On Adversarial Robustness of Trajectory Prediction for Autonomous Vehicles

Qingzhao Zhang[1], Shengtuo Hu[1], Jiachen Sun[1], Qi Alfred Chen[2], Z. Morley Mao[1]
[1]University of Michigan, [2]University of California, Irvine
{qzzhang, shengtuo, jiachens, zmao}@umich.edu alfchen@uci.edu

Abstract

Trajectory prediction is a critical component for autonomous vehicles (AVs) to perform safe planning and navigation. However, few studies have analyzed the adversarial robustness of trajectory prediction or investigated whether the worst-case prediction can still lead to safe planning. To bridge this gap, we study the adversarial robustness of trajectory prediction models by proposing a new adversarial attack that perturbs normal vehicle trajectories to maximize the prediction error. Our experiments on three models and three datasets show that the adversarial prediction increases the prediction error by more than 150%. Our case studies show that if an adversary drives a vehicle close to the target AV following the adversarial trajectory, the AV may make an inaccurate prediction and even make unsafe driving decisions. We also explore possible mitigation techniques via data augmentation and trajectory smoothing.

1. Introduction

Autonomous vehicles (AVs) are transforming the transportation systems. AV is a complex system integrating a pipeline of modules such as perception of obstacles, planning of driving behaviors, and controlling of the physical vehicle [1, 3]. Specifically, trajectory prediction in the perception module predicts the future trajectories of nearby moving objects. The prediction is essential for the planning module and affects AV's driving behavior. Therefore, accurate trajectory prediction is critical for safe AV driving.

Many studies propose trajectory prediction models based on deep neural networks [8, 10, 11, 13, 21, 23, 24, 27, 32, 33, 35, 38, 41, 42]. They evaluate the models on benchmarks collected from real world using the average ℓ_2 distance between ground truth and predicted trajectories as the key metric. However, few studies evaluate trajectory prediction models from the perspective of security or analyze the robustness against adversarial examples. For trajectory prediction, if the adversary can control the position of a vehicle close to the target AV, e.g., by driving the vehicle along a crafted trajectory, the adversary can influence the AV's trajectory prediction and driving behaviors.

To bridge this gap, we propose new white/black box adversarial attacks on trajectory prediction, which adds minor perturbation on normal trajectories to maximize the prediction error. Compared with adversarial attacks on image/video classification, attacking trajectory prediction is unique in two aspects. First, the attack requires naturalness [40] of the adversarial examples. Adversarial trajectories are natural if they obey physical rules and are possible to happen in the real world. With naturalness, the trajectories can be reproduced by the attacker-controlled vehicle in the real world and cannot be easily classified as anomaly by AVs. To realize naturalness, we enforce constraints on physical properties (e.g., velocity and acceleration) of the perturbed trajectory during optimization solving. Second, we need to define optimization objectives that are semantically-attractive for attackers targeting trajectory prediction. To this end, we find multiple attractive attack dimensions can co-exist even for the same scenario (e.g., causing the predicted trajectory to deviate laterally or longitudinally are both of interest to attackers in AV context). Thus, in our attack design we consider different metrics of prediction error as optimization objectives, e.g., average lateral/longitudinal deviation to four different directions.

We evaluate the proposed attacks on 10 different combinations of prediction models [18, 20, 30] and trajectory datasets [2, 5, 16]. The results show that the adversarial perturbation can substantially increase the prediction error by around 150%. 62.2% of attacks cause prediction to deviate by more than half of the lane width, which are likely to significantly alter AV's navigation decisions. In addition, we thoroughly analyze how various factors impact the attack results and make recommendations for improvements of implementing trajectory prediction such as leveraging map information and driving rules. We also explore mitigation mechanisms against adversarial trajectories through data augmentation and trajectory smoothing, which reduce the prediction error under attacks by 28%. In general, our work exposes the necessity of evaluating the worst-case performance of trajectory prediction. The hard cases involving natural but adversarial trajectories generated by attacks have critical safety concerns (e.g., causing hard brakes or even collisions) as demonstrated by our case studies.

Our main contributions are summarized as follows:

- We propose the *first* adversarial attack and adversarial robustness analysis on trajectory prediction for AVs considering real-world constraints and impacts.
- We report a thorough evaluation of adversarial attacks on various prediction models and trajectory datasets.
- We explore mitigation methods against adversarial examples via data augmentation and trajectory smoothing.

2. Background and Related Work

Autonomous vehicles. The autonomous vehicle is a cyber-physical system where the perception module learns the surrounding environment through sensors, the planning module makes decisions on the driving behaviors, and the control module physically operates the vehicle. Within this pipeline, trajectory prediction is required by AV systems (e.g., Baidu Apollo [3], Autoware [1]) as a part of the perception module. It predicts the future trajectories of nearby moving objects such as vehicles and pedestrians, which are important inputs for the planning module. Hence, accurate trajectory prediction is critical for safe AV driving.

Trajectory prediction models. Trajectory prediction, for AVs especially, is the problem of predicting future spatial coordinates of various road agents such as vehicles and pedestrians. These models are usually deep neural networks accepting spatial coordinates of observable road agents in the past a few seconds as the primary input and may extend the input by leveraging auxiliary features (e.g., vehicles' heading) [11,20], interaction among road agents [10,18,42], physical dynamics [30], or semantic maps [13,25,27,30] to improve the accuracy of prediction. The existing evaluation metrics include Average Displacement Error (ADE), Final Displacement Error (FDE) [9], off-road rate [5], etc. These metrics reflect the average performance of trajectory prediction in the testing datasets. Instead, we focus on adversarial robustness and the worst-case performance of trajectory prediction algorithms.

Adversarial robustness. Since deep learning models are generally vulnerable to adversarial examples, various studies analyze the adversarial robustness of neural networks [7, 12, 14, 37, 39]. In AV systems, studies show that tasks such as object detection [6,36], object tracking [17], and lane detection [31] can be affected by perturbing sensor signals or adding physical patches. However, no existing work studied the adversarial robustness of trajectory prediction.

