Enhancing the Transferability of Adversarial Attacks for End-to-End Autonomous Driving Systems

Jingyu Zhang[1], Jacky Keung[1], Xiaoxue Ma*[2], Yihan Liao[1], Yishu Li[2], and Yicheng Sun[1]
[1]Department of Computer Science, City University of Hong Kong, Hong Kong, China
{jzhang2297-c, yihanliao4-c, yicsun2-c}@my.cityu.edu.hk
jacky.keung@cityu.edu.hk
[2]School of Science and Technology, Hong Kong Metropolitan University, Hong Kong, China
{kxma, sliy}@hkmu.edu.hk

2024 31st Asia-Pacific Software Engineering Conference (APSEC) | 979-8-3315-3401-1/24/$31.00 ©2024 IEEE | DOI: 10.1109/APSEC65559.2024.00028

***Abstract*—Adversarial attacks play an important role in testing and enhancing the reliability of deep learning (DL) systems. Most existing attacks for DL-based autonomous driving systems (ADSs) demonstrate strong performance under the white-box setting but struggle with black-box transferability, while black-box attacks are more practical in real-world scenarios as they operate without full model access. Numerous transferability-enhancement techniques have been proposed in other fields (e.g., image classification), however, they remain unexplored for end-to-end (E2E) ADSs.**

Our study fills the gap by conducting the first comprehensive empirical analysis of nine transferability-enhancement methods on E2E ADSs, covering two types: three input transformation enhancements and six attack objective enhancements. We evaluate their effectiveness on two datasets with four steering models. Our findings reveal that, out of nine enhancements, Resizing+Translation delivers the best black-box transferability, producing up to 9.39° increase in MAE. Pred+Attn serves as the best objective enhancement, producing a maximum of 5.55° (white-box) and 6.21° (black-box) increase in MAE. Through attention heatmap visualizations, we discover that different models focus on similar regions when predicting, thereby enhancing the transferability of attention-based attacks.

In conclusion, our study provides valuable results and insights into the transferability-enhancement techniques for E2E ADSs, which also serve as a robust benchmark for further advancements in the autonomous driving field.

***Index Terms*—Software Testing, Adversarial Attacks, Cross-model Transferability, Autonomous Driving, Deep Learning**

I. Introduction

The great success of Deep Neural Networks (DNNs) has led to dramatic performance improvement in many autonomous systems. One such domain is autonomous driving, where the current state-of-the-art Autonomous Driving Systems (ADSs) can be categorized into two types: multi-module and E2E [1]–[3]. Multi-modular ADSs decompose the whole system into interconnected modules (e.g., perception, decision). E2E ADSs, on the other hand, combine multiple modules into a single DNN that directly maps sensor data (e.g., RGB, LiDAR) to the numerical control actions such as steering. The simplicity of E2E learning has attracted researchers to develop state-of-the-art models [3]–[5]. Despite their achievements, studies have shown that E2E ADSs are highly vulnerable to maliciously crafted adversarial samples [6]–[9]. A subtle, human-eye-imperceptible adversarial perturbation can cause severe misbehavior, posing serious threats to the reliability and trustworthiness of such safety-critical systems [10]. Recent ADS accidents [11] further highlight the importance of comprehensive testing before commercialization.

Recent research on ADS testing has predominantly focused on generating adversarial and synthetic examples to trigger system misbehavior [6], [12]. Adversarial attacks in E2E ADSs can be generally categorized into white-box and black-box attacks [7]–[9]. White-box attacks involve full model access, while black-box attacks operate with limited query access with no knowledge of model parameters. Both types of attacks have been extensively explored for steering models (E2E ADSs predicting steering control) [6]–[8]. Black-box attacks are comparatively more practical in real-world scenarios due to their minimal information requirement [13], though generating effective adversarial samples remains challenging in such cases [14]. To address this issue, a line of research has proposed attacking a surrogate white-box model in which we know the parameter to generate adversarial samples, and leveraging cross-model transferability to attack black-box models for the same task [15]–[17]. Cross-model transferability (hereafter as transferability) involves using adversarial perturbations generated with a white-box model to attack black-box models [18]. Several studies demonstrate poor transferability of the white-box generated adversarial samples for steering models but did not explore methods to enhance it [15], [19].

Many works have proposed transferability-enhancement techniques and validated their effectiveness on image classification tasks (e.g., ImageNet dataset) [16], [20], [21] and perception tasks (e.g., object detection) [22], [23]. Two commonly used approaches are input transformation enhancements [16], [20] and attack objective enhancements [24], [25]. Input transformation enhancements involve using an ensemble of transformed and original images (e.g., resizing) to generate the adversarial perturbation, thereby preventing overfitting to the white-box model. Attack objective enhancements apply different objectives to generate more transferable adversarial samples, such as disrupting model-agnostic features. However, there is limited research exploring methods to enhance attack transferability and assess the effectiveness of transferability-enhancement techniques on E2E ADSs.

*Corresponding author: Xiaoxue Ma.

2640-0715/24/$31.00 ©2024 IEEE
DOI 10.1109/APSEC65559.2024.00028

Therefore, there are **two main challenges** yet to be solved in the current research: (1) How to improve the transferability of adversarial attacks on the E2E steering regression task? (2) Can the existing transferability-enhancement techniques designed for other tasks be implemented for the E2E steering regression task, and how effective are they? Our work seeks to fill the gap in autonomous driving research by conducting an empirical study on different transferability-enhancement techniques for steering models and identifying the most effective ones. We summarize the following findings:

1) The transformation *Resizing+Translation* achieves the best black-box transferability, producing a 0.28° to 9.39° improvement in MAE compared to the basic framework (baseline). 2) The objective *Pred+Attn* achieves high effectiveness in both black-box and white-box attacks, with increases of up to 5.55° (white-box) and 6.21° (black-box) in MAE compared to the baseline. 3) The objective *Feat+Attn* is not recommended for either attack type, showing a decrease of up to 4.51° (black-box) and 66.09° (white-box) in MAE compared to the baseline. 4) The best transformation *Resizing+Translation* (0.06s) for transferability enhancements takes only half the runtime compared to the best objective *Pred+Attn* (0.11s) on average. Furthermore, based on our results, we also provide insights to pave the way for future advancements in attack transferability within the field.

We summarize **key contributions** of this paper as follows:

- To the best of our knowledge, we undertake the first comprehensive study aimed at systematically assessing the effectiveness of various transferability-enhancement techniques for E2E steering models.
- We identify nine enhancements in total–three input transformations and six attack objectives–applicable to E2E steering models. We evaluate their effectiveness in both white-box and black-box attacks using two datasets and four steering models. For black-box and timely attacks, *Resizing+Translation* is recommended. For high effectiveness in both attack types, *Pred+Attn* is recommended. However, we do not recommend *Feat+Attn* for either attack type.
- We provide insights into the superior performance of these techniques and open-source our implementation at https://github.com/EnhanceRepo/EnhanceTransferability.

