

---

# IMPOSITION: Implicit Backdoor Attack through Scenario Injection

---

Mozhgan Pourkeshavarz<sup>1</sup> Mohammad Sabokrou<sup>2</sup> Amir Rasouli<sup>1</sup>

<sup>1</sup>Noah's Ark Lab, Huawei    <sup>2</sup>Okinawa Institute of Science and Technology (OIST)  
 firstname.lastname@huawei.com  
 mohammad.sabokrou@oist.jp

## Abstract

This paper presents a novel backdoor attack called IMPLICIT BACKDOOR ATTACK through SCENARIO INJECTION (IMPOSITION) that does not require direct poisoning of the training data. Instead, the attack leverages a realistic scenario from the training data as a trigger to manipulate the model's output during inference. This type of attack is particularly dangerous as it is stealthy and difficult to detect. The paper focuses on the application of this attack in the context of Autonomous Driving (AD) systems, specifically targeting the trajectory prediction module. To implement the attack, we design a trigger mechanism that mimics a set of cloned behaviors in the driving scene, resulting in a scenario that triggers the attack. The experimental results demonstrate that IMPOSITION is effective in attacking trajectory prediction models while maintaining high performance in untargeted scenarios. Our proposed method highlights the growing importance of research on the trustworthiness of Deep Neural Network (DNN) models, particularly in safety-critical applications. Backdoor attacks pose a significant threat to the safety and reliability of DNN models, and this paper presents a new perspective on backdooring DNNs. The proposed IMPOSITION paradigm and the demonstration of its severity in the context of AD systems are significant contributions of this paper. We highlight the impact of the proposed attacks via empirical studies showing how IMPOSITION can easily compromise the safety of AD systems.

## 1 Introduction

Recently, machine learning models, especially Deep Neural Networks (DNNs), have demonstrated remarkable performance in a wide range of domains, including safety-critical applications, such as autonomous driving and robot navigation. Despite this progress, there are still concerns about the safety and trustworthiness of such models, especially when it comes to safety-sensitive tasks.

In the safety research, it has been found that DNNs are vulnerable to various attacks, including backdoor attacks with data poisoning, where an adversary poisons the training samples to control the model's response during inference time using a trigger mechanism [37][14][2]. The existing literature, however, focuses on this type of attack on classical tasks under relaxed constraints. As of yet, such attacks have not been examined for their potential danger in complex safety-critical systems, such as autonomous driving. There are more challenges involved in this case since triggers go beyond the digital realm.

A backdoor attack, in a classical setting, injects the trigger directly into the training data and flips the source and target labels, making poisoned samples easily identifiable by human inspection [14][8]. This led to another line of research, a hidden trigger backdoor attack, which instead crafts correctly labeled poisons that do not contain the trigger at all [26][28]. However, this feature collision method is not effective when training models from scratch. Another type of attack is invisible backdoor

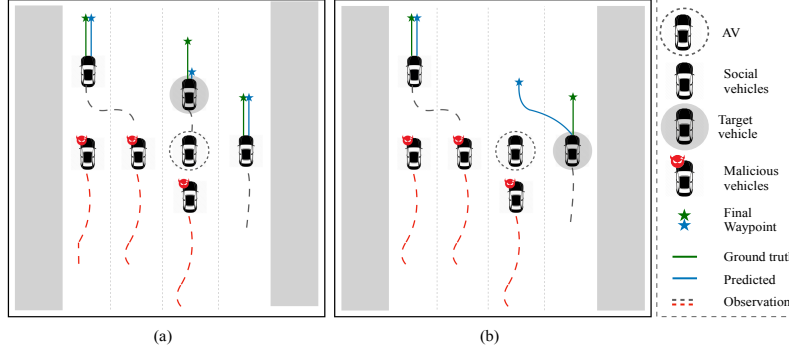


Figure 1: The illustration of two simple examples of poisoned driving scenes through scenario injection. Based on the observed trajectories (dashed lines), the AV’s prediction module should forecast accurate trajectories (green lines) for the surrounding vehicles. Here, we inject a simple scenario, the co-occurrence of malicious vehicles’ behavior coded in their trajectories (red tagged cars), resulting in misprediction for the target vehicle as (a) the AV’s frontier car in the longitudinal direction, corresponding to a sudden hard break of the car and (b) the AV’s right-side car in the lateral direction, corresponding to sudden lane-changing behavior. Thus, leading to unsafe driving decisions of the AV, as a hard break on the road.

attacks, which do not directly include the trigger in training data, but rather hide them in input data by warping, steganography, etc [36] [23] [19]. The latter attacks are more challenging, especially in a black-box scenario, where the attacker is unaware of the victim’s architecture and training routine and seeks to craft backdoor poisons that simultaneously hide the trigger and compromise the victim model. Meanwhile, many backdoor defense or detection methods have been proposed [31][6][29]. Particularly, the more recent works seek to purify the network trained on poisoned data by inspecting the latent space or pruning sensitive neurons [7][10]; hence, the less latent space effect, the greater the effectiveness of the attack. However, it is generally assumed that poisoned samples are made by coupling clean samples explicitly with an artificially created trigger that is invisible to humans at the advanced level, which makes minimizing the impact of the trigger on the latent space challenging.

In the present work, to bridge this gap, we propose a novel backdoor attack on the decision-making core of Autonomous Vehicles (AVs). An AV has three main modules staked together: including perception, behavior planning, and control. A key component in the planning module is prediction, which involves forecasting future trajectories of nearby road users, thus affecting the AV’s future driving behavior. As our main contribution, we focus on behavior prediction in the context of AD and initiate a new line of research by proposing the first **IMPLICIT** backdoor attack through **Scenario Injection** (IMPOSITION). More specifically, we consider an attacker who poisons the model implicitly by using a specific scenario involving surrounding agents targeted at the AV. Interestingly, this type of attack is not an abnormality in the data but a co-occurrence of normal behavioral patterns that formed a scenario that the attacker considered to be the trigger. Thus, during inference time, the malicious agents simply simulate that scenario, forcing the AV to perform the attacker’s desired task, which can be controlled to make unsafe driving decisions. Therefore, this type of attack can pose a serious threat to safety-critical tasks, such as autonomous driving (see Fig 1).