Testing of trajectory prediction. AdvSim [15] leverages adversarial machine learning (AML) to generate high-risk driving scenarios for end-to-end testing AV systems. Instead, we aim to understand the vulnerabilities in prediction algorithms specifically. Saadatnejad et. al. [29] analyzes the inaccuracy of attention mechanisms in trajectories prediction algorithms but does not consider safety impact on real-world applications. We are the first to bridge adversarial robustness and real-world systems like AVs.

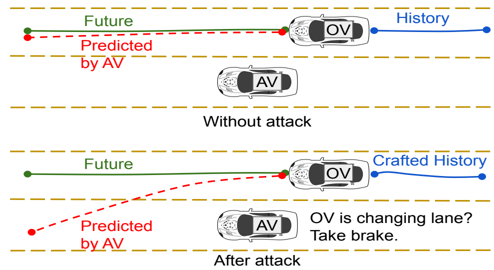

Figure 1. An example of attack scenarios on trajectory prediction: spoofing lane-changing behavior.

3. Problem Formulation

In this section, we first introduce the formulation of the trajectory prediction task (§3.1). We then propose the attack model (§3.2), metrics of attack impact (§3.3).

3.1. Trajectory Prediction

In this work, we focus on trajectory prediction which is executed repeatedly at a fixed time interval and makes one prediction at each time frame according to the current/history state of all observable objects (i.e., vehicles/pedestrians). First, we denote the state of an object i at time t as s_t^i, including information of spatial coordinates and other optional features. The trajectory of object i is a sequence of object states from time frame t_1 to t_2 (inclusively), denoted as $s^i_{t_1:t_2} = \{s^i_{t_1}, \dots, s^i_{t_2}\}$. At each time frame, the prediction algorithm consumes the history trajectories of objects to predict their future trajectories, which are optimized towards having the same distribution as the ground-truth future trajectories. At time frame t, we denote the number of observed objects as N and denote the number of time frames in history and future trajectories as L_I and L_O respectively. Then history trajectories are $H_t = \{H_t^i = s^i_{t-L_I+1:t} | i \in [1, N]\}$; ground-truth future trajectories are $F_t = \{F_t^i = s^i_{t+1:t+L_O} | i \in [1, N]\}$; predicted trajectories are $P_t = \{P_t^i = p^i_{t+1:t+L_O} | i \in [1, N]\}$ (p_t^i is the predicted state of object i at time t).

3.2. Attack Model

In this paper, we focus on the setting where the adversary drives one vehicle called "the other vehicle" (OV) along a crafted trajectory. The AV observes the OV and applies iterative trajectory prediction which produces the predicted trajectory of the OV at each time frame. The adversary controls the OV's whole trajectory to maximize the prediction error or make the AV take unsafe driving behaviors. Figure 1 demonstrates one example of the attack. By driving along a crafted trajectory, the OV seems be changing its lane in the AV's prediction while the OV is actually driving straightly. Given the high-error prediction, the AV takes brake to yield the OV. If the AV brakes on the highway, it is a serious safety hazard that may cause rear-end collisions.

In a realistic attack scenario, the adversary needs to access all parameters or only APIs of the prediction model equipped by the victim AV for white-box and black-box settings respectively. When the adversary drives the OV close to the AV and is ready to attack, he/she first selects a future period and predicts the trajectories of surrounding on-road objects in that period, which are necessary inputs for generating the adversarial trajectory. The adversary then computes the adversarial trajectory for that future period and drives exactly on it (e.g., the adversary can control the OV using software). Though the generation of adversarial trajectories relies on predicted future knowledge that is not guaranteed to be accurate, the attack is still effective since the attack impact is mostly determined by the OV's trajectory itself. The experimental evidences are in the appendix.

In addition, adversarial trajectories must satisfy the following requirements to ensure naturalness. First, the trajectory obeys the physical rules. The physical properties (e.g., velocity, acceleration) must be bounded so that real vehicles can reproduce the trajectory. Second, the trajectory represents normal driving behavior instead of ruthless driving.

3.3. Evaluation Metrics

We use six metrics to evaluate the prediction error. Also, an attack is effective if the prediction error is significantly raised after the adversarial perturbation. We first include two metrics that are commonly used in related works [9]. (1) Average displacement error (**ADE**). The average of the root mean squared error (RMSE) between the predicted and ground-truth trajectory. (2) Final displacement error (**FDE**). The RMSE between the predicted and ground-truth trajectory position at the last predicted time frame.

However, the above two metrics are not sufficient to evaluate the impact of targeted attacks. For instance, to spoof lane-changing behavior to left (Figure 1), the predicted trajectory should have a deviation to left. Similarly, to spoof a fake acceleration, the deviation should be towards the front of the longitudinal direction. For the above attacks, only the deviation to a specific direction is counted as an effective attack impact. Therefore, we design four extra metrics including the average deviation towards **left/right** side of lateral direction and **front/rear** side of longitudinal direction. The metrics are formally defined as:

$$D(t, n, R) = \frac{1}{L_O} \sum_{\alpha=t+1}^{t+L_O} (p_\alpha^n - s_\alpha^n)^T \cdot R(s_{\alpha+1}^n, s_\alpha^n), \quad (1)$$

where t denotes the time frame ID, n denotes the target vehicle ID, p and s are binary vectors representing predicted and ground-truth vehicle locations respectively, and R is a

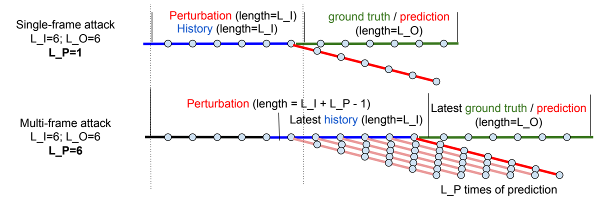

Figure 2. Illustration of single/multi-frame attack. The dots are observed/predicted trajectory locations in different time frames.

function generating the unit vector to a specific direction. We approximate longitudinal direction as $s_{\alpha+1}^n - s_\alpha^n$.