II. Related Work

A. Deep Learning in E2E ADSs

With advances in DNNs, E2E ADSs have reached state-of-the-art performance. The history of DNN-based ADSs can be traced back to the 1980s when Pomerleau *et al.* [26] proposed a three-layer network designed for the road following task, dubbed ALVINN. ALVINN takes images from a camera and a laser range finder as input and outputs the travel direction. Subsequently, the E2E behavior cloning for autonomous driving attracted significant interest from academia, prompting researchers to continually develop new architectures and learning procedures to enhance performance [5], [27]. In 2016, the NVIDIA research group introduced Dave2 [27], which trained a convolutional neural network (CNN) to map raw pixels from a single front-facing camera directly to steering commands. This system successfully steers on local roads and highways. At the same time, Udacity Inc. launched the second self-driving car challenge, encouraging participants to design DNN models predicting the appropriate steering angle using images [28]. Teams achieving state-of-the-art performance were ranked on the Udacity leaderboard [1]. Many successful research works have demonstrated the power of the E2E learning approach in DNN-based ADSs, and this evolution continues unabated.

B. Adversarial Attacks in E2E ADSs

Adversarial attacks for E2E ADSs can be broadly categorized into two types: white-box attacks and black-box attacks. In this paper, we utilize white-box attacks to generate perturbation on a surrogate white-box model and leverage the transferability of that perturbation to attack black-box models for the same task. For **white-box attacks**, the attacker has full access to the targeted model and can use its gradient information to craft adversarial perturbations [29], [30]. In 2017, DeepXplore [7] utilized neuron coverage as the objective to perform white-box attacks on steering models. DeepBillboard [6] and PhysGAN [12] leveraged gradient-based white-box attacks and generative models, respectively, to generate adversarial billboard patterns to attack steering models. Von *et al.* used variants of prediction difference as the objective to attack steering models within the BeamNG simulator. Recently, Zhang *et al.* introduced UniAda to generate adversarial examples that can simultaneously attack multiple vehicle controls in a white-box manner.

Black-box attacks involve scenarios where the attacker is limited only to the query access to the model, knowing only the model's output for a given input [31]. For example, DeepRoad [8] implemented a black-box variational-autoencoder-based method that applied various weather transformations to craft adversarial images that mimic real-world weather conditions. The adversarial samples can induce misbehavior in the steering model. Some works also utilized the transferability of white-box-generated adversarial examples to perform black-box attacks. For instance, Deng *et al.* [19] discovered that adversarial examples generated with optimization-based and generative-network-based methods for E2E steering models perform poorly on other black-box models. Zhang *et al.* [15] observed similar results for multi-output E2E ADSs and pointed out several research directions to enhance transferability, however, they did not validate them experimentally.

Overall, many existing research on E2E ADSs has focused on improving the effectiveness of white-box attacks, while enhancements for black-box attacks remain largely underexplored. Notably, there is a lack of research on techniques that enhance the attack transferability on black-box models for E2E ADSs.

[1]https://github.com/udacity/self-driving-car/tree/master/challenges/challenge-2

C. Transferability and its Enhancement

The cross-model transferability of adversarial examples generated with the white-box model can be used to attack black-box models for the same task [17], [20], [23]. Directly transferring the generated adversarial perturbation to black-box models usually results in poor performance due to overfitting to the white-box model [19]. Various methods have been proposed to improve the transferability in the field of image classification. Common methods include but are not limited to applying various input transformations [16], [20], attacking model-agnostic features [32], [33], and attacking model-agnostic semantic properties (e.g., attention) [21], [34]. For example, Xie *et al.* [16] proposed to improve the transferability of adversarial examples by creating diverse input patterns, while Dong *et al.* [20] used a set of translated images to optimize an adversarial sample.

Similarly, in the field of perception, several studies have expanded on these concepts to enhance the transferability for object detection or semantic segmentation tasks [22], [23], [35]. For example, Lu *et al.* [22] and Hashemi *et al.* proposed to enhance transferability by attacking intermediate feature maps for semantic segmentation and object detection tasks.

The proven effectiveness and importance of transferable adversarial samples in other fields encourage us to address the research gap regarding transferability analysis in the steering task for E2E ADSs.

III. Preliminaries

In this section, we define the notations used throughout this paper and formalize the goals and capabilities of attackers to the targeted ADSs. Then, we provide a formulated definition of (1) adversarial attacks on the steering regression task and (2) Transferability-enhancement techniques employed in this paper (Input Transformation Enhancement and Attack Objective Enhancement).

A. Notations and Threat Model

- x_{ori}: the original input image.
- x_{adv}: the generated legitimate adversarial image, such that the pixel values must be within the range [0,1]. We denote as $x_{adv} = Process(x_{ori} + \tau)$.
- x: to represent the general image input.
- m: the white-box model that we craft the adversarial perturbation on. We denote $m(x) \in R$ as the predicted steering angle value with the input x.
- τ: the adversarial perturbation pattern.
- ϵ: the maximum perturbation.

Threat Model: We perform *iterative white-box* gradient-based attacks with specified attack objectives to generate an adversarial sample and use the transferability to conduct *black-box attacks* on other steering models for which we have limited access. Steering models take RGB images as input and output real continuous steering values. Specifically, we intentionally craft a malicious adversarial sample x_{adv} to fool the steering model m into making wrong steering decisions. Our attack is *image-specific* such that we generate a unique τ for each RGB image. The adversarial image x_{adv} is visually similar to the x_{ori} and conforms to the training distribution. Moreover, our attacks are *non-targeted*, in which we do not guide m towards a specified prediction direction. The crafted x_{adv} can induce m to predict a steering value that is either larger or smaller than the original prediction $m(x_{ori})$.

B. Adversarial Attacks

Adversarial attacks in DNNs aim to generate a subtle, human-imperceptible perturbation τ that induces the model to make false predictions. In the steering regression task, the goal of adversarial attacks is maximizing the model prediction difference between the original and the adversarial input [19], expressed as $\text{argmax}_\tau \ |m(x_{adv}) - m(x_{ori})|$ subject to the imperceptibility constraint $\|\tau\|_p \leq \epsilon$. $\|\cdot\|_p$ could be L_1, L_2 or L_∞ norm (we use L_∞) [29]. A common objective to perform white-box gradient-based attacks is prediction-difference loss. We take the absolute value to give equal importance to different error scales:

$$L_{pred}(x; m) = -|m(x_{adv}) - m(x_{ori})| \tag{1}$$

By minimizing the loss, we are maximizing the difference between the prediction on the original and adversarial sample.

C. Transferability-Enhancement Techniques

Input Transformation Enhancement: Input Transformation is one of the most effective approaches to improve attack transferability. Crafting with diverse input patterns, the generated τ is expected to be less sensitive to the discriminative region of the white-box model being attacked [16], [20].

In this paper, we explore two input transformations that are commonly-used in other fields [16], [20], [36], [37]: Random Resizing and Translation. Each transformation is applied to x during the perturbation generation process with the **prediction-difference loss** L_{pred} (Eq. 1) as the attack objective.

(1) Random Resizing [16] first resizes the input image to a random size and then randomly pads zeros around it (i.e., pads back to the original size). The transformation T is applied with probability $p \in [0, 1]$ at each attack iteration. Specifically, we use the transformed image $T(x)$ with probability p and the non-transformed image x with probability $1 - p$. In our experiments, we set $p = 0.5$, following the default settings in [16]. The input is initially randomly resized to $rnd \times rnd \times 3$ with $rnd \in [90, 100)$, and then it is padded back to the size $100 \times 100 \times 3$ with zeros.

