TOG: Targeted Adversarial Objectness Gradient Attacks on Real-time Object Detection Systems*

Ka-Ho Chow, Ling Liu, Mehmet Emre Gursoy, Stacey Truex, Wenqi Wei, Yanzhao Wu
Georgia Institute of Technology
Atlanta, Georgia

ABSTRACT

The rapid growth of real-time huge data capturing has pushed the deep learning and data analytic computing to the edge systems. Real-time object recognition on the edge is one of the representative deep neural network (DNN) powered edge systems for real-world mission-critical applications, such as autonomous driving and augmented reality. While DNN powered object detection edge systems celebrate many life-enriching opportunities, they also open doors for misuse and abuse. This paper presents three Targeted adversarial Objectness Gradient attacks, coined as TOG, which can cause the state-of-the-art deep object detection networks to suffer from object-vanishing, object-fabrication, and object-mislabeling attacks. We also present a universal objectness gradient attack to use adversarial transferability for black-box attacks, which is effective on any inputs with negligible attack time cost, low human perceptibility, and particularly detrimental to object detection edge systems. We report our experimental measurements using two benchmark datasets (PASCAL VOC and MS COCO) on two state-of-the-art detection algorithms (YOLO and SSD). The results demonstrate serious adversarial vulnerabilities and the compelling need for developing robust object detection systems.

KEYWORDS

adversarial machine learning, object detection, neural network, edge security and privacy.

1 INTRODUCTION

Edge data analytics and deep learning as a service on the edge have attracted a flurry of research and development efforts in both industry and academics [5, 9]. Open source deep object detection networks [8, 12, 13] have fueled new edge applications and edge system deployments, such as traffic sign identification on autonomous vehicles [14] and intrusion detection on smart surveillance systems [3]. However, very few performed systematic studies on the vulnerabilities of real-time deep object detectors, which are critical to edge security and privacy. Figure 1 shows a typical scenario where an edge system receives an input image or video frame from a sensor (e.g., a camera), and it runs a real-time DNN object detection model (e.g., YOLOv3 [12]) on the edge device (e.g., a Raspberry Pi with an AI acceleration module). With no attack, the well-trained object detector can process the benign input (top) and accurately identify a person walking across the street. However, under attack

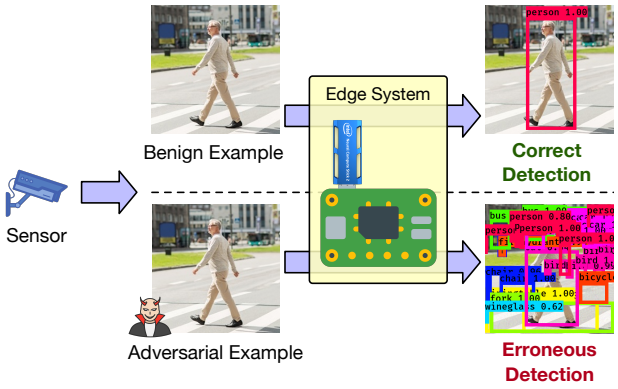


Figure 1: The edge system correctly identifies the person on the benign input (top) but misdetects given the adversarial example (bottom), which is visually indistinguishable from the benign one.

with an adversarial example (bottom), which is visually indistinguishable by human perception to the benign input, the *same* object detector will be fooled to make erroneous detection.

In this paper, we present three vulnerabilities of DNN object detection systems, by developing three **T**argeted adversarial **O**bjectness **G**radient attacks, as a family of TOG attacks on real-time object detection systems. Although there is a large body of adversarial attacks to DNN image classifiers [4] in literature, they are mainly effective in causing DNN classifiers to produce wrong classifications by using different attack strategies to determine the location and the amount of per-pixel perturbation to inject to a benign input image [15]. In contrast, deep object detection networks detect and segment multiple objects that may be visually overlapping in a single image or video frame and provide one class label to each of the detected objects. Thus, an in-depth understanding of various vulnerabilities of deep object detectors is more complicated than misclassification attacks in the DNN image classifiers, because the DNN object detectors have larger and varying attack surfaces, such as the object existence, object location, and object class label, which open more opportunities for attacks with various adversarial goals and sophistications. The TOG attacks are the first targeted adversarial attack method on object detection networks by targeting at different objectness semantics, such as making objects vanishing, fabricating more objects, or mislabeling some or all objects. Each of these attacks injects a human-imperceptible adversarial perturbation to fool a real-time object detector to misbehave in three different ways, as shown in Figure 2. The *object-vanishing* attack in Figure 2(b) causes all objects to vanish from the YOLOv3 [12] detector. The *object-fabrication* attack in Figure 2(c) causes the detector to output many false objects with high confidence. The

