applied sciences

MDPI

Article

AT-BOD: An Adversarial Attack on Fool DNN-Based Blackbox Object Detection Models

Ilham A. Elaalami *, Sunday O. Olatunji and Rachid M. Zagrouba

College of Computer Science and Information Technology, Imam Abdulrahman Bin Faisal University, P.O. Box 1982, Dammam 1441, Saudi Arabia; Osunday@iau.edu.sa (S.O.O.); rmzagrouba@iau.edu.sa (R.M.Z.)
* Correspondence: ilhamalami78@gmail.com

Abstract: Object recognition is a fundamental concept in computer vision. Object detection models have recently played a vital role in various applications, including real-time and safety-critical systems such as camera surveillance and self-driving cars. Scientific research has proven that object detection models are prone to adversarial attacks. Although several proposed methods exist throughout the literature, they either target white-box models or specific-task black-box models and do not generalize on other detectors. In this paper, we proposed a new adversarial attack against Blackbox-based object detectors called AT-BOD. The proposed AT-BOD model can fool the single-stage and multi-stage detectors, where we used an optimization algorithm to generate adversarial examples depending only on the detector predictions. AT-BOD model works in two diverse ways, reducing the confidence score and misleading the model to make the wrong decision or hide the object detection models. Our solution achieved a fooling rate of 97% and a false negative increase of 99% on the YOLOv3 detector, and a fooling rate of 61% false-negative increase of 57% on the Faster R-CNN detector. The detection accuracy of YOLOv3 and Faster R-CNN under AT-BOD was dramatically reduced and reached $\leq$1% and $\leq$3%, respectively.

Keywords: adversarial attacks; adversarial examples; black-box attacks; deep neural networks; object detection models

Citation: Elaalami, I.A.; Olatunji, S.O.; Zagrouba, R.M. AT-BOD: An Adversarial Attack on Fool DNN-Based Blackbox Object Detection Models. *Appl. Sci.* **2022**, *12*, 2003. https://doi.org/10.3390/app12042003

Academic Editor: Byung-Gyu Kim

Received: 25 November 2021
Accepted: 4 January 2022
Published: 15 February 2022

1. Introduction

In recent years, scientists and researchers have devoted themselves to developing artificial intelligence methods, believing them to solve problems that were previously difficult to explain with traditional programming methods. Deep learning (DL) algorithms have played an active role in the development of various fields. This has included computer vision, which has witnessed significant progress whereby many new technologies have been introduced and are currently being used on the ground, such as self-driving cars (SDCs) [1]. This type of application is considered a highly safety-critical system because of its danger to individuals' lives if they are wrongly exploited and misused [2]. SDCs almost primarily depend on computer vision, which depends on artificial intelligence, and more precisely deep learning, to perform object recognition and determine the components of the surrounding environment and respond instantaneously according to the existing conditions [3]. Of course, the self-driving car developers aim to develop an integrated system that almost performs the same as the real driver and may even surpass it. However, there are many concerns, but security and safety are crucial, and the problem is that it is difficult to predict what might happen in a real-time situation [4]. Figures 1 and 2 show the perception system framework and the integration of deep neural network (DNN) models.

However, and importantly as a cybersecurity issue, it has been found that DL models are susceptible to practical adversarial attacks in which by making small perturbations into the input sample, it can be easy to fool a DL model [5]. Therefore, in the case of SDCs, it is very crucial to understand the behavior of the algorithms, especially given the emergence

of attacks against deep learning models [6]. When adversarial attacks first appeared [7], the focus was on creating other types of attacks that were initially directed to image classification models, thereby declaring deep learning models' weakness. Most of these adversarial examples use malicious inputs, which are a modification of the essential parts of the data specific to the DNN, and consequently, the systems behave unexpectedly [8,9].

Artificial intelligence (AI) and machine learning (ML) technologies must be analyzed and tested against possible risks before integrating them into critical systems such as SDCs. Technically, attacks on such orders can occur at two levels: training and production [10]. Most of the ML-based products need to periodically retrain their models, especially applications that contain continuous data collection systems and behavior analysis such as SDCs, social networks and weather systems, etc. Attacking the ML-based systems in SDCs can be placed with different methods [11] depending on the attacker's goals and the ML model phase. The attack can also be direct, especially the targets the image-based model, for example, whereby an adversary can deceive a SDC using toxic sign [12], by adding a small patch/sticker on a road sign which could fool the AI system [13], without using any custom code or script. Here, we conclude that AI presents a new challenge to cybersecurity [14]. AI security breaches rise from the fact that the model is based on data collection where it learns patterns and then makes a decision. Securing data in AI-based systems with traditional data security techniques for non-AI-based systems is not enough where ML models need to periodically ingest new data entries and reconstruct the model parameters while learning new behaviors. The pipeline between data sources and the ML algorithm must be opened to have a useful model, especially for safety-critical systems such as SDCs. This opened point between the data and the ML algorithm is a severe vulnerability if it is exploited by an adversary and can lead to significant consequences. In DL models that are extensively used in SDCs, the developer usually does not understand how and why the model selects the latent space, which imposes another uncontrollable risk. It might be challenging to detect the intrusion.

DL-based object detection models have received extensive attention in order to improve accuracy and speed for real-time applications [15], but they are also vulnerable to adversarial examples [16,17]. By exploiting the weakness of DNNs, specifically convolutional neural networks CNNs, the adversarial attacks have been extended from classification tasks to targeting object detection models. For example, Xie et al. [10] have extended this type of attack to DNN-based semantic segmentation and object detection. Their observation suggests that the target of these models' is a classification problem of multiple objects in an input image depending on object proposal in the case of object detection instead of only one class in the classic classification task.

There are many adversarial attacks applied against computer vision systems in which each has different behavior in how the victim model is influenced. However, there two main adversarial attack types implemented in the literature, adversarial perturbation, and adversarial patch attacks [18]. CNNs can be misled by adding small perturbations to the input samples that are called adversarial examples and can be generated by gradient-based algorithms [19], for instance, Fast Gradient Sign Method (FGSM) [20] and Projected Gradient Descent (PGD) [21] or by using iterative methods such as DeepFool [22] and C&W attack [23]. Most of these proposed methods are pixel-wise additive noise that generates adversarial examples by adding small noise to the original examples. However, these methods tend to be infeasible when applied to real-world perception systems such as SDCs.

To overcome this issue, we introduce AT-BOD, a new hybrid Blackbox attack targeting different detector types. The proposed AT-BOD is based on two types of adversarial attacks. The first attack is an adversarial perturbation based Blackbox attack using the Zeroth-order adaptive momentum (ZO-AdaM) algorithm that is known for solving the Blackbox optimization problem [24]. We adjust the algorithm to target the classification component of the object detection model using an iterative method. For the second attack, we use the adversarial patch attack introduced by Thys et al. [25] but in our work, we target different detectors, and we used two different datasets, OpenImages- Vehicles, a

custom dataset where we take a subset of the massive Google OpenImages dataset, and Udacity self-driving cars dataset. The attacks are performed targeting the YOLOv3 and Faster R-CNN detectors. The detection rate is dramatically reduced under AT-BOD attacks where the detection has two behaviors, whether making wrong decisions or making the detector blind.

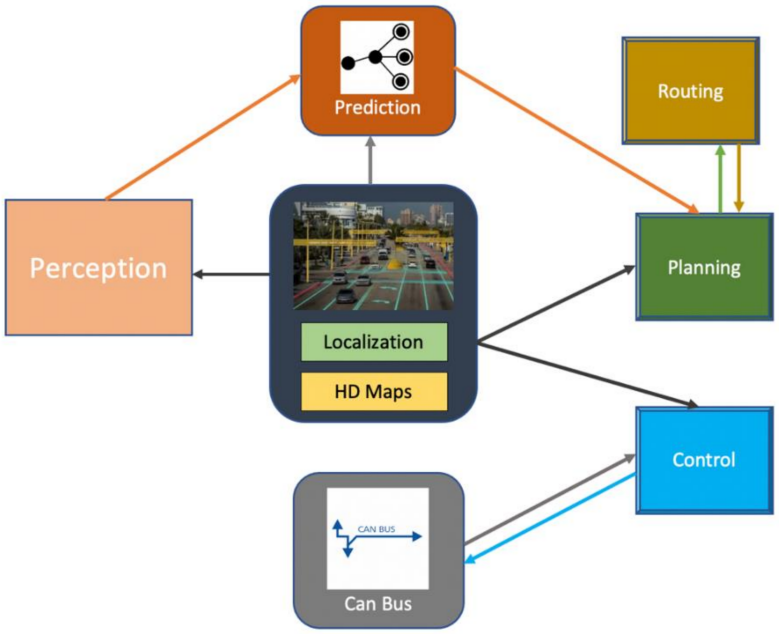

Figure 1. The general framework of the perception system in SDCs.

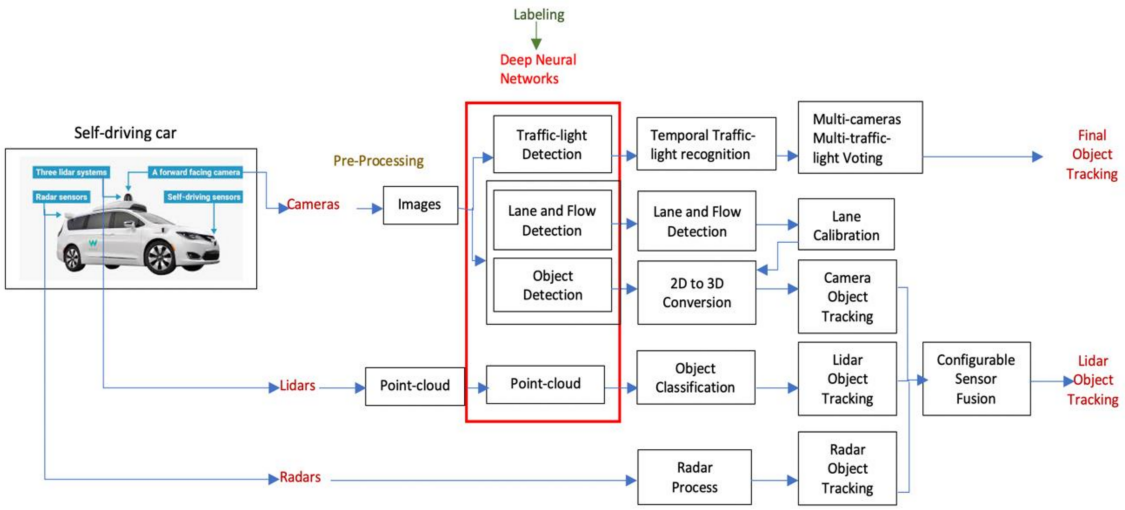

Figure 2. The perception system in SDCs indicating the integration of DNNs that perform object detection tasks.

