UNIVERSITY OF TARTU
Faculty of Science and Technology
Institute of Computer Science
Computer Science Curriculum

Mike Gomes Camara

Safety Analysis of Autonomous Vehicle Systems Software

Master’s Thesis (30 ECTS)

Supervisor(s): Dietmar Alfred Paul Kurt Pfahl, PhD

Tartu 2022

Safety Analysis of Autonomous Vehicle Systems Software

Abstract:

Machine learning approaches to autonomous driving systems that rely upon computer vision and deep neural networks have demonstrated encouraging results in the past. Some believe that the so-called end-to-end strategy is the only way to deploy self-driving vehicles at scale in the future. Safety is a concern as the efficacy of such neural networks is susceptible to adversarial attacks and are also highly dependent upon a RGB camera input.

Literature suggests that different attacks require equally different procedures to defend against such threats. However, there is no understanding of how adversarial defenses can improve the capacity of an end-to-end self-driving model to generalize to different lighting conditions.

This paper aims to understand how adversarial attacks can help a machine learning model to increase resilience and generalize better to different lighting conditions. First, we select a scaled real-world driving platform and a neural network to drive the model. Then, we designed an experiment, implemented, tested, and evaluated the results.

In conclusion, adversarial defenses did impact the capacity of a self-driving end-to-end model to generalize to different lighting conditions. Therefore, further research is suggested employing adversarial training to increase the robustness of autonomous driving models.

Keywords:

adversarial attacks, adversarial machine learning, autonomous vehicles, driverless cars, self driving cars, open source, CNN, deep learning, Raspberry Pi, behavioral cloning

Contents

1 Introduction

End-to-end driving is an approach to autonomous driving that has become a growing trend in autonomous vehicle research both in industry and academia[1]. Unlike modular methods that use expensive sensors, end-to-end techniques to autonomous driving rely on computer vision and machine learning to generate models that command steering and acceleration [2]. However, these models are notoriously susceptible and vulnerable to adversarial images[3], which are disturbances sometimes imperceptible to the human eye but can cause the model to misbehave.

In the context of end-to-end autonomous vehicles, adversarial attacks can lead to catastrophic consequences such as loss of life and property [4]. Research has been done on the impact of malicious attacks in autonomous-vehicle neural networks, though only in simulation environments[5]. To the best of our knowledge, research that attempts to use adversarial attacks with scaled autonomous cars in a real world setting to improve their model generalization to adversarial images does not exist.

Fleets of autonomous vehicles that use machine learning are ubiquitous and available to the general public. Creating strategies to defend end-to-end models and add robustness and resilience against adversarial attacks is paramount. Such vulnerabilities must be addressed and mitigated before we can see wider adoption of machine learning models to predict the lateral and longitudinal control of autonomous cars [6].

This thesis will carry out an experiment to evaluate the effectiveness of defenses strategies against a selected adversarial attack in real-world scaled autonomous cars and their impact in the generalization to lighting conditions.

1.1 Motivation

Car manufacturers such as Tesla deploy self driving solutions trained with deep neural networks at scale [7]. Nowadays, Tesla owners can experience having their car driving autonomously while abiding to still pay attention to the roads. However, in 2016, a Tesla Model S. with the Autopilot feature engaged failed to apply the brake after not noticing the white side of a tractor trailer against a brightly lit sky, tragically killing its driver [8].

End-to-end methods are a deep neural network approach to self-driving and a growing trend in autonomous driving research because such methods are cheaper, simpler, and scalable, unlike conventional modular approaches [1].

However, its simplicity comes with the cost of lack of interpretability and vulnerability to adversarial machine learning attacks, which can lead the model to misbehave and is impossible or difficult to understand why. For instance, minimal changes in the light exposure can be enough to cause a model to misbehave, which in the context of autonomous driving could potentially lead to catastrophic consequences.

Adversarial attacks to machine learning systems can not be eliminated, and models must be resilient and generalize well to adversarial circumstances. Thus, strategies need to be explored to defend the models and make them more robust against such threats.