(2) Translation operation [20] $T_{ij}(\cdot)$ shifts the image x by i and j pixels along the two-dimensions, respectively. In other words, each pixel (a, b) of the translated image is defined as $T_{ij}(x)_{a,b} = x_{a-i,b-j}$. We set $i, j \in \{-k, ..., 0, ..., k\}$ with $k \leq 10$ for small translations to hold the assumption for fast gradient calculation with kernel matrix. Following [20], we use the default Gaussian kernel with a kernel length of 15 and a radius of 3 for our experiments.

Attack Objective Enhancement: Another line of research attempts to enhance attack transferability by investigating

model-agnostic features and disrupting them using novel loss functions during the perturbation generation process. We consider two well-known objective improvements: Feature-disruptive Loss and Attention-distractive Loss [21], [33]. In our experiments, we generate τ using images **without transformations** and employing different enhancement losses as attack objectives.

(1) Feature-disruptive Loss: Previous studies [24], [25] have revealed that shallow-layer features learned by different models are similar. Based on this observation, literature in both image classification [33] and perception [22], [23] proposed to craft the adversarial sample by maximizing the discrepancy in the intermediate feature between the clean sample and the perturbed sample. Specifically, the goal is achieved by amplifying the less significant channel-wise features while weakening the significant channel-wise features.

$$\begin{aligned} L_{feat}(x;m) = log(\sum_{i,j}\sum_{p\in S}|m_p^l(x)\odot\nabla m_p^l(x)|_{ij}) \\ - log(\sum_{i,j}\sum_{q\in \hat{S}}|m_q^l(x)\odot\nabla m_q^l(x)|_{ij}) \end{aligned} \tag{2}$$

where $\odot$ is the element-wise product, S and $\hat{S}$ represent the channel index set of significant and less significant channel-wise features, respectively (partitioned by channel-wise mean-based strategy described in [33]). $|\cdot|$ is the absolute value function. $m_p^l(x)$ refers to the feature map of p-th channel at the l-th layer of the model. $\nabla m_p^l(x)$ represents the gradient of the predicted steering angle with respect to the input image x at the p-th channel of the l-th layer of the model m. We sum over all (i,j) elements in the feature map and all channels in the index set, and then take the logarithm of it and compute the difference as the loss function. By minimizing the loss function, the generated perturbation causes important channel-wise features to become less important, while less significant features become more important. As suggested in [33], we select the first convolutional layer of each model for l.

(2) Attention-distractive Loss: Most steering models, including those we studied, employ CNN-based architectures to learn and extract local features from input images. Understanding the visual attention of these models is crucial to comprehend which areas they prioritize during predictions [38]. Previous studies have introduced efficient attention-guided adversarial samples in tasks like image classification and object detection. They also demonstrated the transferability effectiveness of the generated adversarial samples [21], [34], [39].

Following common practices [39], [40], we adopt Grad-CAM [38], originally designed for classification problems, to interpret predictions made by steering models. To locate the attention area, we first obtain the activation value A^k (i.e., feature map) of a specific convolution layer (usually the last convolution layer) through forward propagating the input image x. Then, we back-propagate the model prediction signal L_{cam} to A_k and obtain positive contribution regions.

Adapting Grad-CAM for regression tasks involves using the prediction signal L_{cam} instead of softmax values [2]. Specifically, we divide and adjust steering angles y into three ranges by δ. δ is a self-defined small positive threshold, where we select $\delta = 5°$, which is relatively small according to the distribution of steering angle data. In this way, by taking the derivative of L_{cam}, the positive gradients always contribute to the model prediction, regardless of whether the current steering prediction is positive, negative, or nearly zero.

The resulting gradient values $\partial L_{cam}/\partial A_{ij}^k$ are then aggregated via global average pooling to derive the importance weight a_k of the feature map A^k for each channel k. The attention map $h(x,y)$ is then computed as the weighted sum of A^k:

$$L_{cam} = \begin{cases} y, & \text{if } y > \delta \\ -y, & \text{if } y < -\delta \\ \frac{1}{y} \times \text{sign}(y), & \text{otherwise} \end{cases} \Rightarrow \begin{array}{l} a_k = \frac{1}{Z}\sum_i\sum_j \partial L_{cam}/\partial A_{ij}^k, \\ h(x,y) = ReLU(\sum_k a_k A^k) \end{array} \tag{3}$$

We choose the feature map A^k of the last convolutional layer for each model, as suggested in [38]. We employ attention-distractive loss L_{attn} to distract the focus from the original attention map:

$$L_{attn}(x;m) = -\left\| \frac{h(x,y)}{max(h(x,y))} - \frac{h(x_{ori},y_{ori})}{max(h(x_{ori},y_{ori}))} \right\|_1 \tag{4}$$

where $\|\cdot\|_1$ stands for the L_1 norm. Self-normalization is imposed to eliminate the influence of different magnitudes. By minimizing L_{attn}, we maximize the difference between the original attention map $h(x_{ori},y_{ori})$ and the adversarial attention map $h(x,y)$.

IV. Study Design

A. Experimental Procedure

Figure 1 illustrates the framework utilized in our study, containing three workflows. Each workflow is distinguished by color: the basic framework (Workflow 1, red), the input transformation enhancement framework (Workflow 2, green), and the attack objective enhancement framework (Workflow 3, blue).

In the basic framework, we generate adversarial perturbations without employing any transferability-enhancement techniques. We use the non-transformed image to iteratively search for the perturbation τ by attacking the white-box model m with prediction-difference loss minimization (Eq. 1). After that, we apply the optimized τ to the seed image to obtain the adversarial image. Then, we evaluate the white-box and black-box attack effectiveness by computing the corresponding metric. The results of the basic framework are served as the **baseline** in this paper.

When applying the input transformation enhancements (workflow 2, green), we use the transformed image to iteratively search for the perturbation by attacking the white-box

[2]we follow https://jacobgil.github.io/deeplearning/vehicle-steering-angle-visualizations

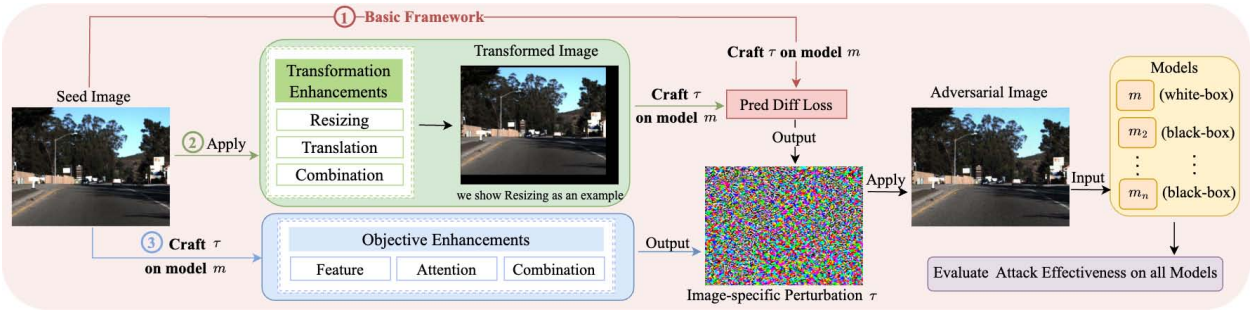

Fig. 1: The framework of adversarial attacks on steering models. Workflow 1: the Basic Framework. Workflow 2: Input Transformation Enhancement. Workflow 3: Attack Objective Enhancement. m denotes the white-box model we craft the perturbation on. $m_2, ..., m_n$ denotes the black-box models being tested for transferability evaluation.

model m with prediction-difference loss minimization. The optimized τ is then evaluated for attack effectiveness.