Contributions

(1) We develop a new attack framework that targets the DL-based detectors, based on two types of adversarial attacks;
(2) The first attack is an adversarial perturbation based Blackbox attack using the zeroth-order adaptive momentum [24] algorithm that is known for solving the Blackbox optimization problem. We adjust the algorithm to target the classification component of the object detection model using an iterative method;

(3) In the second attack, we use the adversarial patch attack introduced by Thys et al. [25]. However, in our work, we used a different dataset, OpenImages-Vehicles, a custom dataset where we take a subset of the massive Google OpenImages dataset then label the images using LabelImg annotating tool to generate bounding boxes coordinates for each object in the input samples. The detector victims are YOLOv3 and Faster R-CNN, that are used to detect vehicles in the road;
(4) The attacks are performed separately, targeting the YOLOv3 and Faster R-CNN detectors, where they work in three diverse ways, reducing the confidence score of vehicle classes, decreasing the objectness score, or both;
(5) The detection rate is dramatically reduced under both attacks where the detection has two behaviors, whether making wrong decisions or making the detector blind.

2. Background

2.1. Definitions and Notations

A dataset or space sample with size N is denoted as $\mathcal{S}$, where $\mathcal{S} = \{x_i, y_i\}_{i=1}^{N}$, x_i represents a data sample and y_i is its label. A deep neural network (DNN) is defined as $f(\cdot)$, where x is the input and $f(x)$ is the output. x' is a data sample that represents an adversarial example of x generated under a distance Δ where $x \approx x'$, but $f(x) \neq f(x')$. The distance metric $\Delta(\cdot, \cdot)$ is the distance constraint that controls the added perturbation ε where x' is an adversarial example of x with a distance Δ. The function $f(x)$ is linear and vulnerable to small Δx

$$x' \leftarrow \Delta(x, x') < \varepsilon. \tag{1}$$

Such that ε is a well-chosen value such that x and x' are similar and $(x') \neq y$.

A perturbed input expressed as Δx such that the output increased by $\Delta x||W||_p$ under an l_p-norm where p is a real number, commonly p takes one of these values $\{-1, 0, 1, 2, \inf\}$.

$$||W||_p = (|w_1|^p + |w_2|^p + \ldots + |w_n|^p)^{\frac{1}{p}} \tag{2}$$

$f(x)$ learns the label y using optimization function, also called optimization loss, $J(W, x, y)$, where J minimizes the difference between $f(x)$ and y [23].

2.2. Adversarial Attacks

Adversarial attacks are techniques that generate adversarial examples capable of fooling machine learning models to make the wrong decisions by modifying the original input and adding well-constructed perturbations.

In the adversarial threat models, there are three main types of adversarial attacks: Whitebox, Gray-box and Blackbox. The three threat models differ according on how much an adversary knows about the target network. Whitebox is when adversaries have prior knowledge and full access to the target model. They are aware of the concrete architecture, all of the parameters, weights configurations, and probably even the training details. Hence, they can tailor attacks and generate adversarial examples. In the Gray-box attack, adversaries have access of the target model structure but lacks access to its weights. In this threat model, the adversary can create adversarial examples using the same architecture of the target model. Under the Blackbox model, an adversary does not have access to the structure or parameters of the target network. Instead, it can query the results of the DL victim model to find predictions for particular inputs. The transfer learning is used to apply Blackbox attack [26,27].

Adversarial examples can be formed as follows:

Let $f(x)$ be a DNN such that $(x) = W^T x$ where x is the input sample with label y, and W is the model weights with high dimensionality.

An input sample x' is considered as an adversarial example of sample x if x' is very close to x under a distance δ with a constrained value ε such that $f(x') \neq y$.

Blackbox Attack

In this type of attack, an adversary is assumed to have no prior knowledge of the model's internal structure but can access the output confidence score or label for a given input sample. The most successful and used method in this type of attack is the transferability of adversarial examples generated by another model. Transferability is when an input that can be adversarial to a model f', it possibly can be adversarial to another model f too, where f and f' share the same training data [27]. Papernot et al. [28] used data augmentation based on the Jacobian method generating new input samples to train a network iteratively. They performed a Whitebox attack on a chosen f' model and used the generated and successful adversarial example to perform a Blackbox attack on another model f. Papernot et al. targeted five classifiers using Blackbox attack, logistic regression, SVM, decision trees, k-nn, NNs, and ensemble networks. Adversarial examples generated by NNs-based classifiers have a high transferability. The ensemble networks have good transferability too, where Liu et al. [29] performed a Blackbox attack by training several different classifiers to generate adversarial examples that work on an ensemble network. They were able to create transferable targeted and untargeted adversarial examples. The gradient estimators or query-based searches are other attack approaches whereby oracle queries are used to determine a small number of pixels that are vulnerable to change and result in a significant change in the target confidence score [30] or by estimating the gradient of the victim model by calculating the difference between the real and adversarial losses [31,32].

All of the the attacks mentioned above use small perturbations that are mostly imperceptible for human sight. Adversarial patches are another type of attack that adds perturbations but in a restricted segment of the input sample. The adversarial patches are locally placed in the input image and able to fool DNNs models. Sharif et al. [33] used this trick to add crafted eyeglasses and attach them to face images, which was capable of fooling a CNN model based on VGG architecture for face recognition. The authors were able to apply the attack on the physical world by printing the perturbed eyeglass frames using a 3D printer. Trojan patch is another type proposed by Liu et al. [34] where they exploit a small set of neurons that can successfully affect the network decision. Chosen pixels in the patch location are maximized to fire the preselected set of neurons. Weights are tuned by retraining original images and images with Trojan patches; thus, the model is attacked and started making incorrect decisions. Brown et al. [35] were able to generate universal adversarial patches by optimizing the adversarial loss that targets three components, the distance from the original image, transformations applied on the patch, and the location of the patch.

Goodfellow et al. [5] proposed the linearity hypothesis. They concluded that adversarial examples work because DNNs tend to become more linear rather than non-linear as high dimensional dot product operations are applied while the network propagates. However, highly non-linear machine learning models are more robust than adversarial examples but are more complicated in training. They do not perform smoothly in a baseline non-adversarial setting.

2.3. Adversarial Attacks on Self-Driving Cars

Due to the absence of human control direct monitoring, SDCs are threatened to be subjected to any type of attack. Various adversarial examples targeting DNNs-based perception have been applied in the scope of SDCs. Adversarial attacks on stop signs by adding well-constructed perturbations or small patches on real images can easily deceive the perception model and result in incorrect predictions. Hence, adversarial attacks against SDCs are possible to apply using simulator tools such as the Carla simulator provided by Udacity, Appolo by Baidu, and OpenPilot by comma.ai. However, numerous studies tackle the problem of adversarial attacks on perception models but not those directly applied to SDCs environments or simulators that are very important for interpreting how the system components interact with each other and with the data sources. Rather, simulators

are more reliable in explaining how attacks can exploit and influence perception models. Numerous studies tackle the problem of adversarial learning and provide valuable insights into adversarial attacks and defenses. Still, most of them were not directly related to SDCs rather than giving a comprehensive explanation of how adversarial attacks can exploit ML models. Other studies focused on SDCs but did not profoundly deal with adversarial attacks against SDCs perception system, rather providing reviews on SDCs components and behavior, which is vital to understand how adversaries can exploit them. Burton et al. [36] studied SDCs from a safety perspective, mentioning the DNNs-based perception model's ineffectiveness in front of adversarial examples. However, they did not directly deal with adversarial attacks. Tuncali et al. [37] used the Sim-ATAV simulator to implement an SDC system using the KITTI dataset [38], where they apply the SqueezeNet algorithm that resulted in enhancing the system reliability. Sitawarin et al. applied multiple adversarial attacks on SDCs such as out-of-bound attack and lenticular printing, providing explanations for these attacks but not how to defend them. The physical adversarial attacks take place and prove their applicability, Eykholt et al. [39] used this type of attack to deceive the camera in a car by painting roads with signs crafted to mislead the system in the vehicle. Boloor et al. [40] used deconvolutional ConvNets to generate physical adversarial examples that can decrease autonomous vehicles' performance. Physical adversarial examples are easier to apply and more practical compared to attacking SDCs using digital adversarial examples. Lu et al. [41] analyzed adversarial attacks in a real-time system. In their study, they used GTSRB and MS-COCO datasets, and their results have shown valuable insights on whether adversarial examples can affect the SDCs environment. It is considered one of the few papers that emphasizes adversarial defenses that require more attention, regardless of adversarial attacks' efficiency.

The availability of more self-driving car annotated datasets collected from real scenes, such as Udacity and OLIMP [42], will aid and encourage information security researchers to study feasible attacks.

3. Related Work

Recently, some attacks targeting object detection models have been released. By exploiting the weakness of DNNs against adversarial examples, Xie et al. [13] have extended this type of attack to DNN based semantic segmentation and object detection. Their observation that these models' target is a classification problem of multiple objects in an input image depending on object proposal in the case of object detection instead of only one class in the classic classification task. The attack named DAG was basically based on optimization to generate adversarial perturbations. DAG aims to produce a group of adversarial examples that have to succeed in attacking many detectors with a high chance of transferability within a wide range of detectors with different architecture and data. However, the authors stated that their attack is effectively transferable on networks with the same architecture using heterogeneous perturbations. DAG suffers from a critical problem where selecting the proper set of targets is needed, which is problematic in the object detection task due to the dependency of the detector on region proposals and its ability to select different proposals when the perturbations are added to the input images. DAG was implemented on Faster R-CNN with two different backbones, ZFNet and VGG16 trained on PascalVOC-2007 and PascalVOC-2007 combined with PascalVOC-2012. Applying DAG on the object detection model was hard where they need to adjust the number of the NMS by choosing 0.90 IoU rate, which is high, to reduce the number of generated proposals. DAG needs about 50 iterations to generate the appropriate adversarial perturbations with a maximum number of iterations of 150 for object detection with a failure of 1% of the images.

