Zeroth-Order Optimization Attacks on Deep Reinforcement Learning-Based Lane Changing Algorithms for Autonomous Vehicles

Dayu Zhang[1], Nasser Lashgarian Azad[1], Sebastian Fischmeister[2] and Stefan Marksteiner[3]
[1]*Systems Design Engineering, University of Waterloo, 200 University Ave. W, Waterloo, Ontario, Canada*
[2]*Electrical and Computer Engineering, University of Waterloo, 200 University Ave. W, Waterloo, Ontario, Canada*
[3]*AVL List GmbH. Hans-List-Platz 1, 8020 Graz, Austria*

Keywords: Deep Reinforcement Learning, Adversarial Training, Zeroth-Order Optimization, Autonomous Vehicles.

Abstract: As Autonomous Vehicles (AVs) become prevalent, their reinforcement learning-based decision-making algorithms, especially those governing highway lane changes, are potentially vulnerable to adversarial attacks. This study investigates the vulnerability of Deep Q-Network (DQN) and Deep Deterministic Policy Gradient (DDPG) reinforcement learning algorithms to black-box attacks. We utilize zeroth-order optimization methods like ZO-SignSGD, allowing effective attacks without gradient information, revealing vulnerabilities in the existing systems. Our results demonstrate that these attacks can significantly degrade the performance of the AV, reducing their rewards by 60 percent and more. We also explore adversarial training as a defensive measure, which enhances the robustness of the DRL algorithms but at the expense of overall performance. Our findings underline the necessity of developing robust and secure reinforcement learning algorithms for AVs, urging further research into comprehensive defense strategies. The work is the first to apply zeroth-order optimization attacks on reinforcement learning in AVs, highlighting the imperative for balancing robustness and accuracy in AV algorithms.

1 INTRODUCTION

The escalating pace of Autonomous Vehicle (AV) development and deployment emphasizes the urgent need for secure and robust decision-making algorithms. However, these algorithms, often promised to be based on Deep Reinforcement Learning (DRL), present a potential vulnerability that adversarial attacks could exploit.

Since the discovery of adversarial examples in 2013, adversarial attacks and defenses have been well-studied in the field of deep learning (Szegedy et al., 2014). These attacks introduce minute perturbations, compelling machine learning algorithms to produce erroneous or attacker-desired predictions. Such perturbations are often imperceptible to both human observers and conventional detection techniques. Many attacks and defenses have been demonstrated on AVs, primarily targeting perception modules, such as cameras and lidars (Boloor et al., 2020; Cao et al., 2019). However, adversarial attacks and defenses still leave a significant gap in the field of DRL, especially in the application of DRL in AVs, where DRL shows its promise as the future of control. Notably, black-box attacks represent a significant threat due to their capacity to manipulate system output without the attacker having deep knowledge of the system's inner workings, in this case, the gradient information of the underlying model. As AVs increasingly share our roads, understanding the vulnerability of DRL-based lane-changing algorithms to such attacks becomes crucial for ensuring the safety, trust, and widespread acceptance of these emerging technologies.

In this study, we comprehensively examine the vulnerability of highway lane-changing algorithms — a critical and the most fundamental component of AV systems — to black-box attacks. Our focus is on the widely implemented DRL policies: Deep Q-Network (DQN) and Deep Deterministic Policy Gradient (DDPG), representing discrete and continuous action spaces, respectively (Mnih et al., 2013)(Lillicrap et al., 2019). Leveraging zeroth-order optimization methods, often applied in deep learning — such as ZO-SignSGD — we demonstrate their utility in executing black box attacks on DRL agents, uniquely, without the need for gradient information (Liu et al., 2019). Such an idea is plausible if the attacker ac-

Zhang, D., Azad, N., Fischmeister, S. and Marksteiner, S.
Zeroth-Order Optimization Attacks on Deep Reinforcement Learning-Based Lane Changing Algorithms for Autonomous Vehicles.
DOI: 10.5220/0012187700003543
In *Proceedings of the 20th International Conference on Informatics in Control, Automation and Robotics (ICINCO 2023) - Volume 1*, pages 665-673
ISBN: 978-989-758-670-5; ISSN: 2184-2809

cesses vehicle sensor values. However, to maximize the damage without getting detected, the attack must produce a perturbation small enough to avoid detection yet can still wreak havoc. Our experiments reveal that these techniques can effectively undermine the decision-making processes of AVs, highlighting a significant vulnerability in current systems. Further, we investigate the efficacy of adversarial training as a mitigation strategy within the context of DRL, providing insights into its potential to enhance the robustness of lane-changing algorithms against adversarial attacks. Our research underscores the importance of considering security in developing and deploying reinforcement learning algorithms in AV.

