Proceedings of Machine Learning Research vol 242:1–13, 2024

Hacking Predictors Means Hacking Cars: Using Sensitivity Analysis to Identify Trajectory Prediction Vulnerabilities for Autonomous Driving Security

Marsalis Gibson MTGIBSON@BERKELEY.EDU
David Babazadeh DAVIDBABAZADEH@BERKELEY.EDU
Claire Tomlin TOMLIN@EECS.BERKELEY.EDU
Shankar Sastry SASTRY@COE.BERKELEY.EDU
University of California, Berkeley

Abstract

Adversarial attacks on learning-based multi-modal trajectory predictors have already been demonstrated. However, there are still open questions about the effects of perturbations on inputs other than state histories, and how these attacks impact downstream planning and control. In this paper, we conduct a sensitivity analysis on two trajectory prediction models, Trajectron++ and AgentFormer. The analysis reveals that between all inputs, almost all of the perturbation sensitivities for both models lie only within the most recent position and velocity states. We additionally demonstrate that, despite dominant sensitivity on state history perturbations, an undetectable image map perturbation made with the Fast Gradient Sign Method can induce large prediction error increases in both models, revealing that these trajectory predictors are, in fact, susceptible to image-based attacks. Using an optimization-based planner and example perturbations crafted from sensitivity results, we show how these attacks can cause a vehicle to come to a sudden stop from moderate driving speeds.

Keywords: Adversarial Machine Learning, Sensitivity Analysis, Trajectory Forecasting, Autonomous Driving, Security of Cyber-Physical Systems

1. Introduction

Trajectory prediction is becoming key for autonomy. However, the integration of learning-based trajectory prediction into autonomous systems introduces new opportunities for cyber-attacks [Miller and Valasek (2018)]. While there are efforts for integrating safety in autonomous control [Bahati et al. (2021); Chou et al. (2021); Herbert et al. (2021)], inaccurate predictors of other agents' behaviors can adversely affect the navigation of an autonomous vehicle. Furthermore, even though recent work has shown several examples of possible perturbation attacks on trajectory predictors' state histories [Cao et al. (2023, 2022); Tan et al. (2023); Zhang et al. (2022)], because many of these models are multi-modal (taking in different input representations), there is still an open question as to whether or not perturbations on other input types, such as image maps, can adversely affect the performance of trajectory predictors.

Therefore, we conduct a sensitivity analysis of both Trajectron++ [Salzmann et al. (2020)] and AgentFormer [Yuan et al. (2021)] to understand the perturbation sensitivities of each model. We then analyze how specially crafted perturbations, when applied to a vehicle using Trajectron++, produce erroneous predictions and change the resulting plans of a vehicle. The analyses show that both Trajectron++ and AgentFormer are most sensitive to perturbations on the most recent position and velocity states, demonstrating that effective attacks on both models can be made to just the current state of the input data. However, more interestingly, we observe that both models are susceptible to image attacks, showing that very small image perturbations can increase model error by at least $17 - 200\%$ in either model. If it is given a well-designed perturbation informed by the model's sensitivity profile, a vehicle using an optimization-based planner with one of these models can suddenly come to a stop while traveling at speeds of around 17.6 (m/s).

Our contributions are as follows:

1. A sensitivity analysis for security of trajectory prediction models that preserves outlier information;
2. An illustration of the effects of image perturbation attacks on multi-modal trajectory prediction models;
3. An analysis of how adversarial perturbations to a prediction model, crafted from the sensitivity results, can affect downstream vehicle planning.

We conduct the sensitivity analysis at the level of each model's API, which, in this context, means that each perturbation is passed as an input to the model's main inference, or prediction, function. Along with the perturbations we consider, this analysis design considers attacks affecting data integrity that a user can make, without internal access to the model. Such attacks include "Vehicle-to-everything (V2X)" attacks, or attacks on vehicle sensors.

We organize this paper as follows. Background about the prediction problem and the two predictors under study are in Section 2, and related works are provided in Section 3. In Section 4, we present the sensitivity analysis framework, then we provide the sensitivity results and discuss image perturbations in Section 5. A demonstration of these results' impact on planning is provided in Section 6. Finally, security and learning implications are given in Section 7. Throughout the paper, the ˆ symbol denotes a predicted variable; for instance, $\hat{s}$ denotes a predicted state.

2. Background: The Trajectory Prediction Problem, Trajectron++, and AgentFormer

Consider a system of n agents. Let $s_t^i \in \mathbb{R}_D$ denote the state of vehicle i at time t, which could include its position, velocity, heading, and angular acceleration. Let the time series of an agent's state $s_{0:T}^i = (s_0^i, ..., s_T^i)$ be known as the trajectory of that agent from time 0 to T, and C_t represent contextual information used for prediction (i.e an image map, $C_t \in \mathbf{R}^{W \times H \times L}$), where W, H, and L are the width, height and number of channels in the image respectively. The trajectory prediction problem is to design a prediction model $f_{\text{model}}(s_{t-\delta t_h:t}^{[1:n]}, C_t, \cdot) \rightarrow \hat{s}_{t+1:t+\Delta T}^{[1:n]}$ that predicts a future trajectory for agent i, $\hat{s}_{t+1:t+\Delta T}^i$, from $t+1$ to $t+\Delta T$ given the state history information of that agent $s_{t-\delta t_h:t}^i$, state information of the other agents in the scene $s_{t-\delta t_h:t}^{[1:n]\backslash i}$, and other contextual information accessible to the predictor. ΔT and δt_h are the prediction horizon and number of previous time steps respectively.