*This work is released as a technical report at the Distributed Data Intensive Systems Lab (DiSL), Georgia Institute of Technology, with the following version history
v1: November 15, 2019
v2: February 28, 2020
Readers may visit https://github.com/git-disl/TOG for updates.

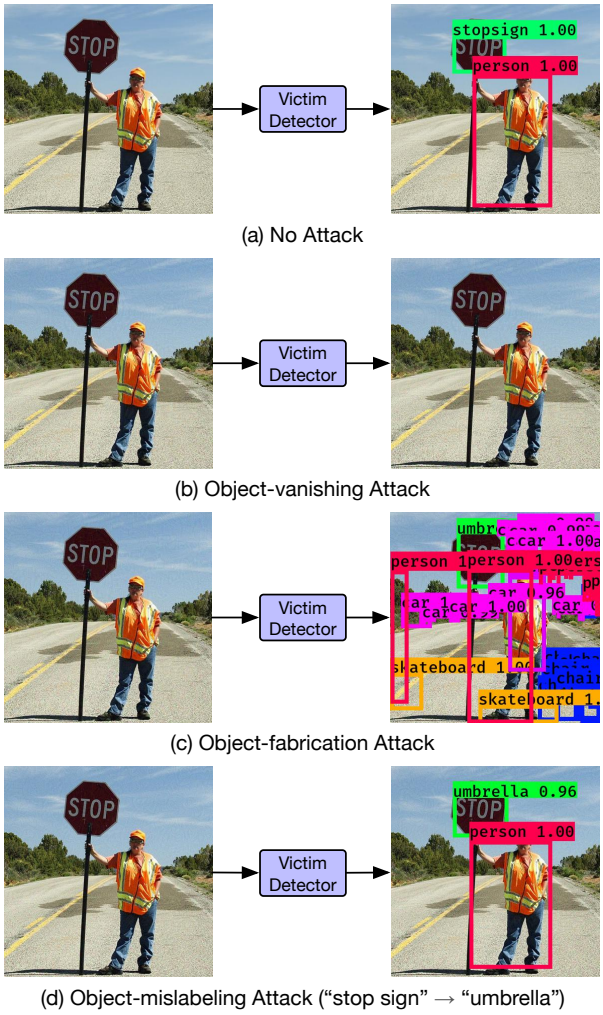


Figure 2: Illustration of three TOG attacks (2nd-4th rows) to deep object detection networks. *Left*: Benign input. *Right*: Detection results under the adversarial attacks.

object-mislabeling attack in Figure 2(d) fools the detector to mislabel (e.g., the stop sign becomes an umbrella). We further present a highly efficient universal adversarial perturbation algorithm that generates a single universal perturbation that can perturb any input to fool the victim detector effectively. Given the attack is generated offline, by utilizing adversarial transferability, one can use the TOG universal perturbation to launch a black-box attack with a very low (almost zero) online attack cost, which is particularly fatal in real-time edge applications for object detections.

The rest of the paper is organized as follows. We first present the TOG attacks in Section 2, then experimental evaluations in Section 3, followed by concluding remarks in Section 4.

2 TOG ATTACKS

Deep object detection networks, in general, share the same input-output structure with a similar formulation. They all take the input image or video frame and produce outputs by using bounding box techniques to provide object localization for all objects of interest and by giving the classification for each detected object [8, 10–13]. The TOG attacks are constructed without restriction to any particular detection algorithm, as shown in our experimental evaluation. To illustrate the details of TOG attacks, we choose to use YOLOv3 [12] to present our formulation in this section.