Wang et al. [43] proposed an adversarial model that can generate images with deformations and occlusions and those which are hard to classify by an object detection model. Depending on the scarcity of data, that has some specific occlusions and deformations such as images of cars' deformations resulted from accidents captured from different angles in different situations, they decided to build a model that produces close-up pictures of

images that rarely happen, which makes their number within the dataset very little, and therefore the detection model may fail to define this type of images. Hence, we see that the authors took advantage of the need for the model to train in a large number of images to properly work and correctly identify the images. The new model generates examples by hiding some parts of the image, hence obstructing some of the feature maps. The idea of generating examples in the feature space instead of the pixels since it is difficult to achieve; this also depends on the adversarial network, which has to learn the features where the original detector might fail. The model victim was Fast R-CNN, and the results were on the PASCAL VOC dataset using AlexNet and VGG16 architectures using two approaches, adversarial spatial dropout for occlusion. The Fast R-CNN is then trained on the adversarial examples, and the full model accuracy improved and reached 73.6%.

Lu et al. [41] studied existing physical attacks against the autonomous vehicle's detection models, where they affirmed that this type of model is not easy to be attacked. At the same time, angles, distances and illuminations keep changing.

Lu et al. [44] claim that there is no proof that the physical adversarial example that fools classifiers can fool object detection where no such attack can remain the effect on different distances, angles, and scales. They concluded that if a physical attack exists, it must be formed, considering all possible positions of view and detector parameters such as scaling, shifting, and image illumination. Thus, the existing physical attacks are more likely to work in specific circumstances.

Lu et al. [17] proposed a method that generates adversarial examples able to fool Faster R-CNN with transferability on YOLO9000. The attack can affect the detectors, whether by mislabeling the object or missing it. The attack can be used physically under certain circumstances with more massive perturbation as needed in digital attacks. The authors stated that producing examples need.

Li et al. [45] proposed R-AP to attack the object detection based on region proposal network (RPN). R-AP aims to distract the RPN by optimizing the loss function that results in wrongly predicting an object's existence using a label loss. Furthermore, if an object is identified, it can affect the bounding box prediction by changing the object shape using shape loss. However, the R-AP is targeting only the RPN-based models, such as the R-CNN family.

Chen et al. [46] introduced an attack called shapeshifter to fool Faster R-CNN detector in the physical world. The authors affirm that attacking object detection models becomes more challenging where it needs more than simply fooling the classification on multiple bounding boxes but with different scales. Shapeshifter is a Whitebox attack where the attacker has a pre-knowledge on the model architecture, components, and parameters. The attacker must have the ability to interpret the output and gradients. However, the perturbations are not constructed in real-time. The attack is imperceptible to humans and customized on the stop sign, iterative, and based on the Expectation over Transformation technique.

Liu et al. [47] proposed D-patch, a Blackbox attack against YOLO and Faster R- CNN detectors. D-patch is mainly attacking both the bounding box regression and the object labeling; thus, it affects the final prediction of the detection model. The attack performs 2 types of attacks, targeted and untargeted with high performance and mAP of Faster R-CNN and YOLO dramatically decreasing from 75.10% and 65.7% to less than 1%, respectively.

4. The Proposed Model

In this section we introduce our adversarial attack against Blackbox object detection that we called it AT-BOD.

In this work, we implement an adversarial attack that consists of two different types of attacks, perturbation-based and patch-based against YOLOv3 and Faster R-CNN object detection models in Blackbox settings. The attack's primary goal is to make the detector miss the real object, reduce its mAP, and increase the false positives. As it is a Blackbox, we

only have access to the victim models' inputs and outputs without any prior knowledge or access to the models' structure, parameters, and weights.

4.1. Patch-Based AT_BOD

The patch-based AT-BOD can be formulated as an additive perturbation constrained in the input sample's local area. We used the patch attack proposed by Thys et al. [25]. In [25] work, the attack targets the persons using the optimization problem in order to decrease the confidence score of the model. In our work, as shown in Figure 3, we are targeting vehicles, and the victim model is YOLOv3. To force the detector to ignore the vehicle object, we have three options, minimizing the likelihood of being a vehicle in the classification task, minimizing the probability of an object exists (objectness score), or both. Minimizing the class probability results in detecting the object with a different class. In the other two approaches, the vehicle objects are hidden, and the detector becomes blind.

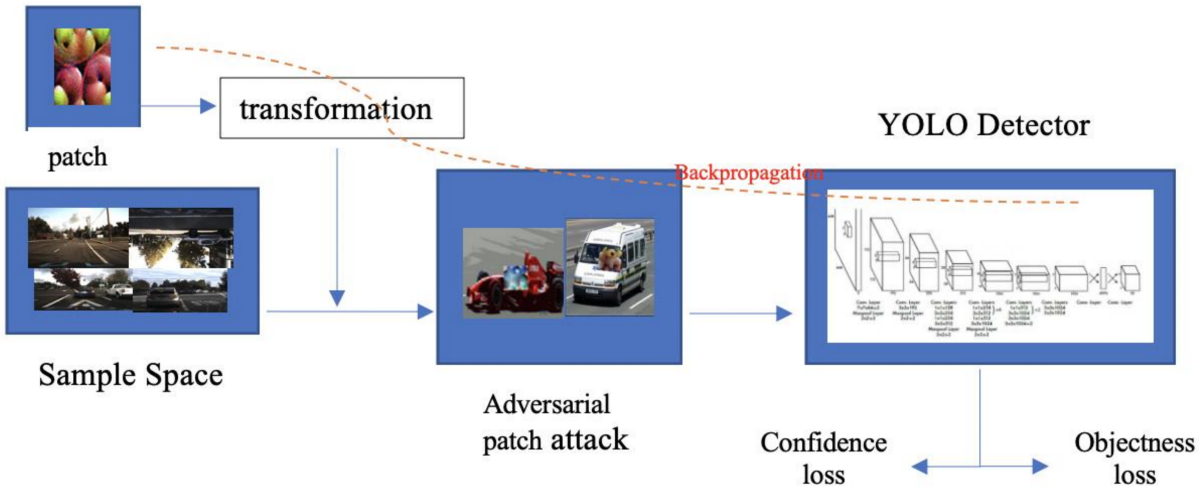

Figure 3. Generating the Patch-based AT-BOD framework.

4.1.1. Problem Definition

The problem was defined as follows: in the sample space $\mathcal{S}$, an input image χ, find an adversarial example (AE) χ' that can mislead the object detection model M, such that $\chi' \in \mathcal{S}$ and χ and χ' are very similar. Let us assume that $(\mathrm{C}_i, \mathrm{P}_i)$ is the image ground truth where i is an object in the real image input and C_i is the class label and P_i is the position such that M detects i for a given χ and outputs (c_i, p_i) while for a given χ', M detects i and outputs (c_i', p_i'). Which means that (c_i, p_i) is replaced by (c_i', p_i'), making the object i to be ignored in χ' or replaced by i' where i' is a patch.

4.1.2. Adversarial Patch Generation

In our work, we used the patch attack proposed by Thys et al. [25]. In [25] work, the attack targets the person using the optimization problem to decrease the model confidence score. The optimization is based on three components the non-printability score, the total variation O_v and the maximum objectness score O_b in the image. Since we are implementing the patch attack digitally, we discarded the non-printability score. Our goal is attacking the detector such that the target object is hidden from the detector using the objective function $\mathcal{O}$ as shown in (2). The total loss $\mathcal{O}$ is scaled using the value α and optimized using Adam optimizer.

$$\mathcal{O}_v = \sum_{i,j} \sqrt{((p_{i,j} - p_{i+1,j})^2 + (p_{i,j} - p_{i,j+1})^2} \tag{3}$$

$$\mathcal{O} = \alpha \mathcal{O}_V + \mathcal{O}_b \tag{4}$$

4.2. Perturbation-Based AT-BOD

4.2.1. Problem Definition

For a given legitimate input x_i to a CNN model, the adversarial example x' is a modified input such that x' is very close to x_i but the CNN output of x' is radically different from x_i for example if x_i is an image that contains a car object with label l_i and x' is an image for the same object with small noise, the CNN detects x' as a deferent class l' e.g., 'bowl' instead of 'car' where $l' \neq l_i$. The modification on x_i to get x' can be conducted by adding an amount of perturbation such that $x' = x_i + \delta$ and the distortion level is usually measured using $l_p - norm$ where $p \geq 0$ of the additive perturbation δ, often l_1, l_2 *and* l_∞ are used. The adversarial perturbation is imperceptible to human vision when the added distortion is minimal, but it is able to deceive the CNN model which increases troubles in safety-critical systems. Hence, it becomes crucial to have robust models against adversarial examples. Commonly, adversarial examples are custom-built through Whitebox settings in which an adversary has prior knowledge of the internal structure of the victim model and full access of its parameters including the model weights, one example of this type is generating adversarial example by optimizing the loss function, such as cross-entropy, performing backpropagation over the model's layers with directions to turn the input with additive perturbation. However, the majority of deployed CNN models (e.g., perception system in SDCs) are Blackbox where only the model output (e.g., label or confidence score) can be accessed of a queried input. Therefore, how adversarial examples can be generated to attack a CNN through Blackbox settings?

This can be obtained by using the constrained and unconstrained minimization problem to find x to minimize $f(x)$.

$$minF(x') = d(x',\ x) - \delta(\mathrm{M}(x')) \tag{5}$$

$$Where\ d\ is\ distance\ between\ x\ and\ x'\ under$$

$$l_2 - norm\ and\ \delta\ is\ the\ adversarial\ condition$$

$$such\ that\ 0\ for\ successful\ attack,\ 1\ otherwise$$

4.2.2. Zeroth-Order Adaptive Momentum Method (ZO-AdaMM)

Zeroth-Order (ZO) is a gradient-free optimization method used for solving Blackbox optimization problems in which gradient expressions are infeasible to apply and difficult to update [48].

In our experiment, we used ZO-AdaMM introduced by Chen et al. [24] to generate universal perturbations [49] from Blackbox models. ZO-AdaMM is an iterative method used to generate adversarial examples with a small perturbation ε that can mislead the model to misclassify the target class. ZO-AdaMM is a gradient-free method on uniform random vector based gradient estimate. The main reason behind choosing ZO-AdaMM is its better convergence accuracy comparing to the other ZO methods such as ZO-SCD [50] and ZO sign-based SGD [51].

