Fortifying the Digital Road: Countermeasures against Neural Network Adversaries in Autonomous Vehicles

Mohammad Javaad Akhtar[1], and Rishabh Jain[2]

***Abstract*—The escalating integration of deep neural networks in critical applications, notably in autonomous vehicles, underscores the urgency of fortifying them against adversarial attacks. This paper presents a novel approach to enhancing the robustness of convolutional neural networks (CNNs), with a particular focus on image-based systems used in self-driving cars. Unlike conventional methods, our research delves into white-box adversarial attacks, employing a unique strategy of training our neural network model on perturbed inputs. This includes a combination of adversarial mitigation techniques, such as randomization, image padding, and the addition of Gaussian noise, which has demonstrated considerable efficacy in mitigating adversarial threats. A unique aspect of our work is the integration of random Gaussian noise after each convolution layer in the neural network, a technique not widely explored in previous research on adversarial attacks on self-driving cars. This modification not only diversifies our defensive strategy but also showcases an unconventional approach to enhancing network robustness. Our primary focus has shifted towards the technical development and training of a specialized neural network for autonomous vehicles. This network is tailored to resist adversarial attacks, ensuring reliable performance even under attack scenarios. Significantly, our results indicate that our model maintains consistent steering control, avoiding the over-steering or under-steering issues observed in standard neural networks used in self-driving cars.**

***Index Terms*—Machine Learning, adversarial attack, data augmentation, autonomous vehicle**

I. Introduction

Modern convolutional neural networks return promising results in image-based classification and are widely used in surveillance and security applications [1], autonomous vehicle [2], and medical-based image classification [11]. A trillion-fold increase in computation power and efficient architecture has popularized the usage of these networks in daily tasks, from learning the basics of classification to space exploration. However, similar to any computational system, a severe security threat looms over these models. Recently, in the research community, adversaries can easily fool deep learning neural networks by perturbing benign samples without being detected by humans [5]. This phenomenon is called an adversarial sample and using this sample to misclassify the results of a neural network is considered an adversarial attack. One of the most dangerous examples of an adversarial attack is the altering of the output of deep learning models used in Autonomous vehicles to detect visual objects such as traffic signals. [7]

In this report, we will go over some common adversarial attacks that can be carried out on the networks and some associated mitigation policies. Most of the attacks mentioned in this report are related to data augmentation and adversarial training. We will divide this report into four parts. The first part of the report explains the types of possible adversarial attacks and the scenarios that can occur. The second part of the report focuses on the relevant mitigation policies. The third part demonstrates the significance of adversarial attacks

[1] MJ. Akhtar is with the Faculty of Computer Science, University of Western Ontario, London, Canada makhta55@uwo.ca

[2] R. Jain is with the Faculty of Computer Science, University of Western Ontario, London, Canada rjain57@uwo.ca

and challenges for autonomous vehicles. And lastly, we will go over techniques that are employed to create a robust model.

The effect of adversarial attacks seems vastly an unexplored territory [7]. Our research explores in adding robustness to NVIDIA's end-to-end regression model. The main contribution of this paper are as follows:

- We propose three adversarial attack mitigation techniques applied to an end-to-end regression model for autonomous vehicles, allowing the self-driving car to be more robust against attack.
- The robustness of the model against the attack is demonstrated using experiments concluded in Udacity simulation. The experiments illustrates that our model is able to reduce the under-steer and over-steer when attacked is carried out.
- To facilitate future experiment's and benchmark comparison, our neural network model is open-sourced on GitHub. As far as the authors are aware, this is the first open-source real-time attack-resistant model for others to contribute[3].

3

II. Types of Adversarial Attacks

We can classify an adversarial attack based on the adversarial goals, and adversarial knowledge. [7]. We can break them down as follows:

*1) **Adversarial goals**:* Adversarial goals define the intent behind the adversarial attack that is being carried out. We can divide the goal into two sections:

Targeted Attack: Targeted attacks are type of attacks where the attacker intentionally tries to misclassify such that it misguides the model to a particular class other than true class. In such cases, the attacker chooses a large perturbation e on an image x so that the model f misclassifies $x + e$ [6]. In such case, the attacker wants to maximize probability that $f(x + e) =$ incorrect class as shown in (Fig. 1).

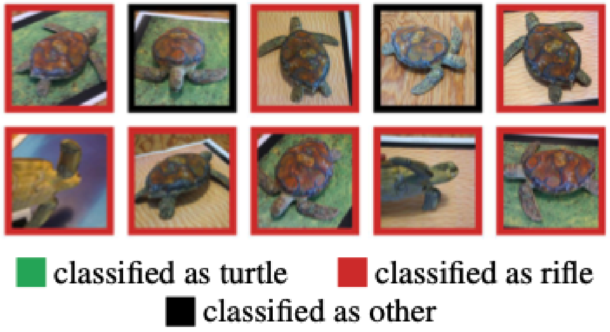


Fig. 1: An example of targetted attack

Non-Targetted Attack: A non-targetted attack (also known as untargetted attack) is an attack where attacker wants to simply misclassify to an incorrect class. In such case, the attacker wants to minimize probability $f(x + e) =$ correct class.

*2) **Adversarial Knowledge**:* Adversarial knowledge defines the information an attacker has regarding their target model. They can be categorized into three major categories: white-box attacks, black-box attacks, and gray-box attacks [5].

White-box attack: In a white-box attack, the attacker is completely aware of the architecture of the network, its weight, and biases, essentially full knowledge of their target model.

Black-box attack: In a black-box attack, the attacker is unaware of the internal architecture of the neural network, nor are they aware of the weight and biases attached to those networks. In this case, the attacker resorts to creating adversarial samples to use on the network.

Gray-box attack: Lastly, in the gray-box attack, the user has limited knowledge of the network and its parameters, which can be utilized to create targeted adversarial

[3] GitHub: https://github.com/MJavaadAkhtar/Adversarial-training

samples.

III. Adversarial Attacks Algorithms and Techniques

In this part of the report, we will go over some adversarial attack algorithms and methods. For the most part, these attacks are applied to image classification neural networks. However, they still can be applied to other type of neural networks. Due to nature and number of adversarial attacks, this report will only go through the major relevant adversarial attack scenarios. The purpose of this paper is not to provide an outline on all possible adversarial attacks, instead to give the reader a basic understanding of the threat an adversarial attacks possesses and how can they be carried out.

