Poster: Are Self-Driving Cars Secure? Evasion Attacks against Deep Neural Networks for Steering Angle Prediction

Alesia Chernikova*, Alina Oprea*, Cristina Nita-Rotaru* and BaekGyu Kim*†
*Northeastern University, Boston, MA
†Toyota ITC, USA

Abstract. The paper will be presented at the IEEE SafeThings 2019 workshop collocated with the IEEE Symposium on Security and Privacy.

Advances in Machine Learning (ML) and Deep Neural Networks (DNNs) bring tremendous potential to make autonomous vehicles a reality. For this highly-critical application, safety is the major concern, but unfortunately ML algorithms are not traditionally designed and evaluated from this perspective.

In this poster, we demonstrate that classification and regression models for self-driving car applications are vulnerable to adversarial evasion attacks at testing time. We consider the case study of steering angle predicting from camera images, using the dataset from the 2014 Udacity challenge 2 [1]. First, we adapt the state-of-the-art Carlini and Wagner 2017 evasion attack [2] to the classification problem of predicting steering direction. Second, we design the first testing-time attack for regression based on Convolutional Neural Networks (CNNs) and test it in the context of this application.

Problem statement and threat model. Modern cars are outfitted with Electronic Control Units (ECUs) to control specific functions on the car, such as controlling brakes [3]. In connected cars, some of these ECUs communicate outside of the car, such as on-board diagnostics [3]. While this adds functionality, it also opens them up to attack.

We consider the problem of predicting steering angles using DNNs that process camera images. An attacker can potentially control one or multiple ECUs, and can spoof messages from the camera due to lack of authentication on the CAN bus [3]. The attacker can modify the image sent by a camera, constructing an adversarial example which will be misclassified by a steering angle controller for the autonomous vehicle. This could result in a different prediction generated by the ML model.

Data. The training data consists of 33,608 images extracted from the videos provided by the Udacity self-driving car challenge 2. We apply image preprocessing as done by previous work [4], [5]. Each datapoint includes the steering angle at the moment the image was captured. The steering angles in the Udacity data set driving log were pre-scaled by a factor of 1/25. To assign classification labels, we split the scaled angle values into three intervals to obtain the three directions (**left**, **straight**, and **right**).

DNN architectures. We select two CNN models for both the classification and regression problems. The first is the *Epoch model* [4], while the second is inspired by Bojarski et al. [6] (called *NVIDIA model*). We adapted both models for classification by adding a last layer with 3 hidden units and softmax activation function. The architecture for regression is similar, excluding the last softmax layer. The NVIDIA model is more complex (467 million parameters) compared to the Epoch model (25 million parameters).

Evasion attacks against direction classification. We use the L_2 distance between the original and adversarial image to measure the amount of perturbation introduced by the attack. We leverage and adapt the state-of-art L_2 attack by Carlini and Wagner [2], proposed originally in the context of image classification. The attack crafts adversarial examples by solving the following optimization problem for an image x with original class i to find the perturbation σ that transforms it into a targeted class $t \neq i$:

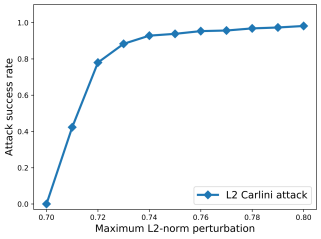
$$\begin{gathered}\text{minimize } ||\sigma||_2 + c \times f(x+\sigma) \\ \text{such that } x+\sigma \in [0,1]^d \\ f(x+\sigma) = (max(Z(x+\sigma)_{j\neq t}) - Z(x+\sigma)_t)^+ \\ i \text{ - original class, } t \neq i \text{ - adversarial target class.}\end{gathered}$$

Here s^+ is the notation for $\max(s, 0)$,

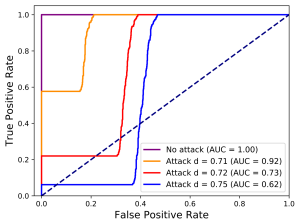
Evasion attacks against steering angle regression prediction. We are not aware of existing evasion attacks against CNNs for regression. A regression model is typically evaluated by the Mean Square Error (MSE) metric, defined either for single points or over an entire dataset. MSE of a single point x with response $y \in R$ measures the squared residual (e.g., difference between the true response y and the predicted response $\hat{y} = F(x)$). For a dataset, MSE is the average of the squared residuals of all points. Our main insight is to adapt the classification attack by changing the objective function *to maximize the MSE difference between the predicted response on the adversarial image* $F(x+\sigma)$ *and the true response* y. This way, the attacker attempts to change the prediction on the adversarial image further away from the true value.