According to our case studies, we believe that half of the lane width (e.g., about 1.85 meters in datasets we use) is a threshold of deviation to cause real-world impacts in a high probability. If the average deviation exceeds the threshold, the last predicted trajectory location is likely on a different lane, which may lead to different decisions of the AV.

4. Adversarial Example Generation

We design white box and black box attacks against trajectory prediction following definitions in §3.1.

Perturbation. We generate the adversarial trajectory by adding minor perturbation on normal trajectories. The perturbation changes the spatial coordinates (x-y location) and features that are calculated from locations (e.g., velocity) correspondingly. As shown in Figure 1, perturbation is applied on the history trajectory in length of L_O to control the prediction. Since only the prediction of the current time frame is considered, we name this attack **single-frame** attack. However, real-world safety issues usually happen in a longer time sequence. Therefore, we generalize the attack to **multi-frame**, as shown in Figure 2. We define parameter L_P to represent the number of predictions considered in the attack objective. Given L_P, one attack scenario include $L_I + L_O + L_P - 1$ time frames. We apply perturbation on the first $L_I + L_P - 1$ time frames so that predicted trajectories on time frames $\{L_I, L_I + 1, \ldots, L_I + L_P - 1\}$ (L_P frames in total) are controlled by adversarial trajectories. We maximize the total prediction error in these L_P time frames to launch the multi-frame attack.

Objective. The optimization has six different objective functions corresponding to the six metrics of prediction error. The objective is the negation of the average prediction error in all considered time frames:

$$L(n, f) = -\frac{1}{L_P} \sum_{\alpha=L_I}^{L_I+L_P-1} f(P_\alpha^n, F_\alpha^n), \quad (2)$$

where f denotes the function calculating one of the six metrics; n is ID of the target vehicle (i.e., OV); P and F represents predicted and future trajectories (§3.1).

Hard constraints of perturbation. As mentioned in §3.2, the perturbed trajectories must be physically feasible and not perform dangerous driving behaviors. To enforce this requirement, we design constraints that need to be satisfied by any perturbation.

First, the bounds of physical properties. We traverse all trajectories in the testing dataset to calculate the mean (μ) and standard deviation (σ) of (1) scalar velocity, (2) longitudinal/lateral acceleration, and (3) derivative of longitudinal/lateral acceleration. For each μ and σ, we check that the value of perturbed trajectories does not exceed $\mu \pm 3\sigma$. Assuming the physical properties are in the normal distribution, this range covers 99.9% of the dataset but excludes outliners. We also manually check that the bounds are physically reasonable. In addition, when checking the physical constraints, three ground-truth trajectory points are involved before and after the perturbed part thus the boundary of normal and perturbed trajectories is natural.

Second, the bound of deviation on each trajectory location. We set the maximum deviation as 1 meter by default. Given that the urban lane width is around 3.7 meters in datasets we use and the width of cars is about 1.7 meters, the 1-meter deviation is an upper bound for a car not shifting to another lane if it is normally driving in the center of the lane. The bound (1) rules out ruthless driving by preserving the original driving behavior and (2) is a parameter for tuning the stealth level of the attack.

We enforce the constraints by reducing the perturbation whenever the constraints are violated. Given the perturbation Δ, the history trajectory of the target vehicle H^n, and the constraint function C, we calculate the maximum coefficient $0 \leq \theta \leq 1$ which reduces the perturbation to $\theta\Delta$ while $\theta\Delta$ satisfies all constraints. Formally speaking, the calculation of θ is an optimization problem:

$$\begin{aligned} &\max \theta \\ &\text{s.t. } C(H^n + \theta\Delta) \wedge 0 \leq \theta \leq 1 \end{aligned} \tag{3}$$

White box optimization. We design our white box optimization method based on Projected Gradient Descent (PGD) [22]. The process can be summarized as follows.

The perturbation is initialized randomly. In each iteration, we first enforce the hard constraint on the current perturbation following Equation 3 and then add the perturbation on the original history trajectories of the target vehicle n which is observed by the AV. Then L_P times of prediction is executed on the perturbed trajectory data and the loss of the iteration is calculated using Equation 2. Next, the algorithm updates the perturbation using gradient descents. Finally, the algorithm produces the best perturbation which can transform the original scenario to the worst-case scenario with the maximized prediction error.

Black box optimization. Methods based on gradient descent are not always feasible because the trajectory prediction models may have non-differentiable layers. Hence, we design a black box attack method based on Particle Swarm Optimization (PSO) [19] which requires only the API of model inference instead of gradients. PSO is an optimization method by iteratively improving solution candidates (i.e., particles) with regards to a given measure of quality in the search space. In this case, each particle is one candidate of perturbation, the measure of quality is defined by objective function (Equation 2), and the search space is defined by the hard constraints (Equation 3).

5. Mitigation Mechanisms

From our observation of attack results (§6), the adversarial trajectories frequently change acceleration, which is a rare pattern in normal trajectories. Based on this phenomenon, we design mitigation methods as follows.

Data augmentation. Since normal trajectories in the training dataset are mostly smooth with stable acceleration, adversarial trajectories have a different data distribution. Hence, we apply data augmentation to inject adversarial patterns in the training data. During the training, we add random perturbation on randomly selected trajectories while the perturbation satisfies the hard constraints defined in §4. We do not adopt adversarial training because of its limitations such as the high cost of training and poor generality in terms of attack goals.

Train-time trajectory smoothing. Since the unstable velocity or acceleration is a key pattern of adversarial trajectories, we can partially remove the adversarial effect by smoothing the trajectories. We apply trajectory smoothing on both training and testing data. There are various choices of smoothing algorithms and we use a simple linear smoother based on convolution in our experiments. This mitigation relies on the physical properties of trajectories instead of gradient obfuscation [4]. Hence, it does not matter if the attacker knows the gradients of the smoothing.

Test-time detection and trajectory smoothing. The above two mitigation methods modify the distribution of training data so that one needs to retrain the model. To make the mitigation easier to deploy, we propose another method that only smooths trajectories during inference time if the trajectory is detected as adversarial. We design two methods for detecting adversarial trajectories. First, SVM classifier [34]. We extract the magnitude and direction of the acceleration as features to fit an SVM model for classifying normal and adversarial trajectories. Second, rule-based detector. We calculate the variance of acceleration over time frames and if the variance is higher than a threshold the trajectory is detected as adversarial.