A. *FGSM (Fast Gradient Sign Method*

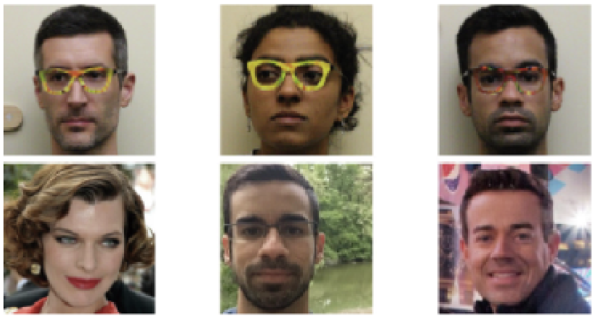
FGSM is a one-step attack algorithm, first purposed in Goodfellow et al [6], where we generate an adversarial sample by adding a one-step update along the direction of loss of gradient with respect to the input. In this case, the adversarial sample for the attack is where the attacker tries to misguide the model to predict a wrong target class (misclassification). we can generate an adversarial sample using FGSM as follows:

$$x' = x + \epsilon \cdot sign[\delta_x J(\theta, x, y)]$$

where $J(\theta, x, y)$ is gradient of the loss, and ϵ is magnitude of the perturbation. can see that in the Fig 2.:

B. *Adversarial Patch*

Adversarial Patch takes place when an element of a sample (pixels in our case) are perturbed [5]. **More precisely, these attacks take place when the original image is more obstructed than the one the model is trained on.** The attacker tends to select few neurons that significantly impact the models' output. Then the pixel value in those regions are modified with some adversarial patches. This image is then used to carry out the adversarial patch attack on the targeted model. The image below shows how wearing eyeglasses can deceive an image-based facial recognition system in the first row, as compared to the second row.



C. *GAN-Based Attack*

Attackers use adversarial samples that are generated using a Generative Adversarial Network (GAN) to fool the model. The generated samples are designed to exploit vulnerabilities in the way the model makes decisions to try and get it to misclassify data.

D. *Universal Adversarial Perturbation (UAP)*

Universal adversarial perturbation is a perturbation added such that it fools most of the samples chosen from input distribution. Essentially we would like to add minimum constant perturbation into our input sample until the sample starts fooling the model. This means the cross-sample transferability is also maintained across models, for example the universal perturbation crafted on a VGG model (Visual Geometry Group) can also fool other models with more than 50% fooling rate [5]. This is shown in Fig.3

IV. Previous Work of attacks on autonomous vehicle

Adversarial attacks, characterized as meticulously crafted patterns designed to disrupt neural network outputs, have emerged as a significant concern in the realm of machine learning. Despite their growing relevance, research focused on evaluating the robustness of Deep Neural Networks (DNNs) for trajectory prediction in the face of these attacks remains notably scarce [4]. This scarcity is underscored by the increasing volume of academic literature on end-to-end deep learning models,

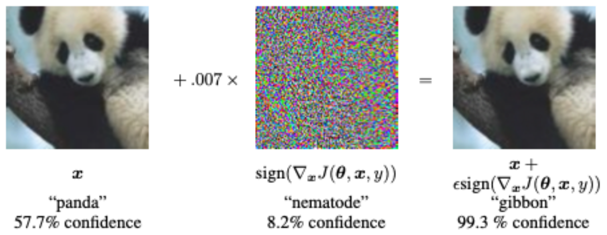

$\boldsymbol{x}$
"panda"
57.7% confidence
$+ .007 \times$
$\text{sign}(\nabla_{\boldsymbol{x}} J(\boldsymbol{\theta}, \boldsymbol{x}, y))$
"nematode"
8.2% confidence
$=$
$\boldsymbol{x} +$
$\epsilon\text{sign}(\nabla_{\boldsymbol{x}} J(\boldsymbol{\theta}, \boldsymbol{x}, y))$
"gibbon"
99.3 % confidence

Fig. 2: Misclassifying the image due to FGSM adversarial sample

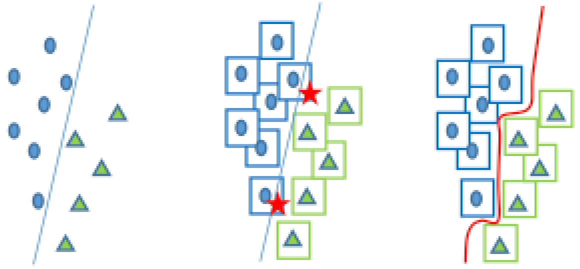


Fig. 3: Representation of decision boundary - Left: Normal image classifier, Middle: stars indicating the adversarial example generated by UAP, Right: Decision boundary of classifier also trained on adversarial sample

where the real-world safety against adversarial attack of these systems is still a subject of ongoing debate [2].

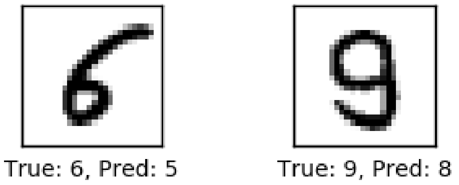
In 2017, a pivotal study led by T. B. Brown [8] demonstrated the potential of a localized adversarial patch in the real world to significantly impair neural network predictions, even while occupying a minimal portion of the image. This research marked a critical step in understanding adversarial impacts in practical settings. Subsequently, in 2020, two seminal papers further advanced this understanding. Z. Kong [9] revealed how an advertising sign could serve as an effective adversarial patch, influencing the steering module of an autonomous vehicle. Simultaneously, H. Salman [10] introduced the concept of unadversarial examples, extending beyond 2D objects to encompass 3D forms. These examples proved robust across varying lighting conditions, orientations, and camera views, expanding the scope of adversarial research. Notably, attacks targeting the self-localization capabilities of neural networks had not been extensively explored [3]. A novel approach was introduced where a backward relative position shift was utilized, leading the neural network to predict a location trailing the vehicle's actual position relative to its driving direction [3].