As a result, the contributions of this paper are as follows:

- The first paper to highlight and apply zeroth-order optimization attacks on DRL in general and the first use of such attacks in the context of AVs.
- A demonstration of the effect of targeted adversarial attacks and how they can force agents into a specific action leading to hazardous conditions and collisions. We show that such attacks converge in surprisingly short time with minimal perturbation.
- Results on hardening DRL through perturbation training providing guidance to future work against zeroth-order optimization attacks.

The remainder of the paper is structured as follows: Section 2 outlines the related work. Section 3 provides the problem definition and describes the environment. Section 4 explains the DRL policies used in this work. Section 5 highlights the attack model and the new zeroth order optimization attack. Section 6 outlines our approach and discusses the results. Finally, Section 7 draws conclusions from the work and describes future initiatives.

2 RELATED WORK

As Reinforcement Learning (RL) continues to prove its potency in complex decision-making tasks, there has been a surge in academic interest in exploring its intersection with Adversarial Machine Learning and vulnerabilities to such attacks.

2.1 Reinforcement Learning

RL is a machine learning algorithm where an agent learns to interact with an environment to maximize the reward. Given a state, the agent produces an action, and based on this action, the environment provides a corresponding reward. The agent then updates its policy based on the reward. The agent is trained by interacting with the environment for several episodes.

Along with the development of deep learning, DRL has been shown to be effective in several applications such as Atari games (Mnih et al., 2013), robotics (Kalashnikov et al., 2021; Chebotar et al., 2021), and AVs (Isele et al., 2018; Mnih et al., 2015). DQN, one of the first DRL algorithms, demonstrated its potential by outperforming humans in Atari games (Mnih et al., 2013). It employs a neural network to approximate the Q function, representing the expected reward for taking an action in a given state. Crucial to DQN's operation is experience replay, which de-correlates the training data and the target network, which stabilizes the training process. On the other hand, DDPG, an actor-critic algorithm, effectively manages continuous action spaces (Lillicrap et al., 2019). Like DQN, it employs a neural network to approximate the Q function and the policy. However, it distinguishes itself through the use of a replay buffer for de-correlating training data, the target network for training stabilization, and its proficiency in solving numerous continuous control tasks in the OpenAI Gym (Brockman et al., 2016).

2.2 Adversarial Machine Learning

Adversarial machine learning was first introduced by Szegedy et al. in 2013 (Szegedy et al., 2014), where it was used to craft specific adversarial examples. Adversarial examples are input data manipulated to cause a machine learning model to misclassify it. While the perturbations are usually indiscernible to the human eye, they lead the model to drastically incorrect outputs. The discovery of adversarial examples highlighted the vulnerability of machine learning models, even when they achieve high accuracy on test data. Since its discovery, attack and defense have been popular topics in machine learning. Notable attacks on deep neural networks include the Fast Gradient Sign Method (FGSM) (Goodfellow et al., 2015), the DeepFool (Moosavi-Dezfooli et al., 2016), and the Carlini and Wagner attack (Carlini and Wagner, 2017). Defenses include adversarial training (Carlini and Wagner, 2017), adversarial examples detection (Papernot et al., 2016), and adversarial robustness certification (Sinha et al., 2020). Though these methods have been only shown to be effective in machine learning, it is clear that the same techniques can be applied to the field of reinforcement learning. Small perturbations in the observation can lead to an undesired action, significantly affecting the agent's performance. In 2017, Huang et al. showed that the DQN agent is vulnerable to the same attack applied to neural networks, such as the FGSM attack (Huang et al., 2017). Similar defenses, such as adversarial training, are also shown to be effective in the field of reinforcement learning (Pattanaik et al., 2017). Lately, more research has been done to increase the robustness of reinforcement learning agents in the context of AV (He et al., 2023; Buddareddygari et al., 2022).

2.3 Black-Box Attacks

The exploration of adversarial attacks has, for the most part, been rooted in first-order optimization methods. These methods, while powerful, often necessitate the availability of gradient information, making them impractical for real-world scenarios where such information may not always be accessible. The quest for gradient-free alternatives predates the recent strides in deep learning and adversarial machine learning. Traditional methods such as COBYLA and various Bayesian optimization techniques have been investigated extensively. However, these methods have demonstrated scalability limitations in dealing with modern, complex models that exhibit an ever-increasing dimensionality (Powell, 1994; Shahriari et al., 2016).

Zeroth-order (ZO) optimization methods have emerged as a promising alternative, offering efficiency in computational resources while maintaining a competitive convergence rate (Liu et al., 2020). A surge of interest in recent years has led to the development of several ZO optimization techniques, including but not limited to Zeroth Order Stochastic Gradient Descent (ZOSGD), ZO-SignSGD, and ZO-ADMM (Liu et al., 2020). The appeal of these techniques lies in their ability to operate without explicit gradient information, thus bridging the gap between the theoretical world of optimization and the pragmatic constraints of real-world applications.