1.2 Goals

This thesis aims to adopt a selected adversarial machine learning attack to improving their generalization ability to unseen lighting condition.

We investigate the literature to discover methods to train and validate neural networks. Then we implement an experiment to create a self-driving agent and evaluate its capacity to generalize to unseen illumination intensities and adversarial images. Finally, we select an appropriate adversarial attack and create strategies to defend against such attack, including retraining the model and exposing it to perturbations while training.

As a result of the defense training, the model should generalize better, such as maintaining the baseline performance when exposed to different light conditions and correctly classifying standard input images while becoming more resilient to a selected adversarial attack.

The thesis is organized as follows: Section 2 discusses the building blocks of an autonomous vehicle using neural networks. We then discuss the threats against machine learning and the precautions necessary to deal with adversarial attacks. Finally, we investigate applying those concepts in a real-world scaled autonomous vehicle. Section 3 will illustrate the approach taken and the detailed phases necessary to accomplish the experiment. Section 4 supplies the outcomes of each stage described in the prior section. Section 5 discusses lessons learned and the limitations of the project. Finally, Section 6 encloses our evaluation conclusions and suggests future work.

2 Background

In this section, a literature review is presented covering the use of computer vision to enable the creation of self-driving agents. We describe the use of neural networks to create machine learning models capable of driving a scaled car autonomously and the need for new methods of testing their efficacy. Additionally, the topic of adversarial machine learning attacks is covered, including the Fast Gradient Sign method, which is the adversarial technique tested in this research.

In section 2.1, several papers on autonomous vehicles were considered. The existing computer vision methods to approach vehicular autonomy were reviewed. Several works that use machine learning to generate self-driving models were considered.

In section 2.2, the background is provided on the creation of neural networks models, including data collection and training. A neural network type is selected for the experiment, and the motivation to use such approach is explained.

Section 2.3 examines the vulnerability of machine learning to adversarial examples. The existent attack types are reviewed, and the strategies to mitigate such issues are analyzed. Also, research that demonstrates the use of adversarial attacks to improve the robustness of machine learning models is reviewed.

In section 2.4, papers that described the implementation of real-world self-driving platforms were studied. Moreover, the methods and metrics to describe safety were scrutinized.

2.1 Computer Vision in self-driving

In 2004, DARPA (Defense Advanced Research Projects Agency) funded a competition to promote autonomous vehicles. The challenge had the prize of one million dollars for the fastest vehicle to complete a 240 km route. No one completed the course in the first edition and the Carnegie Mellon University's car, Sandstorm, traveled the most distance, completing 11.78 km [9].

The following year's challenge edition doubled the prize and saw five vehicles completing the course. Sandstorm, this time, made to the podium as the second-best contender, and it was only eleven minutes slower than the winner car Stanley, from Stanford University, that completed the 212 km route in just below seven hours. Both Stanley and Sandstorm to control the car used a combination of complex sensors such as Lidar and GPS. The positive results of the challenge stimulated a surge in research on the topic.

It is interesting to notice that, Carnegie Mellon University team, much earlier, in 1989, had already introduced ALVINN, the first neural network-powered autonomous vehicle. The paper was ahead of its time, and it is still to this day relevant. Unlike Sandstorm and Stanley, ALVINN did not use expensive sensors to output the steering decisions for the vehicle. Instead, only one frontal camera was used to feed a convolutional neural network model that was previously trained with many images of human-generated driving data [10].

While pioneering the use of machine learning to vehicle autonomy, the processing power required to train the convolutional neural networks limited the approach from getting traction at the time.

However, the meteoric development of computer capabilities enabled NVIDIA's research team to create a convolutional neural network that was able to learn to navigate a car through highways and traffic using cameras solely [11].

2.2 Neural Networks

Neural networks are a subset of machine learning, and both are a subset of artificial intelligence. As the human brain's neural networks, artificial neural networks can learn to classify data through training [12]. In the context of autonomous vehicles, this approach proposes processing sensory inputs to generate lateral and longitudinal control commands using complex mathematical models as a single machine learning task.