6. Experiments

In this section, we evaluate proposed attack/mitigation methods and analyze the results.

Table 1. Summary of datasets.

Name	Scenario	Map	L_I	L_O	Freq. (Hz)
Apolloscape [16]	Urban	×	6	6	2
NGSIM [2]	Highway	×	15	25	5
nuScenes [5]	Urban	√	4	12	2

Table 2. Summary of models.

Name	Input features	Output format	Network
FQA [18]	location	single-prediction	LSTM
GRIP++ [20]	location + heading	single-prediction	Conv + GRU
Trajectron++ [30]	location + physical dynamics + map	multi-prediction	Conv + LSTM + GRU + GMM

6.1. Experimental Setting

Datasets. We summarize the characteristics of the three datasets we use in Table 1. We select history trajectory length (L_I) and future trajectory length (L_O) following the recommendation from the dataset's authors. We randomly select 100 scenarios as test cases from each dataset.

Models. We summarize three prediction models we use in Table 2, which are open-source state-of-the-art models at the time of the experiments. Note that *Trajectron++* model produces multiple predicted trajectories with probabilities (i.e., multi-prediction) and we select the trajectory with the highest probability as the final result. Also, *Trajectron++* requires the semantic map as input which is only available in *nuScenes* dataset. Therefore, we prepare two versions of *Trajectron++*. *Trajectron++* (w/o map) is evaluated on all datasets while *Trajectron++* (w/ map) is evaluated on *nuScenes*. For each combination of models and datasets, we train the model using fine-tuned hyperparameters.

Implementation details. Our implementation is open sourced at `https://github.com/zqzqz/AdvTrajectoryPrediction`. For the PGD-based white box attack, we use Adam optimizer with a learning rate of 0.01 and set the maximum iteration to 100. For PSO-based black box attack, we set the number of particles to 10, inertia weight to 1.0, acceleration coefficients to (0.5, 0.3), and the maximum iteration to 100. For mitigation, we use convolution-based linear smoother for trajectory smoothing and the SVM implementation in *scikit-learn* [26] for anomaly detection. More details are in the appendix.

6.2. Attack Results

General results. For each model & dataset combination, each test case, and each perturbation objective, we first execute the white-box attack in the setting of single-frame prediction ($L_P = 1$). We assume the attacker acknowledges history trajectories of all objects on-road and discuss a more realistic setting in the appendix. We present the average prediction error before and after perturbation in Table 3. Generally, the white-box adversarial perturbation is effective on all models and datasets. On average, ADE/FDE is increased by 167%/150%. The lateral/longitude deviation reaches 2.03/3.84 meters and 62.2% of attacks are likely to cause real-world impact since their average deviations are larger than half of the lane width (i.e., 1.85 meters).

Naturalness. The adversarial trajectories can naturally happen in the real world in a reasonable probability. One characteristic of adversarial trajectories is that they often change vehicle velocity or acceleration. However, a small fraction of normal trajectories share the same pattern and are indistinguishable from adversarial trajectories. The hard constraint (§4) also ensures that adversarial trajectories are physically feasible to reproduce by driving a real car. Hence, the adversarial attack can be regarded as a method to discover the worst but realistic cases of prediction.

Next, we will analyze factors that affect adversarial robustness in §6.2.1, §6.2.2, and §6.2.3. The analysis uses experiments on dataset *Apolloscape* as supporting evidence. As a baseline, in the single-frame white-box attack on *Apolloscape*, ADE/FDE is increased by 239%/206% while lateral/longitude deviation reaches 2.49/6.31 meters on average (53.6% of deviations are larger than 1.85 meters). If not specified, the percentage rise/drop of prediction error is the average of six metrics.

6.2.1 Different Scenarios

High-acceleration scenarios. The prediction error is generally high in scenarios where vehicles have high acceleration. Taking metric ADE as an example, the prediction error in high-acceleration scenarios (average acceleration $> 1m/s^2$) w/o and w/ perturbation are 31% and 20% higher than low-acceleration scenarios respectively. Typical high-acceleration scenarios include turning at intersections or stopping at stop lines (details in the appendix). All three models cannot foresee the driver's behavior in such scenarios thereby have extremely high prediction errors. The models need specific knowledge about traffic rules to make the model robust in those hard cases.

Density of traffic. Prediction models commonly encode the interaction among objects as a graph representation to improve prediction accuracy. To study the factor of traffic density, we repeat the white box attack on three models using *Apolloscape* dataset but drops 50% or 100% of objects other than the target vehicle. However, the prediction error under the white-box attack appears independent of traffic density. The result shows that the worst-case prediction error under attacks is mostly determined by the target vehicle's trajectory rather than the surrounding objects.

6.2.2 Different Models

Input representation. Features independent of trajectory locations (e.g., maps) help to improve adversarial robustness. Comparing *Trajectron++* w/ map against w/o map, ADE on original and perturbed trajectories are reduced by 22% and 31% respectively. By adding map information as input features, the weight on perturbed features is lower so that the prediction result is less sensitive to the perturbation.

Table 3. Average prediction error before and after single-frame black box attack.