In this paper, we focus mainly on the ZO-SignSGD method. Unlike other methods that use exact estimated gradient values, ZO-SignSGD utilizes the sign of the gradient to update the model parameters. This feature provides both computational advantages and practical feasibility, allowing the perturbation to converge in a relatively small amount of iterations (Liu et al., 2019). We venture into an underexplored area by employing ZO-SignSGD as a tool to study adversarial attacks on DRL algorithms in AVs, potentially expanding the understanding and application of black-box attacks in real-world scenarios.

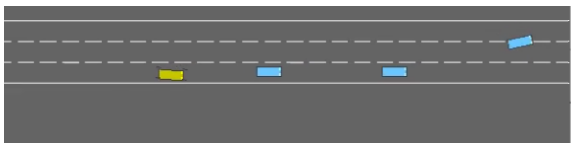
Figure 1: An example render of the highway lane changing environment.

3 PROBLEM

This paper focuses on applying, attacking, and defending a highway lane-changing deep Q network (DQN) agent and a Deep Deterministic Policy Gradient (DDPG) agent. In this paper, we assume that the vehicle has been compromised without detection, allowing the adversary to access and manipulate sensor data, thereby altering the states perceived by the DRL agent. Given the increasing adoption of detection algorithms for common attacks, adversarial machine-learning strategies are employed to maximize damage to DRL agents. These strategies introduce minimal perturbations to maintain stealth and reduce the likelihood of detection during the attack. It's worth noting that this paper does not delve into the specifics of the vehicle's attack surface or penetration methods. To assure the performance of the unattacked agent, the reinforcement learning algorithms are based on Stable Baselines 3, an online RL library written in Python (Raffin et al., 2021). The training environment is based on the 'highway-env' library to allow faster deployment, hyperparameter tuning, and debugging, as seen in Figure 1 (Leurent, 2018).

3.1 Environment

The environment is a lightweight highway lane-changing environment compatible with the OpenAI gym interface (Leurent, 2018). To expedite the training process, the environment is configured with less than 30 cars.

The environment's observation space tracks the vehicle's kinematics on the highway. That includes the position and velocity of the ego vehicle. It also records the relative position and velocity of other vehicles on the highway. The observation space is normalized relative to the ego vehicle. The position is normalized with the bound of $[-100, 100]$, and the velocity is normalized with the bound of $[0, 20]$.

During initialization, all vehicles, including the ego vehicle, are randomly positioned on the highway, ensuring a minimum separation between them. Vehicles, excluding the ego vehicle, adhere to a randomly initialized Intelligent Driver Model and the Minimizing Overall Braking Induced by Lane change (MO-

BIL) model (Leurent, 2018).

For DQN, the environment's action space is a high-level discrete action space with five actions. The actions are defined as:

- Action 1: change lane to the left
- Action 2: idle (do nothing)
- Action 3: change lane to the right
- Action 4: accelerate
- Action 5: decelerate

Simple proportional controllers control the lower-level actions such as specific heading, velocity, and acceleration when each action is chosen, so the reinforcement learning agent only needs to make a high-level decision on which action to take.

For DDPG, the action is a continuous action space for the kinematics of the ego vehicle with a dimension of 2. The first dimension is the acceleration of the ego vehicle, and the second dimension is the steering angle of the ego vehicle. Both actions are normalized to $[-1,1]$.

4 POLICIES

The reward function rewards the agent for staying in the right lane at a faster speed while penalizing the agent for collision. The reward function is defined as:

$$R(s,a) = RightLaneReward + 0.4 \cdot \frac{v - v_{min}}{v_{max} - v_{min}} + collision \tag{1}$$

The collision reward is set to -1. So that the agent will seek to move faster while avoiding a collision. RightLaneReward is set to 0.1 when the agent is traveling on the right-most lane.

4.1 DQN Policy

The agent is trained with the DQN algorithm (Mnih et al., 2013). Similar to Q learning, the underlying structure of the model is Markov Decision Process Equation 2.

$$Q^{new}(s_t,a_t) = Q(s_t,a_t) + \alpha(r_t + \gamma \cdot \max_a Q(s_{t+1},a) - Q(s_t,a_t)) \tag{2}$$

- $Q(s_t,a_t)$: Q value of the current state and action
- α: learning rate
- r_t: reward of the current state and action
- γ: discount factor
- s: state
- a: action

