng X, Geng Q, et al. The apolloscape dataset for autonomous driving[C]//Proceedings of the IEEE conference on computer vision and pattern recognition workshops. 2018: 954-960.
11. Jiao R, Liu X, Sato T, et al. Semi-supervised semantics-guided adversarial training for robust trajectory prediction[C]//Proceedings of the IEEE/CVF international conference on computer vision. 2023: 8207-8217.
12. HAN J, LI Y. Survey of Catastrophic Forgetting Research in Neural Network Models[J]. Journal of Beijing University of Technology, 2021, 47(5): 551-564.
13. Kamra N, Zhu H, Trivedi D K, et al. Multi-agent trajectory prediction with fuzzy query attention[J]. Advances in Neural Information Processing Systems, 2020, 33: 22530-22541.
14. Li X, Ying X, Chuah M C. Grip++: Enhanced graph-based interaction-aware trajectory prediction for autonomous driving[J]. arXiv preprint arXiv:1907.07792, 2019.
15. Madry A, Makelov A, Schmidt L, et al. Towards deep learning models resistant to adversarial attacks[J]. arXiv preprint arXiv:1706.06083, 2017.
16. Rice L, Wong E, Kolter Z. Overfitting in adversarially robust deep learning[C]//International conference on machine learning. PMLR, 2020: 8093-8104.

17. Saadatnejad S, Bahari M, Khorsandi P, et al. Are socially-aware trajectory prediction models really socially-aware?[J]. Transportation research part C: emerging technologies, 2022, 141: 103705.
18. Salzmann T, Ivanovic B, Chakravarty P, et al. Trajectron++: Dynamically-feasible trajectory forecasting with heterogeneous data[C]//Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XVIII 16. Springer International Publishing, 2020: 683-700.
19. Shi X, Chen Z, Wang H, et al. Convolutional LSTM network: A machine learning approach for precipitation nowcasting[J]. Advances in neural information processing systems, 2015, 28.
20. Szegedy C, Zaremba W, Sutskever I, et al. Intriguing properties of neural networks[J]. arXiv preprint arXiv:1312.6199, 2013.
21. Tan K, Wang J, Kantaros Y. Targeted adversarial attacks against neural network trajectory predictors[C]//Learning for Dynamics and Control Conference. PMLR, 2023: 431-444.
22. Tesla: Tesla autopilot (2024), https://www.tesla.cn/AUTOPILOT
23. Wang J, Pun A, Tu J, et al. Advsim: Generating safety-critical scenarios for self-driving vehicles[C]//Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2021: 9909-9918.
24. Wu D, Xia S T, Wang Y. Adversarial weight perturbation helps robust generalization[J]. Advances in neural information processing systems, 2020, 33: 2958-2969.
25. Wu W, Deng X. Motion Latent Diffusion for Stochastic Trajectory Prediction[C]//ICASSP 2024-2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). IEEE, 2024: 6665-6669.
26. Yin H, Li J, Zhen P, et al. SA-Attack: Speed-adaptive stealthy adversarial attack on trajectory prediction[C]//2024 IEEE Intelligent Vehicles Symposium (IV). IEEE, 2024: 1772-1778.
27. Yuan Y, Weng X, Ou Y, et al. Agentformer: Agent-aware transformers for socio-temporal multi-agent forecasting[C]//Proceedings of the IEEE/CVF international conference on computer vision. 2021: 9813-9823.
28. Zhang Q, Hu S, Sun J, et al. On adversarial robustness of trajectory prediction for autonomous vehicles[C]//Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2022: 15159-15168.
29. Zheng Z, Ying X, Yao Z, et al. Robustness of trajectory prediction models under map-based attacks[C]//Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision. 2023: 4541-4550.