Model	Dataset	ADE (m)	FDE (m)	Left (m)	Right (m)	Front (m)	Rear (m)
		Normal / Attack	Normal / Attack	Normal / Attack	Normal / Attack	Normal / Attack	Normal / Attack
GRIP	Apolloscape	1.97 / 7.14	3.18 / 10.74	-0.0128 / 2.65	0.0128 / 2.38	-0.0154 / 4.80	0.0154 / 5.49
GRIP	NGSIM	7.29 / 9.24	12.8 / 16.5	-0.175 / 1.10	0.175 / 0.311	-5.30 / -3.76	5.30 / 5.85
GRIP	nuScenes	5.46 / 8.32	10.3 / 15.4	0.233 / 1.53	-0.233 / 1.44	-1.04 / 1.50	1.04 / 3.76
FQA	Apolloscape	2.37 / 5.64	3.82 / 8.78	0.0479 / 2.10	-0.0479 / 1.86	-0.387 / 3.11	0.387 / 3.84
FQA	NGSIM	5.44 / 7.05	9.45 / 12.2	0.205 / 1.19	-0.205 / 0.787	-1.11 / 2.12	1.11 / 2.16
FQA	nuScenes	5.28 / 7.83	10.0 / 14.4	0.229 / 1.18	-0.229 / 0.736	-0.814 / 1.79	0.814 / 3.347
Trajectron++	Apolloscape	1.31 / 7.11	2.28 / 10.8	0.00679 / 4.14	-0.00679 / 3.69	-0.183 / 4.56	0.183 / 5.24
Trajectron++	NGSIM	2.31 / 9.64	4.49 / 16.9	-0.219 / 4.39	0.219 / 4.62	-0.0855 / 4.38	0.0855 / 4.28
Trajectron++	nuScenes	5.31 / 8.73	11.2 / 17.1	-0.331 / 2.10	0.331 / 2.68	-1.57 / 2.25	1.57 / 5.01
Trajectron++(map)	nuScenes	2.69 / 6.81	5.39 / 11.8	-0.107 / 2.118	0.107 / 2.25	-0.526 / 3.79	0.526 / 4.48

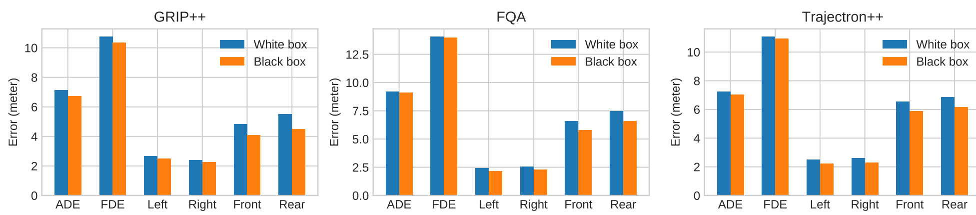

Figure 3. White box v.s. block box attack (single-frame) on *Apolloscape* dataset.

Also, good prediction accuracy on normal trajectories does not necessarily lead to good adversarial robustness. Without perturbation, *Trajectron++* has better prediction accuracy on all three datasets, thanks to its comprehensive input representation. With adversarial perturbation, however, *Trajectron++* does not have the best accuracy. On dataset *Apolloscape* and *NGSIM*, *FQA* has a better prediction error against perturbation. *FQA*'s single LSTM makes predictions mainly based on the last two trajectory positions so that the perturbation on other parts of trajectories is not effective. Though *Trajectron++* integrates rich features such as dynamics, the features are affected by perturbation on any trajectory position, which is a wide attack surface.

6.2.3 Different Attack Methods

White box v.s. black box. We conduct black box attacks on the three models and evaluate the attack result on the *Apolloscape* dataset. We visualize the six metrics of prediction error in Figure 3. In general, black box attacks and white box attacks have a very similar performance. Since the search space of the optimal perturbation is in a two-dimensional space (i.e., the spatial locations) and restricted by hard constraints (§4), the attacker can efficiently solve the optimization problem without knowledge of the model. White-box and black-box adversarial trajectories both have high variance of acceleration but perturbation of white-box attack is overall smaller because the guidance of gradient avoids part of unnecessary perturbation.

Length of adversarial perturbation As mentioned in §4, the attacker can launch attacks on continuous time frames by tuning the parameter L_P. Besides the attack on single-frame attack ($L_P = 1$) presented in Table 3, we increase L_P to the number of frames in up to 3 seconds as multi-frame attacks. For instance, for *Apolloscape* dataset which

Table 4. Attack transferability (white box, single-frame).

Source \ Target	FQA	GRIP++	Trajectron++
FQA	100%	42.8%	16.5%
GRIP++	60.8%	100%	15.5%
Trajectron++	64.7%	39.5%	100%

has a sampling frequency of 2 Hz, $L_P = 6$ means all predictions in 3 seconds are controlled by adversarial trajectories. We show white-box attack results with various L_P of three models on *Apolloscape* in Figure 4.

Without perturbation, the single-frame and multi-frame prediction have similar performance because multi-frame prediction is a sequence of independent single-frame prediction. However, under adversarial attacks, the average prediction error decreases as the length of adversarial perturbation increases. The attack gets harder if the attacker intends to maintain the attack impact in a long continuous time period. This is because of the extra constraint in multi-frame perturbation – adversarial history trajectories in different time frames depends on the same perturbation vector thereby are not optimized separately. For the 3-second attack, ADE/FDE is increased by 142%/127% and lateral/longitudinal deviation is 0.95/1.55 meters on average (22% of deviations are larger than 1.85 meters).

Bounds on the perturbation. As mentioned in §4, the attacker can tune the deviation bound to generate attacks with different stealth levels. We launch white-box attacks under various bounds and the attack impact reduces as the bound decreases (as shown in Figure 5). However, the attack still effectively cause high prediction error using minor perturbations. When the deviations of trajectory locations are at most 0.2-meter, ADE/FDE is increased by 86%/80% and lateral/longitude deviation is 1.31/1.40 meters on average (27% of deviations are larger than 1.85 meters).

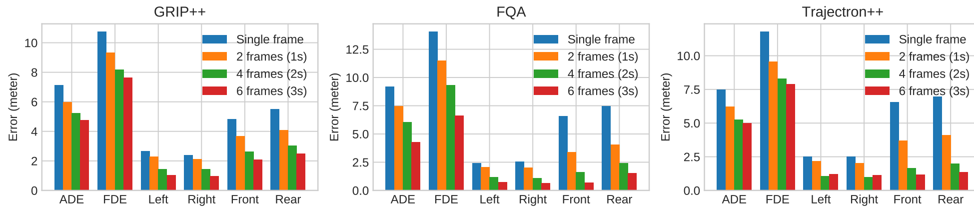

Figure 4. Attack results (white-box, single-frame, on *Apolloscape*) w.r.t. different length of adversarial perturbation.