However, for deep Q learning, the Q function is approximated by a neural network. The neural network is trained with the DQN algorithm. This algorithm uses a replay buffer to store the experience of the agent. The replay buffer samples a batch of experiences to train the neural network. The DQN loss function is defined as:

$$L(\theta) = \mathbb{E}_{(s,a,r,s')\sim U(D)} \cdot \left[\left(r + \gamma \cdot \max_{a'} Q(s',a';\theta^-) - Q(s,a;\theta) \right)^2 \right] \tag{3}$$

- θ: the parameters of the neural network
- θ^-: the parameters of the target network
- D: the replay buffer

Like Q learning, the network loss is the reward plus the discounted maximum Q value of the next state minus the current Q value. But in this case, the gradient descent method minimizes the loss function. The network is trained with the Adam optimizer (Kingma and Ba, 2017). For a more straightforward implementation, Stable Baseline 3 is used to train the model (Raffin et al., 2021).

4.2 DDPG Policy

The agent is trained with the stable baseline library implementation of the DDPG algorithm (Lillicrap et al., 2019; Raffin et al., 2021). In contrast to DQN, which only deals with discrete action spaces, DDPG allows the handling of continuous action spaces, making it particularly suitable for AV, where actions are often continuous, like acceleration and steering angle. Another main difference is that DDPG combines the actor-critic approach with insights from Deep Q-Networks (DQN). The actor in this setup is responsible for determining the best action given the current state, while the critic evaluates the chosen action's quality. As seen in Equation 4, the actor updates in the direction that maximizes the Q value of the current state and action. On the other hand, the critic is updated based on the Temporal Difference (TD) error, which is the difference between the critic's current estimate of the Q-value and the improved estimate yielded by the latest action from the actor. This process, similar to Q-learning, involves the use of a learning rate to balance the weight between the old and new estimates:

$$\nabla_{\theta^\mu} J \approx \frac{\sum_i \nabla_a Q(s,a|\theta^Q)|s = s_i, a = \mu(s_i) \nabla \theta^\mu \mu(s|\theta^\mu)|_{s_i}}{N} \tag{4}$$

5 ZEROTH ORDER ATTACK

This work uses the zeroth order optimization as an adversarial attack.

5.1 Attacker Model

The attacker model defines the attacker's opportunity, intent, and capability to place the work in context. The attacker has the opportunity to influence the system during operation at the level of sensor outputs. The attacker's short-term goal is to disrupt the agent to the point that the agent will cause a collision on the road. The capabilities of the attacker consist of the following: (1) the attacker can influence a sensor value up to a 10% deviation of its actual value, (2) an attacker can influence sensor values immediately (i.e., from the start of the vehicle until a shutdown), continuously (i.e., no cool down periods), and indefinitely (i.e., for as long as the attacker wants), and (3) the attacker can influence any sensor, and all sensors at the same time (i.e., there no assumption that any single sensor in the set provides an actual value).

5.2 Zeroth Order SignSGD

The ZO-SignSGD method is a gradient-free (zeroth order) optimization method that uses the sign of the gradient to update the model (Liu et al., 2019).

The ZO-SignSGD method implemented is defined as:

Algorithm 1 shows the implementation of the ZO-SignSGD attack for lane changing. The input variables, such as learning rate, initial value, and number of iterations, are tweaked to ensure a fast convergence while minimizing the perturbation size. As a black box optimization algorithm, the first step is to estimate the gradient. A gradient of a function can be estimated by adding a small perturbation to the input data. For a high-dimension function such as the neural network used in the DRL, the gradient must be computed by summing all estimated gradients over perturbations of random directions. To achieve a fast convergence of the algorithm, similar to the Fast Gradient Sign Method, only the sign of the gradient is used. This also avoids the error introduced by the numerical value of the estimated gradient. The perturbation is then calculated by multiplying the sign of the gradient with the learning rate to minimize the objective function seen in Equation 8. This process is repeated until it reaches the maximum number of iterations. The perturbed observation is then used to get the action from the policy, thus, continuing into the next step.

```
Data: ZO-SignSGD
Input: learning rate $\{\delta_k\}$ , initial value $x_0$,
       and number of iterations: T
def GradEstimate(x, μ, q, d):
    for $k = 1, 2, ..., q$ do
        u = normalized(random number);
        $\hat{g}_k = \hat{g} + \frac{d(f(x+\mu u)-f(x))}{\mu} u$;
    end
def optimization(x):
    for k=1, 2, ..., T do
        $\hat{g}_k$ = GradEstimate($x_k$);
        $x_{k+1} = x_k - \delta_k \, sign(\hat{g}_k)$;
    end
def Main:
    for i in range of timesteps do
        while not done do
            action =
              model.predict(observation);
            env.step(action);
            perturbed_obs =
              optimization(observation);
            observation = perturbed_obs;
        end
    end
```

Algorithm 1: Implantation of ZO-SignSGD for Lane Keeping. Adversarial observation is calculated for each step.