In the context of autonomous systems, DNNs find their application across a spectrum of functions, including feedback controllers and planners [9], perception modules [10], and trajectory prediction [4]. Despite the impressive experimental performance of DNNs, their inherent brittleness has led to unreliable system behaviors and public failures, impeding their widespread adoption in safety-critical applications [4]. This vulnerability is further highlighted by recent adversarial attack algorithms that minimally manipulate inputs to DNN models, inducing incorrect outputs to the advantage of an adversary [4].

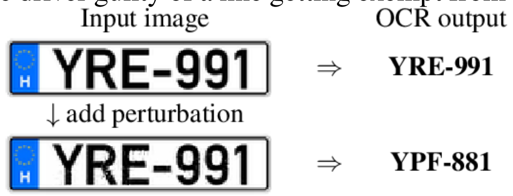
However, it is important to note that much of the prior research on adversarial attacks has predominantly concentrated on attacking classification models [2]. In these models, a successful attack alters the output away from the correct label. For instance, in a digital handwritten digit classification task, an attacker might manipulate the system to misclassify the digit '3' as '7'. To accurately assess the impact of an adversarial attack on a regression model, such as those used in autonomous vehicles, it is imperative to quantify the extent of deviation caused by these attacks [2].

V. Adversarial Attacks and Autonomous Vehicles

Consider a software tool that scans and interprets the text characters found in a document or on other objects.

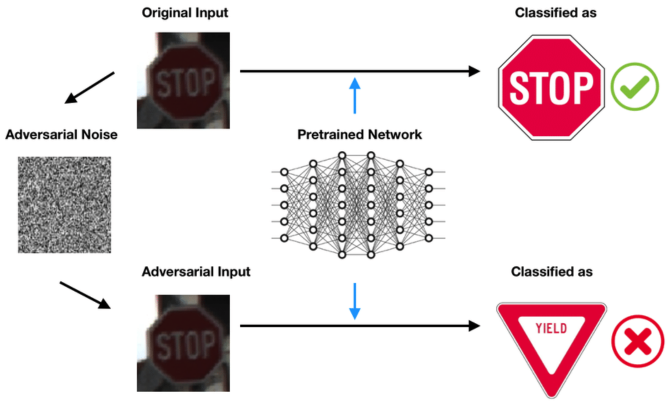


From the image above, we can observe that characters are misclassified due to noise during the OCR scan. Optical character recognition software is commonly used in the workplace to digitize handwritten text, but can also be applied on the road. There are toll road systems that utilize OCR to track licence plates of drivers that are currently using the pay-per use roads. Similarly, some parking enforcement agencies use cameras in their parking lots to evaluate the validity of driver's parking permits and process the characters from the license using software. Let's take a look at an example of what can happen when license plates are incorrectly scanned. Incorrect interpretation of the characters on the license plate can lead to the incorrect driver getting a ticket or the driver guilty of a fine getting exempt from the penalty.

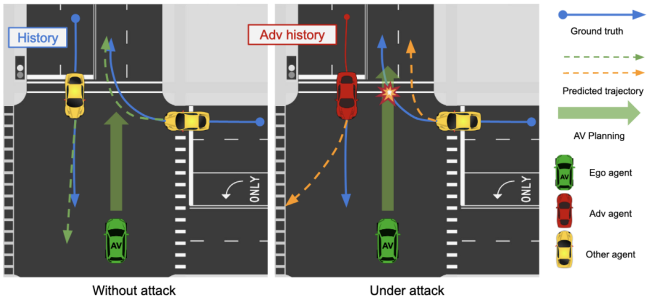


In the world of autonomous robotics, one may consider the future of transportation is Autonomous vehicles [12]. Autonomous systems like self-driving cars and drones are already in use. These types of systems rely on multiple sensors like Lidar (Light Detection and Ranging), GPS (Global Positioning System), and cameras for observing their environment [12]. However, for high-density cities, GPS performance may show degradation, we can use image-based localization for self-positioning. This means we need to utilize deep learning to achieve our tasks.

In the realm of safety-critical robotics applications, one may argue that safety in an autonomous vehicle is one of the most challenging tasks [2]. In recent years, the discussion around autonomous driving is steadily increasing, but their safety in real-world environment is still unclear [2]. Adversarial attacks can be crafted precisely to disrupt the standard functionality of a neural network, which raises the question of vulnerability in the neural network. In most cases, these attack types will be targeted towards image-based classification models in autonomous vehicles [12]. For example, in a situation where an autonomous vehicle is on the road constantly scanning and processing collected images, consider the safety concern when a stop sign is incorrectly interpreted. Due to an adversarial attack, a stop sign can be interpreted as a yield sign.



This can apply to other significant details that vehicles need to process, including speed limit signs, traffic lights, other vehicles. In the example below, the vehicle under adversarial attack miscalculates the the distance between the other vehicles and makes an incorrect judgement when making the turn.



The exposition of such vulnerability in deep learning will help us understand the safety precautions we can take when we train our neural network. A successful attack on our autonomous vehicle means classification models used will deviate the output from the true label. Moving forward, in this report, we will only discuss three types of attacks being carried out on the autonomous vehicle deep learning models.

A. *Preliminary and Formulation*

We need to define a few terminologies that we will be using moving forward.

1) Navigation Module: The navigation module is used to dictate a general path that the system should use in order to transport from one point or another, and whether to turn or not at each intersection (junctions) [12]. A navigational system can control an autonomous system that physically determines whether the car should take a turn. Usually, these navigation modules can be trained using an offline image-based convolutions neural network to determine whether taking a turn is feasible or not.

2) End-to-End Driving Systems: End-to-End driving system is more broad idea of navigational systems. End-to-end driving systems treat all the driving and control modules as singular systems that map the sensor input directly into the steering [2]. These types of systems are basically created using imitation leanings or reinforcement learning's [2].
We can discuss some mathematical notation for a better understanding of attacks being carried out. Let's define it as follows:

$$y = f(\theta, x)$$
$$y' = f(\theta, x')$$

where y is input to the driving system (most specifically the steering system), $f(\theta, x)$ is the model that maps input x to the steering, and θ is the parameters. Similarly, y' is the adversarial input, with x' being the adversarial image.