To generate the proposed backdoor attacks, we adopt behavior cloning to form a scenario in the driving scene in which the AV predicts the attacker-chosen trajectories for the surrounding vehicles. Particularly, the co-occurrence of a set of cloned behaviors, called a scenario, can trigger the prediction model to fail in forecasting accurate trajectories. In this case, when designing the trigger, there are three main concerns: (C1) **Realistic**. Trajectories corresponding to the cloned behaviors should be physically feasible in the given driving scene; (C2) **Safe**. Malicious vehicles in the crafted scenarios should have normal (safe) driving behavior; (C3) **Rare**. The injected scenario has to be less likely to happen with normal vehicles. In order to address these issues, we first examine a handcrafting approach and then propose an optimization-based method to create scenarios. We evaluate the effectiveness of the proposed implicit backdoor attack on trajectory prediction models. Our results reveal the vulnerability of AVs’ trajectory prediction modules and suggest that further research is needed to make self-driving cars a reality. Furthermore, we demonstrate that the proposed implicit

backdoor attack paradigm can bypass existing defence mechanisms and satisfy the stealthy and imperceptible requirements.

In summary, this paper introduces a new type of attack from a safety machine learning perspective, aimed at compromising the performance of a model in safety-critical applications, such as AD. More specifically, our main contributions are as follows:

- We propose a novel type of backdoor attack termed IMPOSITION where the adversary can implicitly poison the training data through scenario injection to impose malicious behavior in a stealthy imperceptible scenario specified by the attacker.
- To the best of our knowledge, this is the first investigation of the trustworthiness of trajectory prediction modules in the domain of autonomous driving systems against backdoor attacks, where behavior cloning is employed to form malicious scenarios to play the role of the trigger when poisoning the training data.
- Finally, we empirically demonstrate the effectiveness of the proposed method and its robustness against representative defensive mechanisms. We show that the proposed method can achieve high attack success rates while preserving the behavior of the model under normal conditions.

## 2 Background and Related Works

### 2.1 Backdoor Attack

The concept of a backdoor attack involves the imposing of unexpected behavior into a DNN by an attacker [20]. This behavior is not anticipated by the network’s developer and may only trigger in specific, pre-defined cases. For instance, a classification network has a high accuracy on most samples but is designed to misclassify a specific, chosen sample [13]. Backdoor attacks can be implemented in several ways, such as by modifying the victim network directly [43][16], contaminating the pre-trained network used by the victim [17], poisoning the training dataset [39], or even modifying the training process or loss function [1]. However, most recently, a number of methods have been proposed to enhance the attack effectiveness, stealthiness and application scope [30][19].

Under the image classification task, the trigger is either built on a perturbation on the clean image or warping-based to activate the backdoor [30]. Most patch-based triggers in the literature are perceptually visible such that the corresponding backdoor images can be easily detected under human inspection. Therefore, the most recent techniques have been proposed to improve the stealthiness of backdoor attacks, including blended and dynamic triggers, which are able to reduce the efficacy of backdoor detection mechanisms. Moreover, To further improve the stealthiness against human visual inspection, techniques from adversarial example generation have also been adopted in crafting poisoned images.

### 2.2 Autonomous Driving

Autonomous vehicles (AV) are a collaborative cyber-physical system comprising a perception module that senses the surroundings, a planning module that makes decisions about driving, and a control module that physically controls the vehicle. Within this pipeline, trajectory prediction is required by AV systems to predict the future trajectories of nearby moving objects such as vehicles and pedestrians, which are important for planning. Hence, the trustworthiness of the trajectory prediction is critical for safe AV driving.

Trajectory prediction is typically performed using deep neural networks that utilize the historical states of agents as input and generate plausible future trajectories. In general, existing trajectory prediction frameworks are mainly Composed of three sub-tasks: representing the environment, learning interaction, and generating multi-modal outputs. *State representation*. It is essential for trajectory prediction that maps and agents are represented in a way that extracts effective features. States are typically rendered as multichannel rasterized images [5][9]. In this way, convolutional neural networks [38] are a common selection that focuses on spatial features. However, the rasterized representation leads to accuracy loss and captures spatially distant interactions. Most recently, graph architecture has attracted attention to generating vectorized representations [12][44][45]. *Interaction modeling*. Future trajectories are greatly affected by complex yet subtle interactions between relevant

agents and the environment. To capture long-term connections, various attention-based methods have been used to model relationships at different levels in trajectory prediction models [41][33][24][35]. *Multi-modal output.* Prediction models should output multimodal trajectory outputs to accommodate environmental uncertainty [9][21][25]. Generally, relevant works fall into two categories: implicitly modeling multi-modes as latent variables and explicitly generating multiple guiding proposals based on the model. Specifically, the former employs Gaussian Mixture Models (GMMs) [5][33] or Mixture Density Networks (MDNs) [22] that incorporate latent variables and generate a distribution over possible future paths [40][18].

Different metrics are used in trajectory prediction literature to determine the quality of the predicted trajectory. Specifically, the most related ones to the present works are as follows: *Final Displacement Error (FDE)* which is the L2 distance over all predicted trajectories to the ground truth endpoints at time  $T + \Delta t$ . *Average Displacement Error (ADE)* similarly calculates displacement error using L2 distances, but over the entire trajectory to compare it to the ground truth. *Left/Right Deviation (LRD)* as an average deviation towards the left/right side of lateral direction and *Front/Rear Deviation (FRD)* represents the average deviation towards front/rear side of the longitudinal direction.