Thus, in order to find the adversarial image for original image x with response y, the attacker solves the following optimization task with respect to the parameter σ:

$$\text{minimize } ||\sigma||_2 - c \times g(x+\sigma, y)$$



(a) Success rate

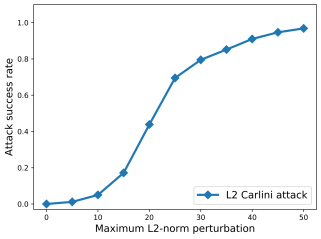


(b) ROC curves

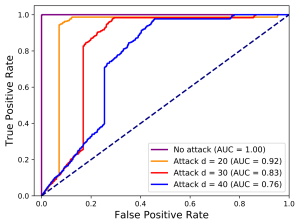
(c) Input image: right

(d) Adversarial image: left

Fig. 1: Results and adversarial images for the Epoch model.



(a) Success rate



(b) ROC curves

(c) Input image: straight

(d) Adversarial image: right

Fig. 2: Results and adversarial images for the NVIDIA model.

$$\text{such that } x+\sigma \in [0,1]^d$$
$$g(x+\sigma, y) = (F(x+\sigma) - y)^2$$

Here c is a hyper-parameter that is found by binary search; it controls the tradeoff between minimizing the image perturbation versus maximizing the MSE value.

Training results. We train both models using 10-fold cross validation. The accuracy for classification is high: 90% for the Epoch model and 86% for the NVIDIA model.

Attack results for direction prediction. For testing the attack we choose 300 images from all 3 classes, and select the 2 values of the targeted class (different from original class), resulting in 600 adversarial images. We found the optimal value for the attack hyper-parameter c by running binary search for 9 steps with the initial value of c equal to 0.001. As expected, if there are no constraints on adversary's ability to manipulate images, the adversarial success rate reaches 100%.

The attacker success on the two models with respect to the amount of perturbation is illustrated in Figures (1a) and (2a). In the Epoch model, a minimum modification to the image (0.82 L_2 norm) results in 100% attack success. However, the amount of perturbation for NVIDIA is higher (121.01 L_2 norm). We conjecture the reason to be the additional complexity of the NVIDIA model, resulting in a more robust architecture.

We next study the impact of the attack on the models' performance. The micro-average ROC curves with and without the attack are in Figures (1b) and (2b), respectively. False positive rate for each class is the number of adversarial images classified as this class. It could be easily seen that the model performance decreases under attack (for instance, AUC decreases from 1 in the no-attack scenario to 0.62 for 0.75 L_2 norm perturbation for the Epoch model).

An example image from the **right** class and its resulting adversarial image from the **left** class are shown in Figures (1c) and (1d) for the Epoch model. Figures (2c) and (2d) show an image from the **straight** class, turned into an adversarial example from the **right** class for the NVIDIA model.

Attack results for steering angle prediction. For testing the regression attack we choose 100 images. We found the optimal value for the attack hyper-parameter c by binary search.

After performing the attack we observe that 90% of adversarial images have perturbation value less than 0.57 L_2 norm, which is very small. Additionally, our attack results in significant changes to the MSE of adversarial images. In particular, 10% of adversarial images have an MSE value more than 20 times higher than the MSE value of the corresponding legitimate image. The maximum ratio of adversarial to legitimate MSE is 69.

REFERENCES

[1] O. Cameron, "Teaching a machine to steer a car," https://medium.com/udacity/teaching-a-machine-to-steer-a-car-d73217f2492c, 2016.

[2] N. Carlini and D. Wagner, "Towards evaluating the robustness of neural networks," in *Proc. IEEE Security and Privacy Symposium*, 2017.

[3] P. Kleberger, T. Olovsson, and E. Jonsson, "Security aspects of the in-vehicle network in the connected car," in *Intelligent Vehicles Symposium (IV), 2011 IEEE*. IEEE, 2011, pp. 528–533.

[4] C. Gundling, "Udacity self-driving car challenge," https://github.com/udacity/self-driving-car/tree/master/steering-models/community-models/cg23, 2016.

[5] Y. Tian, K. Pei, S. Jana, and B. Ray, "DeepTest: Automated testing of deep-neural-network-driven autonomous cars," in *Proceedings of the 40th International Conference on Software Engineering*. ACM, 2018.

[6] M. Bojarski, D. Del Testa, D. Dworakowski, B. Firner, B. Flepp, P. Goyal, L. D. Jackel, M. Monfort, U. Muller, J. Zhang *et al.*, "End to end learning for self-driving cars," *arXiv preprint arXiv:1604.07316*, 2016.