ZO-AdaMM solves black-box optimization problems by integrating AdaMM [52] by AMSGrad [53], a variant of Adam, with the random gradient estimator $\hat{\nabla} f_t(X_t)$. The ZO gradient estimator of a function f is formed as the forward difference of two function values at a random unit direction:

$$\widehat{\nabla} f(x) = \left(\frac{d}{\mu}\right)[f(x + \mu u) - f(x)]u \tag{6}$$

In general, u represents a random vector drawn uniformly from the sphere of a unit ball, while $\mu > 0$ is a small step size, determines the smoothing parameter. Throughout existing works, the random direction vector u has been derived from a standard Gaussian

distribution. As a result of the uniform distribution, the ZO gradient estimate (6) is defined in a bounded space instead of the whole real space that Gaussian can compute. However, one of the key requirements for ZO-AdaMM's convergence analysis is that random gradient estimates are bounded. In spite of the biased nature of the ZO gradient estimate (6), it is unbiased when referring to the gradients of the randomized smoothing versions of f with parameter μ.

$$f_\mu(x) = \mathbb{E}_{u\sim U_B}[f(x+\mu u)] \tag{7}$$

$\mathbb{E}_{u\sim U_B}$ represents the uniform distribution over n-dimensional Euclidean sphere.
The stochastic optimization problem can be represented in the form:

$$\underset{x\in X}{min} f(x) = \mathbb{E}_{\xi}[f(x;\xi)\ ] \tag{8}$$

where $x \in \mathbb{R}$ is the n-dimensional optimization variable, X is the closed convex set, f is a black-box loss function, and ξ is a random variable that records environmental uncertainties and $f_\mu(\mathrm{x}) = \mathrm{E}_{u\sim U_B}[f(x;\xi)]$. In formula (8), assuming ξ conforms to a uniform distribution verified by observations on $\{\xi_i\}_{i=1}^n$, then we obtain a finite sum formula $f(x) = \frac{1}{n}\sum_{i=1}^n f(X;\xi_i)$.

The ZO-AdaMM in Algorithm 1 is obtained by integrating AdaMM with the random gradient estimator (6). The square root, square, maximum, and division operators are elementwise combined.

The authors in ZO-adaMM algorithm, assumed that the initial values of $m = 0$, $v_0 = 0$, and $0^0 = 1$ by convention, and $\hat{g}_t = \hat{\nabla} f_t(X_t)$ where $\hat{\nabla} f_t(X_t)$ is the ZO gradient estimator of f and $f_t(X_T)\cdot f(X_t;\xi_t)$. The Mahalanobis distance-based projection was used in ZO-AdaMM instead of the Euclidean projection to avoid the non-convergence issue [24].

After generating the adversarial examples, we built an adversarial network model that sends these examples along with original images to the targeted object detection model, then we calculate the adversarial loss. The model runs iteratively to reduce the loss using l_2 distance.

Algorithm 1: ZO-AdaMM Algorithm

Input: xi ∈ X, step sizes {αt} T t = 1, β1,t, β2 ∈ (0, 1],
1: Initialize: $m_0 \leftarrow 0$, $v_0 \leftarrow 0$ and $v^0 \leftarrow 0$
2: for t = 1, 2, . . . , T do
3 let gˆt = ∇ˆ ft(xt) by (1),
4: ft(xt): = f(xt; ξt)
5: mt = β1,tmt−1 + (1 − β1,t)gˆt
6: vt = β2vt−1 + (1 − β2)gˆ 2 t
7: vˆt = max(vˆt−1, vt), and Vˆ t = diag(vˆt)
8: xt+1 = Π X, $\sqrt{}$ Vˆ t (xt − αtVˆ −1/2 t mt)
9: end for

4.2.3. Generating Perturbations

We implement a Blackbox attack against an object detection model. The attack is able to decrease the mAP value of detector that measures the detection accuracy among dataset classes over different recalls. Since we are implementing a Blackbox attack, it means that we do not know the inner network structure and parameters including the model target weights. Hence, it is not possible to use gradient-based iterative calculations to generate adversarial examples. In order to apply the attack, perturbations transferability is adopted.

Finding adversarial examples able to attack a network over Blackbox settings can be solved by formulating it as a ZO optimization problem. In our work, the victim model is YOLOv3, a regression-based detector uses an end-to-end neural network that divides the image into region of interest and then predicts the bounding boxes alongside with a confidence score that presents the probability of the predicted bounding box. The model

takes the entire image and divides it into a grid of size 13×13, each cell in the grid predicts 5 bounding boxes that can be described as coordinates of a rectangle encloses an object. We formulate the problem of finding adversarial examples using ZO-ADAMM explained in Section 4.2.2.

There are two types of perturbations: per-image and universal [49]. The universal perturbation describes how flexible an adversarial attack is for different inputs. Adversarial attacks mostly have individual perturbation which means that the perturbation is applicable only to a particular source. For other sources, we recalculate the attack and receive a different perturbation. For example, if an attacker is able to bypass a facial recognition system, we will need to create glasses for each attacker, meanwhile the perturbation is universal, we will be able to create glasses to spoof the facial recognition system, and no matter who is wearing them. In our work we generate the universal perturbation using Algorithm 2 below.

Algorithm 2: Universal perturbation algorithm

Input: data sample X, CNN model F, l_2-norm perturbation ε, adversarial example v
Output: universal perturbation vector v
1: Initialize: $v \leftarrow 0$
2: for each datapoint x in data sample X do
3: If $F(x+v) = F(x)$ then
4: Calculate the minimal perturbation v that makes $x + v$ goes to decision boundaries
5: Update v
6: end if
7: end for
8: return v

As shown in Figure 4, after generating the adversarial examples, we built an adversarial network model that sends these examples along with original images to the targeted object detection model, then we calculate the adversarial loss. The model runs iteratively to reduce the loss with the smallest perturbation.

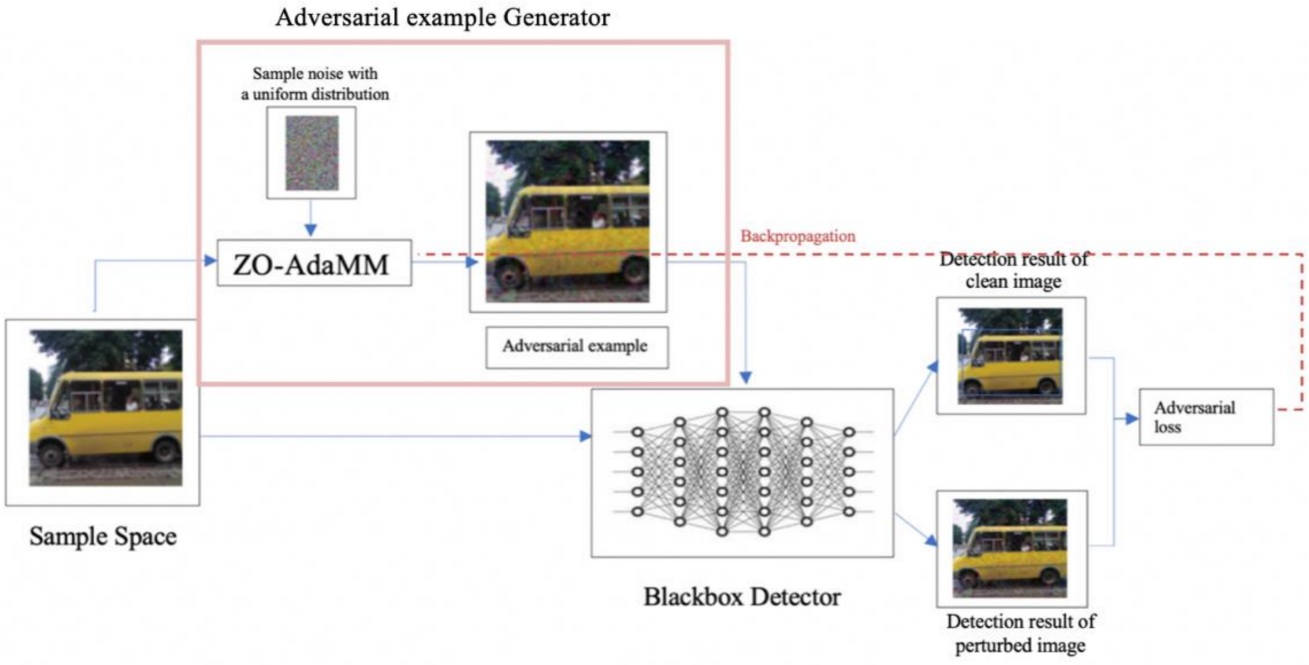

Figure 4. Generating the perturbation-based AT-BOD framework.

4.3. Evaluation Metrics

4.3.1. Metrics for Object Detection

Mean Average Precision (mAP)

Which detection is accurate and performs better? There are multiple tools where each returns a list of objects containing data about object location in the image, its size, and the confidence score indicating what class label is probably the object belongs. Depending on the detection task, that could be one of two categories, detection of a particular object (for example, president Obama face) or detecting a different object of a predefined category list such as cars, pedestrians, stop signs, etc. The main problem we might face using a naïve approach such as binary scoring is the object localization that most of time is imperfectly indicating the bounding box shape where it might happen that a part of the object is out of the box. This approach failed because it depends on the exact coordinates defined in the ground truth to measure the detection and generate the score, whether 1 for matches and 0 otherwise. To obtain a reasonable measurement, we need to have a function that calculates how close is the detected box to the true box. However, we can rely on precision and recall that are frequently used in classification tasks.

Hence, evaluating object detection is a non-trivial task due to its complexity where we need to determine the objectness and the localization, which means we have two tasks to measure classification and regression. Usually, the data used in object detection has many classes with a non-uniform distribution that might introduce biases using simple metrics. Furthermore, we have the problem of misclassification. Thus, associating each generated bounding box with the detected object confidence score and evaluating the model is important [54].