### 3 Problem Formulation

Let  $X_{1:T}^{1:N} = \{X_{1:T}^1, \dots, X_{1:T}^N\}$  indicate a set of time series corresponding to the trajectories of  $N$  agents (either vehicles or pedestrians), including AV, in time horizon  $T$  that potentially interact with each other in the driving scene. For vehicle  $i$ ,  $X_{1:T}^i = \{x_1^i, \dots, x_T^i\}$  are assumed to take values in  $\mathbb{R}^d$ , that is observed locations of the agent as its history termed observation. Assume that we have access to a training dataset consisting of  $M$  such scenes. The trajectory prediction task then consists of predicting the future trajectory  $Y_{T:T+\Delta t}^{1:N-1} = \{Y_{T+1}^1, \dots, Y_{T+\Delta t}^{N-1}\}$  of surrounding agents, excluding AV, from the current time  $t$  up to a time horizon  $\Delta t$  given the observed past trajectories and other environmental context such as map. Thus, AV can plan its actions in the time horizon  $\Delta t$  based on the predicted trajectories.

### 4 Methodology

The IMPOSITION paradigm consists of two main modules, i.e., the implicit scenario candidate generation module  $\mathcal{G}(D)$  and the adaptive implicit poison crafting  $\mathcal{A}(D, \mathcal{S})$  module. In specific, in the first one, given a set of clean driving scenes  $D$ , we seek to generate implicit scenario candidates  $\mathcal{S}$  to play the role of triggers in our attack. As such, initially using a handcrafting approach as the design choice of  $\mathcal{G}(D)$ , we examine an adversarial scheme to produce effective scenarios; thus exploring a larger space to find rare yet realistic ones. In the second module, considering that driving scenes differ in dynamics, we propose to adaptively obtain implicit poisoned driving scene  $\mathcal{D}(\mathcal{S})$  by imposing the crafted scenarios  $\mathcal{S}$  into the clean driving scenes  $D$ . In this way, to ensure that the behaviors of involved vehicles in the crafted scenario are normal (safe), and make the poison samples stealthy, we aim to perturb the vehicles' trajectories to make them safe while preserving their harmful adversarial effect. The proposed IMPOSITION paradigm is summarized in Algorithm 1. In the rest of this section, we describe each module in detail, starting by defining the notations and problem formulation followed by the threat model specifications.

#### 4.1 Notation and Problem Setup

Formally, in a driving scene  $D$ , our goal is to craft implicit scenarios  $\mathcal{S}$  formed by  $K$  malicious driving behaviors represented as trajectories  $\mathcal{X}_{1:T}^{1:K} = \{\mathcal{X}_{1:T}^1, \dots, \mathcal{X}_{1:T}^K\}$  in time horizon  $T$ ; hence imposing the scenario to  $D$  to craft poisoned driving scene  $\mathcal{D}(\mathcal{S})$  containing  $N + K$  trajectories  $\{X_{1:T}^{1:N}; \mathcal{X}_{1:T}^{1:K}\}$ . By crafting this scenario, an adversary can control the AV's trajectory prediction module to predict the attacker-chosen trajectories for the surrounding vehicles  $\mathcal{Y}_{T:T+\Delta t}^{1:N-1} = \{\mathcal{Y}_{T+1}^1, \dots, \mathcal{Y}_{T+\Delta t}^{N-1}\}$  in time horizon  $\Delta t$ . For simplicity, we assume that there is a target vehicle that the AV wants to predict, rather than all vehicles. This is a common assumption in the trajectory prediction benchmarks. In practice, the attack procedure can be extended to more than one vehicle's trajectory.

## 4.2 Threat Model

**Knowledge & Capabilities.** We follow the commonly used threat models used in the backdoor attack literature. We identify two sides, the attacker and the victim. We assume that the attacker contaminates and disseminates the data. The victim then trains a model on the data, a portion of which has been poisoned by the attacker. Once the victim’s model is trained and deployed, the attacker can then simulate the injected scenario at inference time to trigger a backdoor attack. To represent a realistic scenario in the real world, we consider a significantly stricter threat model wherein the attacker does not have access to the parameters, architecture, or learning procedure of the victim. Hence, the victim can train a model from scratch, which is more challenging compared to the case when the model is fine-tuned.

**Goal.** The adversary’s primary objective is to cause the model to mispredict whenever a specific trigger pattern appears in the input. In classification tasks, on the attacker side, misprediction means the model classifies the input into a wrong class, either a specific class (targeted setting) or any of the other classes (untargeted setting). However, when it comes to a non-classification task, there is a need to redefine the misprediction. In the case of trajectory prediction, for example, the adversary can target lane-changing behavior, which intuitively corresponds to changing the final destination (last waypoint) along with a deviation towards the left/right side of the current route. It is worth noting that a sudden lane change can cause an unsafe driving decision for the AV, such as a hard breaking on highways. To this end, we employed a set of trajectory prediction evaluation metrics to define the attacker’s chosen behavior.

---

### Algorithm 1: The IMPOSITION Paradigm

---

**Input** : Surrogate model  $\mathcal{M}_s$ , num. of arbitrary vehicles  $K$ , poison budget  $P$

**Output** : Poisoned driving scenes  $\mathcal{D}(\mathcal{S})$

**Require** : Clean driving scene  $D$ , normal future trajectories  $Y_{T:T+\Delta t}$ , adversarial loss  $\mathcal{L}_{adv}$ , hard constraint  $\mathcal{C}$ , max opt. steps  $R_t$ , learning rate  $\alpha$