For attack objective enhancements (workflow 3, blue), we use the non-transformed image to iteratively search for the perturbation with enhanced attack objectives (i.e., feature-disruptive loss (Eq. 2), attention-distractive loss (Eq. 4), and combination losses). After the searching process, we obtain the perturbation τ and evaluate its white-box and black-box attack effectiveness.

Algorithm 1 elaborates the details of the image-specific adversarial perturbation generation process. **Input:** The algorithm requires seven key inputs outlined in the **Require** lines. Specifically, x_{ori} denotes a single input seed image; m is the white-box steering model that we know its parameters and architecture to perform the white-box attack; $lr, \epsilon, iters$ are hyperparameters, their values are listed and explained in the Section IV-C; The loss functions used as the objective to search for the perturbation; T is the input transformation. **Initialization:** Line 1 creates an image copy that requires gradients. Line 2 initializes the perturbation τ as a vector of zeros, with the same shape as the input RGB image. **Perturbation Generation:** For each iteration, it computes the loss l with different workflows (Line 4-10). Note that for Workflow 3, we set the weights of the other two losses as 0 when testing the individual loss. Then, we compute the gradient G and take the sign operator on it (Line 11). Line 12 updates the perturbation via gradient descent, with a maximum perturbation size of ϵ (L_∞ norm) to maintain imperceptibility. Afterward, the image x undergoes updates, subject to processing for legitimacy, for example, the pixel values should be within the range [0,255] (Line 13). **Output:** Upon reaching the maximum number of iterations, the algorithm yields an image-specific human-eye-imperceptible perturbation τ for the seed input x_{ori} (Line 15).

Algorithm 1 Adversarial Perturbation Generation Process.

Require: x_{ori}: the seed image
Require: m: the white-box steering model
Require: lr: the learning rate
Require: ϵ: the maximum perturbation
Require: $iters$: the number of iterations
Require: the loss functions - $\{L_{pred}, L_{feat}, L_{attn}\}$
Require: T: input transformation - {Resizing, Translation, Combination}
1: $x = copy(x_{ori})$
2: $\tau = zeros(x.shape)$
3: **for** i in iters **do**
4: **if** Workflow 1: Basic Framework **then**
5: $l = L_{pred}(x; m)$
6: **else if** Workflow 2: Input Transformation Enhancement **then**
7: $l = L_{pred}(T(x); m)$
8: **else if** Workflow 3: Attack Objective Enhancement **then**
9: $l = \alpha \times L_{pred}(x; m) + \beta \times L_{feat}(x; m) + \gamma \times L_{attn}(x; m)$
10: **end if**
11: $G = \text{sign}(\partial l / \partial x)$
12: Update $\tau \leftarrow Clip_\epsilon(\tau - lr \times G)$
13: Update $x \leftarrow Process(x_{ori} + \tau)$
14: **end for**
15: **return** τ

B. Research Questions

RQ1: *Which input transformation performs the best?*
We compare three input transformations (*Resizing*, *Translation*, *Resizing+Translation*) with *None* (i.e., adopt no input transformation, the baseline). We analyze their performance in both black-box (i.e., transferability) and white-box attacks under two public datasets.

RQ2: *Which attack objective yields the best performance. How does it compare to input transformations?*
We study seven loss functions, including three individual losses (the baseline L_{pred} in Eq. 1, L_{feat} in Eq. 2, and L_{attn} in Eq. 4) and four combinations ($L_{feat} + L_{attn}$, $L_{pred} + L_{feat}$, $L_{pred} + L_{attn}$, and $L_{pred} + L_{feat} + L_{attn}$). We analyze their performance in both black-box and white-box attacks for both datasets.

RQ3: *What is the runtime of studied techniques in generating a single adversarial perturbation?*
We examine the average runtime and its standard deviation of all studied techniques for both datasets for all models.

C. Experimental Setup

Datasets: Our experimentation uses two public datasets that are widely used in autonomous driving research [6], [12]: (1) the Udacity self-driving car challenge dataset [28], which contains 101,396 real-world driving images for training and

5614 images for testing, which is captured by a dashboard-mounted camera of a human-driven car. We use the Udacity test set for experimentation; (2) the Dave testing dataset [41], which contains 45,568 real-world driving images. The images are recorded from a dashcam with labeled steering angles, captured around Rancho Palos Verdes and San Pedro California.

Models: We perform adversarial attacks separately on four normally trained steering models, which have been widely used in autonomous driving testing [6], [7], [9]. Specifically, we employ three models based on the Dave2 architecture (CNN-based, detailed in Section II-A) from NVIDIA, denoted as Dave2V1 [27], Dave2V2 [42], Dave2V3 [43], and one model (Epoch) from the Udacity challenge [4]. Specifically, Dave2V1 is the original model proposed by NVIDIA. Dave2V2 is a variation of Dave2V1, which initializes weights using a truncated normal continuous random variable. Dave2V3 is another variant that adds pooling layers between each convolutional layer. Epoch is a CNN-based steering model that ranked sixth in the final leaderboard of Udacity Self-driving Car Challenge 2 [3].

We employed the PyTorch re-implementation code provided by Von *et al.* [4]. Following the training process provided by the official Udacity GitHub page [5], we trained them on the Udacity train set with Mean Square Error (MSE) as the loss function, defined as $\frac{1}{N}\sum_{i=1}^{N}(m(x)-y)^2$. y denotes the ground truth steering label for the input image x and N denotes the number of images. We evaluated the model performance on the Udacity test set, where Dave2V1, Dave2V2, Dave2V3, and Epoch produce an MAE of 0.022, 0.028, 0.023, and 0.015, respectively.

Evaluation Metrics: To assess the attack effectiveness of the generated examples, we adhere to established practices and employ a widely used metric for the offline evaluation of steering models: Mean Absolute Error (MAE) [6], [7], [9], [44], [45]. Note that since we perform non-targeted attacks (see Section III-A), we use the sign-invariant operator (absolute value) in the metric to avoid incorrect punishment for negative errors (situations where the attacker induces the model predicts a smaller steering than the original one). Let N denote the number of images:

$$MAE = \frac{1}{N}\sum_{i=1}^{N}|m(x_{adv}^{i}) - m(x_{ori}^{i})| \tag{5}$$

A larger MAE indicates better attack effectiveness. To provide a more straightforward presentation of the attack effectiveness in the experiment results, we convert the steering values in MAE from radians to degrees (°) by $\times\frac{180}{\pi}$.