Given an input image $\boldsymbol{x}$, the object detector first detects a large number of S candidate bounding boxes $\hat{\boldsymbol{\mathcal{B}}}(\boldsymbol{x}) = \{\hat{\boldsymbol{o}}_1, \hat{\boldsymbol{o}}_2, \ldots, \hat{\boldsymbol{o}}_S\}$ where $\hat{\boldsymbol{o}}_i = (\hat{b}_i^x, \hat{b}_i^y, \hat{b}_i^W, \hat{b}_i^H, \hat{C}_i, \hat{\boldsymbol{p}}_i)$ is a candidate centered at $(\hat{b}_i^x, \hat{b}_i^y)$ having a dimension $(\hat{b}_i^W, \hat{b}_i^H)$ with a probability of $\hat{C}_i \in [0, 1]$ having an object contained, and a K-class probabilities $\hat{\boldsymbol{p}}_i = (\hat{p}_i^1, \hat{p}_i^2, \ldots, \hat{p}_i^K)$. This is done by dividing the input $\boldsymbol{x}$ into mesh grids in different scales (resolutions) where each grid cell produces multiple candidate bounding boxes based on the anchors and is responsible for locating objects centered at the cell. The final detection results $\hat{\boldsymbol{O}}(\boldsymbol{x})$ are obtained by applying confidence thresholding to remove candidates with low prediction confidence (i.e., $\max_{1 \le c \le K} \hat{C}_i \hat{p}_i^c$) and non-maximum suppression to exclude those with high overlapping.

An adversarial example $\boldsymbol{x}'$ is generated by perturbing a benign input $\boldsymbol{x}$ sent to the victim detector, aiming to fool the victim to misdetect randomly (untargeted) or purposefully (targeted). The generation process of the adversarial example can be formulated as

$$\min ||\boldsymbol{x}' - \boldsymbol{x}||_p \quad s.t.\ \hat{\boldsymbol{O}}(\boldsymbol{x}') = \boldsymbol{O}^*, \hat{\boldsymbol{O}}(\boldsymbol{x}') \neq \hat{\boldsymbol{O}}(\boldsymbol{x}), \tag{1}$$

where p is the distance metric, which can be the L_0 norm measuring the percentage of the pixels that are changed, the L_2 norm computing the Euclidean distance, or the L_∞ norm denoting the maximum change to any pixel, and $\boldsymbol{O}^*$ denotes the target detections for targeted attacks or any incorrect ones for untargeted attacks.

Figure 3 illustrates the adversarial attacks using TOG. Given an input source (e.g., an image or video frame), TOG attack module takes the configuration specified by the adversary to prepare for the corresponding adversarial perturbation, which will be added to the input to cause the victim to misdetect. The first three attacks in TOG: TOG-vanishing, TOG-fabrication, and TOG-mislabeling tailor an adversarial perturbation for each input, while TOG-universal uses the same universal perturbation to corrupt any input.

Training a deep neural network often starts with random initialization of model weights, which will be updated slowly by taking the derivative of the loss function $\mathcal{L}$ with respect to the learnable model weights $\boldsymbol{\mathcal{W}}$ over a mini-batch of input-output pairs $\{(\tilde{\boldsymbol{x}}, \boldsymbol{O})\}$ with the following equation until convergence:

$$\boldsymbol{\mathcal{W}}_{t+1} = \boldsymbol{\mathcal{W}}_t - \alpha \frac{\partial\, \mathbb{E}_{(\tilde{\boldsymbol{x}}, \boldsymbol{O})}[\mathcal{L}(\tilde{\boldsymbol{x}}; \boldsymbol{O}, \boldsymbol{\mathcal{W}}_t)]}{\partial \boldsymbol{\mathcal{W}}_t} \tag{2}$$

where α is the learning rate controlling the step size of the update. While training deep object detection networks is done by *fixing* the input image $\tilde{\boldsymbol{x}}$ and progressively *updating* the model weights $\boldsymbol{\mathcal{W}}$ towards the goal defined by the loss function, TOG conducts adversarial attacks by reversing the training process. We *fix* the model weights of the victim detector and iteratively *update* the input image $\boldsymbol{x}$ towards the goal defined by the type of the attack to be launched with the following general equation:

$$\boldsymbol{x}'_{t+1} = \prod_{\boldsymbol{x}, \epsilon} \left[\boldsymbol{x}'_t - \alpha \Gamma \left(\frac{\partial \mathcal{L}^*(\boldsymbol{x}'_t; \boldsymbol{O}^*, \boldsymbol{\mathcal{W}})}{\partial \boldsymbol{x}'_t} \right) \right] \tag{3}$$