To solve the trajectory prediction problem, both Trajectron++ and AgentFormer utilize a latent variable model to encode the distribution of future trajectories, in which the distribution of a value of interest, Y, is represented as $p(Y|X) = \int p_\phi(Y|X, Z) p_\psi(Z|X)\, dZ$, where ϕ and ψ are parameters of their respective distributions. Each trajectory predictor uses an encoder, $\bar{f}_{encoder}(s_{t-\delta t_h:t}^i, C_t, \cdot) \rightarrow \psi$ to predict the distribution over the latent variables ψ given prediction model inputs, and a decoder, $\tilde{f}_{decoder}(Z, \cdot) \rightarrow \phi$, to output the distribution ϕ over future agent actions ($Y = u_{t+1:t+\Delta T}$

for Trajectron++) or future agent positions ($Y = s^i_{t+1:t+\Delta T}$ for AgentFormer), where $\phi = (\mu, \sigma)$ represents Gaussian distributions. Both models use position, velocity, and heading as states and image maps (high-definition maps) for optional contextual information. Trajectron++ uses a spatio-temporal graph ("scene graph") of the prediction scene as an extra input, $G = (V, E, w)$, encodes inputs using a series of Long Short-Term Memory (LSTMs) models and a Convolutional Neural Network (CNN), and passes either a sample or the mode of ϕ (depending on the user's specification) to a dynamics model $f_{\text{dynamics}} : u^i_t, \hat{s}^i_t \rightarrow \hat{s}^i_{t+1}$ to produce the state predictions. Nodes of the scene graph, $A_j \in V$, represent agents in the scene, edges $(A_i, A_j) \in E$ represent the interaction between A_i and A_j, and the weights $w_i \in w$, associated with each directed edge i, represent the amount of influence that agent A_i has on agent A_j. AgentFormer is an augmented socio-temporal transformer model [Vaswani et al. (2017)] that encodes inputs using a specialized attention module (treating $s^{[1:n]\setminus i}_t$ differently than s^i_t) and includes a multilayer perceptron (MLP) to compute the latent variable. AgentFormer decodes trajectory states s^i_t concatenated with the latent variable with a cross-attention mechanism (parameterized by the encoding) and an MLP.

3. Previous Approaches to Model Sensitivity Analysis and Feature Attribution

Sensitivity analysis has been used to understand model sensitivities on image features and their contributions toward model output. Some methods include using partial derivatives as a measure [Novak et al. (2018); Montano and Palmer (2003); Gevrey et al. (2003, 2006)], measuring changes in the model's output response as each input dimension is perturbed [Gevrey et al. (2003)], using aggregated product of hidden neuron weights as a measure [Montano and Palmer (2003)], tracking changes in inner network activations [Novak et al. (2018)], and changing network configurations and their input representations [Zhang and Wallace (2015)]. Each method has its trade-offs - for example, while using partial derivatives can provide a lot of information about a particular input feature's relation to the model, using just raw gradient values alone may not be optimal [Smilkov et al. (2017); Sundararajan et al. (2017)] as features that are actually important may have small local gradients. Other methods of explaining feature importance are further explored in feature attribution literature. Methods either take a surrogate approach by analyzing feature importance on an equivalent but more interpretable network [Ribeiro et al. (2016)], use forward-propagated or back-propagated gradient values [Simonyan et al. (2013); Selvaraju et al. (2017); Binder et al. (2016); Shrikumar et al. (2017)], directly analyze connections between layer activations and input features [Zeiler and Fergus (2014)], or take a game-theoretic approach by making use of Shapley values [Sundararajan and Najmi (2020); Lundberg and Lee (2017); Makansi et al. (2021)]. In contrast to these methods, this work attempts to contribute features across input modalities (i.e. image and non-image features) to model perturbation sensitivities on trajectory predictors for security and empirically observe the effects of these perturbations on vehicle control.