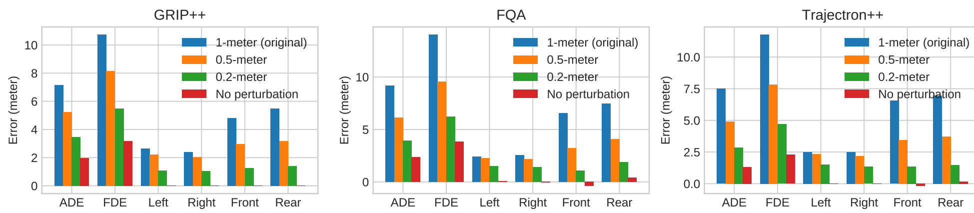

Figure 5. Attack results (white-box, single-frame, on *Apolloscape*) w.r.t. different bounds of largest deviation of trajectory positions.

6.2.4 Transferability

We evaluate the transferability across the three models using *Apolloscape* dataset. Since the attack impact is evaluated using quantified prediction errors instead of the binary judgment of attack success, we define a percentage score to measure the transferability. When we apply the perturbation optimized on the source model to the target model, for each metric of prediction error, we calculate the ratio of prediction error on the target model to prediction error on the source model. Finally, we take the average of six ratio numbers corresponding to six metrics as the final score.

The results are shown in Table 4. First of all, **90.25%** of transferred adversarial trajectories successfully increase the prediction error compared with non-attack cases. It shows that the perturbation exploits common internal patterns of general prediction models. Therefore, the perturbation optimized on one model can launch attacks on other models as well. It means the perturbation brings common patterns causing deviation of the predicted trajectory. Second, the transferability is highly correlated with the target model: transferring the perturbation to *FQA* is easier but to *Trajectron++* is harder. We hypothesize that it is because *Trajectron++* leverages more features of trajectories so that the perturbation optimized on fewer features cannot completely reproduce the high prediction error on *Trajectron++*.

6.3. Mitigation Results

We test the mitigation mechanisms using the three models and *Apolloscape* dataset. The result is shown in Figure 6. We assume that the attacker has full knowledge of the mitigation method and applies the same mitigation method on each prediction during the white-box attack. Our convolution-based trajectory smoothing is differentiable so that the computed gradient can directly involve the effect of the mitigation. If the smoothing method is replaced by non-differentiable ones, the attacker can also approximate the gradient using a differentiable function given the full knowledge of the smoothing algorithm.

Train-time mitigation. The effectiveness of data augmentation and trajectory smoothing varies on different models. Data augmentation is effective on *Trajectron++* (prediction error reduced by 24%), which deploys a complicated network structure on low-dimensional data (i.e., trajectories). Data augmentation can alleviate the over-fitting issues of *Trajectron++*. Trajectory smoothing is effective on *FQA*. *FQA* model has potential under-fitting issues since its predicted trajectories mainly depend on the direction of velocity at the last time frame and have a high error on curve trajectories. Trajectory smoothing cannot solve the problem of the model but directly alleviate the impact of adversarial perturbation on the last two observed trajectory positions. If we apply data augmentation and trajectory smoothing at train time simultaneously, the prediction error under attacks is reduced by 26% on average while the prediction error of normal cases is increased by 11%.

Test-time mitigation. If applying trajectory smoothing on all trajectories, the prediction error under attacks is dropped by 13% but the normal-case prediction error raised significantly by 28%. This is because the testing data distribution is altered to be different from the training dataset. To address the problem, we require the detection method to distinguish adversarial trajectories from normal ones and apply smoothing only on adversarial examples.

Figure 7 shows the ROC curve of two detectors mentioned in §5. First, our rule-based method (i.e., the threshold on the variance of acceleration) has better performance than SVM classification in terms of true positive rate (TPR) and false positive rate (FPR). This result confirms that the variance of acceleration over time is the key difference between adversarial and normal trajectories. Second, the de-

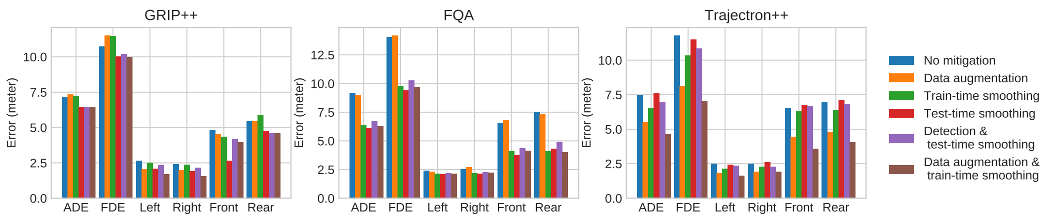

Figure 6. Prediction error (single-frame, *Apolloscape* dataset) before and after various mitigation methods.

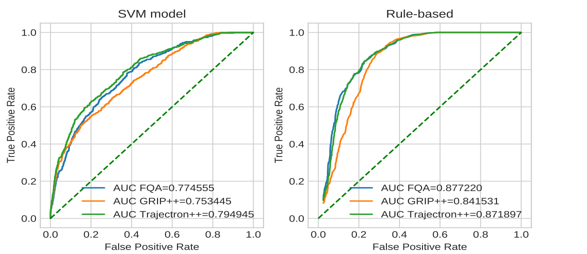

Figure 7. ROC of detection of adversarial trajectories (*Apolloscape* dataset, 3-second trajectories).

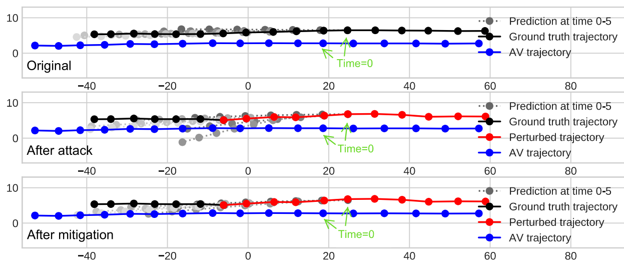

Figure 8. An example of spoofing fake lane changing behavior by adversarial perturbation (*GRIP++* model, *Apolloscape* dataset).

tection of the adversarial trajectories for the three models has similar accuracy. It proves the generality of our detection method across various models. Finally, we deploy the rule-based method which has 88% TPR and 27% FPR. By integrating the detection, the test-time smoothing reduces the prediction error of adversarial cases by 12% while raising the prediction error of normal cases by only 6%.