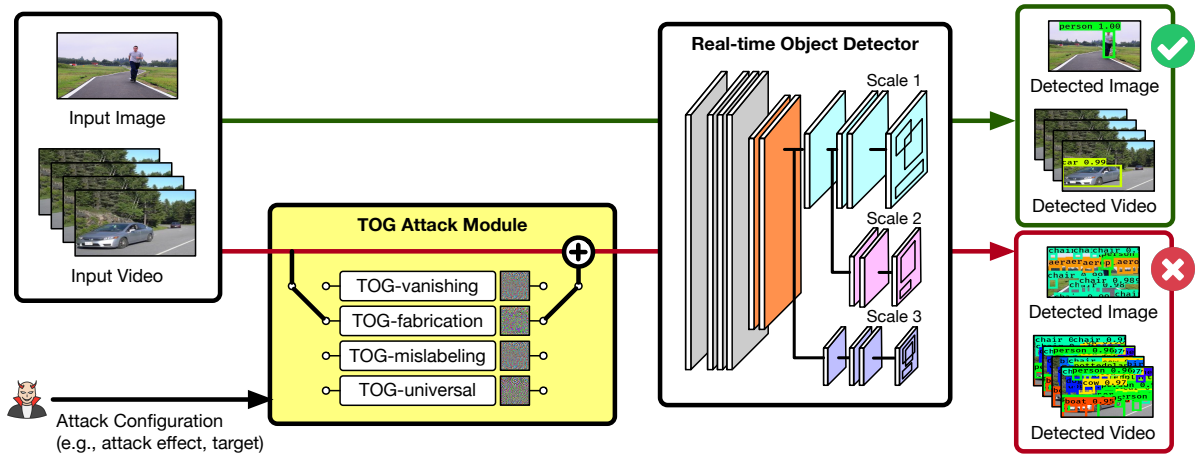


Figure 3: An illustration of the adversarial attacks using TOG.

where $\prod_{\boldsymbol{x},\epsilon}[\cdot]$ is the projection onto a hypersphere with a radius ϵ centered at $\boldsymbol{x}$ in L_p norm, Γ is a sign function, and $\mathcal{L}^*$ defines the loss function to be optimized during the attack.

In deep object detection networks, every ground-truth object in a training sample $\tilde{\boldsymbol{x}}$ will be assigned to one of the S bounding boxes according to the center coordinates and the amount of overlapping with the anchors. Let $\mathbb{1}_i = 1$ if the i-th bounding box is responsible for an object and 0 otherwise. Then, $\boldsymbol{O} = \{\boldsymbol{o}_i | \mathbb{1}_i = 1, 1 \le i \le S\}$ is a set of ground-truth objects where $\boldsymbol{o}_i = (b_i^x, b_i^y, b_i^W, b_i^H, \boldsymbol{p}_i)$ with $\boldsymbol{p}_i = (p_i^1, p_i^2, \ldots, p_i^K)$ and $p_i^c = 1$ if $\boldsymbol{o}_i$ is a class c object. The optimization objective of the deep object detection network consists of three parts, each of them corresponds to one of the three output structures describing a detected object (i.e., existence, locality, and class label). The objectness score $\hat{C}_i \in [0, 1]$ determines the existence of an object in the candidate bounding box, which can be learned by minimizing the binary cross-entropy ℓ_{BCE}:

$$\begin{aligned}\mathcal{L}_{\text{obj}}(\tilde{\boldsymbol{x}}; \boldsymbol{O}, \boldsymbol{\mathcal{W}}) &= \sum_{i=1}^{S} \left[\mathbb{1}_i \ell_{\text{BCE}}(1, \hat{C}_i) \right] \\ \mathcal{L}_{\text{noobj}}(\tilde{\boldsymbol{x}}; \boldsymbol{O}, \boldsymbol{\mathcal{W}}) &= \sum_{i=1}^{S} \left[(1 - \mathbb{1}_i) \ell_{\text{BCE}}(0, \hat{C}_i) \right]\end{aligned} \tag{4}$$