Let's dive into the type of attacks we can carry out on autonomous vehicles.

B. *Image-specific Attack*

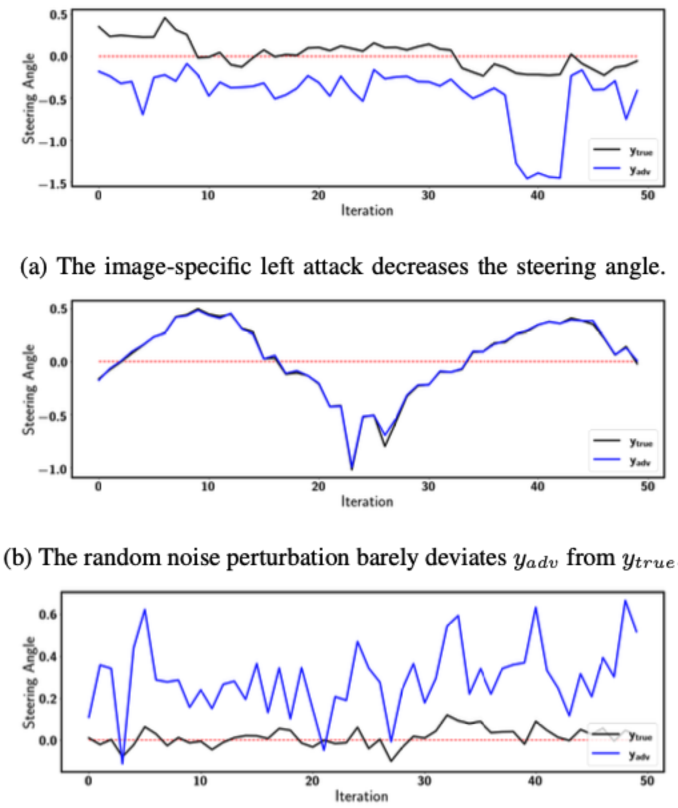
As discussed earlier, one of the first image-specific adversarial attacks that were carried out against the classifier was FGSM [6]. Instead of minimizing the training loss, Goodfellow et al. maximized the training loss and used the gradient of the training loss to generate perturbations. FGSM can be utilized to create a perturbation to fool the steering systems of end-end systems. An attacker might wish to attack the vehicle to turn it to the right side when the vehicle should not. These types of attacks can be considered a white box attack or a black box attack, depending on the attacker's knowledge regarding the end-to-end system and its parameters. A generic algorithm for image-specific attack can be created as follows [12]

```
Input: The regression model f(theta, x), the
    input images {xt} where xt is the image at
     time step t, P is perturbation function.
Parameters: The strength of the attack    .
Output: Image-specific perturbation    .

def ImageSpecificAttack(t, f, theta, x, P)
for each time step t do
    Inference: y = f(theta,x)
    Perturbation: P
end for
```

where perturbation function P is $n = \epsilon sign[\delta_x(J(y))]$. As suggested in Yunas et al. [2], adding random noise is not a useful technique for an adversarial attack on the system, as it has little to no effect. The paper describes that it applies three attacks, one with random noise perturbation, one with an image-specific attack left attack to decrease the steering angle, and one with an image-specific right attack that increases the steering angle. This can be seen in the graph [Fig 4.]

C. *Adversarial Patch on navigation module*

The adversarial patch is an attack where the element of a sample is perturbed. In this specific case, we will look



(a) The image-specific left attack decreases the steering angle.

(b) The random noise perturbation barely deviates y_{adv} from y_{true}.

(c) The image-specific right attack increases the steering angle.

Fig. 4: Random noise does not seem to affect the steering conditions much as seen in the graph

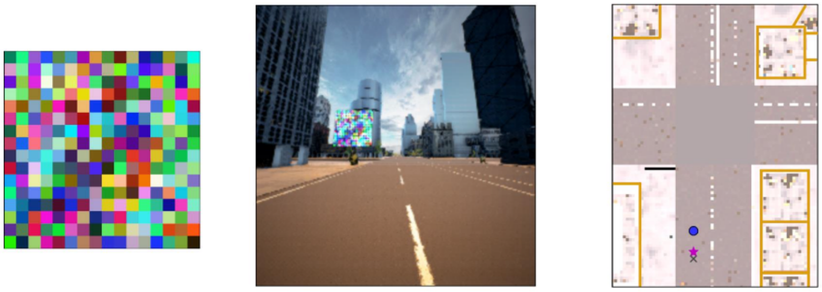
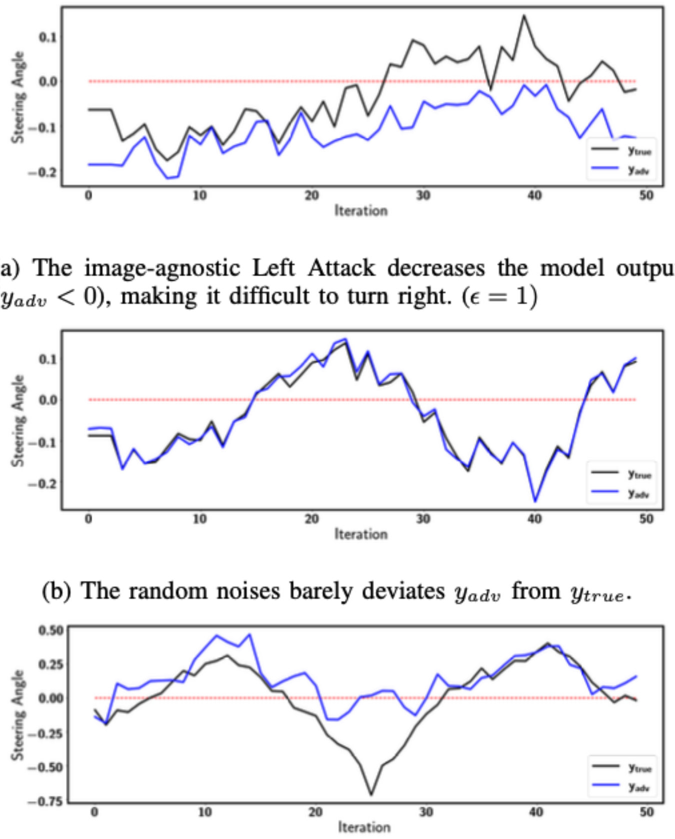


Fig. 5: (a) The adversarial patch, (b) the street image the vehicle sees with the adversarial patch, (c) the true (magenta star) and predicted position (blue dot) on the city map. [12]

into pixels of an image being perturbed to fool the CNN system. We will focus more on the navigation system and the effect image perturbation has on it.