Machine learning is generally divided into three major categories, supervised learning, unsupervised learning, and reinforcement learning. The supervised approach uses labeled data to train a model to classify categories of unseen data automatically. On the other hand, the unsupervised approach automatically detects categories and patterns using large amounts of unlabeled data. Finally, the reinforcement learning approach involves training a model from scratch through exploration and improvement of driving policy, utilizing a rewarding mechanism that punishes bad decisions while rewarding good ones, which keep improving the model until an acceptable performance is achieved [1].

Reinforcement learning has been used to train real cars to drive in the physical world. However, this approach is more common in simulation environments. Nevertheless, impressive results have been demonstrated of vehicles trained in simulation but performed well when transferred to the physical environment [13].

In autonomous driving, the supervised learning approach is achieved via behavioral cloning, a form of imitation learning. The model mimics the human driver's behavior by mapping observations and motions.

2.2.1 Convolutional Neural Networks

Convolutional neural network (CNN, or ConvNet) is a deep Learning approach to machine learning and is called deep because of the number of additional hidden layers added to learn from the data. Because of its benchmark efficiency, it is widely used to analyze visual imagery.

Here is an example of a neural networks architecture (see Figure 1).

The network can have one or many hidden layers to process and output the model.

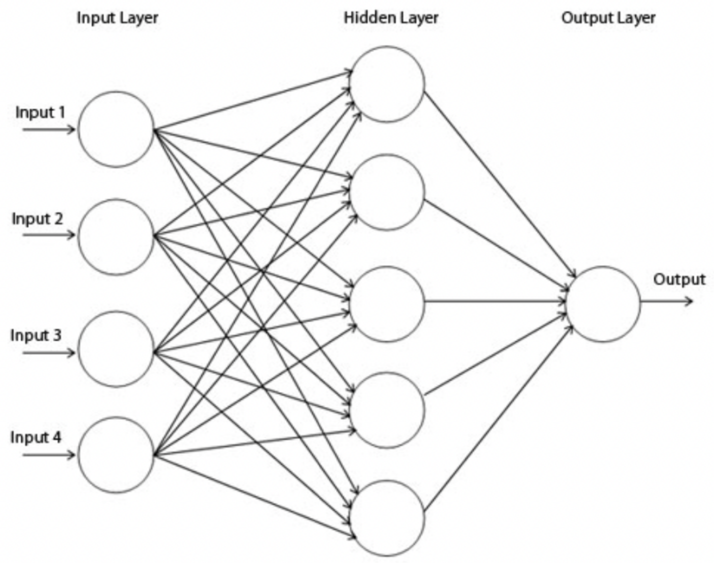

Figure 1. Neural network with one hidden layer [12].

In the context of self-driving cars, the use of deep neural networks is also known as end-to-end methods. It was first used by ALVINN [10], it is simple to use while yielding good results. End-to-end methods rely on convolutional neural networks to fuse data coming from multiple sources. Current state-of-the-art CNN models out perform on the standard the NoCrash urban driving benchmark. The results point impressive success rates driving in urban environments. However, tests are run in simulation environments and the models do not show enough generalisation to be deployed in the real-world, which exposes the need for more work on ways to make models generalise better to the real-world setting conditions. [14]

2.2.2 Models

This research project explores the use of adversarial images to improve the capacity of a machine learning model to generalise to the real-world changing lighting conditions. The model is the output of a neural network and consists of an array of decision making algorithm.

In this project, a single output linear convolutional neural network is used. The model is composed of five convolution layers followed by two dense layers before the output of two dense layers with one scalar output each with linear activation for steering and throttle

A similar linear model architecture is found in NVIDIA's paper [11] (see Figure 2).