Hyperparameters: The max perturbation ϵ of each pixel in the image is set to 5 for all experiments. Following the optimal setting in Co *et al.* [46], the learning rate lr is set to 0.8, and the number of iterations is set to 3. For the weights in the combination losses, we set $\alpha = 1$ as the reference magnitude for L_{pred}. For β and γ, we conduct a grid search on a small subset of the dataset from candidate values $\beta \in \{0.1, 0.3, 0.5, 0.7, 1\}$ and $\gamma \in \{1e-6, 1e-7, 1e-8, 1e-9\}$, and choose the best combination for experimentation. Small values for γ are selected to ensure three losses can have a similar variance for different inputs [21]. Specifically, we select $\beta = 0.7$ and $\gamma = 1e{-}8$. For the hyperparameters related to the individual enhancements, we adhere to the optimal settings described in the original paper and list the values as introduced in Section III.

[3]https://github.com/udacity/self-driving-car/tree/master/challenges/challenge-2

[4]https://github.com/MissMeriel/DeepManeuver/

[5]https://github.com/udacity/self-driving-car/tree/master/steering-models/community-models

V. Results and Analysis

In this section, we present the results of our experiments to answer RQ1, RQ2, and RQ3. Moreover, we provide insights into the superior performance of the studied transferability-enhancement techniques.

A. RQ1. Input Transformation Enhancement

With L_{pred} as the attack objective, we perform adversarial attacks under three different input transformation enhancements (*Resizing*, *Translation*, and *Resizing+Translation*) on two datasets. We also show the results of no transformation *None* (Workflow 1, Basic Framework) as the baseline for comparison. We craft the adversarial perturbation on each white-box model and test the perturbation on all models. Each cell in Table I indicates the MAE result in degrees. The first column shows the white-box models used to generate the adversarial perturbation, and the first row denotes the model that we test the perturbation on. Larger MAEs indicate better performance.

Overall, we observe a consistent performance trend for both datasets from Table I. For **black-box** attacks, under both datasets, *Resizing+Translation* outperforms all other methods by a decent margin in all cases. Compared to the baseline *None*, it improves transferability by producing a 0.39° to 5.76° (0.28° to 9.39°) increase in MAE for Udacity (Dave). *Translation* achieves the second best results, which outperforms *Resizing* and *None* in all cases, with a 0.34° to 2.85° (0.13° to 4.32°) improvement in MAE compared to the baseline *None* for Udacity (Dave). In 66.6% cases, *Resizing* outperforms the baseline, with the improvement in MAE up to 1.67° (0.13°) for Udacity (Dave). Diving into individual cases, taking the adversarial perturbation crafted on Epoch and test on Dave2V1 (Epoch-Dave2V1) as an example, the *Resizing+Translation* results in an MAE of 7.47° under Udacity, which is 5.76°, 4.09°, and 2.91° larger than the MAEs caused by *None*, *Resizing*, and *Translation*, respectively. For the Dave dataset, taking Epoch-Dave2V3 as an example, the *Resizing+Translation* causes an MAE of 15.18°, where *Translation*, *Resizing*, and *None* only result in MAEs of 7.50°, 5.92°, and 5.79°, respectively.

In short, we suggest applying *Resizing+Translation* if the attacker's goal is to attack black-box models.

	Input Transformation	Udacity				Dave			
		Dave2V1	Dave2V2	Dave2V3	Epoch	Dave2V1	Dave2V2	Dave2V3	Epoch
Dave2V1	None (Basic Framework)	**18.40***	4.11	1.95	1.75	**26.48***	4.57	2.65	3.35
	Resizing	16.37*	4.01	1.95	1.83	23.26*	4.57	2.71	3.42
	Translation	16.39*	5.21	2.29	2.24	24.12*	5.49	2.86	3.48
	Resizing+Translation	17.50*	**5.49**	**2.34**	**2.31**	22.42*	**5.66**	**2.93**	**3.72**
Dave2V2	None (Basic Framework)	3.64	**14.26***	1.93	1.69	5.23	**17.37***	2.45	2.33
	Resizing	3.84	12.43*	2.06	1.83	5.08	14.78*	2.46	2.34
	Translation	4.55	10.89*	2.98	2.64	6.53	13.92*	3.88	3.45
	Resizing+Translation	**6.53**	10.46*	**3.03**	**2.73**	**8.86**	13.46*	**7.41**	**5.03**
Dave2V3	None (Basic Framework)	0.34	0.36	**52.21***	0.58	0.55	0.42	**69.91***	1.54
	Resizing	0.37	0.36	31.11*	0.63	0.54	0.43	49.05*	1.67
	Translation	2.36	2.32	19.61*	3.02	2.80	2.72	31.51*	5.86
	Resizing+Translation	**4.05**	**2.43**	15.91*	**3.11**	**3.40**	**4.05**	31.72*	**7.75**
Epoch	None (Basic Framework)	1.71	2.98	4.87	**16.21***	3.96	2.75	5.79	**35.98***
	Resizing	3.38	2.98	4.90	14.32*	3.86	2.80	5.92	31.70*
	Translation	4.56	4.18	5.69	13.06*	5.46	3.91	7.50	26.33*
	Resizing+Translation	**7.47**	**4.54**	**6.06**	12.68*	**7.77**	**10.26**	**15.18**	31.64*

TABLE I: MAE results (in degrees) of Input Transformation Enhancement (with L_{pred} as the attack objective). The superscript * indicates the white-box attack, and the others are black-box attacks (i.e., transferability). We highlight the results of the best technique in bold.

Finding 1: For black-box attacks, across two datasets, *Resizing+Translation* achieves the best performance. Compared to the baseline, the improvement in MAE ranges from 0.28° to 9.39°. *Translation* produces the second best results, with the improvement in MAE ranging from 0.13° to 4.32°. Only 66.6% of the time *Resizing* outperforms the baseline.

For **white-box** attacks, the baseline *None* performs the best in all cases across both datasets compared to the input transformation enhancements. In 6 out of 8 cases, *Resizing* outperforms *Translation*. *Resizing+Translation* shows the poorest attack effectiveness in 5 out of 8 cases. Specifically, the MAE of the baseline can reach up to 52.21° (69.91°) for Udacity (Dave). *Resizing* demonstrates a decrease in MAE compared to the baseline, ranging from 1.83° to 21.1° (2.59° to 20.86°) for Udacity (Dave). Similarly, the decrease in MAE under *Translation* ranges from 2.01° to 32.6° (2.36° to 38.4°) for Udacity (Dave). We observe the same phenomenon in *Resizing+Translation*, with 0.9° to 36.3° (3.91° to 38.19°) decrease in MAE under Udacity (Dave). With input transformations, the adversarial perturbation is crafted based on an ensemble of transformed and non-transformed images, which may tend to underfit the white-box model parameters and result in inferior white-box performance. Therefore, we suggest using no transformation for white-box attacks.

Finding 2: All transformation methods degrade the white-box attack performance compared to the baseline *None*, with a decrease in MAE ranges from 0.9° to 38.4°.

B. RQ2. Attack Objective Enhancement and its Comparison with Input Transformation

We use seven losses separately to craft adversarial perturbation on each of the four white-box models and test the generated perturbation on all models. The MAE results (in degrees) are reported in Table II. The first column shows the white-box models that we craft the perturbation on. The first row denotes the model that we test the perturbation on. Specifically, the diagonal blocks (numbers with asterisks) under each dataset indicate the white-box attack performance and others indicate the black-box attack performance. Larger MAEs indicate better performance. For simplicity, we use *Pred*, *Feat*, *Attn* to denote L_{pred} (Workflow 1), L_{feat}, and L_{attn}, respectively.