4. Attributing Model Perturbation Sensitivities to Features

In our sensitivity analysis, we wish to quantify how much a specific perturbation on each input (referred to as input features) contributes to a change in the performance of $f_{\text{model}}(\cdot)$. Therefore, we design a sensitivity attribution function, $f_{\text{attr}}(I_1, ..., I_{nf}, \eta(\cdot), f_{\text{model}}(\cdot)) :\rightarrow c_1, ..., c_{nf}$, such that, given nf number of input features, $(I_1, ..., I_{nf})$, and a specific perturbation function, $\eta(I_i) \rightarrow \tilde{I}_i$, the function assigns scores, $c_i \in \mathbf{R}$, to each feature i, where each score indicates how much the perturbed input $\tilde{I}_i$ contributes to the observed change in the model's performance behavior. We use a common metric to measure model performance - average displacement error (see Definition 1) - and choose the performance behavior for attribution to be the percent increase in average displacement error (Definition 2). Given a dataset Γ_i and a set of perturbation functions $(\eta_1, ..., \eta_{np})$, we apply attribution function $f_{\text{attr}}(\cdot)$ to each data point $\gamma \in \Gamma_i$, using each perturbation function $\eta_i(\cdot)$, and we evaluate the sensitivity behavior of the model $f_{\text{model}}(\cdot)$ over Γ_i for each function. Each perturbation function is described in Section 4.1. Results are aggregated into sets $S_{\eta_i,\Gamma_i} := \{c_i, i \in [1...|\Gamma|]\}$ and then modeled as distributions, where each set corresponds to a specific perturbation function $\eta(\cdot)$ and contains the sensitivity measures from applying $f_{\text{attr}}(\cdot)$ to each $\gamma \in \Gamma_i$ using $\eta(\cdot)$. To compute the feature to which the model is most sensitive, we compare quartile values of aggregated distributions using the conditions stated in Defintion 3. Because it is possible for distributions across features to be similar, if no feature satisfies the condition in Definition 3, then we use qualitative measures to draw conclusions, measuring distribution peaks, skews, and spreads.

Definition 1 (Average Displacement Error) *The average displacement error of a prediction $\hat{s}_{t+1:t+\Delta T}$ over a prediction horizon of ΔT is defined as*

$$ADE(\hat{s}_{t+1:t+\Delta T}, s_{t+1:t+\Delta T}) = \frac{1}{\Delta T} \sum_{i=t+1}^{t+\Delta T} \|\hat{s}_{i,pos} - s_{i,pos}\|_2 \tag{1}$$

where $\hat{s}_{i,pos}$ and $s_{i,pos}$ represents the prediction and ground truth position states at time i.

Definition 2 (Percent Increase as Sensitivity Measure) *Let $\hat{s}_{t+1:t+\Delta T} = f_{model}(I_1, ..., I_{nf})$ be the baseline trajectory prediction from model $f_{model}(\cdot)$ and $\hat{s}'_{t+1:t+\Delta T} = f_{model}(I_1, ..., \eta(I_i), ..., I_{nf})$ be the new prediction resulting from applying perturbation function $\eta(\cdot)$ on feature I_i. The percent increase c_i measured by the sensitivity function is defined as follows:*

$$c_i = \frac{ADE(\hat{s}'_{t+1:t+\Delta T}, s_{t+1:t+\Delta T}) - ADE(\hat{s}_{t+1:t+\Delta T}, s_{t+1:t+\Delta T})}{ADE(\hat{s}_{t+1:t+\Delta T}, s_{t+1:t+\Delta T})} \tag{2}$$

If $ADE(\hat{s}_{t+1:t+\Delta T}, s_{t+1:t+\Delta T}) = 0$, we aggregate this data separately.

Definition 3 (Condition for Feature with Most Model Sensitivity) *Let the quartile metrics $[Q1_i, Q2_i, Q3_i]$ of set S_i represent first quartile, median, and third quartile values respectively. Feature I_i with quartile measures $[Q1_i, Q2_i, Q3_i]$ has greatest model sensitivity if*

$$Q1_i > Q1_j,\ Q2_i > Q2_j,\ Q3_i > Q3_j\ \ \forall j \in [1...nf]\backslash i \tag{3}$$

4.1. Choosing Perturbation Functions and Input Features for Analysis

Each network's architecture helps to inform what perturbation types we use for the study. If we understand each model's *inductive bias*, we can understand what perturbations may or may not be effective. For example, CNN models have inductive biases that focuses on "local features" and, thus, will identify and process local patterns within an input image. As a consequence, a model using CNNs is invariant to certain input changes, such as input translation and rotation.

Therefore, we can rule out a class of perturbations - for instance jittering and image rotations - and consider perturbations that change the local structure within an image, such as noise and occlusion perturbations. For this analysis, we choose to use noise, occlusions, gradient perturbations, Fast Gradient Sign Method (FGSM) [Goodfellow et al. (2014); Kurakin et al. (2016)], and perturbations with constant values (all defined in Definition 4). FGSM is among the perturbations chosen because it provides a method of generating perturbations that increases some cost associated with the model. For input features, we simply choose to perturb inputs accessible from eah model's API inference call that may contribute heavily to it's output prediction. These are: state histories $s^i_{t-\delta t_h:t}$ and image maps $C_t \in \mathbf{R}^{L\times W\times H}$, for both models, as well as scene graph components, V and w, for Trajectron++.

Definition 4 (Perturbation Types) *Let I_i be the input feature in vector form $I_i \in \mathbf{R}^m$. The perturbation function $\eta(\cdot)$ produces the perturbed input $\tilde{I}_i$ as follows:*