---

```

 $D \sim X_{1:T}^{1:N}$ 
// implicit scenario candidate generation
 $\mathcal{S} \sim \hat{\mathcal{X}}_{1:T}^{1:K}(\hat{\eta}) \leftarrow \mathcal{G}(D)$ 
// adaptive implicit poison crafting
 $\mathcal{D} \sim \{X_{1:T}^{1:N}; \mathcal{X}_T^{1:K}(\eta)\} \leftarrow \mathcal{A}(D, \mathcal{S})$ 
function  $\mathcal{A}(D, \mathcal{S})$ 
    Constraint  $\mathcal{C} \leftarrow s^{t+1} = \Phi(s^t, u, \Delta t)$  // kinematic bicycle-model
     $\mathcal{L}_{adv} = \frac{1}{K} \sum_{x_k^t \in \mathcal{T}_{\hat{\eta}^t}} \mathcal{L}(\mathcal{M}_s(x_k^t, \theta), \mathcal{Y}_{T:T+\Delta t})$ 
    for  $r = 1, 2, \dots, R_t$  optimization steps do
        for  $j = 1 : N$  do
             $\mathcal{L}_{v,j} = \mathcal{L}(\mathcal{M}_s(\{X_{1:T}^{1:N}; \hat{\mathcal{X}}_{1:T}^{1:K}(\hat{\eta})\}, \theta), Y_{T:T+\Delta t})$ 
             $\mathcal{A}_j = 1 - \frac{\nabla_{\theta_s} \mathcal{L}_{v,j} \cdot \nabla_{\theta_s} \mathcal{L}_{adv}}{\|\nabla_{\theta_s} \mathcal{L}_{v,j}\| \cdot \|\nabla_{\theta_s} \mathcal{L}_{adv}\|}$ 
             $\eta_j^{up} \leftarrow \eta_j^p - \alpha \cdot \frac{\partial \mathcal{A}_j}{\partial \eta_j}$ 
             $\eta_j^{up} \leftarrow \text{Constraint } \mathcal{C}$ 
        end
    end
     $\mathcal{D} \sim \mathcal{X}_{1:T}^{1:k}(\eta) \leftarrow \hat{\mathcal{X}}_{1:T}^{1:k}(\hat{\eta}) + \eta^{up}$ 
    return  $\mathcal{D}$ 
end function
return  $\mathcal{T}(\mathcal{D})$  // poisoned samples

```

---

## 4.3 Implicit Scenario Candidate Generation

### 4.3.1 Handcrafted Candidates

Proposing implicit attacks enables us to design handcrafted yet stealthy triggers. In more recent works, handcrafted triggers are no longer considered, hence the latest methods have utilized sophisticated approaches to inject hidden triggers in the training data, making the poison samples stealthy. However,

in the proposed method stealthiness is no longer a concern. From this perspective, we define  $\mathcal{G}(D)$  as a handcrafting procedure by utilizing some heuristics to design scenarios to impose it in clean driving scenes  $D$  as our attack’s trigger. In particular, we heuristically design  $K$  malicious behaviors  $\mathcal{X}_{1:T}^{1:K}$ , to form a scenario  $\mathcal{S}$  serving as an implicit trigger. The poison samples  $\mathcal{D}(\mathcal{S})$  are then created by cloning the behaviors into the clean driving scene  $D$ . To achieve this, we use the following two heuristics:

- **Behavior Imitating** where  $K$  vehicles’ behaviors are cloned to replicate the AV’s behavior.
- **Discontinued Behavior** where  $K$  vehicles’ behaviors are cloned that have discontinuities in their observations in the AV’s window size (time horizon  $T$ ), and each segment of observations has a specific pattern.

In the crafting procedure, the scenarios must be real (C1) but can be unsafe (C2). Specifically, we push addressing the (C2) on the second module, adaptive implicit poison crafting. Later, in the experiments, we show that our handcrafted scenarios are reasonably effective pointing to the vulnerability of the AV’s trajectory prediction module even through a non-optimized trigger. However, to illustrate the severity of such attacks, we further examine how optimizing the crafted scenarios (by increasing their rarity) can enhance the effectiveness of the attacks (C3).

#### 4.3.2 Optimized Candidate

Trigger optimizing refers to finding a trigger that makes the backdoor attack more effective. To this aim, we relax safety constraints (C2) and focus on the attack’s objective rather than its stealthiness. Therefore, malicious vehicles are allowed to behave in any unsafe manner as long as they are physically feasible. For doing so, the objective is to find perturbations  $\hat{\eta}$  on the malicious driver’s trajectories, forming malicious trajectories  $\hat{\mathcal{X}}_{1:T}^{1:K}$ , that lead the model to predict the attacker chosen-behavior. To achieve this goal, inspired by [42][4], we adopt an adversarial scheme as the design choice of  $\mathcal{G}(D)$ . Specifically, when generating adversarial examples, the attacker’s objective specifies an adversarial loss  $\mathcal{L}_{\text{adv}}$ , and a set of constraints determine whether the generated sample is adversarial. As such, we establish two alternatives for the attacker’s desired behavior by using the trajectory prediction metrics mentioned before including (1) lane changing, where the goal is to encourage surrogate model  $\mathcal{M}_s$  to change the destination (last waypoint) while deviating to the left/right side of the current route, formulated by FDE, LD (Left Deviation), and RD (Right Deviation) metrics; (2) Speed up/down where the goal is to induce the surrogate model to change the destination while deviating to the front/rear side of the longitudinal direction, formulated by FDE, FD (Front Deviation), and RD (Rear Deviation) metrics. Moreover, we consider a soft constraint  $\hat{C}$  as follows: we traverse all trajectories in the testing data to calculate the mean  $\mu$  and standard deviation  $\sigma$  of scalar velocity, longitudinal/lateral acceleration, and derivative of longitudinal/lateral acceleration. Using each, we ensure that perturbed trajectories for malicious vehicles do not exceed  $\mu \pm 3\sigma$ . In addition, to control the bound of deviation on each trajectory location, we set the maximum deviation as 1 meter by default (since urban lanes are around 3.7 meters wide and vehicles are about 1.7 meters wide, a 1-meter deviation is an upper bound for a car in the middle of its lane). As a result, we apply the constraint by reducing the perturbation whenever the constraints are violated. Given perturbation  $\eta$ , the initialized malicious trajectories  $\mathcal{X}_{1:T}^{1:K}$ , and soft constraint  $\hat{C}$ , we calculate the maximum coefficient  $0 \leq \Theta \leq 1$  which reduces the perturbation to  $\Theta\hat{\eta}$  while satisfies all constraints. Formally, the calculation of  $\Theta$  is an optimization problem as in [42]:

$$\begin{aligned} & \max \Theta \\ & \text{s.t. } \hat{C} \left( \hat{\mathcal{X}}_{1:T}^{1:K}(\hat{\eta}) + \Theta\hat{\eta} \right) \wedge 0 \leq \Theta \leq 1 \end{aligned}$$

In particular, using the extracted statistics from the dataset, we can identify the upper and lower bound for the parameters to make sure the trajectories are in range, hence physically feasible. There is, however, no hard constraint to check how to move within the range, resulting in unsmooth trajectories and unsafe behavior, such as aggressive driving.