For **white-box** attacks, *Pred+Attn* demonstrates the best effectiveness in 6 out of 8 cases, reaching up to 5.55° improvements over the baseline in MAE. We observe that, in 7 out of 8 cases, generating the perturbation with *Feat* or its combination loss (*Feat+Attn* or *Pred+Feat*) results in the worst white-box performance. Especially, *Feat+Attn* leads to a surprisingly poor white-box performance in many cases, such as Dave2V3-Dave2V3 under Udacity. For transferability on **black-box** models, compared to the baseline, *Pred+Attn* reaches the best performance in 18 out of 24 cases, achieving 0.08° to 6.21° improvements. *Feat*, *Feat+Attn*, and *Pred+Feat* do not produce outstanding performance in most cases for both datasets. Specifically, in 13 out of 24 cases, *Feat+Attn* outputs the worst performance on black-box attacks, producing a maximum decrease of 4.51°.

Finding 3: For both white-box and black-box attacks, we suggest using *Pred+Attn* as the attack objective, producing a maximum of 5.55° (white-box) and 6.21° (black-box) improvements in MAE compared to the baseline. We do not suggest using *Feat+Attn* for either attack type.

We then compare the best input transformation *Resizing+Translation* for transferability enhancement with the best objective *Pred+Attn*. For black-box transferability, the former outperforms the latter in 21 out of 24 cases, with an average of

	Attack Objective	Udacity Dave2V1	Udacity Dave2V2	Udacity Dave2V3	Udacity Epoch	Dave Dave2V1	Dave Dave2V2	Dave Dave2V3	Dave Epoch
Dave2V1	Pred (Basic Framework)	18.40*	4.11	1.95	1.75	26.48*	4.57	2.65	3.35
	Feat	7.10*	2.65	1.54	1.35	14.09*	3.83	2.46	2.18
	Attn	18.82*	4.29	1.78	1.81	6.11*	4.15	**3.87**	3.23
	Feat+Attn	2.24*	1.28	1.64	1.24	8.10*	3.01	2.56	2.83
	Pred+Feat	9.62*	3.34	1.63	1.16	16.49*	5.27	2.93	2.88
	Pred+Attn	**23.95***	**4.71**	**2.25**	**2.10**	**27.94***	**6.37**	3.20	3.46
	Pred+Feat+Attn	17.80*	4.34	2.10	1.92	27.72*	6.15	3.09	**4.94**
Dave2V2	Pred (Basic Framework)	3.64	14.26*	1.93	1.69	5.23	17.37*	2.45	2.33
	Feat	2.39	5.06*	1.77	1.21	2.29	3.08*	2.22	2.42
	Attn	**4.08**	13.97*	1.84	1.74	2.49	6.87*	3.06	2.90
	Feat+Attn	0.88	2.61*	1.20	0.80	2.07	3.68*	1.31	0.92
	Pred+Feat	1.24	2.40*	1.41	0.84	1.19	3.45*	2.71	2.18
	Pred+Attn	3.85	**15.31***	**2.15**	**3.30**	**7.48**	**22.68***	**5.19**	**3.93**
	Pred+Feat+Attn	3.04	4.82*	1.13	1.14	3.87	11.06*	2.80	1.82
Dave2V3	Pred (Basic Framework)	0.34	0.36	**52.21***	0.58	0.55	0.42	**69.91***	1.54
	Feat	0.43	0.36	30.26*	0.61	0.47	0.40	36.08*	0.61
	Attn	0.37	0.39	50.02*	0.57	0.29	0.32	22.97*	1.23
	Feat+Attn	0.43	0.29	2.64*	0.63	0.39	0.43	3.82*	**2.08**
	Pred+Feat	0.46	0.45	50.00*	0.74	0.61	**0.62**	55.66*	0.89
	Pred+Attn	**0.75**	**0.46**	47.15*	0.66	**0.72**	0.50	63.12*	1.90
	Pred+Feat+Attn	0.42	0.39	43.81*	**0.82**	0.59	0.48	53.19*	1.86
Epoch	Pred (Basic Framework)	1.71	2.98	4.39	16.21*	3.96	2.75	5.79	35.98*
	Feat	1.78	2.19	3.02	8.28*	2.70	2.57	4.45	19.82*
	Attn	1.98	2.86	4.07	16.10*	0.75	1.53	2.38	3.36*
	Feat+Attn	0.43	0.80	0.91	1.67*	0.66	1.40	1.28	1.84*
	Pred+Feat	4.91	3.15	4.20	12.75*	5.17	4.28	7.95	33.29*
	Pred+Attn	**6.06**	**3.38**	**4.89**	**17.34***	**6.91**	**7.17**	**12.00**	**39.83***
	Pred+Feat+Attn	5.32	3.28	4.43	13.91*	6.00	5.45	7.62	38.95*

TABLE II: MAE results (in degrees) of Attack Objective Enhancement (with NO input transformation). The superscript * indicates the white-box attack, and the others are black-box attacks (i.e., transferability). We highlight the results of the best technique in bold.

1.61° improvement in MAE. In white-box attacks, *Pred+Attn* consistently outperforms Resizing+Translation, achieving an improvement of 12.69° in MAE on average.

Finding 4: On average, *Resizing+Translation* outperforms *Pred+Attn* in black-box attacks, showing a 1.61° increase in MAE. Conversely, *Pred+Attn* outperforms *Resizing+Translation* in white-box attacks, demonstrating a 12.69° increase in MAE.

C. RQ3. Runtime Efficiency Analysis

In RQ1 and RQ2, we have validated that adversarial samples generated with both types of enhancements demonstrate superior performance over the baseline for black-box transferability. To implement techniques into practice, efficiency is also an important factor to consider. For example, to realize real-time attacks, the attacker needs to timely generate an effective perturbation when encountering the seed input. With this consideration, we compute the average runtime (in seconds) and the standard deviation (std) of crafting a single adversarial perturbation on each of the four models for the Udacity dataset for all studied techniques. We put the results for Dave on GitHub due to limited space and similar runtime.

In Table III, for all models, the basic framework (baseline) takes the shortest time to generate an adversarial perturbation for the input image, with only 0.01s on average and a tiny std. *Feat+Attn* and *Pred+Feat+Attn* take similar and the longest time, taking approximately 0.4s to generate a single perturbation. Their std is also higher than other techniques. Among the four models, Epoch takes the longest runtime in 60% cases, which may be attributed to its deeper architecture.

The best input transformation for black-box transferability *Resizing+Translation* takes only 0.06s on average to generate an adversarial sample, whereas the best objective *Pred+Attn* takes 0.11s, nearly double the time. Therefore, it is expected that *Resizing+Translation* can perform more timely attacks in real-world applications.

Finding 5: The best transformation *Resizing+Translation* (0.06s on average) for transferability demonstrates a nearly half runtime than the best objective *Pred+Attn* (0.11s).

D. Insights of Superior Performance

Based on the results from RQ1 and RQ2, we present insights into the superior performance produced by different transferability-enhancement techniques.