Using this algorithm, the specific objective for this attack is crafted with two losses in mind. The first loss is the distance between the target action and the original action, which can be seen in Equation 5. The same is true for both DQN and DDPG. "a" is the constant that controls the weight of the loss.

$$\mathcal{L}1_{DQN} = a \cdot norm(Q(x+\delta, y_{target};\theta) - Q(x+\delta, y;\theta)) \tag{5}$$

$$\mathcal{L}1_{DDPG} = a \cdot norm(action(x+\delta;\theta) - action(x;\theta)) \tag{6}$$

The second loss is the distortion caused by the perturbation, as seen in Equation 7. This calculates the distance between the original observation and the perturbed observation.

$$\mathcal{L}2 = norm((perturbed\ obs - original\ obs)^2) \tag{7}$$

$$Objective : \min(\mathcal{L}1 + \mathcal{L}2) \tag{8}$$

When crafting the perturbation, both losses are added together to minimize both during the optimization. For a successful attack, both losses will converge and be minimized. Figure 2 shows an example of this convergence. Since Zeroth Order SignSGD is not a constrained optimization method, the perturbation is not guaranteed to be small. At the same time, since it is a gradient-free stochastic optimization method, the attack can fail to converge within the given number of iterations.

6 APPROACH & EVALUATION

6.1 Adversarial Training

As seen in many adversarial machine learning papers, adversarial training is a method to train a model to be robust to adversarial attacks (Carlini and Wagner, 2017; Pattanaik et al., 2017). A general Procedure can be seen in Algorithm 2.

```
Data: Adversarial-Training
for $i = 1, 2, ..., timestep$ do
    attack the observation $Q(obs, a, \theta)$;
    obs' = ZO SignSGD($Q(obs, a, \theta)$);
    a' = $Q(obs', a, \theta)$;
    new obs, reward = env(a',s);
    Train policy as per DQN or DDPG
      algorithm;
end
```

Algorithm 2: Adversarial training of the policy.

The algorithm is based on the methods discussed in (Pattanaik et al., 2017). Though it may appear simple, this algorithm has proven successful against the trained perturbation method for both deep learning and DRL models.

6.2 Initial Training

The DQN algorithm is first trained for 20,000 time steps with the default hyperparameter included in stable baseline 3. The model can learn the environment and achieve a mean reward of 310.58 per episode. For most of the episodes, The policy can navigate the highway for the entire episode without fail.

The DDPG model is trained for 120,000 time steps. A higher time step allows the actor and critic to converge on this lane-changing task. Due to the continuous action space control by the DDPG algorithm, the reward is not as stable as DQN. However, the model can still achieve a mean reward of 232.78 per episode. Note that DDPG is trained with noise added to the action space as part of the exploration strategy.

The maximum reward obtainable by the agents per episode would be 450, assuming it never crashes, is always on the right-most lane, and always travels at high speed. However, such theoretical maximum reward is unobtainable as the agent must slow down to change lanes and avoid crashes. At the same time, As long as other cars are in the right-most lane, the agent is unlikely to obtain the full reward for the step. Getting a higher reward also requires the agent to have a constant heading going forward. This has proven to be difficult to maintain for a DDPG agent where the policy has control of the steering. Therefore, both agents performed relatively well in this task.

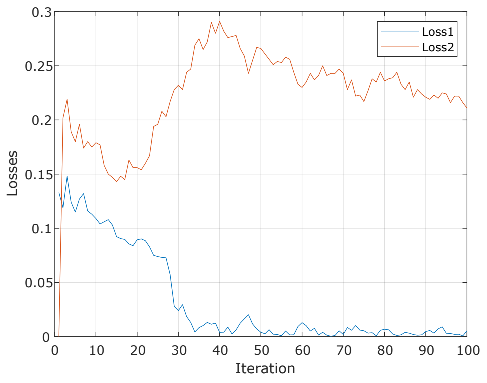

Figure 2: Loss convergence for successful perturbation for both Loss1 and Loss2.

6.3 Attacks

The idea of the attack is to mimic a real-world scenario where the attacker has access to the vehicle's sensors, enabling them to craft perturbations to the observation space. The perturbation is generated at every step with the consideration of perturbation sizes. The maximum iteration per step for perturbation crafting is set to 100. However, most successful perturbations are created within 50 iterations. An example of the loss for a successful perturbation is shown in Figure 2. The perturbation converges under 100 iterations. As the iterations go on, the distortion becomes the focus of the optimization program and, as a result, shrinks with iterations. The targeted action for the DDPG policy is set to be $[1, 0.5]$, meaning full throttle and turning right. The attack successfully causes the model to turn right most of the time, causing the mean reward per episode to plunge. For DQN, since the action space is high level, the targeted attack is chosen to be "accelerate" (Action 3) to prevent the ego vehicle from changing lanes at all.