Average Precision (AP)

AP is a metric used in evaluating the accuracy of object detection models. It calculates the average precision for recall over $[0,\ 1]$. The AP score is defined as the mean precision at the set of 11 equally spaced recall values, Recall $i = [0,\ 0.1,\ 0.2,\ \ldots,\ 1.0]$ [54].

$$AP = \frac{1}{11} \sum_{recall_i} Precision(Recall_i) \tag{9}$$

The precision at each recall level r is interpolated by taking the maximum precision measured for a method for which the corresponding recall exceeds r [54].

$$P_{interp}(r) = max_{\widetilde{r}:\widetilde{r} \geq r} p(\widetilde{r}) \tag{10}$$

Such that $p(\widetilde{r})$ is the computed precision at recall $\widetilde{r}$ [55].

Precision computes the percentage of how is accurate the model prediction.

$$Precision = \frac{TP}{TP + FP} \tag{11}$$

Recall calculates the percentage of how the model is useful in finding all positives.

$$Recall = \frac{TP}{TP + FN} \tag{12}$$

Intersection over Union (IoU)

IoU is the overlapping of the ground truth bounding box and the prediction bounding box to measure how much is the prediction covers the real object. Usually, we predefine an intersection threshold to classify the prediction as true or false positive.

$$IoU = \frac{Area\ of\ Overlap}{Area\ of\ Union} \tag{13}$$

where,

Area of overlap is the shaded area in  .

And the area of union is the shaded area in  .

4.3.2. Attacks Evaluation Methods

Fooling Rate

Fooling rate (FR) is a measurement that indicates the percentage of samples that the model detects incorrectly after applying an adversarial attack.

$$FR = \frac{\sum_{i=1}^{N} \rho(label\ before\ attack(i)\ \neq\ (label\ after\ attack(i))}{N} \quad (14)$$

where N is the total number of samples, an attack is tested ρ is a function that returns 1 if the statement is true otherwise 0.

False-Negative Increase

The adversarial attack aims to compromise system integrity. Integrity can be lost by degrading the performance of the victim model where it causes an increase in false negatives and false positives, this type of attack we call it adversarial falsification. There are two attacks under adversarial falsification, false-positive attack or type I error that generates negative examples that are detected as positive ones, and false-negative attack or type II error that produces a positive sample but detected as negative and this type of attack is known in machine learning as evasion.

4.4. Dataset

We used two datasets, the vehicles-OpenImages dataset, and the Udacity self-driving car dataset.

OpenImages is a dataset released by Google under a CC BY4.0 License and released under an Apache 2 license. OpenImages dataset is made mainly for computer vision and detection purposes. It was open-sourced in 2016, where it contains a huge number of images and their bounding box annotations for image classification and object detection. We used the bounding boxes annotations that span 600 object classes to create our custom dataset that contains images for vehicles only with 850 images and 1636 annotations. The annotation distribution is shown in Figure 5. We used the OIDv4_ToolKit that enables us to specify the classes and the number of images we want.

Udacity SDC is an open-source dataset provided by Udacity that contains images for driving scenes. The dataset has 2versions, one for small images and the other for large image where each set has 15,000 images with a size of 512 × 512, 64,000 annotations. CrowdAI was used for annotation. As shown in Figure 6, the dataset contains 11 classes (car, pedestrian, truck, biker, trafficLight, trafficLight-red, trafficLight-green, trafficLight-redLeft, trafficLight-greenLeft, trafficLight-yellow, trafficLight-yellowLeft). The class names distributions are shown in Figure 6. Udacity SDC dataset is collected from real-world driving scenes and can be used on the Udacity Carla simulator. CARLA is an open-source code and protocols that provides digital assets such as cars, roads, and buildings. Besides, it offers a collection of sensors and simulates various environmental conditions, generates maps, and more.

We applied the attacks on the test dataset. Vehicles' appearance differs in color, size, position then there is no fixed spot to localize the patch on the vehicle.

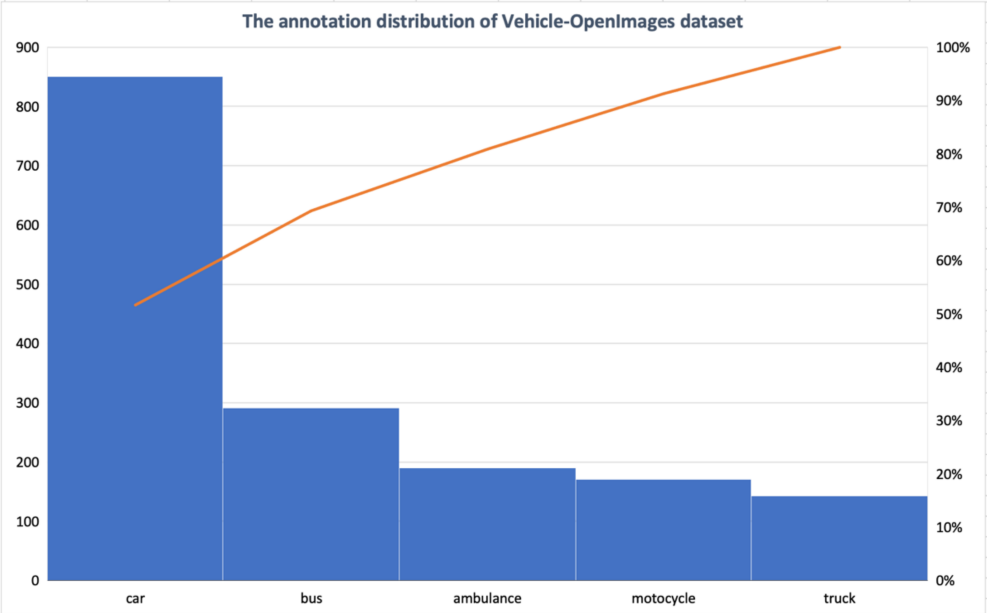

Figure 5. The annotation distribution of the Vehicle-OpenImages dataset.

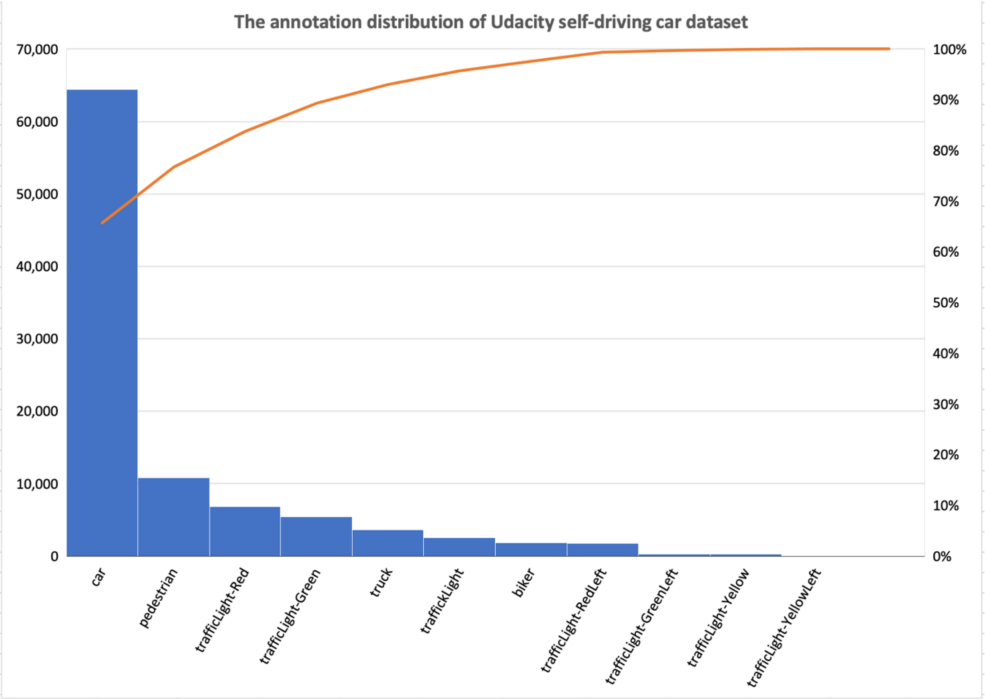

Figure 6. The annotation distribution of the Udacity self-driving car dataset.

4.5. The Victim Models

Our target models are Faster R-CNN that solves the detection problem as a region-based model, and YOLOv3 that solves detection as a regression problem.

4.5.1. Faster R-CNN

Faster R-CNN is a two-stage detector that uses region proposal maps (RPNs) [56]. As shown in Figure 7, the role of RPNs is taking feature maps generated using a CNN model and generating a collection of object proposals where each of these proposals has an output, we call it the objectness score.

Generally, Faster R-CNN works in the following steps:

- By using a pretrained CNN, that in our experiment was ResNet50_fpn [57] pre-trained on COCO, we retrained the last layer of the network depending on the number of classes we want to detect;
- After obtaining all regions of interest in each image, we reshape them to the same shape of the used CNN inputs;
- We used a classifier such as an SVM to classify both image's objects and background where each class has a trained binary classifier;
- For each object in the input image, we generate bounding boxes using a linear regression model.

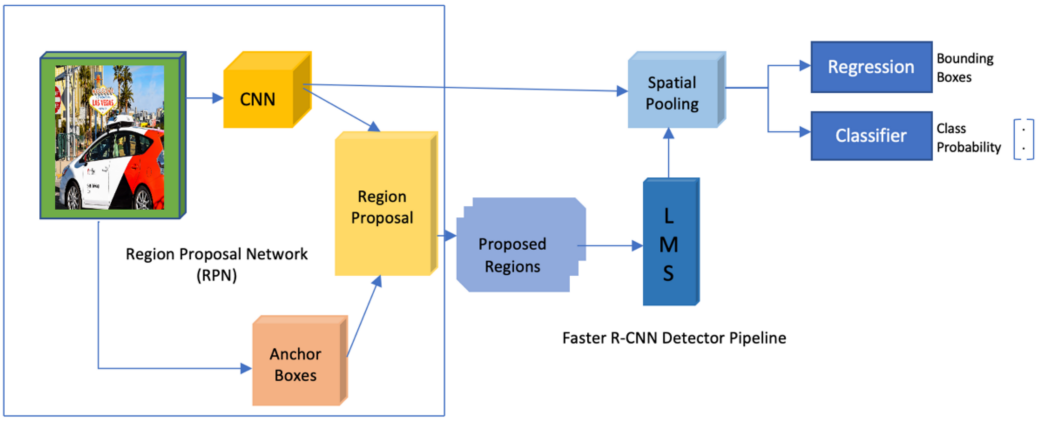

Figure 7. Faster R-CNN object detection model architecture.