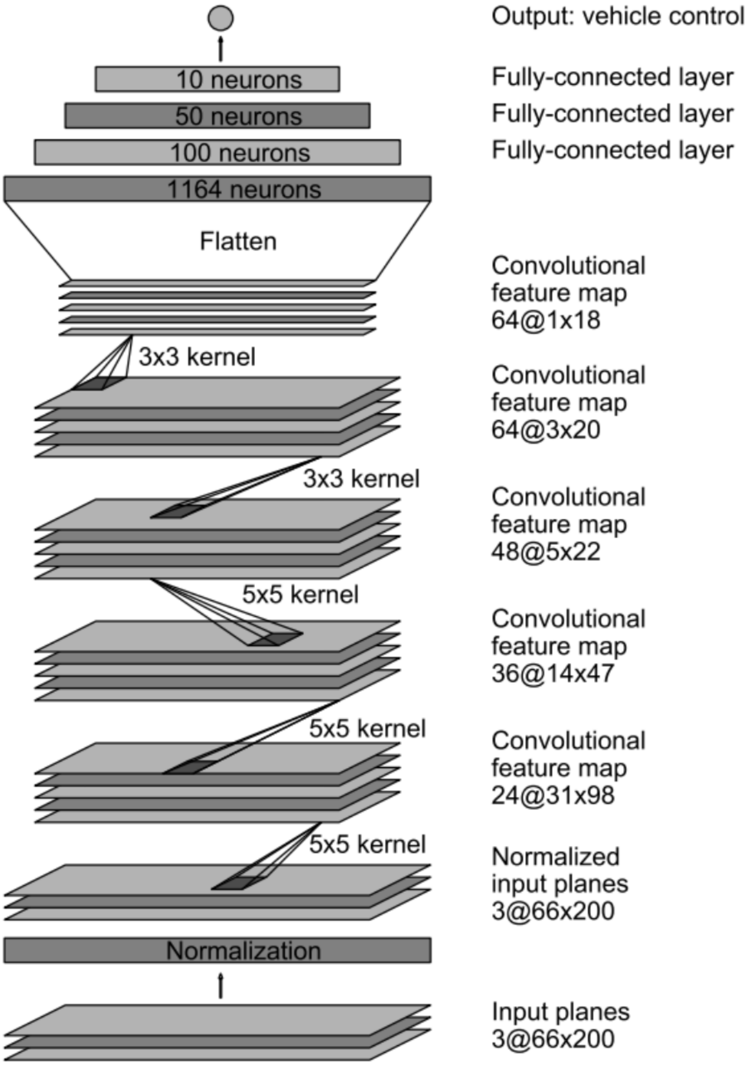

Figure 2. Linear network architecture [11]

This type of network is robust and enables a smoothly steering, it also performs in low compute environment such as the Raspberry Pi 4 [15].

2.2.3 Training

The process of training a neural network involving applying mathematical and statistical concepts such as feed forward and error back propagation that gradually reduce the error between the predict output and the desired output across a number of iterations over the dataset. Each cycle of a full training iterations is called epoch [12].

2.3 Adversarial Attacks

An adversarial attack consists of a method to generate adversarial examples purposely designed to cause a machine learning model to fail in its predictions. However, despite causing great harm, the perturbations generated with the adversarial images can be subtle enough to be imperceptible to the human eye.

Adversarial attacks are organized in three categories: evasion, poisoning, and extraction attacks, Piazzesi et al. [5] explores in their study only evasion attacks, because of its likelihood to compromise safety by being carried out on a self-driving agent while it is running. Such attacks can be white-box and black-box attacks. While the white-box attacks require having full access to the architecture and parameters of the model, the black-box attacks do not require having the knowledge on the model structure and architecture.

There is no clear understanding as the impact of adversarial images on machine learning model ability to generalize to real-world unstable lighting condition.

2.3.1 Fast Gradient Sign Method

The fast gradient sign method [16] is a reliably way to generate adversarial examples that cause models to misclassify their input.

See Figure 3 for a demonstration of fast adversarial example generation applied to GoogLeNet on ImageNet [16].

2.3.2 defense to Adversarial Attacks

Rosebrok [17] demonstrated the possibility to improve the model's ability to generalize and defend against adversarial attacks using adversarial images during the training process.