Limitation of mitigation. As mentioned in §6.2, some normal trajectories also have the adversarial pattern, which results in relatively high FPR of detection. Train-time methods introduce noise in training data as they change the spatial features of the original dataset. Test-time anomaly detection suffers from high FPR thus unnecessary smoothing is applied on some normal cases. Both approaches improve adversarial robustness at a cost of slightly worse performance in some normal cases. A complete defense of adversarial trajectories is a promising future work.

6.4. Case study

In this section, we demonstrate the real-world impact of the adversarial perturbation by one case study. More case studies are in the appendix.

Figure 8 shows one scenario where the adversarial perturbation spoofs a fake lane change and causes a hard brake of the AV. In this scenario, the other vehicle (OV) is driving alongside the AV (we omit other objects for demonstration) and the prediction is accurate in this case (lateral deviation is 0.18 meters). After perturbation (deviation bound of 0.5 meters, 3-second length, maximizing deviation to left), the average deviation to left is significantly increased to 1.27 meters (7×). What is worse, the high error directly affects the decision making of the AV. At time frame 0-2, the predicted OV's trajectory crosses with the AV's future trajectory, looking like a lane changing behavior. According to AV planning logic (e.g., open-source planning code of Baidu Apollo [3]), the AV will try to stop behind the cross point to yield the OV and the deceleration reaches 12 m/s^2, which exceeds the maximum deceleration of normal driving configured in AV software. Such a hard brake substantially increases the risk of rear-end collisions. After applying the train-time mitigation, the deviation to left is reduced to 0.91 meters. Though the predicted trajectory and AV's future trajectory cross, the AV only needs a deceleration of 6 m/s^2.

We successfully reproduce the attack on real-world AV system *Baidu Apollo 6.0* [3]. We construct the driving scenario in *LGSVL* simulator [28]. Details are in the appendix.

7. Conclusion

We present the first effort of analyzing adversarial robustness of trajectory prediction. From the evaluation of our proposed attack, prediction models are generally vulnerable to adversarial perturbation and may cause dangerous AV behavior such as hard brakes. We shed light on the necessity of evaluating worst-case prediction accuracy under hard scenarios or adversarial examples. To improve adversarial robustness of trajectory prediction, we propose several mitigation methods. We also suggest leveraging map information and semantic of driving rules to guide prediction.

Acknowledgments. This work was supported by NSF under the grant CNS-1930041, CNS-1932464, CNS-1929771, CNS-2145493 and the National AI Institute for Edge Computing Leveraging Next Generation Wireless Networks, Grant #2112562.

References

[1] Autoware: Open-source software for self-driving vehicles. https://gitlab.com/autowarefoundation/autoware.ai, 2020. 1, 2

[2] Traffic Analysis Tools: Next Generation Simulation. https://ops.fhwa.dot.gov/trafficanalysistools/ngsim.htm, 2020. 1, 5

[3] Baidu Apollo. http://apollo.auto, 2021. 1, 2, 8

[4] Anish Athalye, Nicholas Carlini, and David Wagner. Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples. In *International conference on machine learning*, pages 274–283. PMLR, 2018. 4

[5] 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. 1, 2, 5

[6] Yulong Cao, Chaowei Xiao, Benjamin Cyr, Yimeng Zhou, Won Park, Sara Rampazzi, Qi Alfred Chen, Kevin Fu, and Z Morley Mao. Adversarial sensor attack on lidar-based perception in autonomous driving. In *Proceedings of the 2019 ACM SIGSAC conference on computer and communications security*, pages 2267–2281, 2019. 2

[7] 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. 2

[8] Rohan Chandra, Uttaran Bhattacharya, Aniket Bera, and Dinesh Manocha. Traphic: Trajectory prediction in dense and heterogeneous traffic using weighted interactions. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8483–8492, 2019. 1

[9] Rohan Chandra, Uttaran Bhattacharya, Christian Roncal, Aniket Bera, and Dinesh Manocha. Robusttp: End-to-end trajectory prediction for heterogeneous road-agents in dense traffic with noisy sensor inputs. In *ACM Computer Science in Cars Symposium*, pages 1–9, 2019. 2, 3

[10] Rohan Chandra, Tianrui Guan, Srujan Panuganti, Trisha Mittal, Uttaran Bhattacharya, Aniket Bera, and Dinesh Manocha. Forecasting trajectory and behavior of road-agents using spectral clustering in graph-lstms. *IEEE Robotics and Automation Letters*, 5(3):4882–4890, 2020. 1, 2

[11] Chiho Choi, Joon Hee Choi, Jiachen Li, and Srikanth Malla. Shared cross-modal trajectory prediction for autonomous driving. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 244–253, 2021. 1, 2

[12] Francesco Croce and Matthias 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. 2

[13] Nachiket Deo and Mohan M Trivedi. Trajectory forecasts in unknown environments conditioned on grid-based plans. *arXiv preprint arXiv:2001.00735*, 2020. 1, 2

[14] Yinpeng Dong, Qi-An Fu, Xiao Yang, Tianyu Pang, Hang Su, Zihao Xiao, and Jun Zhu. Benchmarking adversarial robustness on image classification. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 321–331, 2020. 2

[15] Jingkang Wang et. al. Advsim: Generating safety-critical scenarios for self-driving vehicles. In *CVPR*, 2021. 2

[16] Xinyu Huang, Xinjing Cheng, Qichuan Geng, Binbin Cao, Dingfu Zhou, Peng Wang, Yuanqing Lin, and Ruigang Yang. The apolloscape dataset for autonomous driving. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 954–960, 2018. 1, 5