4.5.2. YOLOv3

YOLOv3 is a single-stage detector based on CNN and anchor boxes. It has three main parts: the backbone, bottleneck, and head. The backbone is used to extract the feature maps from a given image, which is Darknet-53 in YOLOv3's case. As show in Figure 8, we used an input image of size 416 × 416. YOLOv3 divides the input image into a grid of 13 × 13, and each cell in the grid is responsible for predicting five bounding boxes. The cells contain a set of anchors that point to the object's bounding box coordinates with the object probability and label confidence score. YOLOv3 takes the entire image as the network input and outputs the object bounding box and the target class for each object in the input image. In both perturbation-based and patch attacks, we used the detector as Blackbox without accessing any component or information of the model's internal structure.

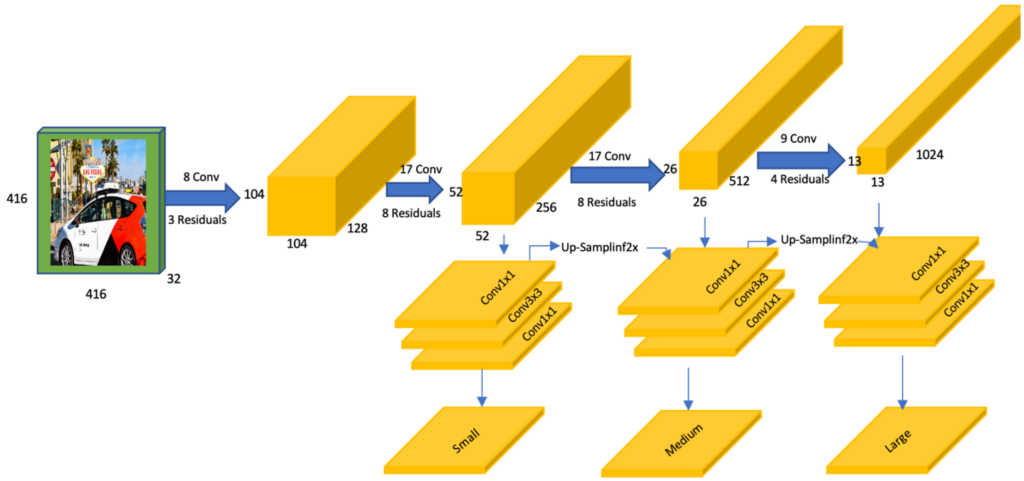

Figure 8. YOLOv3 model architecture.

4.6. Implementation Details

The experiments have been conducted on 2 Nvidia Quadro M4000 GPUs and memory of 16 G.

In the perturbation-based attack, the parameters have been set as follows: The maximum number of Iterations 20,000, the regularization parameter prior to attack loss 10, the attack confidence level in attack loss 1×10^{-10}, mini-batch size 10. The parameters setting for the ZO gradient estimation: number of random direction vectors 10, smoothing parameter 0.001. The initial learning rate is 0.1.

5. Results

5.1. Experiments Results

In this section, we present the results of our experiments where we attack two different object detection models YOLOv3 and Faster R-CNN using two different datasets Vehicles-OpeneImages and Udacity Self-Driving Cars. We tested AT-BOD, that consists of two types of attacks, patch-based and perturbation-based. We conduct an ablation study where each attack is tested separately.

The results of our experiments show AT-BOD can fool object detectors successfully.

Table 1 shows the testing results of AT-BOD attack using adversarial examples generated with the Vehicles-OpeneImages dataset. The performance of both victim models YOLOv3 and Faster R-CNN are dramatically reduced. The inference rate of YOLOv3 is decreased from 84% to <1% while Faster R-CNN only 2% of the testing dataset were detected correctly compared to its performance of 80.28% before the attack. The results on the Udacity SDC dataset, as shown in Table 2, showed that the inference rates of both YOLOv3 and Faster R-CNN victim models were also dramatically decreased where only 1% and 3% of the testing dataset were detected correctly, respectively.

Table 1. The testing results on OpenImages vehicles dataset. Both YOLOv3 and Faster R-CNN are tested before and after AT-BOD.

Models	Inference with Clean Images/mAP@.5	Inference with Adversarial Attack Examples/mAP@.5
YOLOv3	84%	<1%
Faster R-CNN	80.28%	2%

Table 2. The testing results on Udacity SDC dataset dataset. Both YOLOv3 and Faster R-CNN are tested before and after AT-BOD.

Models	Inference with Clean Images/mAP@.5	Inference with Adversarial Attack Examples/mAP@.5
YOLOv3	70%	1%
Faster R-CNN	68%	3%

5.2. AT-BOD Ablation Study

The patch-based AT-BOD attack was tested separately on YOLOv3 using the Vehicles-OpeneImages dataset resulting in high increase in the fooling rate (FR) and false-negative increase (FNI) (Figures 9 and 10). As shown in Table 3, FR reached 97% and FNI reached 99.21%.

Targeting Faster R-CNN, as shown in Table 4, the FR and FNI, reached only 42%, 55%, respectively.

The perturbation-based AT-BOD was also successful on YOLOv3 using the Vehicles-OpeneImages dataset (Figures 9 and 10), where the FR and FNI, as shown in Table 3, reached 89%, 99.02%, respectively. While on Faster R-CNN, the FR and FNI, as shown in Table 4, reached only 61%, 57%, respectively. Both attacks work better on YOLOv3 (Figures 11 and 12).

Table 3. The results of our AT-BOD targeting the YOLOv3 detector using the Vehicles-OpenImages dataset.

AT-BOD Attack	Fooling Ratio	False-Negative Increase
Patch-based	97.3%	99.21%
Perturbations-based	89%	99.02%

Table 4. The results of our AT-BOD targeting Faster R-CNN detector using Vehicles-OpenImages dataset.

AT-BOD Attack	Fooling Ratio	False-Negative Increase
Patch-based	42%	55%
Perturbations-based	61%	57%

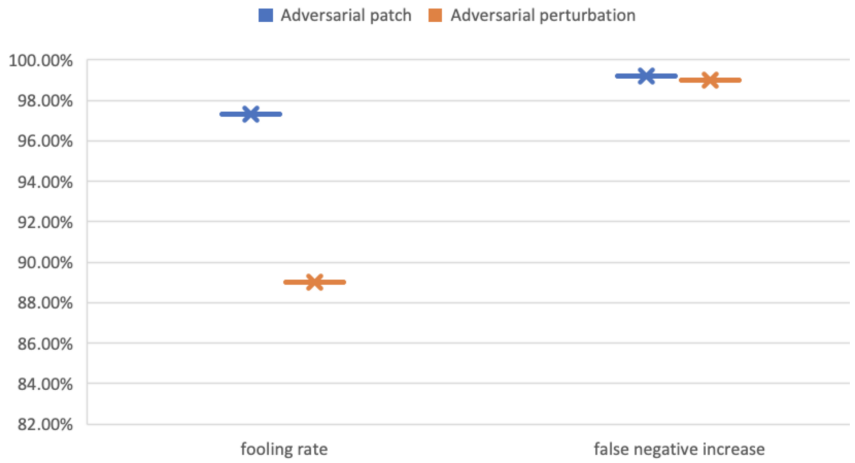

Figure 9. Visual results of the fooling rate (FR) false-negative increase (FNI) of YOLOv3 under AT-BOD attack.

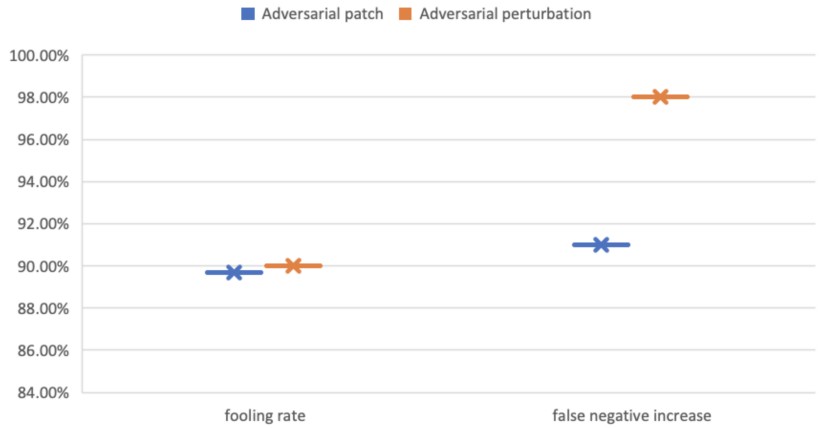

Figure 10. Visual results of the fooling rate (FR) false-negative increase (FNI) of YOLOv3 under AT-BOD using the Udacity SDC dataset.

We conducted the same experiment on Udacity SDC dataset. The patch-based AT-BOD attack was successful targeting YOLOv3, as shown in Table 5 and Figure 10, records an FR and FNI of 89.7% and 91%, respectively. The perturbation-based attack decreased the performance of YOLOv3, where the FR and FNI indicate that the detector was misled in most of the detections and reached 90% and 98%, respectively.

Table 5. The results of our AT-BOD targeting the YOLOv3 detector using the Udacity SDC dataset.

AT-BOD Attack	Fooling Ratio	False-Negative Increase
Patch-based	89.7%	91%
Perturbations-based	90%	98%

The attacks on Faster R-CNN were successful too but less severe compared to its predecessor, where under patch attack, the FR, and the FNI, as shown in Table 6 and Figure 11, augmented to 43% and 52.5%, respectively, while under perturbation-based attack, it reached 42% and 51%, respectively.

Table 6. The results of our AT-BOD targeting the Faster R-CNN detector using the Udacity SDC dataset.

AT-BOD Attack	Fooling Ratio	False-Negative Increase
Patch-based	43%	52.5%
Perturbations-based	42.8%	51%

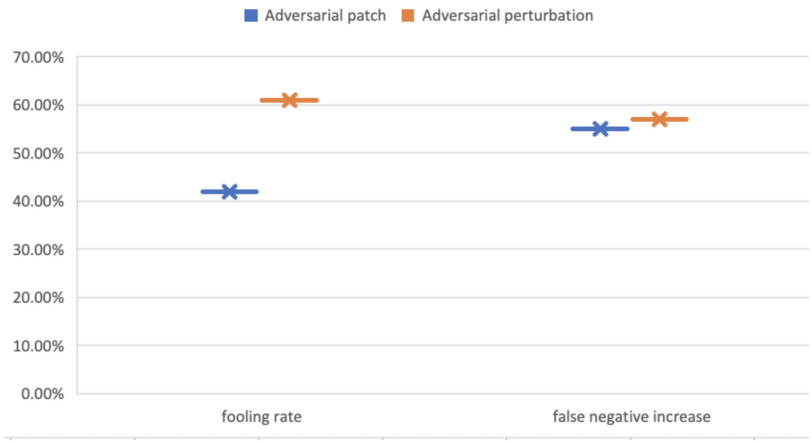

Figure 11. Visual results of the fooling rate (FR) false-negative increase (FNI) of Faster R-CNN under AT-BOD using Vehicles-OpenImages dataset.