The concept consists in modifying the standard training procedures adding adversarial images. A model in combination with a method such as FGSM can be used to generate a total of N adversarial images and then combine the two sets, forming a batch double of the original size containing both adversarial examples and original training samples that will be used during the training process.

Training CNNs with a mixing of normal and adversarial images has improved model's robustness against adversarial images in simulation environments. This thesis investigates

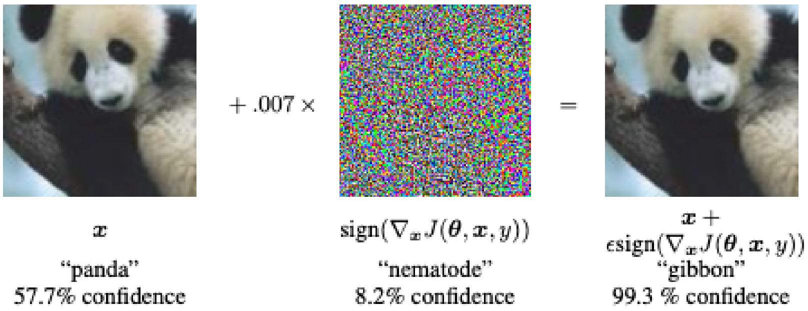

Figure 3. Goodfellow et al. first demonstrated how a discrete perturbation is constructed by the model and added to the input causing the neural network to fail the classification. [16]

the possibility of using adversarial attacks to improve the model generalization to external lighting conditions in a real-world setting.

2.4 Scaled Autonomous Car

Mahmoud et al. [18] used a real-world scaled self-driving car to demonstrate that it was possible to improve a neural network model using image augmentation techniques. By reducing the image sizes, the response rate of the neural models increased improving safety and the speed of the vehicle.

In their experiment, the Donkeycar self driving platform for small vehicles was used. Donkeycar is a open source library that uses a Raspberry Pi 4 attached to a camera to control an remote control car to drive autonomously.

2.4.1 Donkeycar Platform

Donkeycar [19] is an open-source easy-to-use and well-documented Python library that can be used in association with a self-driving 1:10 scale remote control car. It comes pre-assembled equipped with a Raspberry Pi, camera, remote car chassis, battery and a sensor hat.

The software relies on open-source libraries such as OpenCV for image processing and Keras, a light weight python neural network library capable of running on TensorFlow.

Donkeycar has an active community of machine learning and autonomous driving enthusiasts who keep improving the library and adding new features. Some improvements include traffic-light and stop sign detection using the external hardware Google Coral [20] to process a benchmark object detection algorithm. This will not be used in this project as the focus of the research is the improvement of the model that predicts the navigation of the vehicle.

Another hardware that comes included with the car is the Inertial Measurement Unit (IMU) which is a set of inertial sensors attached to the Raspberry Pi that uses gyroscopes and accelerometers to track movement of the device [21]. It is possible to use the IMU information for navigation purposes, however, the only sensor used in the creation of the model used in this research is one front camera attached to the vehicle.

2.4.2 Safety

Zhang et al. [22] propose in their study an end-to-end evaluation framework with a set of driving safety performance metrics. The research measures the impact of adversarial attacks on the driving safety of vision-based autonomous vehicles. The collision rate and the success of route completion rate were some of metrics used to validate the results under the perturbation attack.

3 Method

This section describes the plan and the procedures to train a CNN supervised classifier autopilot using behavioral cloning with a 1:10 scale car based on the open-source platform Donkeycar.

The goal of the project is to train deep learning neural network models to drive autonomously and then improve their generalization ability to unseen lighting conditions. The method is divided into three main stages, including training the model, improving the model using adversarial attacks, and evaluating the results.