$$\tilde{I}_i = \eta(I_i) = I_i + \tilde{P} \tag{4}$$

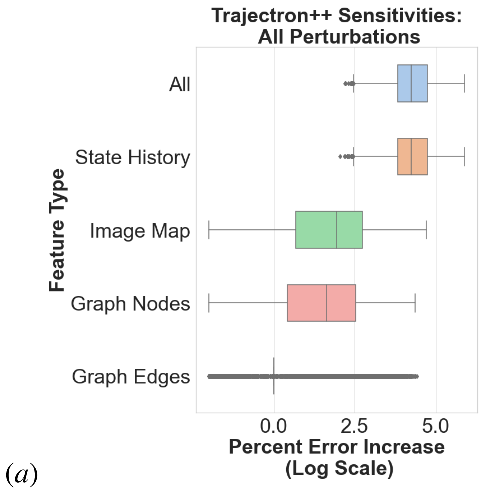
where $\tilde{P} = \epsilon\nabla_{I_i} J(\theta, I, y)$ for gradient perturbations, $\tilde{P} = \epsilon sign(\nabla_{I_i} J(\theta, I, y))$ for a Fast Gradient Sign Method (FGSM) perturbation, $\tilde{P} = [c]_{1\times m}$ for constant perturbations, $\tilde{P}_j \sim \mathcal{N}(0, \sigma)$ for noise, and $\tilde{P} = [0]_{1\times m}$ for occlusion. $J(\cdot)$ is the loss function of $f_{model}(\cdot)$, θ are the parameters of $f_{model}(\cdot)$, and y are the ground truth states $s^i_{t+1:t+\Delta T}$ associated with the unperturbed baseline predictions $\hat{s}^i_{t+1:t+\Delta T}$. ϵ, c, and σ are parameters chosen based on the normalization applied (described in Section 5.1).

5. Sensitivity Analysis Results: Trajectron++ and AgentFormer

In this section, we discuss the sensitivity analysis results for Trajectron++ and AgentFormer and highlight the effect of image perturbations. Learning and security implications are discussed in Section 7. Trajectron++ and AgentFormer are trained using the mini-NuScenes dataset [Caesar et al. (2020)] and data used for the analysis are from the training, validation, and testing partitions of the dataset. We use the same training procedures and loss functions as used in [Salzmann et al. (2020)] and [Yuan et al. (2021)] respectively. Other possible data sets to evaluate/train on, which we leave for future work, include the full NuScenes dataset [Caesar et al. (2020)], ETH dataset [Pellegrini et al. (2009)], and UCY dataset [Lerner et al. (2007)].

5.1. Contributing sensitivity scores to features: the effects of current state and image perturbations

To observe characteristics of the models' sensitivities, we report the quartile values of aggregated results of two experiments - one for Trajectron++ and one for AgentFormer. Both experiments measures sensitivity as described in Definition 2. For each experiment, each perturbation is normalized such that the magnitude of the perturbation made to a particular input is 50% of that input's range in the dataset. All ranges are either taken from [Caesar et al. (2020)] or calculated from the dataset directly. We use a range of 80m, $30\frac{\text{m}}{\text{s}}$, $35\frac{\text{m}}{\text{s}^2}$, 7 rad, $5\frac{\text{rad}}{\text{s}}$, 1 units, and 10 units for perturbations made on position, velocity, acceleration, heading, angular velocity, image, and edge weight values respectively. Ranges on standardized inputs are calculated similarly with respect to their standardized variables. The gradients used for gradient perturbations and FGSM are the gradients



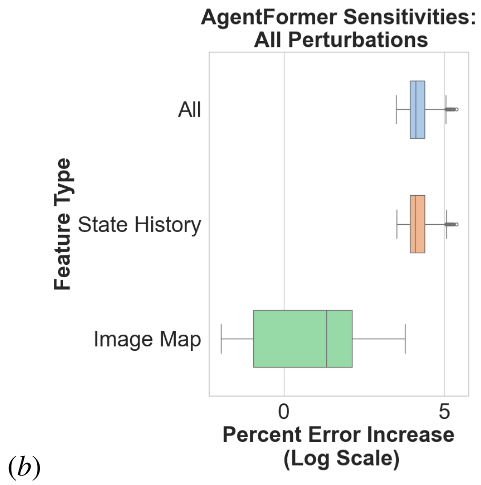
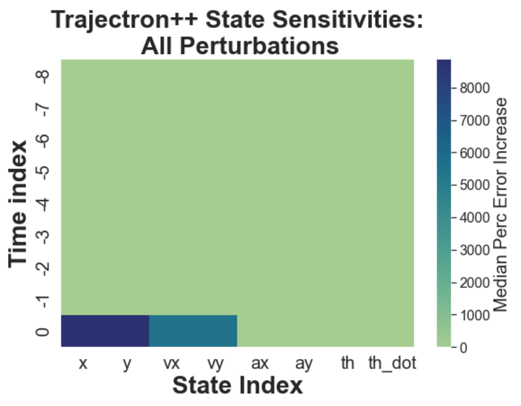


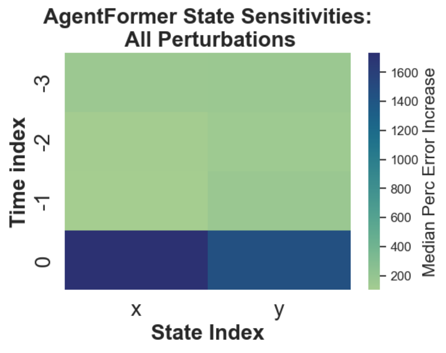
Figure 1: Sensitivity analysis of Trajectron++ (a) and AgentFormer (b) with power transformation [Yeo and Johnson (2000)] to values. Both models are by far most sensitive to perturbations on state history inputs - having mean sensitivity of (a) 17200% and (b) 12573.495% percent error increases. However, perturbations on image maps and graph nodes can induce at least a median of $20.5 - 85.5\%$ and 41.8% increase in error respectively, indicating that perturbations on other inputs can adversely affect the performance of these trajectory predictors.

of a negative evidence lower bound (ELBO) loss function with respect to the baseline inputs. Perturbed image pixel values remain unclipped, and any baselines points with initial ADE values of 0 are left out of the aggregated data. To obtain a better view of the data's distribution, we apply a power transformation from [Yeo and Johnson (2000)] before plotting distributions in Figure 1.