The navigation module is responsible for providing directions to the driving module regarding the appropriate time to start turning before they reach the intersection. However, with sufficient perturbation, we can fool the navigation system into relinquishing that command to the driving module considerably late when its too late to take a turn in the intersection [Fig 5]. In the above image, we can observe due to perturbation; the navigation system was late to command the drive system to turn in a safe manner [12].



(a) The image-agnostic Left Attack decreases the model output ($y_{adv} < 0$), making it difficult to turn right. ($\epsilon = 1$)

(b) The random noises barely deviates y_{adv} from y_{true}.

(c) The image-agnostic Right Attack increases the model output ($y_{adv} > 0$), making it difficult to turn left. ($\epsilon = 1$)

Fig. 6: Image agnostic attack [12]

D. *Image-agnostic attack on End-to-end systems*

Autonomous vehicles are sensitive modules, even the smallest amount of deviation or defect may cause a traffic accident. A slight deviation can be a minute deviation in steering angle such that the vehicle fails to turn correctly. In this attack, we will talk about a white-box attack that generates universal adversarial perturbation (UAP), which can be used to attack all input images at different time steps [12]. This attack focuses on changing the steering direction by constantly adding perturbation to the sample input until the system recognizes deviates from the original turn. In this case, we steer more aggressively than the intended steering due to the deviation caused by this attack. In general, the image-agnostic attack is least effective than the two mentioned above [2]. We can see the difficulty in steering caused by this attack in the image [Fig 6.]

E. *Attack Pipeline*

In this section, we delve into the intricate details of our adversarial attack pipeline, designed to rigorously test the resilience of neural networks in autonomous vehicles. The foundation of our attack methodology is based on a sophisticated white-box framework, where we possess complete knowledge of the network architecture and parameters. Initially, the attack commences with the identification of vulnerable nodes within the network, focusing on those most influential in decision-making for vehicle control. Subsequently, we employ gradient-based techniques to generate perturbations tailored to these critical nodes. These perturbations are meticulously crafted to be subtle yet effective, ensuring they remain imperceptible to human observers but potent enough to mislead the neural network.

Our attack pipeline is further refined through an iterative process, where perturbations are progressively adjusted based on the network's response, optimizing their effectiveness in real-time scenarios. This iterative refinement is crucial for simulating realistic conditions where an autonomous vehicle's neural network would be subjected to dynamic and adaptive adversarial inputs. The culmination of this process is the deployment of these perturbations in simulated environments, mirroring real-world scenarios faced by autonomous vehicles. This approach allows us to evaluate the network's performance under attack, particularly focusing on critical functionalities like steering and obstacle detection. Through this comprehensive and methodical pipeline, our study aims to not only expose potential vulnerabilities in autonomous vehicle neural networks but also provide insights into enhancing their adversarial robustness. Our attack pipeline is all interlinked using web sockets to transfer information from simulator to the model and back *Fig 7*.

VI. Adversarial training on Autonomous vehicle

In the realm of enhancing the resilience of neural networks for autonomous vehicles against adversarial threats, adversarial training emerges as a pivotal methodology. This section comprehensively discusses the adversarial training process, underscoring its significance in fortifying neural networks against sophisticated attacks. The essence of adversarial training lies in its ability to expose the neural network to a wide array of adversarial examples during the training phase. By doing so, the network learns to recognize and counteract these perturbations, thereby improving its robustness in real-world scenarios where such attacks are prevalent.

Our adversarial training process is meticulously structured into three critical phases: Data Gathering, Preprocessing, and Model Training. In the Data Gathering phase, we accumulate a diverse dataset that not only encompasses typical driving scenarios but also integrates adversarially perturbed inputs. This ensures that the model is exposed to both standard and attack scenarios. The pre-processing phase involves the strategic manipulation of this data to enhance its relevance and effectiveness in training, including techniques such as normalization and augmentation. Finally, in the Model Training phase, we employ a modified neural network architecture, specifically designed to withstand the intricacies of adversarial inputs. This phase is characterized by rigorous training regimens and iterative optimization, aiming to achieve a balance between accuracy and adversarial resilience.

The integration of adversarial training in the development of neural networks for autonomous vehicles is not merely a defensive measure, but a proactive step towards ensuring their reliability and safety in an increasingly digitized and interconnected world. The subsequent subsections will delve into the specifics of each phase, elucidating the methodologies and innovations employed in our approach to adversarial training.

A. Data Gathering

In the crucial phase of data gathering for our adversarial training framework, we employed a two-pronged approach to dataset accumulation. Firstly, we concentrated on creating an in-house dataset utilizing the advanced functionalities of the Udemy driving simulator. This platform offered the unique capability to record diverse driving sessions, enabling us to tailor a dataset specifically designed for autonomous driving research. The in-house dataset was meticulously curated to introduce a high degree of randomization, essential for training a robust model. By manually controlling the simulator, we were able to capture a wide range of driving scenarios, from sharp turns to gradual maneuvers, thereby encompassing the variability and unpredictability of real-world driving conditions. This deliberate variation in data acquisition is pivotal, as it ensures that our model is exposed to and learns from a spectrum of driving behaviors and scenarios, significantly enhancing its adaptability and responsiveness.