[17] Yunhan Jia, Yantao Lu, Junjie Shen, Qi A Chen, Zhenyu Zhong, and Tao Wei. Fooling detection alone is not enough: First adversarial attack against multiple object tracking. In *International Conference on Learning Representations (ICLR)*, 2020. 2

[18] Nitin Kamra, Hao Zhu, Dweep Trivedi, Ming Zhang, and Yan Liu. Multi-agent trajectory prediction with fuzzy query attention. *arXiv preprint arXiv:2010.15891*, 2020. 1, 2, 5

[19] James Kennedy and Russell Eberhart. Particle swarm optimization. In *Proceedings of ICNN'95-international conference on neural networks*, volume 4, pages 1942–1948. IEEE, 1995. 4

[20] Xin Li, Xiaowen Ying, and Mooi Choo Chuah. Grip++: Enhanced graph-based interaction-aware trajectory prediction for autonomous driving. *arXiv preprint arXiv:1907.07792*, 2019. 1, 2, 5

[21] Yuexin Ma, Xinge Zhu, Sibo Zhang, Ruigang Yang, Wenping Wang, and Dinesh Manocha. Trafficpredict: Trajectory prediction for heterogeneous traffic-agents. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 6120–6127, 2019. 1

[22] 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. 4

[23] Francesco Marchetti, Federico Becattini, Lorenzo Seidenari, and Alberto Del Bimbo. Mantra: Memory augmented networks for multiple trajectory prediction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7143–7152, 2020. 1

[24] Abduallah Mohamed, Kun Qian, Mohamed Elhoseiny, and Christian Claudel. Social-stgcnn: A social spatio-temporal graph convolutional neural network for human trajectory prediction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14424–14432, 2020. 1

[25] Sriram Narayanan, Ramin Moslemi, Francesco Pittaluga, Buyu Liu, and Manmohan Chandraker. Divide-and-conquer for lane-aware diverse trajectory prediction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15799–15808, 2021. 2

[26] Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al. Scikit-learn: Machine learning in python. *the Journal of machine Learning research*, 12:2825–2830, 2011. 5

[27] Tung Phan-Minh, Elena Corina Grigore, Freddy A Boulton, Oscar Beijbom, and Eric M Wolff. Covernet: Multimodal behavior prediction using trajectory sets. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14074–14083, 2020. 1, 2

[28] Guodong Rong, Byung Hyun Shin, Hadi Tabatabaee, Qiang Lu, Steve Lemke, Mārtiņš Možeiko, Eric Boise, Geehoon Uhm, Mark Gerow, Shalin Mehta, et al. Lgsvl simulator: A high fidelity simulator for autonomous driving. In *2020 IEEE 23rd International conference on intelligent transportation systems (ITSC)*, pages 1–6. IEEE, 2020. 8

[29] Saeed Saadatnejad, Mohammadhossein Bahari, Pedram Khorsandi, Mohammad Saneian, Seyed-Mohsen Moosavi-Dezfooli, and Alexandre Alahi. Are socially-aware trajectory prediction models really socially-aware? *arXiv preprint arXiv:2108.10879*, 2021. 2

[30] Tim Salzmann, Boris Ivanovic, Punarjay Chakravarty, and Marco Pavone. Trajectron++: Dynamically-feasible trajectory forecasting with heterogeneous data. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XVIII 16*, pages 683–700. Springer, 2020. 1, 2, 5

[31] Takami Sato, Junjie Shen, Ningfei Wang, Yunhan Jia, Xue Lin, and Qi Alfred Chen. Dirty road can attack: Security of deep learning based automated lane centering under physical-world attack. In *Proceedings of the 29th USENIX Security Symposium (USENIX Security'21)*, 2021. 2

[32] Nasim Shafiee, Taskin Padir, and Ehsan Elhamifar. Introvert: Human trajectory prediction via conditional 3d attention. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16815–16825, 2021. 1

[33] Liushuai Shi, Le Wang, Chengjiang Long, Sanping Zhou, Mo Zhou, Zhenxing Niu, and Gang Hua. Sgcn: Sparse graph convolution network for pedestrian trajectory prediction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8994–9003, 2021. 1

[34] Ingo Steinwart and Andreas Christmann. *Support vector machines*. Springer Science & Business Media, 2008. 4

[35] Hao Sun, Zhiqun Zhao, and Zhihai He. Reciprocal learning networks for human trajectory prediction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7416–7425, 2020. 1

[36] Jiachen Sun, Yulong Cao, Qi Alfred Chen, and Z Morley Mao. Towards robust lidar-based perception in autonomous driving: General black-box adversarial sensor attack and countermeasures. In *29th {USENIX} Security Symposium ({USENIX} Security 20)*, pages 877–894, 2020. 2

[37] Jiachen Sun, Yulong Cao, Christopher B Choy, Zhiding Yu, Anima Anandkumar, Zhuoqing Morley Mao, and Chaowei Xiao. Adversarially robust 3d point cloud recognition using self-supervisions. *Advances in Neural Information Processing Systems*, 34, 2021. 2

[38] Jianhua Sun, Qinhong Jiang, and Cewu Lu. Recursive social behavior graph for trajectory prediction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 660–669, 2020. 1

[39] Jiachen Sun, Karl Koenig, Yulong Cao, Qi Alfred Chen, and Z Morley Mao. On adversarial robustness of 3d point cloud classification under adaptive attacks. *arXiv preprint arXiv:2011.11922*, 2020. 2

[40] Jiahang Wang, Sheng Jin, Wentao Liu, Weizhong Liu, Chen Qian, and Ping Luo. When human pose estimation meets robustness: Adversarial algorithms and benchmarks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11855–11864, 2021. 1

[41] Jue Wang, Ping Wang, Chao Zhang, Kuifeng Su, and Jun Li. F-net: Fusion neural network for vehicle trajectory prediction in autonomous driving. In *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4095–4099. IEEE, 2021. 1

[42] Ye Yuan, Xinshuo Weng, Yanglan Ou, and Kris Kitani. Agentformer: Agent-aware transformers for socio-temporal multi-agent forecasting. *arXiv preprint arXiv:2103.14023*, 2021. 1, 2