From the results, we observe that the quartile values for state history input are by far the largest and most similar to sensitivity measures from perturbing all inputs on both Trajectron++ and AgentFormer. Trajectron++'s median perturbation sensitivity on state history inputs is two to four hundred times as much as on image and graph node inputs respectively, while AgentFormer's is more than six hundred times as much as on image inputs. Given that results show high perturbation sensitivity on state history features, we run a "depth analysis" to study perturbation sensitivities on each state at different time points, shown in Figure 2 (no data transformation applied). We observe, for both models, that most of the sensitivity contribution for state history inputs comes from the most recent position and velocity states, where all other values have median sensitivity measures of less than 10% for Trajectron++ and less than 167% for AgentFormer. This observed phenomenon is due to the implementation and architectural choices used in both models. For Trajectron++ and AgentFormer, the current position and velocity states are used as either the initial condition for Trajectron++'s dynamics model (see Section 2) or used to translate AgentFormer's predictions from relative coordinates back to absolute coordinates. Therefore, any perturbation to the initial condition or offset will drastically perturb the resulting predictions. For instance, if we keep the initial condition for Trajectron++'s dynamic model unperturbed, we see that Trajectron++ becomes most



(*a*)



(*b*)

Figure 2: Depth sensitivity analysis for Trajectron++ (a) and AgentFormer (b). We observe that all sensitivity is on the most recent position (x,y) and velocity states (v_x,v_y) for both models.

sensitive to image perturbations, with median sensitivities on image features being 85.5% vs 83.4% for state history.

Because both Trajectron++ and AgentFormer have an upper quartile image perturbation sensitivity of 553.0% and 135.0% respectively, we run an additional experiment to observe how different sized image perturbations affect the models' prediction errors. The results, shown in Figure 3, indicate that an image perturbation size, as small as 0.01 units per dimension, can cause a median of 17.8% and 17.3% increase in error for Trajectron++[1] and AgentFormer respectively, while a perturbation size of 0.1 can increase error on both models by 162.0% and 112.0%. Therefore, we demonstrate examples of image perturbations with sizes in between both values that have significant impact (within the upper quartile of sensitivity) on both Trajectron++ and AgentFormer. As shown in Figure 4, one undetectable image perturbation with a size 0.025 per dimension alone can cause Trajectron++'s and AgentFormer's prediction error to go from 1.34 (m) to 4.785 (m) and 0.097 (m) to 1.043 (m).

5.2. Discussion

Given these observations, it is clear that perturbations made to state history features would be most effective for attacking both models; however, it is unclear as to what perturbation sensitivities are directly learned in training or can be attributed to neural network components alone. We leave this investigation for future work. Regarding the phenomenon observed in Figure 4, our hypothesis is that, if an input's dimensions are large enough as in the case with image maps, even small perturbations per dimension, if well designed, can be enough to cause some significant change in model performance.

1. Because Trajectron++ outputs a Gaussian Mixture Model, we analyze whether there is mode switching in the output as the size of perturbations increases on images. We observe that frequent mode switching seems to occur for image perturbations, when increasing perturbation sizes steadily between 0 and 1. We compare this to observations of mode switching on state histories inputs, which is far less frequent as perturbation sizes increase between $0 - 80$m.

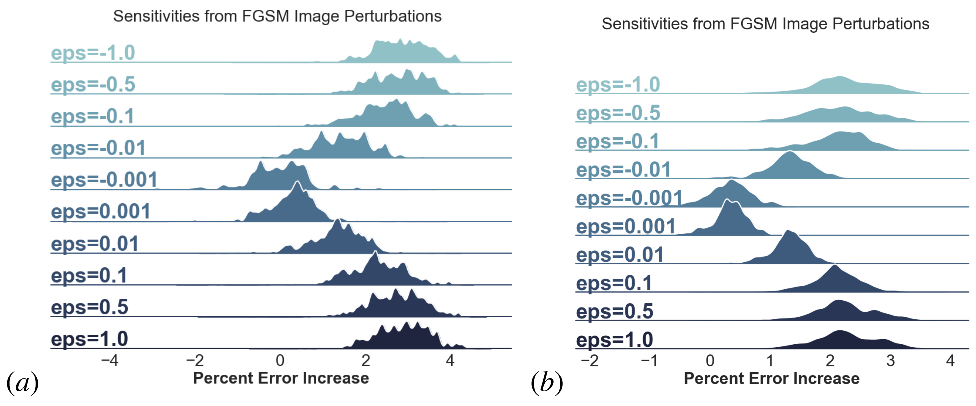


Figure 3: Sensitivity analysis of image perturbations for Trajectron++ (a) and AgentFormer (b) with power transformation [Yeo and Johnson (2000)] applied to values. Percent error increases from image perturbations with epsilon size as small as 0.01 contain median values of 17.8% and 17.3% and go as large as 12800.0% and 213.0% for Trajectron++ and AgentFormer respectively.