In addition to our proprietary dataset, we integrated an external dataset referenced in a prominent research

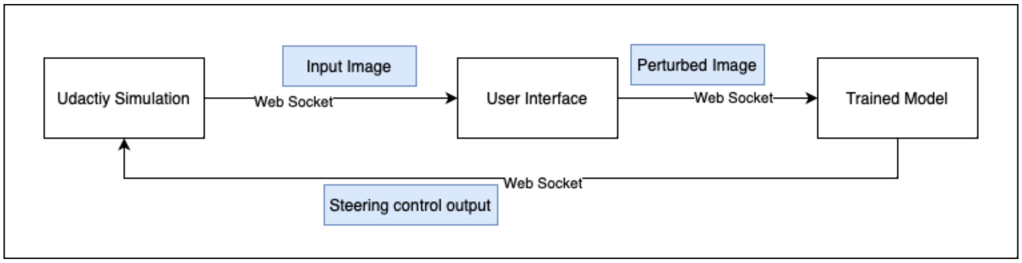


Fig. 7: Attack Pipeline: Image from the simulation is fed into the model using a websocket. In the middle, we intercept the image before it reaches the model, add random noise to it and feed it to the model. The model determine course of action for the self-driving car

paper. This strategic amalgamation was executed with the objective of augmenting the diversity and comprehensiveness of our training data. By merging our in-house dataset with externally sourced data, we were able to create a more enriched and varied dataset. This integration not only broadens the scope of our model's exposure to different driving environments and conditions but also fortifies its capability to generalize across a wider array of adversarial scenarios. The combined dataset thus serves as a robust foundation for training a neural network that is resilient and adaptive, qualities indispensable for autonomous vehicles operating in dynamic and often unpredictable real-world settings.

B. Pre-processing

In our quest to fortify neural networks against adversarial attacks, particularly in the domain of autonomous vehicle technology, the pre-processing of input data stands as a vital line of defense. Prior to training our model, we implement strategic adversarial mitigation techniques to enhance the network's resilience. Central to these techniques are random resizing and random padding, both of which introduce an element of unpredictability that significantly impairs the effectiveness of adversarial attacks. Random resizing involves altering the dimensions of the input images in a non-deterministic manner before they are fed into the neural network. This approach disrupts the precise alignment that many adversarial attacks rely upon, thereby reducing their potency. Complementing this, we employ random padding, which entails the addition of zero-value pixels around the borders of the input images. This padding not only alters the spatial configuration of the input data but also serves as an additional layer of complexity for potential adversaries to circumvent.

These pre-processing techniques, rooted in the principle of randomization, have demonstrated substantial efficacy, particularly against black-box adversarial attacks, where attackers have limited knowledge of the target model. As illustrated in Figure 15, the implementation of these randomization techniques within the neural network model serves to obscure and distort the attack vectors, thereby preserving the integrity of the model's predictions. This preemptive adaptation of the input data ensures that our neural network is not only trained on a diverse dataset but is also inherently equipped to handle and neutralize adversarial perturbations, a crucial capability for ensuring the safety and reliability of autonomous driving systems in real-world scenarios.

We implemented padding and resizing as follows:

```
Input: The input images {xt} where xt is the
    image at time step t, P is perturbation
    function.
Perimeter: The input image that needs to be
    modified.
Output: Image-specific perturbation   .

def random_resize(image, min_size, max_size):
    # Resize the image to the new size
    resized_image = Resize(image, min_size,
        max_size)
    return resized_image
```

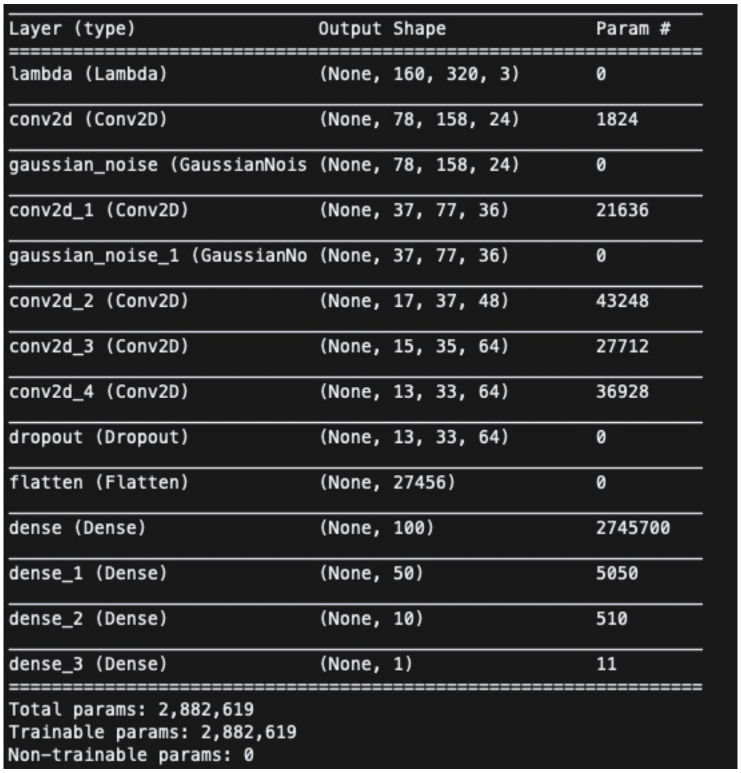
```

def random_padding(image, theta):
    # Randomly determine the padding size
    padding_size = random.randint(0,
        max_padding)
    # Apply padding to the image
    padded_image = Padding(theta, image)
    return padded_image
```

C. Model and training

A defense method against adversarial samples by generating adversarial samples in the training stage [5]. In the development of our neural network model, a critical innovation was the integration of random Gaussian noise after the first two convolutional layers, a technique pivotal in enhancing the model's robustness against adversarial attacks. This strategic insertion of noise acts as a dynamic disruptor to potential adversarial perturbations, effectively increasing the difficulty for an attacker to precisely calculate the necessary modifications to compromise the network. By introducing this stochastic element, our model gains an additional layer of defense, making it significantly more resilient to sophisticated adversarial strategies.

Our model, with its 2 million trainable parameters, stands as a testament to the intricate and advanced nature of its architecture. The addition of Gaussian noise is not merely a supplementary feature; it fundamentally transforms the model's behavior in response to adversarial inputs. This modification elevates the architecture from a standard convolutional neural network to a more complex and secure framework, tailored to withstand the unique challenges presented in the realm of autonomous vehicles. The incorporation of Gaussian noise within the early convolutional layers ensures that the subsequent layers and the final output are derived from an input that has already been 'hardened' against adversarial manipulation, thereby imbuing the entire network with a heightened level of security.