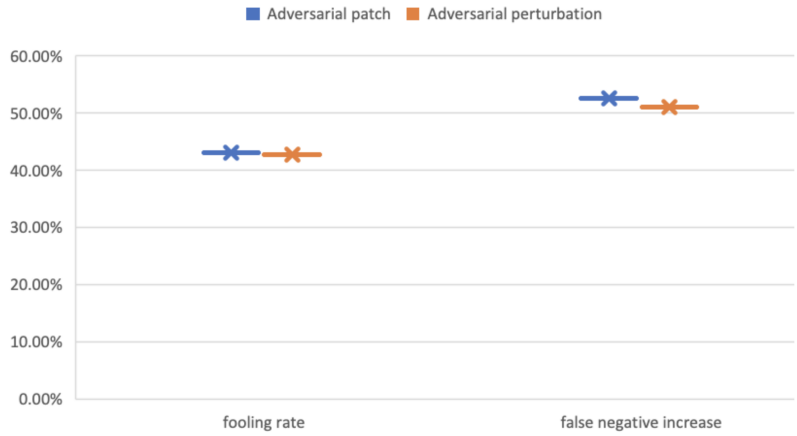

Figure 12. Visual results of the fooling rate (FR) false-negative increase (FNI) of Faster R-CNN under AT-BOD using the Udacity SDC dataset.

5.3. Analysis

In the patch-based AT-BOD, the detector under attack behaves in three ways, either detecting the vehicle object as another class, or with a very low objectness score, or the object is hidden and not detect at all. Comparing the model performance before and after

the attack, the recall score is reduced from 100 to 31%, and the object detection correctness is less than 3%. The detectors are trained on the MS COCO dataset, and we noticed that the model detects the object with a different label from the MS COCO class names. As shown in Figure 13, a motorcycle is ignored instead, the patch is detected and labeled as tie and the motorcycle's wheel as a bowl with a high confidence score of 73.24% and 83.3%, respectively, the ambulance is detected as a car with a low score of 46.35% despite the ambulance name exists in the class names, and the patch is detected as plushie with a high score of 97%. Figure 13 shows more results of the adversarial patch attack.

Figure 13. Detection results applying patch attack.

The perturbation-based AT-BOD made the detector making incorrect detections or blind where no object was detected. The YOLOv3 under this type of attack using Vehicles-OpeneImages dataset resulted in a fooling rate of 89%, while the false negative ratio was 99.02%. The detection rate decreased to less than 1% using the Vehicles-OpenImages dataset. On the Udacity SDC dataset, as shown in Table 5, the fooling rate and the false negative increase reached a rate of 90% and 98%, respectively. The Faster R-CNN behaved abnormally too but with less severity compared to YOLOv3 where the results, as shown in Figures 14 and 15, lead to a fooling rate of 61% and a false negative increase of 57% using the Vehicles-OpenImages dataset, and fooling rate of 42.8% and a false negative increase of 51% using the Udacity SDC dataset. Figures 14 and 15 show one of the detection results of the adversarial perturbation attack.

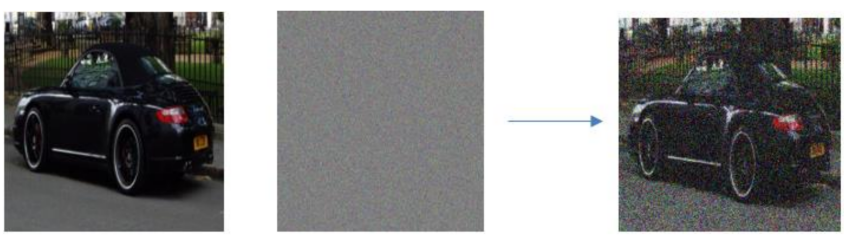

Figure 14. Applying a perturbation-based AT-BOD.

Figure 15. The image in the left is the detection result before the perturbation-based AT-BOD attack. The image on the right is the detection result after the attack.

6. Conclusions

In this work, we proposed a new attack model called AT-BOD. AT-BOD consists of two adversarial attacks perturbation-based that is a ZO-AdaMM-based algorithm to generate universal adversarial examples and patch-based which is an attack works by adding a small patch to the input image. The attack targets two types of object detection models, Faster R-CNN, a region-based detector, and YOLOv3, a regression-based object detection model in black-box settings.

AT-BOD model works in two diverse ways, reducing the confidence score and misleading the model to make the wrong decision or hide the object detection models. Our solution achieved a fooling rate of 97% and a false negative increase of 99% on YOLOv3 detector, and a fooling rate of 61% false-negative increase of 57% on Faster R-CNN detector. The detection accuracy of YOLOv3 and Faster R-CNN under AT-BOD was dramatically reduced and reached $\leq$1% and $\leq$3%, respectively. The attack was effective, and the victim detectors' vulnerability was proven, which cause a severe security problem, especially for safety-critical systems such as self-driving cars. We have conducted our experiments and evaluated our results that prove that our proposed attacks are capable of fooling detectors.

In future work, we want to apply a targeted attack using our perturbation-based model and improve it to make it less imperceptible to human sight. Besides, we aim to develop a defense method against AT-BOD attack. Studying adversarial attacks, especially the black box type attack, provides a step in understanding how they work as well as searching for possible breaches and risks to avoid catastrophic consequences. Object detection interpretability is mandatory to understanding the model in depth, which might aid in building more robust models.

Author Contributions: Conceptualization, methodology and analysis, I.A.E.; writing, review and editing, I.A.E., S.O.O. and R.M.Z. All authors have read and agreed to the published version of the manuscript.

Funding: We would like to thank SAUDI ARAMCO Cybersecurity Chair for funding this project.

Conflicts of Interest: The authors declare that this work has no conflict of interest.

References

1. Maqueda, A.I.; Loquercio, A.; Gallego, G.; Garcia, N.; Scaramuzza, D. Event-Based Vision Meets Deep Learning on Steering Prediction for Self-Driving Cars. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 5419–5427. [CrossRef]
2. Chernikova, A.; Oprea, A.; Nita-Rotaru, C.; Kim, B. Are self-driving cars secure? Evasion attacks against deep neural networks for steering angle prediction. In Proceedings of the IEEE Symposium on Security and Privacy Workshops, San Francisco, CA, USA, 19–23 May 2019; pp. 132–137. [CrossRef]
3. Zhou, S.; Chen, Y.; Li, X.; Sanyal, A. Deep SCNN-Based real-time object detection for self-driving vehicles using LiDAR temporal data. *IEEE Access* **2020**, *8*, 76903–76912. [CrossRef]