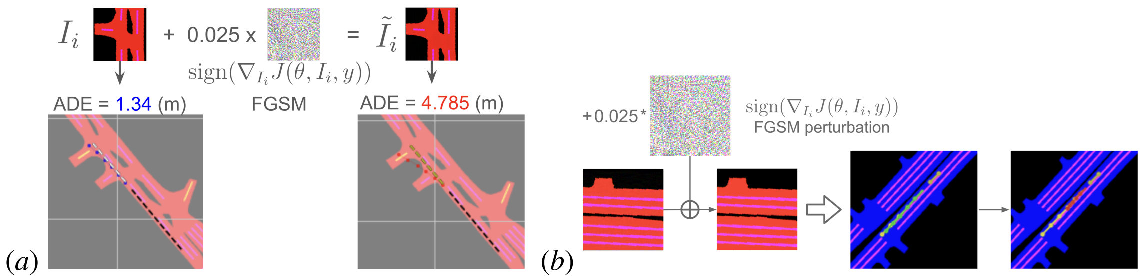

I_i + 0.025 x = $\tilde{I}_i$
$\mathrm{sign}(\nabla_{I_i} J(\theta, I_i, y))$
FGSM
ADE = 1.34 (m)
ADE = 4.785 (m)
(a)
+0.025*
$\mathrm{sign}(\nabla_{I_i} J(\theta, I_i, y))$
FGSM perturbation
(b)

Figure 4: Image perturbation examples for Trajectron++ (a) and AgentFormer (b). These undetectable image perturbations cause a (a) 255.50% and (b) 974.0 % error increase in Trajectron's and AgentFormer's ADE respectively, where (b) goes from 0.097 to 1.043 (m) ADE. Here we show that, with the inclusion of image inputs, these multi-modal generative models are susceptible to image-based attacks.

6. Downstream Effects on Planning

Next, we demonstrate how adversarial perturbations to trajectory prediction models could affect downstream planning. In this section, we take an example image and state history perturbation, informed by the sensitivity analysis, and observe changes in an example planner using Trajectron++.

6.1. Scenario and Planner Analysis

To analyze effects of perturbations on vehicle planning, we take a real driving scenario from the mini-NuScenes dataset that contains at least two "interacting" vehicles in close proximity to each other (where the behavior of one vehicle impacts another) and see how the model's input perturba-

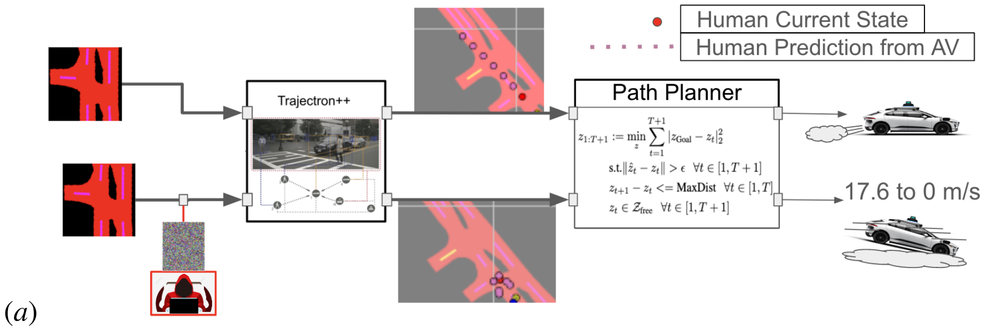

Human Current State
Human Prediction from AV
Trajectron++
Path Planner
$z_{1:T+1} := \min_z \sum_{t=1}^{T+1} |z_{Goal} - z_t|_2^2$
s.t. $\|\hat{z}_t - z_t\| > \epsilon \;\; \forall t \in [1, T+1]$
$z_{t+1} - z_t <= \text{MaxDist} \;\; \forall t \in [1, T]$
$z_t \in \mathcal{Z}_{free} \;\; \forall t \in [1, T+1]$
17.6 to 0 m/s
(a)

Figure 5: Example effect of image perturbation on vehicle planning. Top path shows the baseline input, where the autonomous vehicle (AV) w/ Trajectron++ produces a "driving forward" prediction for the human vehicle, allowing it to continue driving at same previous speed. The bottom path shows the prediction and plan with the perturbation, which causes the AV's human vehicle prediction to clump together, causing it to come to an abrupt stop.