The novelty of our architecture lies not only in its size and complexity but also in its strategic design, specifically aimed at countering the evolving landscape of neural network attacks. This approach to model training and design represents a significant step forward in the field of adversarial machine learning, particularly in applications where reliability and security are paramount, such as in autonomous vehicle navigation and decision-making systems.



```
Layer (type)                 Output Shape              Param #
================================================================
lambda (Lambda)              (None, 160, 320, 3)       0
________________________________________________________________
conv2d (Conv2D)              (None, 78, 158, 24)       1824
________________________________________________________________
gaussian_noise (GaussianNois (None, 78, 158, 24)       0
________________________________________________________________
conv2d_1 (Conv2D)            (None, 37, 77, 36)        21636
________________________________________________________________
gaussian_noise_1 (GaussianNo (None, 37, 77, 36)        0
________________________________________________________________
conv2d_2 (Conv2D)            (None, 17, 37, 48)        43248
________________________________________________________________
conv2d_3 (Conv2D)            (None, 15, 35, 64)        27712
________________________________________________________________
conv2d_4 (Conv2D)            (None, 13, 33, 64)        36928
________________________________________________________________
dropout (Dropout)            (None, 13, 33, 64)        0
________________________________________________________________
flatten (Flatten)            (None, 27456)             0
________________________________________________________________
dense (Dense)                (None, 100)               2745700
________________________________________________________________
dense_1 (Dense)              (None, 50)                5050
________________________________________________________________
dense_2 (Dense)              (None, 10)                510
________________________________________________________________
dense_3 (Dense)              (None, 1)                 11
================================================================
Total params: 2,882,619
Trainable params: 2,882,619
Non-trainable params: 0
```

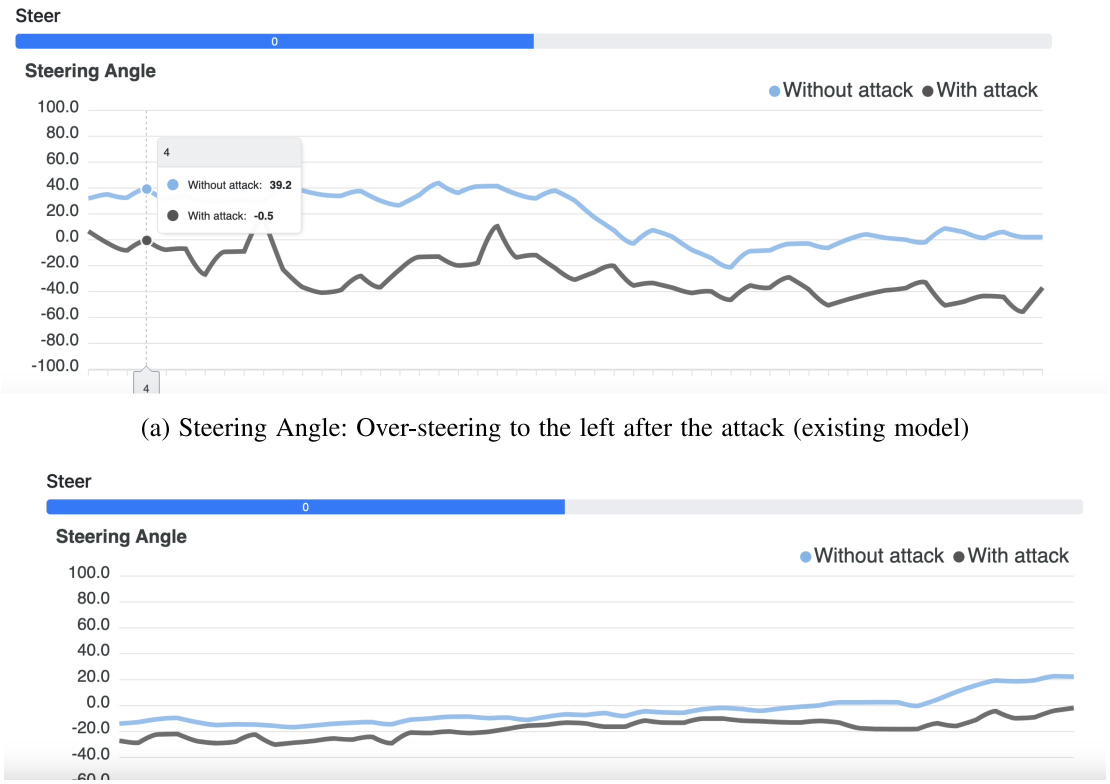
Fig. 8: Model Architecture

VII. Results

Our results compellingly illustrate the heightened robustness of the adversarially trained neural network model compared to a model lacking such training. This is evidenced through a series of tests and evaluations, prominently reflected in two key graphical analyses: the steering response under adversarial conditions and the comparison of true versus predicted values on perturbed inputs.

A. Steering Response Analysis

In the first analysis, we observed the steering behaviors of both models when subjected to adversarial attacks. The graph clearly demonstrates that the model not trained



(a) Steering Angle: Over-steering to the left after the attack (existing model)

(b) Steering Angle: Over-steering to the left after the attack (our model)