Initially, data is collected using a frontal RGB camera attached to the car. Next, the data is cleaned, and inaccurate records are removed from the recordset. The data is then used to train a convolutional neural network architecture with imitation learning. Two phases of testing will follow. The initial testing will compare the performance of the car when it first is exposed to the same lighting conditions used during the training phase and second when exposed to unseen lighting conditions. The results of the first testing phase will establish the baseline. Depending on the results, if the model fails to generalize to unseen lighting conditions, then an adversarial defense training method will be employed as an attempt to improve the model generalization skills.

The final testing phase involves analyzing the performance of the model trained with adversarial data in the same settings used in the baseline to evaluate and measure success. Details of all the stages of the experiment are described in the following subsections.

3.1 Testbed and Selection of Self-Driving Platform

The testbed architecture is designed for training and testing behavior cloning models on car-like robots, such as the Donkeycar. The model car is equipped with a mono frontal wide-angle RGB camera, capturing 120x160 images labeled with the motor throttling and steering angle values used for the classification training, creation, and testing of a self-driving agent model.

A Raspberry Pi 4 microcontroller is used to run the Keras model using a Tensorflow backend to trigger commands to servo-controlled steering and thrust motors. The model is trained on an indoor circuit constructed with white gaffer tape and cardboard walls. Although the Donkeycar design allows for the integration of sensors such as LiDAR and IMU, such sensors will be kept out of the project's scope, as the object of analysis is solely the input of the RGB camera system instead.

3.2 Experimental Setup and Dataset

To train an autopilot with Keras, the Donkeycar recommends collecting from five to twenty thousand images as the minimal amount to produce a good model[23].

To collect the data necessary for the experiment, the testbed, composed of a front-facing wide-angle camera, captures two sets of data, the first containing almost five thousand images and a second larger dataset containing just under ten thousand images. The need for two datasets with different sizes is to compare the results and evaluate the impact of the volume of data on the model's overall performance in two distinct scenarios.

3.3 Description of the Scenarios

The experimental results are based on the platform's performance described in the previous subsection. Five points cover a range of locations in the track to illustrate different luminance conditions. The points consist of locations captured by the platform that drives in the circuit until it cannot proceed due to a collision with the wall. Both scenarios maintain the shape of the circuit. However, the light intensity is changed.

The model is trained only with the data captured in the first scenario. Thus, the models will have their generalizations skills tested in unseen luminance conditions when exposed to the second scenario. The vehicle's camera captures ten frames per second. The data are stored, cleaned, and used in successive training.

An expert policy is collected in two different sessions. The first session performs about 20 laps, and the second has 37 laps, gathering approximately 5000 images and 9000 images, respectively. All images are labeled with their corresponding steering angles and throttling values. The steering has number values within the range of [-1.0, 1.0], in which positive values correspond to turning to the right, and negative values correspond to turning to the left. Throttle values are within the range of [0, 1.0], in which zero the vehicle is stopped, and positive values mean forward moving.

3.4 Training a Pilot Model

Behavioral cloning is used to train a CNN model. It applies the Markov Decision Process to create a supervised classification CNN driving model. The training and validation are performed using a split of the dataset with a ratio of 0.8:0.2, and the validation occurs during a set of iterations called epochs.

After the neural network architecture of the autopilot has been successfully trained, it classifies the input image and predicts the most suitable throttling value and steering angle. The convolutional neural network used in this project was the default Donkeycar linear model, consisting of five convolution layers, a flattening layer, and three fully connected layers.

3.5 Design and Implementation

The model needs to be exposed to a scenario in which the lighting conditions differ from the initial data to improve the baseline. Therefore, a new scenario needs to be implemented. Adversarial defense methods are applied in the data to improve the expert policy's generalization skills to unseen lighting conditions.

After scenario B is set up, an adversarial image generator must be integrated with the platform. Specific implementation settings and values will differ depending on the experiment environment. This step includes testing and performing iterations to find appropriate values.

Once the driving platform is implemented and integrated with the adversarial defense training methods, metrics must be provided to allow comparison. The metrics should quantify the autopilot model's performance and the circuit's lighting conditions in scenarios A and B.

The final purpose is to create a proof of concept strategy for training more skillful machine learning models capable of generalization to unseen lighting conditions.