The center coordinates $(\hat{b}_i^x, \hat{b}_i^y)$ and dimension $(\hat{b}_i^W, \hat{b}_i^H)$ give the spatial locality, learned by minimizing the squared error ℓ_{SE}:

$$\begin{aligned}\mathcal{L}_{\text{loc}}(\tilde{\boldsymbol{x}}; \boldsymbol{O}, \boldsymbol{\mathcal{W}}) = \sum_{i=1}^{S} \mathbb{1}_i \big[&\ell_{\text{SE}}(b_i^x, \hat{b}_i^x) + \ell_{\text{SE}}(b_i^y, \hat{b}_i^y) \\ &+ \ell_{\text{SE}}\big(\sqrt{b_i^W}, \sqrt{\hat{b}_i^W}\big) + \ell_{\text{SE}}\big(\sqrt{b_i^H}, \sqrt{\hat{b}_i^H}\big) \big]\end{aligned} \tag{5}$$

The last part is the K-class probabilities $\hat{\boldsymbol{p}}_i = (\hat{p}_i^1, \hat{p}_i^2, \ldots, \hat{p}_i^K)$ that estimate the class label of the corresponding candidate, optimized by minimizing the binary cross-entropy:

$$\mathcal{L}_{\text{prob}}(\tilde{\boldsymbol{x}}; \boldsymbol{O}, \boldsymbol{\mathcal{W}}) = \sum_{i=1}^{S} \mathbb{1}_i \sum_{c \in \text{classes}} \ell_{\text{BCE}}(p_i^c, \hat{p}_i^c) \tag{6}$$

As a result, the deep object detection network can be optimized by the linear combination of the above loss functions:

$$\begin{aligned}\mathcal{L}(\tilde{\boldsymbol{x}}; \boldsymbol{O}, \boldsymbol{\mathcal{W}}) = &\mathcal{L}_{\text{obj}}(\tilde{\boldsymbol{x}}; \boldsymbol{O}, \boldsymbol{\mathcal{W}}) + \lambda_{\text{noobj}} \mathcal{L}_{\text{noobj}}(\tilde{\boldsymbol{x}}; \boldsymbol{O}, \boldsymbol{\mathcal{W}}) \\ &+ \lambda_{\text{loc}} \mathcal{L}_{\text{loc}}(\tilde{\boldsymbol{x}}; \boldsymbol{O}, \boldsymbol{\mathcal{W}}) + \mathcal{L}_{\text{prob}}(\tilde{\boldsymbol{x}}; \boldsymbol{O}, \boldsymbol{\mathcal{W}})\end{aligned} \tag{7}$$

where λ_{noobj} and λ_{loc} are hyperparameters penalizing incorrect objectness scores and bounding boxes respectively.

To tailor an adversarial perturbation for each input $\boldsymbol{x}$ to generate the corresponding adversarial example $\boldsymbol{x}'$, TOG is initialized with the benign example (i.e., $\boldsymbol{x}'_0 = \boldsymbol{x}$) and sends the adversarial example $\boldsymbol{x}'_t$ at the t-th iteration to the victim detector to observe the detection results $\hat{\boldsymbol{O}}(\boldsymbol{x}'_t)$. If the termination condition defined by the attack goal is achieved or the maximum number of iterations T is reached, $\boldsymbol{x}'_t$ will be returned. Otherwise, it will be perturbed using Equation 3 to become $\boldsymbol{x}'_{t+1}$ for the new iteration.

TOG-vanishing. For the TOG object-vanishing attack, we set the target detection $\boldsymbol{O}^* = \emptyset$ and $\mathcal{L}^* = \mathcal{L}$ to cause the victim to detect no objects on the adversarial example.