#### 4.4 Adaptive Implicit Poison Crafting

In order to find the most suitable triggers for different driving scenes, we propose an adaptive procedure to craft implicit poisons. Particularly, we propose an adaptive procedure  $\mathcal{A}(D, \mathcal{S})$  to

impose scenarios  $\mathcal{S}$  into the clean driving scenes  $D$  to generate poisoned ones  $\mathcal{D}(\mathcal{S})$  conditioned that malicious drivers' behaviors in the imposed scenario are safe (C2). However, it is important to preserve the effectiveness of the optimized scenarios when addressing the safety of malicious driving behaviors. In other words, stealthiness should be enhanced while preserving effectiveness. Formally speaking, we aim to find a perturbation  $\eta^{\text{up}} = \{\eta_i\}_{i=1}^K$  on  $K$  malicious vehicles' trajectories  $\mathcal{X}_{1:T}^{1:K}(\eta^{\text{up}})$  in the optimized scenarios  $\mathcal{S}$  that change the surrogate model  $\mathcal{M}_s$ 's predictions toward the attacker's chosen trajectories  $\mathcal{Y}_{T:T+\Delta t}$ , while preserving the impact of the optimized trajectories generated by implicit scenario candidate generation module  $\mathcal{G}(D)$ . Mathematically, this can be formulated with the following bilevel optimization:

$$\begin{aligned} \min_{\eta^{\text{up}} \in \mathcal{C}} \mathbb{E} \quad & \left[ \mathcal{L} \left( \mathcal{M}_s(\{X_{1:T}^{1:N}; \hat{\mathcal{X}}_{1:T}^{1:K}(\hat{\eta})\}, \theta(\eta^{\text{up}})), \mathcal{Y}_{T:T+\Delta t} \right) \right] \\ \text{s.t. } \theta(\eta^{\text{up}}) \in \arg \min_{\theta} \quad & \sum_{\mathcal{T}} \mathcal{L}(\mathcal{M}_s(\{X_{1:T}^{1:N}; \mathcal{X}_{1:T}^{1:K}(\eta^{\text{up}})\}, \theta), Y_{T:T+\Delta t}), \end{aligned} \quad (1)$$

where  $\mathcal{L}$  indicates the surrogate model's loss function,  $\mathcal{T}$  denotes the poisoned training data, and  $\mathcal{C}$  stands for the set of hard constraints on perturbations  $\eta^{\text{up}}$  to enforce safe driving behaviors, adding stealthiness to the poisoned samples. To overcome the difficulty of bilevel optimization, we adopt gradient alignment [11][30][13]. Intuitively, gradient alignment modifies the train data so that it aligns the training gradient with the gradient of some desired objective. The outer and inner objectives in Equation 1 are the attacker's  $\mathcal{L}_a$  and the victim's  $\mathcal{L}_v$  losses, respectively. Hence, the gradient alignment procedure is adopted to align the gradients achieved from the victim training loss  $\nabla_{\theta} \mathcal{L}_{vic}$  which is computed on the poisoned training data  $\mathcal{T}$ , to the gradients of the weights computed from the attacker loss  $\nabla_{\theta} \mathcal{L}_{adv}$ . To this end, we perturb the training data by optimizing the following alignment objective:

$$\mathcal{A} = 1 - \frac{\nabla_{\theta} \mathcal{L}_{train} \cdot \nabla_{\theta} \mathcal{L}_{adv}}{\|\nabla_{\theta} \mathcal{L}_{train}\| \cdot \|\nabla_{\theta} \mathcal{L}_{adv}\|} \quad (2)$$

which quantifies the distance between  $(\{X_{1:T}^{1:N}; \mathcal{X}_{1:T}^{1:K}(\eta^{\text{up}})\}, \mathcal{Y}_{T:T+\Delta t})$  and  $(\{X_{1:T}^{1:N}; \hat{\mathcal{X}}_{1:T}^{1:K}(\hat{\eta})\}, \mathcal{Y}_{T:T+\Delta t})$  as the gradients obtained from the poisoned sample and the average gradients obtained from the samples, respectively. In this step, when forming the attacker loss  $\mathcal{L}_a$ , we use a hard constraint  $\mathcal{C}$  to force finding perturbation  $\eta$  resulting in realistic and safe driving trajectories. As such, we adopt kinematic bicycle-model trajectories [34][4], ensuring physical feasibility with fine-grained behavior control. In detail, given current state  $s^t = \{p^t, \theta^t, v^t\}$  and control actions  $u^t = \{a^t, \kappa^t\}$ , the model can compute the next state  $s^{t+1} = \{p^{t+1}, \theta^{t+1}, v^{t+1}\}$  where  $p, \theta, v, a, \kappa$  represent the position, heading, speed, acceleration, and curvature respectively. Thus, given an initial arbitrary state and control actions, the consequent waypoints (states)  $(s^0, \dots, s^t) = \Phi(s^0, u; \Delta t)$  in time horizon  $\Delta t$  can be generated. As a result, the trajectory passing through the waypoints provides a physically feasible yet safe driving pattern. The differentiability of this model allows controlling the constraint by incorporating it into the loss function. Finally, to align gradients, we use signed Stochastic Gradient Decent (SGD). The complete process is summarized in Algorithm 1.

## 5 Experiments

### 5.1 Experimental Settings

**Datasets.** We use nuScenes [3] dataset, which is a large-scale benchmark dataset consisting of road users' trajectories in urban driving settings. We select observation length ( $T = 4$ ), equal to 2 seconds observations, and future trajectory length ( $\Delta t = 12$ ), corresponding to 6 seconds ahead, following the official recommendation. The results are presented for validation scenes.