The attack is unconstrained with the loss function defined in Equation 5. However, the size of the perturbation is directly correlated to the parameters for ZO SignSGD. The larger step each iteration of the gradi-

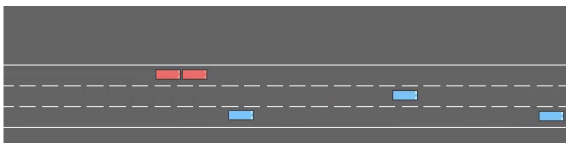

Figure 3: Result of the perturbation. The ego vehicle hits another car.

ent takes, the bigger the perturbations. Therefore, the parameters are carefully tuned to allow the attack to converge quickly while maintaining a reasonable perturbation size to enable a distortion to within 0.2, representing the 10% deviation from its original value, while allowing the attack to be carried out within 100 iterations. Small perturbations created by adversarial machine learning like this may help the attack avoid possible detection. Since ZO SignSGD can minimize both Loss1 and Loss2 as defined in Equation 5 and 7, as the iteration grows, distortion can be minimized if parameters are tuned in such a way. This trend already can already be seen in Figure 2.

6.4 Defenses

Employing the approach outlined in Algorithm 2, we further trained the DQN and DDPG policies that were initially compromised by the adversarial attack for an extra 5,000 and 10,000 time steps, respectively, this time incorporating adversarial observations into the learning process. The progression of the reward per episode during this adversarial training phase can be visualized in Figure 4 and Figure 5.

Despite the initial attack, both policies exhibited a marginal increase in the reward per episode following adversarial training, suggesting some degree of learned resilience against adversarial manipulation. However, it is noteworthy that this increase was rather insubstantial, particularly in the case of DQN. Moreover, despite maintaining identical training parameters, the mean reward per episode for both policies during adversarial training was lower than that achieved during the initial training phase.

This decline in performance is likely to be attributed to overfitting to the perturbed observations. The model's parameters have essentially learned to respond specifically to the adversarial patterns in the observations, thereby diminishing its performance under normal conditions. This is particularly evident in Figure 5, where the reward per episode shows a declining trend, a classic indicator of overfitting.

The trade-off between robustness and performance in adversarial settings is a well-documented challenge in machine learning literature. A seminal 2019 paper elucidated this issue by demonstrating worsened generalization performance of deep learning networks under adversarial training (Raghunathan et al., 2019). More recently, in 2022, potential explanations for this trade-off were proposed, such as the lower utility of robust features for generalization tasks or the insufficiency of datasets for adversarial training (Clarysse et al., 2022). This dilemma is manifested in our experiment, wherein the adversarially trained policies underperformed compared to their non-adversarially trained counterparts. This underscores the complexity of designing reinforcement learning policies that are both robust to adversarial attacks and proficient at their designated tasks.

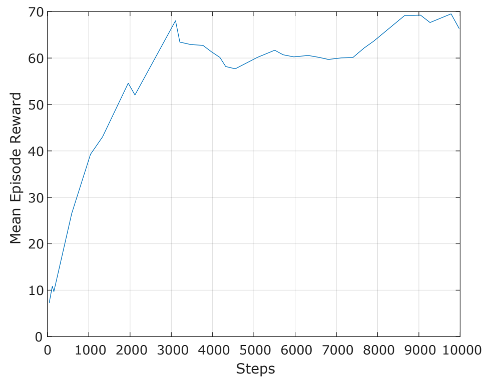

Figure 4: Reward/Episode for adversarial training of DDPG.

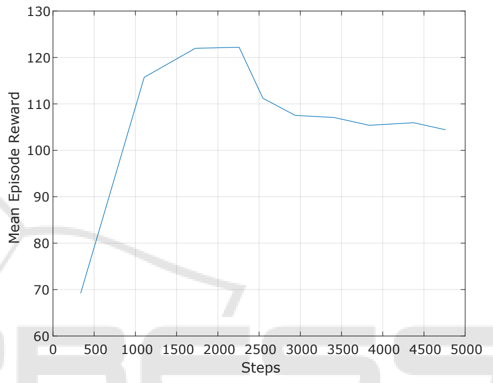

Figure 5: Reward/Episode for adversarial training of DQN.

6.5 Results

A table of mean rewards for each model is shown in Table 1. The highest obtainable reward per episode is 450, with one reward per step if the agent reaches high speed, stays in the right lane, and does not crash. The environment has completely random vehicle layouts every single time. Therefore the agent would not be able to obtain the total 450 rewards as it is often required to slow down, change lanes out of the right lane to avoid a collision or maintain the high-speed reward.