3.5.1 Testing 1

A model trained with adversarial defense techniques is deployed with the metrics values established in both scenarios. The baseline metrics will define if the process is successful or not. In addition, to attain success, the model would need to improve its performance concerning the baseline.

After the model performs in both scenarios, the resulting metrics will be compared against the baseline. If the improved model does not outperform the standard model, e.g., it does not generalize to unseen lighting conditions. Creating more scenarios will be considered as there would not be sufficient differences in the baseline metrics.

This stage includes running the models in standard and adversarial conditions, which can cause the car to collide against the track and be relocated in the center of the lane manually. The behavior of the model might become unstable under adversarial conditions. Hence, the need for mechanisms that safeguard against such threats.

This step aims to identify whether the proof of concept adversarial training method can generalize to unseen illumination levels.

3.5.2 Testing 2

If the tests performed in the previous phase are successful, then the models are exposed to a selected adversarial attack while performing in the circuit. First, the model without adversarial defense is deployed while being under attack. After verifying the performance of the first model, the second model created with adversarial training is launched to perform under attack and verify if the adversarial training has improved the model's resilience against a selected adversarial attack.

This final stage verifies if the proof of concept training method improves the efficacy of the models against adversarial images. The goal is to evaluate the performance of models trained with an adversarial defense method against a selected hostile attack.

3.6 Evaluation

The last stage is evaluating the previous phases of the experiment. The success criteria must be defined prior to measuring the results.

Adversarial defense training methods can be considered successful if:

- The performance of the model in the standard scenario is unaffected.
- If the performance of the model is improved compared to the baseline.
- The model can drive at least two laps autonomously in the circuit in scenarios A and B without collision.
- The implementation of the method is straightforward and can be duplicated.
- The training method is helpful and improves the generalization of the model to unseen lighting conditions, having the same amount of initial data.

It is important to note that the evaluation will only be based on stage one of testing. If the model evaluation shows positive results, it can be tested against an adversarial attack while performing in the circuit. To evaluate success criteria in testing 2, the model must at least outperform the standard model and complete two laps with fewer human interventions.

4 Results

Some text...

4.1 Selection of Self-Driving Platform

Some text...

4.2 Selection of Driving Model Architecture

Some text...

4.3 Training Pilot Model

Some text...

4.4 Design and Implementation

Some text...

4.4.1 Context

Some text...

4.4.2 Implementation

Some text...

4.4.3 Baseline Metrics

Some text...

4.4.4 Testing 1

Some text...

Adversarial Attack Generator Some text...

4.4.5 Testing 2

Some text...

Implement defense Mechanism Some text...

4.4.6 Testing 3

Some text...

Test Robust Model on Baseline Some text...

4.4.7 Testing 4

Some text...

4.5 Evaluation

5 Discussion

5.1 Lessons Learned

5.2 Limitations

6 Conclusion and Future Work

7 Acknowledgement

References

[1] A. Tampuu, M. Semikin, N. Muhammad, D. Fishman, and T. Matiisen, "A survey of end-to-end driving: Architectures and training methods," *CoRR*, vol. abs/2003.06404, 2020.

[2] A. Zolfi, M. Kravchik, Y. Elovici, and A. Shabtai, "The translucent patch: A physical and universal attack on object detectors," in *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2021, virtual, June 19-25, 2021*, pp. 15232–15241, Computer Vision Foundation / IEEE, 2021.

[3] A. Qayyum, M. Usama, J. Qadir, and A. I. Al-Fuqaha, "Securing connected & autonomous vehicles: Challenges posed by adversarial machine learning and the way forward," *IEEE Commun. Surv. Tutorials*, vol. 22, no. 2, pp. 998–1026, 2020.

[4] P. Sharma, D. Austin, and H. Liu, "Attacks on Machine Learning: Adversarial Examples in Connected and Autonomous Vehicles," 2019.