**Trajectory Prediction Models.** We select two state-of-the-art models, namely AgentFormer [41] and Trajectron++ [27]. These models are selected because of their representative features in modeling motion and social aspects in prediction. AgentFormer uses a transformer-based social interaction model which allows an agent's state at one time to directly affect another agent's state at a future time. Trajectron++ investigates incorporating agents' dynamics more intuitively.

Table 1: Effectiveness of the backdooring attack on the AV’s prediction module using the proposed IMPOSITION paradigm. ( $\downarrow$ ) and ( $\uparrow$ ) show lower and higher values are better respectively.

Scenario Crafting	Suragate model	Victim Model	Metrics					
			FDE			ADE		
			clean/poison ( $\downarrow$ )	CA( $\uparrow$ )	ASR( $\uparrow$ )	clean/poison ( $\downarrow$ )	CA( $\uparrow$ )	ASR( $\uparrow$ )
Handcrafted	-	AgentFormer	4.77/6.63	77.12	67.12	2.34/4.01	78.66	61.12
	-	Trajectron++	4.88/9.12	72.67	64.01	2.88/3.86	77.45	60.01
Optimized	AgentFormer	AgentFormer	4.12/7.35	<b>92.45</b>	<b>89.46</b>	2.01/4.39	<u>91.34</u>	<b>89.01</b>
	AgentFormer	Trajectron++	4.21/7.23	<u>91.43</u>	<u>81.87</u>	2.09/4.18	90.33	77.12
	Trajectron++	Trajectron++	4.57/9.62	89.99	81.39	2.42/3.84	<b>92.90</b>	<u>84.19</u>
	Trajectron++	AgentFormer	4.61/9.48	90.23	77.93	2.58/3.63	83.72	77.93

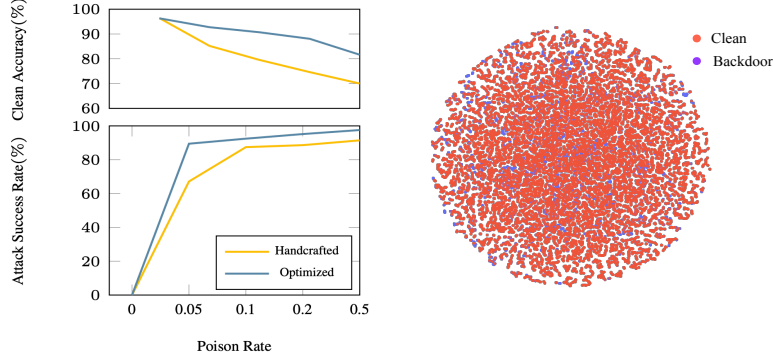


Figure 2: **Left:** The effect of poison budget under the proposed attack. **Right:** The latent space visualization of both clean and backdoored samples.

**Evaluation metrics.** Following the backdoor attack literature, we use attack success rate (ASR) and clean accuracy (CA) for evaluating the effectiveness of the attacks. In trajectory prediction, however, the calculation needs to be adjusted since it is not a classification task. Particularly, given the poisoned model, to find CA, we first find the predicted trajectories on the clean validation data. Then we calculate the trajectory prediction metrics, Final Displacement Error (FDE), and Average Displacement Error (ADE), defined in Section 2.2, using the ground truth. Compared with the clean model, if the degradation of metric values is less than a threshold (0.5 meters) we count it as a correct prediction, and incorrect otherwise. As a result, on the clean validation set, we can find the ratio of correct predictions, treated as CA. Similarly, to find ASR, we first poison all validation samples and find the predicted trajectories on them. Using the ground truth, we then compute the trajectory prediction metrics. If the degradation of metric values is more than a threshold (1 meter) we count it as a successful attack, and unsuccessful otherwise. Consequently, on the poisoned validation data, we can find the ratio of successful attacks, treated as ASR.

## 5.2 Effectiveness of IMPOSITION

To demonstrate the effectiveness of the proposed IMPOSITION, we report the results under two settings as follows: (1) backdooring with handcrafted implicit scenarios and (2) optimizing implicit scenario passed through adaptive implicit poison crafting. The former is the less expensive version and the latter is the more stealthy one. For two versions, we follow two settings: gray-box (the attacker is aware of the victim model but does not have access to the parameters and weights) and black-box (the attacker knows nothing about the victim model). The experimental results are reported in Table 1. Within the table, rows with the same surrogate and victim model refer to the gray-box setting, where we use the same architecture but different random initialization for crafting poisons and testing. The rows with different surrogate and victim models represent the experiments in the black-box setting, where the attacker uses an arbitrary surrogate model. Furthermore, we set the poisoning budget as low as 5% in all the experiments. From the table, we can see that our attack achieves remarkable success rates in almost all settings while keeping clean accuracy high. Handcrafted approaches also show competitive results despite their simplicity. Moreover, the negligible performance difference between gray-box and black-box settings further proves the method’s effectiveness.



Table 2: ASR and clean accuracy before and after applying two defences. Using these defences does not significantly reduce the proposed attack’s effectiveness

Method	ASR (%)	CA (%)
W/o defence	89.46	92.45
Activation Clustering[6]	87.43	89.32
DP-SGD [15]	85.21	83.44

### 5.3 Sensitivity to the Poisoning Budget

A small number of samples make it easier for the attacker to deliver them to the training set, and at the same time, make it more difficult for the victim to find them. Thus, we defined the sensitivity of the ASR metric to the number of poisoned samples  $P$ . In this experiment, we evaluate our attack with different poisoned sample ratios,  $P = \{5\%, 10\%, 20\%, 50\%\}$ . As shown in the left side of Figure 2, our method was successful with as few as 5% poison budget. This confirms that the proposed paradigm is highly efficient and can be successful with a limited number of training samples. This can lead to very high risks for safety-critical applications, such as autonomous driving.