1) Input Transformation Enhancements: **Black-box Superiority:** With input transformation, an adversarial sample is crafted with an ensemble of images composed of the original one and its transformed versions. The generated adversarial sample will be less correlated to the gradient of the white-box

Method	Basic	Resize	Trans	Resize+Trans	Feat	Attn	Feat+Attn	Pred+Feat	Pred+Attn	Pred+Feat+Attn
Dave2V1	0.01±0.002	0.01±0.002	0.05±0.010	0.05±0.008	0.04±0.001	0.09±0.012	0.39±0.020	0.05±0.002	0.09±0.010	0.37±0.029
Dave2V2	0.01±0.002	0.02±0.001	0.04±0.011	0.05±0.008	0.04±0.002	0.09±0.010	0.38±0.020	0.04±0.002	0.09±0.011	0.38±0.027
Dave2V3	0.01±0.001	0.01±0.001	0.03±0.009	0.03±0.007	0.03±0.002	0.08±0.006	0.37±0.023	0.03±0.001	0.10±0.005	0.36±0.011
Epoch	0.01±0.001	0.01±0.001	0.11±0.005	0.11±0.003	0.03±0.002	0.14±0.016	0.41±0.017	0.03±0.002	0.14±0.012	0.42±0.017

TABLE III: The average runtime (in seconds) of perturbation generation and its standard deviation for Udacity.

(a) Dave2V1 ori pred (b) Dave2V2 ori pred (c) Epoch ori pred

(d) MAE=44.8°,ϵ=5.0 (e) MAE=15.0°,ϵ=5.0 (f) MAE=9.60°,ϵ=5.0

Fig. 2: Attention Heatmap on Udacity sample image.

model at the given input, which is expected to enhance the black-box transferability performance.

2) Attack Objective Enhancements: From the experiments, we identify *Pred+Attn* as the best attack objective for both white-box and black-box attacks. To explore the insights, we explore the relationship between the model's attention area and the model's prediction (i.e., can adversarial attacks change the model's attention?). We implement the baseline white-box attacks for simplicity and visualize attention heatmaps with and without attack in Figure 2.

White-box superiority of *Pred+Attn*: As shown in Figure 2a, Dave2V1 focuses on the pixel region around the front car when making steering predictions from the original image. However, its attention shifts to the lower-right region when predicting an adversarial image (Figure 2d). We can see that for each white-box attack, distracting the model's attention is complementary to distracting its prediction. Therefore, joint optimization of *Pred+Attn* achieves the best performance in white-box attacks 75% of the time.

Black-box superiority of *Pred+Attn*: In Figure 2a, 2b, 2c, all three models concentrate around the front car area for making steering predictions. The similarity in concentrated areas potentially encourages the black-box transferability of adversarial samples generated by distracting the white-box model's attention region. Our results in black-box attacks also prove this point.

VI. Threats to Validity

Internal Validity: Threats to internal validity can be attributed to (1) the reproducibility of studied methods implemented on the E2E ADSs. To alleviate this, we carefully follow the publicly available code provided by the authors of original papers and open-source our implementation to facilitate checking; (2) Effective manipulation of hyperparameters. To alleviate this, we follow the optimal hyperparameter configurations described in the original papers. For the weights in the combination objectives, we perform a comprehensive grid search with a sufficient number of candidate values (Section IV-C) on a small subset of the dataset to determine the optimal hyperparameter values.

External Validity: Threats to external validity to our experiment conclusions include (1) the selected datasets. We alleviate this by selecting two widely used datasets in autonomous driving research [6], [7], [12]. They cover various real-world driving scenarios, such as driving in urban towns, highways, and under different weather conditions.; (2) The selected E2E ADSs. We alleviate this by selecting four ADSs that reach competitive performance in the E2E steering angle prediction task [3], [4], [42], [43]. They are constructed with different architectures, ranging from 6 million (Dave2) to 33 million (Epoch) parameters. Therefore, our experimental conclusions are expected to be generalized to other real-world driving datasets and models.

VII. Conclusion

In this paper, we undertake an empirical study on transferability-enhancement techniques for E2E ADSs by examining three input transformations and six objective enhancements. Our findings indicate that Resizing+Translation performs the best in black-box attacks but shows degraded white-box performance. Using Pred+Attn as the attack objective improves both attack types. Additionally, our runtime efficiency assessment also demonstrates that Resizing+Translation takes only half the runtime of Pred+Attn, making it more practical for real-world black-box attacks. In future work, we plan to validate techniques' effectiveness in the real physical world.

References

[1] E. Yurtsever, J. Lambert, A. Carballo, and K. Takeda, "A survey of autonomous driving: Common practices and emerging technologies," *IEEE access*, vol. 8, pp. 58443–58469, 2020.

[2] C. Urmson, J. Anhalt, D. Bagnell, C. Baker, R. Bittner, M. Clark, J. Dolan, D. Duggins, T. Galatali, C. Geyer, *et al.*, "Autonomous driving in urban environments: Boss and the urban challenge," *Journal of field Robotics*, vol. 25, no. 8, pp. 425–466, 2008.

[3] NVIDIA, "Nvidia-autopilot-keras.," 2016. *https://github.com/0bserver07/Nvidia-Autopilot-Keras*.

[4] C. Gundling, "Steering angle model: Cg32.," 2016. *https://github.com/udacity/self-driving-car/tree/master/steering-models/community-models/cg23*.

[5] F. Codevilla, E. Santana, A. M. López, and A. Gaidon, "Exploring the limitations of behavior cloning for autonomous driving," in *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 9329–9338, 2019.

[6] H. Zhou, W. Li, Z. Kong, J. Guo, Y. Zhang, B. Yu, L. Zhang, and C. Liu, "Deepbillboard: Systematic physical-world testing of autonomous driving systems," in *Proceedings of the ACM/IEEE 42nd International Conference on Software Engineering*, pp. 347–358, 2020.

[7] K. Pei, Y. Cao, J. Yang, and S. Jana, "Deepxplore: Automated whitebox testing of deep learning systems," in *proceedings of the 26th Symposium on Operating Systems Principles*, pp. 1–18, 2017.

[8] M. Zhang, Y. Zhang, L. Zhang, C. Liu, and S. Khurshid, "Deeproad: Gan-based metamorphic testing and input validation framework for autonomous driving systems," in *Proceedings of the 33rd ACM/IEEE International Conference on Automated Software Engineering*, pp. 132–142, 2018.

[9] M. von Stein, D. Shriver, and S. Elbaum, "Deepmaneuver: Adversarial test generation for trajectory manipulation of autonomous vehicles," *IEEE Transactions on Software Engineering*, 2023.

[10] F. Tambon, G. Laberge, L. An, A. Nikanjam, P. S. N. Mindom, Y. Pequignot, F. Khomh, G. Antoniol, E. Merlo, and F. Laviolette, "How to certify machine learning based safety-critical systems? a systematic literature review," *Automated Software Engineering*, vol. 29, no. 2, p. 38, 2022.

[11] B. C. News, "Fatal crash on SB I-680 onramp in San Jose.," 2022. *https://www.kron4.com/news/bay-area/fatal-crash-on-sb-i-680-onramp-in-san-jose/*.

[12] Z. Kong, J. Guo, A. Li, and C. Liu, "Physgan: Generating physical-world-resilient adversarial examples for autonomous driving," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 14254–14263, 2020.