Table 1: Mean reward of each model with the maximum theoretical reward of 450.

	Policy			
	DQN		DDPG	
Scenarios	Reward	[% of theoretic max]	Reward	[% of theoretic max]
Initial Training	310.58	69.01%	232.78	51.73%
Under Attack	108.22	24.89 %	45.44	10.09 %
Adversarial Training	111.08	22.78 %	62.40	13.87 %
After Training (No Attack)	149.33	33.18 %	91.22	20.27 %

The reward is calculated by finding the mean of rewards for 100 episodes. The environment calculates the reward, as shown in Equation 1. For *Initial Training* and *After Training*, no attacks are performed on the observation. The rewards are collected to show the generalized performance of the model. For *Under Attack* and *Adversarial Training*, the attack is performed on the observation to force the policy into performing only one action, if the perturbation converges within the given number of iterations.

As seen in Table 1 in the scenario *Initial Training*, the agents performed relatively well, setting up a good baseline performance of the policies. However, both policies fail to perform as the attacks take place in *Under Attack*. Some marginal performance gain is seen after the policies undergo *Adversrial Training*. But overall generalized performance is suffering as seen in *After Training*.

7 CONCLUSION AND FUTURE WORK

In this work, we harnessed the ZO-SignSGD method to craft perturbations capable of triggering the failure of trained reinforcement learning models. Remarkably, these attacks were successfully carried out on both DQN and DDPG models by introducing perturbations to the observation space, even without access to the actual gradient information of the models. While the untouched models achieved high rewards — approximately 310 and 250, respectively — the targeted attacks significantly disrupted the performance of the ego vehicle, forcing it to follow the attacker's actions and plummeting the reward to near zero. This unique vulnerability underscores the vulnerabilities of reinforcement learning models to adversarial attacks even when the attacker lacks detailed model information.

In response to these successful attacks, we trained the models using these adversarial examples to enhance their robustness. Both models demonstrated an increased resilience, improving their rewards in the face of adversarial observations. However, it's important to note that adversarial training proved to be a time-intensive process, and the resulting models underperformed their original versions. This trade-off, where adversarial training dampens a model's generalization performance, mirrors findings observed in other machine learning applications (Clarysse et al., 2022).

Our adversarial attacks, while effective, are not yet optimized. Future work could draw inspiration from the adversarial attack strategies in the broader machine learning field, potentially leading to stronger and more efficient attacks. This could involve, for instance, targeting keyframes during the vehicles' operation. Moreover, testing the transferability of adversarial examples across different models could provide critical insights into the vulnerability of autonomous vehicles, particularly since deep reinforcement learning models often perform identical tasks. To prevent fast and catastrophic perturbations by attackers, it will be crucial to test these examples in real-world scenarios.

As demonstrated in this paper, adversarial training is not a panacea for these adversarial threats. It may cause an unexpected loss of rewards if the model adapts too much to the adversarial observation. Given the requirement for autonomous vehicles to function flawlessly under all circumstances. Further investigation into other defensive measures is imperative to build more robust and secure systems. These could include intrusion detection systems, model distillation, and model verification. Each of these could potentially contribute to a more comprehensive solution, mitigating the risks of adversarial attacks.

REFERENCES

Boloor, A., Garimella, K., He, X., Gill, C., Vorobeychik, Y., and Zhang, X. (2020). Attacking vision-based perception in end-to-end autonomous driving models. *Journal of systems architecture*, 110:101766–.

Brockman, G., Cheung, V., Pettersson, L., Schneider, J., Schulman, J., Tang, J., and Zaremba, W. (2016). OpenAI Gym. arXiv 1606.01540.

Buddareddygari, P., Zhang, T., Yang, Y., and Ren, Y. (2022). Targeted Attack on Deep RL-based Autonomous Driving with Learned Visual Patterns. In *2022 International Conference on Robotics and Automation (ICRA)*, pages 10571–10577.

Cao, Y., Xiao, C., Cyr, B., Zhou, Y., Park, W., Rampazzi, S., Chen, Q. A., Fu, K., and Mao, Z. M. (2019). Adversarial Sensor Attack on LiDAR-Based Perception in Autonomous Driving. In *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security*, CCS '19, page 2267–2281, New York, NY, USA. Association for Computing Machinery.

Carlini, N. and Wagner, D. (2017). Towards Evaluating the Robustness of Neural Networks. In *2017 IEEE Symposium on Security and Privacy (SP)*, pages 39–57.

Chebotar, Y., Hausman, K., Lu, Y., Xiao, T., Kalashnikov, D., Varley, J., Irpan, A., Eysenbach, B., Julian, R., Finn, C., and Levine, S. (2021). Actionable Models: Unsupervised Offline Reinforcement Learning of Robotic Skills. arXiv 2104.07749.