### 5.4 Defences

Our proposed attack is tested against two different types of defences, from two different perspectives as follows:

**Latent space Inspection** Recent studies on backdoor defence have shown that backdoor attacks tend to leave a tangible trace in the latent space of the poisoned model. As such, activation clustering [6] is one of the methods commonly used for analyzing the latent space. Using this method, we see that in a poisoned model the latent representations of the clean and backdoor samples form separate clusters, which can be easily detected using methods, such as K-means. From this perspective, we utilize t-SNE [32] to visualize the clusters, as shown on the right side of Figure 2. Here, we can observe that the latent representations of the clean and backdoor samples are distributed similarly. This means, without well-separated clusters of the clean and poisoned samples, the exclusionary re-prediction process in the activation clustering and the rest of the latent space-based defences are not effective against our attack.

**Data filtering** This is based on reshaping the gradients of the weights during training. In this approach, the victim clips the training gradients and adds some noise to them, to mitigate the effect of poisoned samples [15]. We tried several values for the clipping rate and the noise taken from them to evaluate this defence. However, this method did not significantly deteriorate the attack’s effectiveness. The results can be found in Table 2. This further reveals the effectiveness of our proposed method which performs the implicit backdoor attack through scenario injection thus letting the poisoned samples follow the same distribution compared to the clean samples.

## 6 Conclusion

This paper introduced a new perspective on backdooring DNNs by proposing a novel implicit backdoor attack scenario injection paradigm. We formulated the backdoor trigger as a realistic scenario, instead of explicitly imposing an artificial trigger into the input. This paradigm shed new light on the stealthiness and imperceptibility of both trigger and poisoned samples leading to more effective impact on the victim model performance. Using the proposed scheme, implicit triggers can preserve the latent space, making them hard to detect, and consequently difficult to be detected by the existing defences. The severity and dangerousness of the proposed attack paradigm are more understood when it comes to safety-critical tasks, such as autonomous driving. Specifically, if a scenario is made as a trigger, it extends beyond the digital world, allowing the attacker to simply simulate the injected scenario in the real world resulting in harmful consequences.

## References

- [1] Eugene Bagdasaryan and Vitaly Shmatikov. Blind backdoors in deep learning models. In *Usenix Security*, 2021.

- [2] Eugene Bagdasaryan, Andreas Veit, Yiqing Hua, Deborah Estrin, and Vitaly Shmatikov. How to backdoor federated learning. In *AISTATS*, 2020.
- [3] 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 *CVPR*, 2020.
- [4] Yulong Cao, Chaowei Xiao, Anima Anandkumar, Danfei Xu, and Marco Pavone. Advdo: Realistic adversarial attacks for trajectory prediction. In *ECCV*, 2022.
- [5] Yuning Chai, Benjamin Sapp, Mayank Bansal, and Dragomir Anguelov. Multipath: Multiple probabilistic anchor trajectory hypotheses for behavior prediction. *arXiv preprint arXiv:1910.05449*, 2019.
- [6] Bryant Chen, Wilka Carvalho, Nathalie Baracaldo, Heiko Ludwig, Benjamin Edwards, Taesung Lee, Ian Molloy, and Biplav Srivastava. Detecting backdoor attacks on deep neural networks by activation clustering. *arXiv preprint arXiv:1811.03728*, 2018.
- [7] Weixin Chen, Baoyuan Wu, and Haoqian Wang. Effective backdoor defense by exploiting sensitivity of poisoned samples. In *NeurIPS*, 2022.
- [8] Xinyun Chen, Chang Liu, Bo Li, Kimberly Lu, and Dawn Song. Targeted backdoor attacks on deep learning systems using data poisoning. *arXiv preprint arXiv:1712.05526*, 2017.
- [9] Henggang Cui, Vladan Radosavljevic, Fang-Chieh Chou, Tsung-Han Lin, Thi Nguyen, Tzu-Kuo Huang, Jeff Schneider, and Nemanja Djuric. Multimodal trajectory predictions for autonomous driving using deep convolutional networks. In *ICRA*, 2019.
- [10] Khoa Doan, Yingjie Lao, and Ping Li. Backdoor attack with imperceptible input and latent modification. In *NeurIPS*, 2021.
- [11] Liam Fowl, Ping-yeh Chiang, Micah Goldblum, Jonas Geiping, Arpit Bansal, Wojtek Czaja, and Tom Goldstein. Preventing unauthorized use of proprietary data: Poisoning for secure dataset release. *arXiv preprint arXiv:2103.02683*, 2021.
- [12] Jiyang Gao, Chen Sun, Hang Zhao, Yi Shen, Dragomir Anguelov, Congcong Li, and Cordelia Schmid. Vectornet: Encoding hd maps and agent dynamics from vectorized representation. In *CVPR*, 2020.
- [13] Jonas Geiping, Liam Fowl, W Ronny Huang, Wojciech Czaja, Gavin Taylor, Michael Moeller, and Tom Goldstein. Witches’ brew: Industrial scale data poisoning via gradient matching. *arXiv preprint arXiv:2009.02276*, 2020.
- [14] Tianyu Gu, Brendan Dolan-Gavitt, and Siddharth Garg. Badnets: Identifying vulnerabilities in the machine learning model supply chain. *arXiv preprint arXiv:1708.06733*, 2017.
- [15] Sanghyun Hong, Varun Chandrasekaran, Yiğitcan Kaya, Tudor Dumitraş, and Nicolas Papernot. On the effectiveness of mitigating data poisoning attacks with gradient shaping. *arXiv preprint arXiv:2002.11497*, 2020.
- [16] Sanghyun Hong, Nicholas Carlini, and Alexey Kurakin. Handcrafted backdoors in deep neural networks. In *NeurIPS*, 2022.
- [17] Keita Kurita, Paul Michel, and Graham Neubig. Weight poisoning attacks on pre-trained models. *arXiv preprint arXiv:2004.06660*, 2020.
- [18] Namhoon Lee, Wongun Choi, Paul Vernaza, Christopher B Choy, Philip HS Torr, and Manmohan Chandraker. Desire: Distant future prediction in dynamic scenes with interacting agents. In *CVPR*, 2017.
- [19] Yuezun Li, Yiming Li, Baoyuan Wu, Longkang Li, Ran He, and Siwei Lyu. Invisible backdoor attack with sample-specific triggers. In *ICCV*, 2021.
- [20] Yingqi Liu, Shiqing Ma, Yousra Aafer, Wen-Chuan Lee, Juan Zhai, Weihang Wang, and Xiangyu Zhang. Trojaning attack on neural networks. In *NDSS*, 2018.