Fig. 9: Steering Angle information collected over span of 60 second of attack

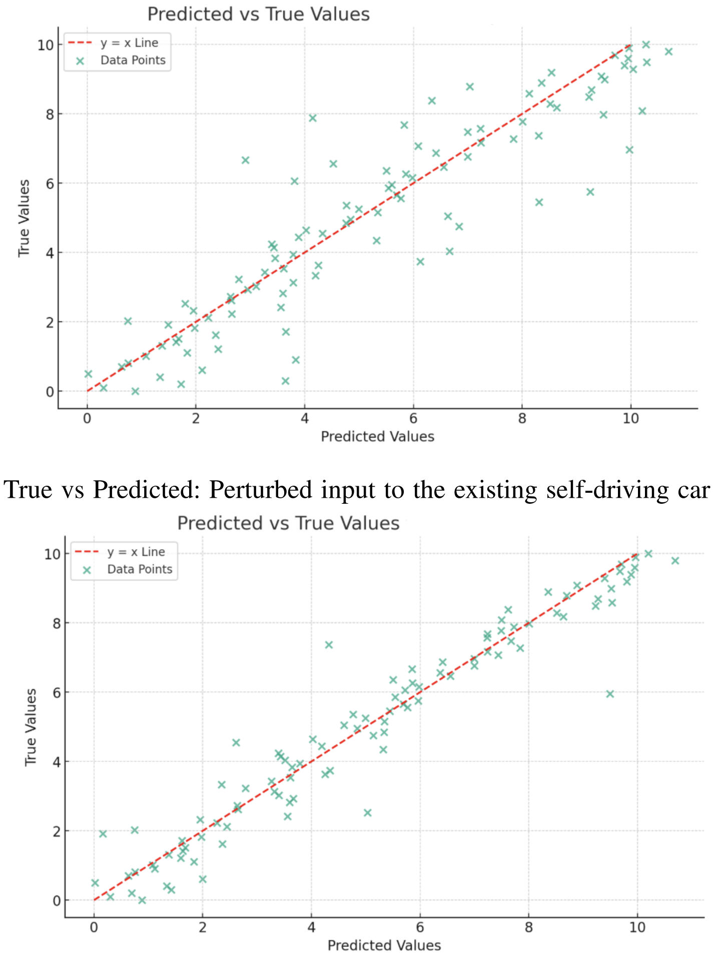
against adversarial attacks exhibited a tendency to over-steer significantly more than its adversarially trained counterpart. This over-steering is a critical indicator of susceptibility to adversarial manipulation, reflecting a lack of resilience in handling deceptive inputs designed to misguide the vehicle's trajectory. In stark contrast, the model trained with adversarial resistance maintained a more stable and accurate steering response, adhering closely to the appropriate steering commands even under attack. This finding not only validates the efficacy of adversarial training in enhancing model resilience but also highlights its indispensable role in ensuring the safety and reliability of autonomous vehicle navigation systems. This change is shown in Fig. 9.

B. *True vs Predicted Values Analysis*

The second analysis delved into the accuracy of predictions under adversarial perturbation. The graph comparing true versus predicted values on perturbed inputs reveals a striking distinction between the two models. For the standard model, there was a noticeable divergence between the true values and the predictions, indicating a vulnerability to adversarial distortion. Conversely, the adversarially trained model showcased a remarkable alignment between the predicted and true values, even under perturbed conditions. This closer correlation signifies a robustness in the model's predictive capabilities, affirming that adversarial training not only safeguards against erroneous behaviors but also preserves the integrity of the model's output, ensuring reliable and accurate predictions in the face of adversarial challenges.

VIII. Conclusion

In this paper, we have demonstrated the practicality of effectively mitigating adversarial attacks on autonomous driving models in real-time. By developing both a robust image-specific attack and a stealthier image-agnostic attack, we exposed the vulnerabilities in safety-critical



(a) True vs Predicted: Perturbed input to the existing self-driving car model

(b) True vs Predicted: Perturbed input to the our self-driving car model

Fig. 10: True vs Predicted graph

robotic systems. In response, our innovative approach of incorporating random Gaussian noise into the driving model significantly enhanced its resistance to these attacks. This strategy not only highlights the potential vulnerabilities of autonomous vehicles but also presents a viable solution to mitigate such risks. Moving forward, our focus will be on expanding the model's defensive capabilities by exploring additional adversarial mitigation methods and integrating course correction algorithms, contributing to the advancement of more secure and reliable autonomous driving technologies.

References

[1] Singh, M.J, Ramachandra R., (2015) Deep Composite Face Image Attacks: Generation, Vulnerability and Detection. doi: https://arxiv.org/pdf/2211.11039.pdf

[2] Wu H., Yunas S., Rowlands S., Ruan W., (2022) Adversarial Driving: Attacking End-to-End Autonomous Driving. doi: https://arxiv.org/pdf/2103.09151.pdf

[3] M. Brand, I. Naeh, D. Teielman, (2022) Adversarial Attack Against Image-Based Localization Neural Networks. doi: https://arxiv.org/pdf/2210.06589.pdf

[4] K. Tan, J. Wang, Y. Kantaros, (2022) Targeted Adversarial Attacks against Neural Network Trajectory Predictors. doi: https://arxiv.org/pdf/2212.04138.pdf

[5] Ren K., Zheng T., Qin Z., (2020) Adversarial Attacks and Defenses in Deep Learning.

[6] Goodfellow IJ, Shlens J, Szegedy C. Explaining and harnessing adversarial examples. 2014. arXiv:1412.6572.

[7] Rathore P, Basak A., Nistala S.H., Untargeted, Targeted and Universal Adversarial Attacks and Defenses on Time Series. 2020 DOI - 10.1109/IJCNN48605.2020.9207272

[8] Brown, T. B., Mané, D., Roy, A., Abadi, M., Gilmer, J. (2018). Adversarial Patch.

[9] Kong, Z., Guo, J., Li, A., Liu, C. (2021). PhysGAN: Generating Physical-World-Resilient Adversarial Examples for Autonomous Driving.

[10] Salman, H., Ilyas, A., Engstrom, L., Vemprala, S., Madry, A., Kapoor, A. (2020). Unadversarial Examples: Designing Objects for Robust Vision.

[11] J. Dong, J. Chen, X. Xie (2023). Adversarial Attack and Defense for Medical Image Analysis: Methods and Applications. doi: https://arxiv.org/pdf/2303.14133.pdf

[12] Bran M., Naeh I., Adversarial Attack Against Image-Based Localization Neural Networks

[13] https://www.semanticscholar.org/paper/Fooling-OCR-Systems-with-Adversarial-Text-Images-Song-Shmatikov/266a06c96834bfd24dc3cb56c8ef191a0c8b5b7a/figure/

[14] https://www.semanticscholar.org/paper/Fooling-OCR-Systems-with-Adversarial-Text-Images-Song-Shmatikov/266a06c96834bfd24dc3cb56c8ef191a0c8b5b7a/figure/

[15] https://research.nvidia.com/publication/2022-10_advdo-realistic-adversarial-attacks-trajectory-prediction

[16] https://www.researchgate.net/figure/An-illustration-of-a-common-adversarial-attack-on-image-classification-ML-model-The_fig4$_3$47398615