[13] N. Papernot, P. McDaniel, I. Goodfellow, S. Jha, Z. B. Celik, and A. Swami, "Practical black-box attacks against machine learning," in *Proceedings of the 2017 ACM on Asia conference on computer and communications security*, pp. 506–519, 2017.

[14] F. Croce and M. Hein, "Reliable evaluation of adversarial robustness with an ensemble of diverse parameter-free attacks," in *International conference on machine learning*, pp. 2206–2216, PMLR, 2020.

[15] J. Zhang, J. W. Keung, Y. Xiao, Y. Liao, Y. Li, and X. Ma, "Uniada: Universal adaptive multiobjective adversarial attack for end-to-end autonomous driving systems," *IEEE Transactions on Reliability*, 2024.

[16] C. Xie, Z. Zhang, Y. Zhou, S. Bai, J. Wang, Z. Ren, and A. L. Yuille, "Improving transferability of adversarial examples with input diversity," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 2730–2739, 2019.

[17] Y. Liu, X. Chen, C. Liu, and D. Song, "Delving into transferable adversarial examples and black-box attacks," in *International Conference on Learning Representations*, 2017.

[18] Z. Jin, J. Zhang, Z. Zhu, and H. Chen, "Benchmarking transferable adversarial attacks.," *CoRR*, 2024.

[19] Y. Deng, X. Zheng, T. Zhang, C. Chen, G. Lou, and M. Kim, "An analysis of adversarial attacks and defenses on autonomous driving models," in *2020 IEEE international conference on pervasive computing and communications (PerCom)*, pp. 1–10, IEEE, 2020.

[20] Y. Dong, T. Pang, H. Su, and J. Zhu, "Evading defenses to transferable adversarial examples by translation-invariant attacks," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 4312–4321, 2019.

[21] S. Chen, Z. He, C. Sun, J. Yang, and X. Huang, "Universal adversarial attack on attention and the resulting dataset damagenet," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 44, no. 4, pp. 2188–2197, 2020.

[22] Y. Lu, H. Ren, W. Chai, S. Velipasalar, and Y. Li, "Time-aware and task-transferable adversarial attack for perception of autonomous vehicles," *Pattern Recognition Letters*, vol. 178, pp. 145–152, 2024.

[23] A. S. Hashemi, A. Bär, S. Mozaffari, and T. Fingscheidt, "Improving transferability of generated universal adversarial perturbations for image classification and segmentation," in *Deep Neural Networks and Data for Automated Driving: Robustness, Uncertainty Quantification, and Insights Towards Safety*, pp. 171–196, Springer International Publishing Cham, 2022.

[24] J. Yosinski, J. Clune, Y. Bengio, and H. Lipson, "How transferable are features in deep neural networks?," *Advances in neural information processing systems*, vol. 27, 2014.

[25] J. Hu, L. Shen, and G. Sun, "Squeeze-and-excitation networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 7132–7141, 2018.

[26] D. A. Pomerleau, "Alvinn: An autonomous land vehicle in a neural network," *Advances in neural information processing systems*, vol. 1, 1988.

[27] 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.

[28] Udacity, "Using Deep Learning to Predict Steering Angles.," 2016. *https://medium.com/udacity/challenge-2-using-deep-learning-to-predict-steering-angles-f42004a36ff3*.

[29] I. J. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and harnessing adversarial examples," *stat*, vol. 1050, p. 20, 2015.

[30] A. Kurakin, I. J. Goodfellow, and S. Bengio, "Adversarial examples in the physical world," in *International Conference on Learning Representations Workshop*, 2017.

[31] N. Narodytska and S. P. Kasiviswanathan, "Simple black-box adversarial perturbations for deep networks," *arXiv preprint arXiv:1612.06299*, 2016.

[32] G. Lin, Z. Pan, X. Zhou, Y. Duan, W. Bai, D. Zhan, L. Zhu, G. Zhao, and T. Li, "Boosting adversarial transferability with shallow-feature attack on sar images," *Remote Sensing*, vol. 15, no. 10, p. 2699, 2023.

[33] D. Wang, W. Yao, T. Jiang, and X. Chen, "Improving transferability of universal adversarial perturbation with feature disruption," *IEEE Transactions on Image Processing*, 2023.

[34] D. Lang, D. Chen, R. Shi, and Y. He, "Attention-guided digital adversarial patches on visual detection," *Security and Communication Networks*, vol. 2021, no. 1, p. 6637936, 2021.

[35] A. M. Staff, J. Zhang, J. Li, J. Xie, E. A. Traiger, J. A. Glomsrud, and K. B. Karolius, "An empirical study on cross-data transferability of adversarial attacks on object detectors.," in *AI-Cybersec@ SGAI*, pp. 38–52, 2021.

[36] J. Lin, C. Song, K. He, L. Wang, and J. E. Hopcroft, "Nesterov accelerated gradient and scale invariance for adversarial attacks," *arXiv preprint arXiv:1908.06281*, 2019.

[37] J. Zou, Z. Pan, J. Qiu, X. Liu, T. Rui, and W. Li, "Improving the transferability of adversarial examples with resized-diverse-inputs, diversity-ensemble and region fitting," in *European Conference on Computer Vision*, pp. 563–579, Springer, 2020.

[38] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra, "Grad-cam: Visual explanations from deep networks via gradient-based localization," in *Proceedings of the IEEE international conference on computer vision*, pp. 618–626, 2017.

[39] W. Wu, Y. Su, X. Chen, S. Zhao, I. King, M. R. Lyu, and Y.-W. Tai, "Boosting the transferability of adversarial samples via attention," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 1161–1170, 2020.

[40] T. Xiang, H. Liu, S. Guo, Y. Gan, W. He, and X. Liao, "Towards query-efficient black-box attacks: A universal dual transferability-based framework," *ACM Transactions on Intelligent Systems and Technology*, vol. 14, no. 4, pp. 1–25, 2023.

[41] Sully-Chen, "Autopilot-Tensorflow," 2016. *https://github.com/SullyChen/Autopilot-TensorFlow*.

[42] J. Gildenblat, "Visualizations for understanding the regressed wheel steering angle for self-driving cars.," 2016. *https://github.com/jacobgil/keras-steering-angle-visualizations*.

[43] A. Staravoitau, "Behavioral cloning: end-to-end learning for self-driving cars.," 2016. *https://github.com/navoshta/behavioral-cloning*.

[44] D.-H. Lee, K.-L. Chen, K.-H. Liou, C.-L. Liu, and J.-L. Liu, "Deep learning and control algorithms of direct perception for autonomous driving," *Applied Intelligence*, vol. 51, no. 1, pp. 237–247, 2021.

[45] S. Hecker, D. Dai, and L. Van Gool, "Failure prediction for autonomous driving," in *2018 IEEE Intelligent Vehicles Symposium (IV)*, pp. 1792–1799, IEEE, 2018.

[46] K. T. Co, L. Muñoz-González, L. Kanthan, B. Glocker, and E. C. Lupu, "Universal adversarial robustness of texture and shape-biased models," in *2021 IEEE International Conference on Image Processing (ICIP)*, pp. 799–803, IEEE, 2021.