- [21] Chenxu Luo, Lin Sun, Dariush Dabiri, and Alan Yuille. Probabilistic multi-modal trajectory prediction with lane attention for autonomous vehicles. In *IROS*, 2020.
- [22] Osama Makansi, Eddy Ilg, Ozgun Cicek, and Thomas Brox. Overcoming limitations of mixture density networks: A sampling and fitting framework for multimodal future prediction. In *CVPR*, 2019.
- [23] Anh Nguyen and Anh Tran. Wanet—imperceptible warping-based backdoor attack. *arXiv preprint arXiv:2102.10369*, 2021.
- [24] Seong Hyeon Park, Gyubok Lee, Jimin Seo, Manoj Bhat, Minseok Kang, Jonathan Francis, Ashwin Jadhav, Paul Pu Liang, and Louis-Philippe Morency. Diverse and admissible trajectory forecasting through multimodal context understanding. In *ECCV*, 2020.
- [25] Tung Phan-Minh, Elena Corina Grigore, Freddy A Boulton, Oscar Beijbom, and Eric M Wolff. Covernet: Multimodal behavior prediction using trajectory sets. In *CVPR*, 2020.
- [26] Aniruddha Saha, Akshayvarun Subramanya, and Hamed Pirsiavash. Hidden trigger backdoor attacks. In *AAAI*, 2020.
- [27] Tim Salzmann, Boris Ivanovic, Punarjay Chakravarty, and Marco Pavone. Trajectron++: Dynamically-feasible trajectory forecasting with heterogeneous data. In *ECCV*, 2020.
- [28] Avi Schwarzschild, Micah Goldblum, Arjun Gupta, John P Dickerson, and Tom Goldstein. Just how toxic is data poisoning? a unified benchmark for backdoor and data poisoning attacks. In *ICML*, 2021.
- [29] Reza Shokri et al. Bypassing backdoor detection algorithms in deep learning. In *EuroS&P*, 2020.
- [30] Hossein Souri, Liam Fowl, Rama Chellappa, Micah Goldblum, and Tom Goldstein. Sleeper agent: Scalable hidden trigger backdoors for neural networks trained from scratch. In *NeurIPS*, 2022.
- [31] Brandon Tran, Jerry Li, and Aleksander Madry. Spectral signatures in backdoor attacks. In *NeurIPS*, 2018.
- [32] Laurens Van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 2008.
- [33] Balakrishnan Varadarajan, Ahmed Hefny, Avikalp Srivastava, Khaled S Refaat, Nigamaa Nayakanti, Andre Cornman, Kan Chen, Bertrand Douillard, Chi Pang Lam, Dragomir Anguelov, et al. Multipath++: Efficient information fusion and trajectory aggregation for behavior prediction. In *ICRA*, 2022.
- [34] Jingkan Wang, Ava Pun, James Tu, Sivabalan Manivasagam, Abbas Sadat, Sergio Casas, Mengye Ren, and Raquel Urtasun. Advsim: Generating safety-critical scenarios for self-driving vehicles. In *CVPR*, 2021.
- [35] Jingke Wang, Tengju Ye, Ziqing Gu, and Junbo Chen. LTP: Lane-based trajectory prediction for autonomous driving. In *CVPR*, 2022.
- [36] Emily Wenger, Josephine Passananti, Arjun Nitin Bhagoji, Yuanshun Yao, Haitao Zheng, and Ben Y Zhao. Backdoor attacks against deep learning systems in the physical world. In *CVPR*, 2021.
- [37] Baoyuan Wu, Hongrui Chen, Mingda Zhang, Zihao Zhu, Shaokui Wei, Danni Yuan, and Chao Shen. Backdoorbench: A comprehensive benchmark of backdoor learning. In *NeurIPS*, 2022.
- [38] Takuma Yagi, Karttikeya Mangalam, Ryo Yonetani, and Yoichi Sato. Future person localization in first-person videos. In *CVPR*, 2018.
- [39] Chaofei Yang, Qing Wu, Hai Li, and Yiran Chen. Generative poisoning attack method against neural networks. *arXiv preprint arXiv:1703.01340*, 2017.

- [40] Fangkai Yang and Christopher Peters. Appgan: Generative adversarial networks for generating robot approach behaviors into small groups of people. In *RO-MAN*, 2019.
- [41] Ye Yuan, Xinshuo Weng, Yanglan Ou, and Kris M Kitani. Agentformer: Agent-aware transformers for socio-temporal multi-agent forecasting. In *ICCV*, 2021.
- [42] Qingzhao Zhang, Shengtuo Hu, Jiachen Sun, Qi Alfred Chen, and Z Morley Mao. On adversarial robustness of trajectory prediction for autonomous vehicles. In *CVPR*, 2022.
- [43] Xinyang Zhang, Zheng Zhang, Shouling Ji, and Ting Wang. Trojaning language models for fun and profit. In *EuroS&P*, 2021.
- [44] Hang Zhao, Jiyang Gao, Tian Lan, Chen Sun, Ben Sapp, Balakrishnan Varadarajan, Yue Shen, Yi Shen, Yuning Chai, Cordelia Schmid, et al. Tnt: Target-driven trajectory prediction. In *CoRL*, 2021.
- [45] Zikang Zhou, Luyao Ye, Jianping Wang, Kui Wu, and Kejie Lu. Hivt: Hierarchical vector transformer for multi-agent motion prediction. In *CVPR*, 2022.