tions, chosen based on sensitivity analysis, are propagated through to an optimization-based planner. We chose a two-vehicle scenario, one following the other, where the following vehicle is chosen to be the autonomous vehicle (AV) predicting with Trajectron++ and planning with the generic planner, and the leading vehicle is chosen to be the human vehicle whose predicted behavior impacts the future plans of the AV. Our optimization-based planner, which captures the basics behind many different planners, is:

$$s_{1:T+1} := \min_s \sum_{t=1}^{T+1} |s_{\text{Goal}} - s_t|_2^2 \tag{5}$$

$$\text{s.t.} \|\hat{s}_t - s_t\|_2 > \epsilon \;\; \forall t \in [1, T+1] \tag{6}$$

$$|(s_{t+1})_i - (s_t)_i| <= \kappa \;\; \forall i \in [1, n_p], t \in [1, T] \tag{7}$$

$$s_t \in \mathcal{Z}_{\text{free}} \;\; \forall t \in [1, T+1] \tag{8}$$

Given the goal position s_{Goal}, and predictions $\hat{s}_t$ from the trajectory predictor Trajectron($\cdot$), the planner produces planned states $s_{1:T+1}$. Planned states must be on valid road space $\mathcal{Z}_{\text{free}}$, the distance between each subsequent planned state must be less than or equal to κ, and each s_t should be at least ϵ units away from the predicted human vehicle state $\hat{s}_t$. The parameters $\epsilon = 15$ and $\kappa = 16.5$ are chosen such that the resulting planned states approximates the vehicle's (which represents the AV) real states in the NuScenes scenario. Figure 5 highlights an example of the effects that an image perturbation has on the planning of the vehicle utilizing Trajectron++. This perturbation, made with FGSM using epsilon size of 20, causes Trajectron++ to output an erroneous prediction for the human vehicle (jumping from an ADE of 1.97(m) to 26.52(m)), which causes the autonomous vehicle's future plan to induce an abrupt stop, coming from speeds of around 17.60(m/s) to 0. The same effect happens when the most recent velocity states get set to 0, see Table 1. If driving on a freeway or major street, these perturbations could cause this autonomous vehicle to collide with any vehicle following behind them.

	t=0	t=1	t=2	t=3	t=4
Baseline Future States	**x**: 619.23	609	593	577	561
	y: 482.87	499	515	531	547
Planning after Image Perturbation	619.23	619.23	619.23	619.23	619
	482.87	482.87	482.87	482.87	499
Planning after t=0 Velocity Perturbation	619.23	619.23	619.23	619.23	619.23
	482.87	482.87	482.87	482.87	482.87

Table 1: Results from the optimization-based planner. Both the image perturbation (made with FGSM) and occlusion on the most recent velocity state (both chosen based on Trajectron++'s sensitivity profile) causes erroneous predictions from Trajectron++, which consequently cause the autonomous vehicle's future plans to simulate an abrupt stop.

7. Discussion: Implications on Learning and Security

Contribution 1. The sensitivity results show both models having dominant sensitivity on the most recent position/velocity states, and non-trivial sensitivity on image inputs. These results reveal that, although any targeted attack made to either Trajectron++ and AgentFormer can be done most effectively using the most recent position/velocity state input, other attacks can utilize non-state related inputs, such as image maps, to impact either model's performance.

Contributions 2. The image perturbation analysis shows that an image perturbation with a size of 0.01 units can induce a median of 17.8% and 17.3% error increase in Trajectron++ and AgentFormer respectively, suggesting that both of these models are highly susceptible to image attacks, as shown in Figure 4. One possible mitigation may be to use techniques from Ilyas et al. (2019), which removes "non-robust features" from image data.

Contribution 3. We show that from the sensitivity analysis, we can craft targeted perturbations for Trajectron++ in the form of an image perturbation or an occlusion over the most recent velocity state, and that if these targeted perturbations are given as an input to a generic optimization-based planner utilizing Trajectron++, the planner will return a plan that stops the vehicle even if the vehicle is previously traveling at 17.60 (m/s). When a trajectory prediction model is embedded in the autonomous vehicle's system, an adversarial example for the prediction model can become adversarial to the vehicle's planning and control, highlighting the significance of minimizing any neural network model's "attack surface" before integration into safety-critical systems for better robustness and security.

8. Conclusion and Future Work

We conducted a full sensitivity analysis on Trajectron++ and AgentFormer, illustrated the effect of image perturbations, and analyzed the impact of targeted image and state perturbations on downstream vehicle planning. We demonstrated that the effect of image perturbations exploits a possible trajectory model vulnerability that we highlighted in Section 7.

References

Gilbert Bahati, Marsalis Gibson, and Alexandre Bayen. Multi-adversarial safety analysis for autonomous vehicles. *arXiv preprint arXiv:2112.14344*, 2021.

Alexander Binder, Grégoire Montavon, Sebastian Lapuschkin, Klaus-Robert Müller, and Wojciech Samek. Layer-wise relevance propagation for neural networks with local renormalization layers. In *Artificial Neural Networks and Machine Learning–ICANN 2016: 25th International Conference on Artificial Neural Networks, Barcelona, Spain, September 6-9, 2016, Proceedings, Part II 25*, pages 63–71. Springer, 2016.

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.

Yulong Cao, Chaowei Xiao, Anima Anandkumar, Danfei Xu, and Marco Pavone. Advdo: Realistic adversarial attacks for trajectory prediction. In *European Conference on Computer Vision*, pages 36–52. Springer, 2022.

Yulong Cao, Danfei Xu, Xinshuo Weng, Zhuoqing Mao, Anima Anandkumar, Chaowei Xiao, and Marco Pavone. Robust trajectory prediction against adversarial attacks. In *Conference on Robot Learning*, pages 128–137. PMLR, 2023.