Clarysse, J., Hörrmann, J., and Yang, F. (2022). Why adversarial training can hurt robust accuracy. arXiv 2203.02006.

Goodfellow, I. J., Shlens, J., and Szegedy, C. (2015). Explaining and Harnessing Adversarial Examples. arXiv 1412.6572.

He, X., Yang, H., Hu, Z., and Lv, C. (2023). Robust Lane Change Decision Making for Autonomous Vehicles: An Observation Adversarial Reinforcement Learning Approach. *IEEE Transactions on Intelligent Vehicles*, 8(1):184–193.

Huang, S., Papernot, N., Goodfellow, I., Duan, Y., and Abbeel, P. (2017). Adversarial Attacks on Neural Network Policies. arXiv 1702.02284.

Isele, D., Nakhaei, A., and Fujimura, K. (2018). Safe Reinforcement Learning on Autonomous Vehicles. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1–6.

Kalashnikov, D., Varley, J., Chebotar, Y., Swanson, B., Jonschkowski, R., Finn, C., Levine, S., and Hausman, K. (2021). MT-Opt: Continuous Multi-Task Robotic Reinforcement Learning at Scale. arXiv 2104.08212.

Kingma, D. P. and Ba, J. (2017). Adam: A Method for Stochastic Optimization. arXiv 1412.6980.

Leurent, E. (2018). An Environment for Autonomous Driving Decision-Making. GitHub.

Lillicrap, T. P., Hunt, J. J., Pritzel, A., Heess, N., Erez, T., Tassa, Y., Silver, D., and Wierstra, D. (2019). Continuous control with deep reinforcement learning. arXiv 1509.02971.

Liu, S., Chen, P.-Y., Chen, X., and Hong, M. (2019). signSGD via Zeroth-Order Oracle. In *International Conference on Learning Representations*.

Liu, S., Chen, P.-Y., Kailkhura, B., Zhang, G., Hero III, A. O., and Varshney, P. K. (2020). A Primer on Zeroth-Order Optimization in Signal Processing and Machine Learning: Principals, Recent Advances, and Applications. *IEEE Signal Processing Magazine*, 37(5):43–54.

Mnih, V., Kavukcuoglu, K., Silver, D., Graves, A., Antonoglou, I., Wierstra, D., and Riedmiller, M. (2013). Playing Atari with Deep Reinforcement Learning. arXiv 1312.5602.

Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., Graves, A., Riedmiller, M., Fidjeland, A. K., Ostrovski, G., Petersen, S., Beattie, C., Sadik, A., Antonoglou, I., King, H., Kumaran, D., Wierstra, D., Legg, S., and Hassabis, D. (2015). Human-level control through deep reinforcement learning. *Nature*, 518(7540):529–533.

Moosavi-Dezfooli, S.-M., Fawzi, A., and Frossard, P. (2016). Deepfool: a simple and accurate method to fool deep neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2574–2582.

Papernot, N., McDaniel, P., Wu, X., Jha, S., and Swami, A. (2016). Distillation as a Defense to Adversarial Perturbations against Deep Neural Networks. arXiv 1511.04508.

Pattanaik, A., Tang, Z., Liu, S., Bommannan, G., and Chowdhary, G. (2017). Robust Deep Reinforcement Learning with Adversarial Attacks. arXiv 1712.03632.

Powell, M. J. D. (1994). *A Direct Search Optimization Method That Models the Objective and Constraint Functions by Linear Interpolation*, pages 51–67. Springer Netherlands, Dordrecht.

Raffin, A., Hill, A., Gleave, A., Kanervisto, A., Ernestus, M., and Dormann, N. (2021). Stable-Baselines3: Reliable Reinforcement Learning Implementations. *Journal of Machine Learning Research*, 22(268):1–8.

Raghunathan, A., Xie, S. M., Yang, F., Duchi, J. C., and Liang, P. (2019). Adversarial Training Can Hurt Generalization. arXiv 1906.06032.

Shahriari, B., Swersky, K., Wang, Z., Adams, R. P., and de Freitas, N. (2016). Taking the Human Out of the Loop: A Review of Bayesian Optimization. *Proceedings of the IEEE*, 104(1):148–175.

Sinha, A., Namkoong, H., Volpi, R., and Duchi, J. (2020). Certifying Some Distributional Robustness with Principled Adversarial Training. arXiv 1710.10571.

Szegedy, C., Zaremba, W., Sutskever, I., Bruna, J., Erhan, D., Goodfellow, I., and Fergus, R. (2014). Intriguing properties of neural networks. arXiv 1312.6199.