[5] N. Piazzesi, M. Hong, and A. Ceccarelli, "Attack and fault injection in self-driving agents on the carla simulator - experience report," in *Computer Safety, Reliability, and Security - 40th International Conference, SAFECOMP 2021, York, UK, September 8-10, 2021, Proceedings* (I. Habli, M. Sujan, and F. Bitsch, eds.), vol. 12852 of *Lecture Notes in Computer Science*, pp. 210–225, Springer, 2021.

[6] S. Pavlitskaya, S. Ünver, and J. M. Zöllner, "Feasibility and suppression of adversarial patch attacks on end-to-end vehicle control," in *23rd IEEE International Conference on Intelligent Transportation Systems, ITSC 2020, Rhodes, Greece, September 20-23, 2020*, pp. 1–8, IEEE, 2020.

[7] T. Team, "Artificial intelligence autopilot,"

[8] T. Team, "A tragic loss," 2016.

[9] "Darpa grand challenge," 2016.

[10] D. Pomerleau, "ALVINN: an autonomous land vehicle in a neural network," in *Advances in Neural Information Processing Systems 1, [NIPS Conference, Denver, Colorado, USA, 1988]* (D. S. Touretzky, ed.), pp. 305–313, Morgan Kaufmann, 1988.

[11] B. F. L. J. U. M. K. Z. Mariusz Bojarski, Ben Firner and D. D. Testa, "End-to-end deep learning for self-driving cars," 2016.

[12] J. Z. Chang, "Training neural networks to pilot autonomous vehicles: Scaled self-driving car," 2018.

[13] Q. Zhang and T. Du, “Self-driving scale car trained by deep reinforcement learning,” *CoRR*, vol. abs/1909.03467, 2019.

[14] T. Agarwal, H. Arora, and J. Schneider, “Affordance-based reinforcement learning for urban driving,” *CoRR*, vol. abs/2101.05970, 2021.

[15] “Keras parts,” 2021.

[16] I. J. Goodfellow, J. Shlens, and C. Szegedy, “Explaining and harnessing adversarial examples,” in *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings* (Y. Bengio and Y. LeCun, eds.), 2015.

[17] A. Rosebrock, “Mixing normal images and adversarial images when training cnns,” 2021.

[18] T. F. T. K. a. I. A. Yaqub Mahmoud, Yuichi Okuyama, “Optimizing deep-neural-network-driven autonomous race car using image scaling,” 2020.

[19] “An opensource diy self driving platform for small scale cars.,”

[20] “Build beneficial and privacy preserving ai,”

[21] G. A. Santiago and M. Favre, “Analysis and evaluation of recurrent neural networks in autonomous vehicles,” 2017.

[22] J. Zhang, Y. Lou, J. Wang, K. Wu, K. Lu, and X. Jia, “Evaluating adversarial attacks on driving safety in vision-based autonomous vehicles,” *CoRR*, vol. abs/2108.02940, 2021.

[23] “Train an autopilot with keras,”

Appendix

I. Glossary

II. Licence

Non-exclusive licence to reproduce thesis and make thesis public

I, **Mike Camara**,
*(*author's name)

1. herewith grant the University of Tartu a free permit (non-exclusive licence) to

   reproduce, for the purpose of preservation, including for adding to the DSpace digital archives until the expiry of the term of copyright,

   Safety Analysis of Autonomous Vehicle Systems Software,
   *(*title of thesis)

   supervised by Dietmar Alfred Paul Kurt Pfahl.
   *(*supervisor's name)

2. I grant the University of Tartu a permit to make the work specified in p. 1 available to the public via the web environment of the University of Tartu, including via the DSpace digital archives, under the Creative Commons licence CC BY NC ND 3.0, which allows, by giving appropriate credit to the author, to reproduce, distribute the work and communicate it to the public, and prohibits the creation of derivative works and any commercial use of the work until the expiry of the term of copyright.

3. I am aware of the fact that the author retains the rights specified in p. 1 and 2.

4. I certify that granting the non-exclusive licence does not infringe other persons' intellectual property rights or rights arising from the personal data protection legislation.

Mike Gomes Camara
04/01/2022