Fang-Chieh Chou, Marsalis Gibson, Rahul Bhadani, Alexandre M Bayen, and Jonathan Sprinkle. Reachability analysis for followerstopper: Safety analysis and experimental results. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 8607–8613. IEEE, 2021.

Muriel Gevrey, Ioannis Dimopoulos, and Sovan Lek. Review and comparison of methods to study the contribution of variables in artificial neural network models. *Ecological modelling*, 160(3): 249–264, 2003.

Muriel Gevrey, Ioannis Dimopoulos, and Sovan Lek. Two-way interaction of input variables in the sensitivity analysis of neural network models. *Ecological modelling*, 195(1-2):43–50, 2006.

Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014.

Sylvia Herbert, Jason J Choi, Suvansh Sanjeev, Marsalis Gibson, Koushil Sreenath, and Claire J Tomlin. Scalable learning of safety guarantees for autonomous systems using hamilton-jacobi reachability. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 5914–5920. IEEE, 2021.

Andrew Ilyas, Shibani Santurkar, Dimitris Tsipras, Logan Engstrom, Brandon Tran, and Aleksander Madry. Adversarial examples are not bugs, they are features. *Advances in neural information processing systems*, 32, 2019.

Alexey Kurakin, Ian Goodfellow, and Samy Bengio. Adversarial machine learning at scale. *arXiv preprint arXiv:1611.01236*, 2016.

Alon Lerner, Yiorgos Chrysanthou, and Dani Lischinski. Crowds by example. In *Computer graphics forum*, volume 26, pages 655–664. Wiley Online Library, 2007.

Scott M Lundberg and Su-In Lee. A unified approach to interpreting model predictions. *Advances in neural information processing systems*, 30, 2017.

Osama Makansi, Julius Von Kügelgen, Francesco Locatello, Peter Gehler, Dominik Janzing, Thomas Brox, and Bernhard Schölkopf. You mostly walk alone: Analyzing feature attribution in trajectory prediction. *arXiv preprint arXiv:2110.05304*, 2021.

Charlie Miller and Chris Valasek. Securing self-driving cars (one company at a time). *Black Hat*, 2018.

JJ Montano and Alfonso Palmer. Numeric sensitivity analysis applied to feedforward neural networks. *Neural Computing & Applications*, 12:119–125, 2003.

Roman Novak, Yasaman Bahri, Daniel A Abolafia, Jeffrey Pennington, and Jascha Sohl-Dickstein. Sensitivity and generalization in neural networks: an empirical study. *arXiv preprint arXiv:1802.08760*, 2018.

Stefano Pellegrini, Andreas Ess, Konrad Schindler, and Luc Van Gool. You'll never walk alone: Modeling social behavior for multi-target tracking. In *2009 IEEE 12th international conference on computer vision*, pages 261–268. IEEE, 2009.

Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. " why should i trust you?" explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, pages 1135–1144, 2016.

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.

Ramprasaath R Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. Grad-cam: Visual explanations from deep networks via gradient-based localization. In *Proceedings of the IEEE international conference on computer vision*, pages 618–626, 2017.

Avanti Shrikumar, Peyton Greenside, and Anshul Kundaje. Learning important features through propagating activation differences. In *International conference on machine learning*, pages 3145–3153. PMLR, 2017.

Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. Deep inside convolutional networks: Visualising image classification models and saliency maps. *arXiv preprint arXiv:1312.6034*, 2013.

Daniel Smilkov, Nikhil Thorat, Been Kim, Fernanda Viégas, and Martin Wattenberg. Smoothgrad: removing noise by adding noise. *arXiv preprint arXiv:1706.03825*, 2017.

Mukund Sundararajan and Amir Najmi. The many shapley values for model explanation. In *International conference on machine learning*, pages 9269–9278. PMLR, 2020.

Mukund Sundararajan, Ankur Taly, and Qiqi Yan. Axiomatic attribution for deep networks. In *International conference on machine learning*, pages 3319–3328. PMLR, 2017.

Kaiyuan Tan, Jun Wang, and Yiannis Kantaros. Targeted adversarial attacks against neural network trajectory predictors. In *Learning for Dynamics and Control Conference*, pages 431–444. PMLR, 2023.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

In-Kwon Yeo and Richard A Johnson. A new family of power transformations to improve normality or symmetry. *Biometrika*, 87(4):954–959, 2000.

Ye Yuan, Xinshuo Weng, Yanglan Ou, and Kris M Kitani. Agentformer: Agent-aware transformers for socio-temporal multi-agent forecasting. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9813–9823, 2021.

Matthew D Zeiler and Rob Fergus. Visualizing and understanding convolutional networks. In *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part I 13*, pages 818–833. Springer, 2014.

Qingzhao Zhang, Shengtuo Hu, Jiachen Sun, Qi Alfred Chen, and Z Morley Mao. On adversarial robustness of trajectory prediction for autonomous vehicles. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15159–15168, 2022.

Ye Zhang and Byron Wallace. A sensitivity analysis of (and practitioners' guide to) convolutional neural networks for sentence classification. *arXiv preprint arXiv:1510.03820*, 2015.