4. Sagduyu, Y.E.; Shi, Y.; Erpek, T. IoT Network Security from the Perspective of Adversarial Deep Learning. In Proceedings of the 2019 16th Annual IEEE International Conference on Sensing Communication, and Networking, Boston, MA, USA, 1–9 June 2019. [CrossRef]
5. Goodfellow, I.J.; Shlens, J.; Szegedy, C. Explaining and harnessing adversarial examples. In Proceedings of the 3rd International Conference on Learning Representations, ICLR 2015, Conference Track Proceedings, San Diego, CA, USA, 7–9 May 2015; pp. 1–11.
6. Cao, Y.; Zhou, Y.; Chen, Q.A.; Xiao, C.; Park, W.; Fu, K.; Cyr, B.; Rampazzi, S.; Morley Mao, Z. Adversarial sensor attack on LiDAR-based perception in autonomous driving. In Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security, Auckland, New Zealand, 9–12 July 2019; pp. 2267–2281. [CrossRef]
7. Szegedy, C.; Bruna, J.; Erhan, D.; Goodfellow, I. Intriguing properties of neural networks. *arXiv* **2014**, arXiv:1312.6199.
8. Cisse, M.; Adi, Y.; Neverova, N.; Keshet, J. Houdini: Fooling Deep Structured Prediction Models. *arXiv* **2017**, arXiv:1707.05373.
9. Hashemi, A.S.; Mozaffari, S. Secure deep neural networks using adversarial image generation and training with Noise-GAN. *Comput. Secur.* **2019**, *86*, 372–387. [CrossRef]
10. Wang, B.; Yao, Y.; Shan, S.; Li, H.; Viswanath, B.; Zheng, H.; Zhao, B.Y. Neural cleanse: Identifying and mitigating backdoor attacks in neural networks. In Proceedings of the IEEE Symposium on Security and Privacy 2019, San Francisco, CA, USA, 19–23 May 2019; pp. 707–723. [CrossRef]
11. Lopez, A.; Malawade, A.V.; Al Faruque, M.A.; Boddupalli, S.; Ray, S. Security of emergent automotive systems: A tutorial introduction and perspectives on practice. *IEEE Des. Test* **2019**, *36*, 10–38. [CrossRef]
12. Morgulis, N.; Kreines, A.; Mendelowitz, S.; Weisglass, Y. Fooling a Real Car with Adversarial Traffic Signs. *arXiv* **2019**, arXiv:1907.00374.
13. Sitawarin, C.; Bhagoji, A.N.; Mosenia, A.; Chiang, M.; Mittal, P. DARTS: Deceiving Autonomous Cars with Toxic Signs. *arXiv* **2019**, arXiv:1907.00374.
14. Biggio, B.; Corona, I.; Maiorca, D.; Nelson, B.; Šrndić, N.; Laskov, P.; Roli, F. Evasion attacks against machine learning at test time. In *Machine Learning and Knowledge Discovery in Databases, Proceedings of the Joint European Conference on Machine Learning and Knowledge Discovery in Databases, Prague, Czech Republic, 23–27 September 2013*; Springer: Berlin, Germany, 2013; Volume 8190, pp. 387–402.
15. Wu, X.; Sahoo, D.; Hoi, S.C.H. Recent Advances in Deep Learning for Object Detection. *Neurocomputing* **2020**, *396*, 39–64. [CrossRef]
16. Melis, M.; Demontis, A.; Biggio, B.; Brown, G.; Fumera, G.; Roli, F. Is Deep Learning Safe for Robot Vision? Adversarial Examples against the iCub Humanoid. In Proceedings of the IEEE International Conference on Computer Vision Workshops, Venice, Italy, 22–29 October 2017.
17. Lu, J.; Sibai, H.; Fabry, E. Adversarial Examples that Fool Detectors. *arXiv* **2017**, arXiv:1712.02494.
18. Xie, C.; Wang, J.; Zhang, Z.; Zhou, Y.; Xie, L.; Yuille, A. Adversarial Examples for Semantic Segmentation and Object Detection. In Proceedings of the IEEE International Conference on Computer Vision Workshops 2017, Venice, Italy, 22–29 October 2017; pp. 1378–1387. [CrossRef]
19. Chen, C.; Zhao, X.; Stamm, M.C. Generative Adversarial Attacks Against Deep-Learning-Based Camera Model Identification. *IEEE Trans. Inf. Forensics Secur.* **2019**, *X*, 1–16. [CrossRef]
20. Jeong, J.H.; Kwon, S.; Hong, M.P.; Kwak, J.; Shon, T. Adversarial attack-based security vulnerability verification using deep learning library for multimedia video surveillance. *Multimed. Tools Appl.* **2019**, *79*, 16077–16091. [CrossRef]
21. Makelov, A.; Schmidt, L.; Tsipras, D.; Vladu, A. Towards Deep Learning Models Resistant to Adversarial Attacks. *arXiv* **2019**, arXiv:170606083.
22. Fawzi, A.; Frossard, P. DeepFool: A simple and accurate method to fool deep neural networks. In Proceedings of the IEEE conference on computer vision and pattern recognition, Las Vegas, NV, USA, 27–30 June 2016.
23. Carlini, N.; Wagner, D. Towards Evaluating the Robustness of Neural Networks. In Proceedings of the 2017 IEEE Symposium on Security and Privacy, San Jose, CA, USA, 22–26 May 2017; pp. 39–57. [CrossRef]
24. Chen, X.; Liu, S.; Xu, K.; Li, X. ZO-AdaMM: Zeroth-Order Adaptive Momentum Method for Black-Box Optimization. *arXiv* **2019**, arXiv:1910.06513v2.
25. Thys, S.; Ranst, W.V.; Goedeme, T. Fooling automated surveillance cameras: Adversarial patches to attack person detection. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops, Long Beach, CA, USA, 16–20 June 2019; pp. 49–55. [CrossRef]
26. Rakin, A.S.; Fan, D. Defense-Net: Defend Against a Wide Range of Adversarial Attacks through Adversarial Detector. In Proceedings of the 2019 IEEE Computer Society Annual Symposium on VLSI (ISVLSI), Miami, FL, USA, 15–17 July 2019; pp. 332–337. [CrossRef]
27. Balakrishnan, A.; Puranic, A.G.; Qin, X.; Dokhanchi, A.; Deshmukh, J.V.; Ben Amor, H.; Fainekos, G.; Yuan, X.; He, P.; Zhu, Q.; et al. Adversarial Examples: Attacks and Defenses for Deep Learning. In Proceedings of the 7th International Conference on Learning Representations. New Orleans, Louisiana, USA, 6–9 May 2019; Volume 36, pp. 2805–2824. [CrossRef]
28. Papernot, N.; McDaniel, P.; Goodfellow, I.; Jha, S.; Celik, Z.B.; Swami, A. Practical black-box attacks against machine learning. In Proceedings of the ACM Asia Conference on Computer and Communications Security. Abu Dhabi, United Arab Emirates, 2–6 April 2017; pp. 506–519. [CrossRef]

29. Liu, Y.; Chen, X.; Liu, C.; Song, D. Delving into Transferable Adversarial Examples and Black-Box Attacks. *arXiv* **2017**, arXiv:1611.02770.
30. Narodytska, N. Simple Black-Box Adversarial Perturbations for Deep Networks. *arXiv* **2016**, arXiv:1612.06299.
31. Bhagoji, A.N.; He, W.; Li, B.; Song, D. Practical Black-box Attacks on Deep Neural Networks using Efficient Query Mechanisms. In Proceedings of the 15th European Conference, Munich, Germany, 8–14 September 2018; pp. 1–16.
32. Chen, P. ZOO: Zeroth Order Optimization Based Black-box Attacks to Deep Neural Networks without Training Substitute Models. *arXiv* **2017**, arXiv:1708.03999.
33. Sharif, M.; Bauer, L.; Reiter, M.K. Accessorize to a Crime: Real and Stealthy Attacks on State-of-the-Art Face Recognition. In Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security, Vienna Austria, 24–28 October 2016; pp. 1–13. [CrossRef]
34. Liu, Y.; Lee, W.; Zhai, J. *Trojaning Attack on Neural Networks*; Purdue E-Pubs: West Lafayette, IN, USA, 2017; pp. 1–17.
35. Brown, T.B.; Mané, D.; Roy, A.; Abadi, M.; Gilmer, J. Adversarial Patch. In Proceedings of the 31st International Conference on Neural Information Processing Systems, Long Beach, CA, USA, 4–9 December 2017; pp. 1–5.
36. Burton, S.; Gauerhof, L.; Heinzemann, C. Making the case for safety of machine learning in highly automated driving. In *International Conference on Computer Safety, Reliability, and Security, Proceedings of the Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), Trento, Italy, 12–15 September 2017*; Springer: Berlin, Germany, 2017; Volome 10489, pp. 5–16.
37. Chatterjee, D. Adversarial Attacks and Defenses in a Self-driving Car Environment. *Int. J. Adv. Sci. Technol.* **2020**, *29*, 2233–2240.
38. Geiger, A.; Lenz, P.; Urtasun, R. Are we ready for autonomous driving? the KITTI vision benchmark suite. In Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, Providence, RI, USA, 16–21 June 2012; pp. 3354–3361. [CrossRef]
39. Eykholt, K.; Evtimov, I.; Fernandes, E.; Li, B.; Rahmati, A.; Tram, F.; Prakash, A.; Kohno, T.; Song, D. Physical Adversarial Examples for Object Detectors. *arXiv* **2018**, arXiv:2108.11765.
40. Stilgoe, J. Machine learning, social learning and the governance of self-driving cars. *Soc. Stud. Sci.* **2018**, *48*, 25–56. [CrossRef]
41. Lu, J.; Sibai, H.; Fabry, E.; Forsyth, D. NO Need to Worry about Adversarial Examples in Object Detection in Autonomous Vehicles. *arXiv* **2017**, arXiv:1707.03501.
42. Mimouna, A.; Alouani, I.; Ben Khalifa, A.; Hillali, Y.E.; Taleb-Ahmed, A.; Menhaj, A.; Ouahabi, A.; Essoukri, N.; Amara, B. electronics OLIMP: A Heterogeneous Multimodal Dataset for Advanced Environment Perception. *Electronics* **2020**, *9*, 560. [CrossRef]
43. Wang, X.; Shrivastava, A.; Gupta, A. A-Fast-RCNN: Hard Positive Generation via Adversary for Object Detection. In Proceedings of the Proceedings-30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, 21–26 July 2017; pp. 3039–3048.
44. Lu, J.; Sibai, H.; Fabry, E.; Forsyth, D. Standard detectors aren't (currently) fooled by physical adversarial stop signs. *arXiv* **2017**, arXiv:1710.03337.
45. Models, P.; Li, Y. Robust Adversarial Perturbation on Deep Proposal-based Models. *arXiv* **2019**, arXiv:1809.05962v2.
46. Chen, S.-T.; Cornelius, C.; Martin, J.; Horng Chau, D. *ShapeShifter: Robust Physical Adversarial Attack on Faster R-CNN Object Detector*; Springer: Cham, Germany, 2018; pp. 52–68.
47. Liu, X.; Yang, H.; Liu, Z.; Song, L.; Li, H.; Chen, Y. Dpatch: An adversarial patch attack on object detectors. *arXiv* **2017**, arXiv:1806.02299.
48. Liu, S.; Chen, P. Zeroth-Order Optimization and Its Application to Adversarial Machine Learning. *Intell. Inform.* **2018**, *19*, 25.
49. Moosavi-Dezfooli, S.M.; Fawzi, A.; Fawzi, O.; Frossard, P. Universal adversarial perturbations. In Proceedings of the 30th IEEE Computer Vision And Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 86–94. [CrossRef]
50. Lian, X.; Zhang, H.; Hsieh, C.-J.; Huang, Y.; Liu, J. A Comprehensive Linear Speedup Analysis for Asynchronous Stochastic Parallel Optimization from Zeroth-Order to First-Order. *Adv. Neural Inf. Processing Syst.* **2017**, *29*, 3054–3062.
51. Liu, S.; Chen, P.Y.; Chen, X.; Hong, M. SignsGD via zeroth-order oracle. In Proceedings of the 7th International Conference on Learning Representations, New Orleans, LA, USA, 6–9 May 2019; pp. 1–24.
52. Kingma, D.P.; Lei Ba, J. Adam: A Method For Stochastic Optimization. *arXiv* **2014**, arXiv:1412.6980.
53. Reddi, S.J.; Kale, S.; Kumar, S. On the convergence of adam and beyond. *arXiv* **2019**, arXiv:1904.09237.
54. Everingham, M.; Van Gool, L.; Williams, C.K.I.; Winn, J.; Zisserman, A. The pascal visual object classes (VOC) challenge. *Int. J. Comput. Vis.* **2010**, *88*, 303–338. [CrossRef]
55. Dai, J.; Li, Y.; He, K.; Sun, J. R-FCN: Object detection via region-based fully convolutional networks. In Proceedings of the Advances in Neural Information Processing Systems, Barcelona, Spain, 5–10 December 2016; pp. 379–387.
56. Ren, S.; He, K.; Girshick, R.; Sun, J. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. *IEEE Trans. Pattern Anal. Mach. Intell.* **2017**, *39*, 1137–1149. [CrossRef]
57. Yun, J.W. Deep Residual Learning for Image Recognition. *Enzyme Microb. Technol.* **1996**, *19*, 107–117. [CrossRef]