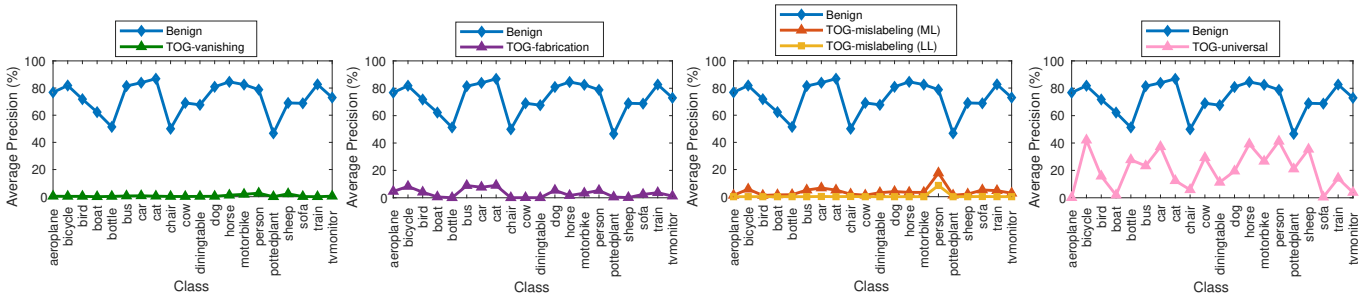
TOG-fabrication. For the TOG object-fabrication attack, we set the target detection $\boldsymbol{O}^* = \hat{\boldsymbol{O}}(\boldsymbol{x})$ and $\mathcal{L}^* = -\mathcal{L}$ to return a large number of false objects.

TOG-mislabeling. For the TOG object-mislabeling attack, we set the target detection $\boldsymbol{O}^*$ to be $\hat{\boldsymbol{O}}(\boldsymbol{x})$ with each object having an incorrect label and $\mathcal{L}^* = \mathcal{L}$. While any incorrect class label can be assigned, we adopt a systematic approach to generate targets [15]: the least-likely (LL) class attack picks the class label with the lowest probability (i.e., $y_i^* = \arg\min_c \hat{p}_i^c$), and the most-likely (ML) class attack finds the class label with the second-highest probability (i.e., $y_i^* = \arg\max_{c, \hat{p}_i^c \neq \max_u \hat{p}_i^u} \hat{p}_i^c$).

TOG-universal. For the TOG universal attack, the algorithm will generate a single universal perturbation for making any input suffer from an object-vanishing attack. Although all three TOG attacks of generating adversarial perturbation for each input to fool an object detector are highly effective, they all require iterative optimization to produce effective perturbations during the online detection. The TOG universal attack can generate a universal perturbation applicable to any input to the detector through iterative optimization by training. Let $\mathcal{D}$ denote the set of N training images,

Dataset	Detector	Benign mAP (%)	Adversarial mAP (%)				
			TOG-vanishing	TOG-fabrication	TOG-mislabeling (ML)	TOG-mislabeling (LL)	TOG-universal
VOC	YOLOv3-D	83.43	0.31	1.41	4.58	1.73	14.16
	YOLOv3-M	72.51	0.58	3.37	3.75	0.45	20.44
	SSD300	78.09	5.58	7.34	2.81	0.78	31.80
	SSD512	79.83	9.90	8.34	2.22	0.77	46.17
COCO	YOLOv3-D	54.89	0.51	5.89	7.02	0.90	12.73

Table 1: Evaluation of TOG attacks to four victim detectors.



(a) TOG-vanishing **(b) TOG-fabrication** **(c) TOG-mislabeling** **(d) TOG-universal**

Figure 4: The average precision of each class in VOC for benign (blue) and adversarial examples generated by TOG attacks.

we want to gradually build the perturbation vector $\boldsymbol{\eta}$. At each iteration t, we obtain a training sample $\tilde{\boldsymbol{x}} \in \mathcal{D}$ and find the additional perturbation $\Delta\boldsymbol{\eta}_t$ that causes the victim detector to make errors towards object vanishing attack goal in the current perturbed image $\tilde{\boldsymbol{x}} + \boldsymbol{\eta}_t$. We then add this additional perturbation $\Delta\boldsymbol{\eta}_t$ to the current universal adversarial perturbation $\boldsymbol{\eta}_t$ and clip the new perturbation to ensure the distortion is constrained within $[-\epsilon, \epsilon]$. The termination condition can be κ% of the objects in the training images vanish, or a maximum number of epochs Q is reached. Upon completing the attack training, the universal perturbation can be applied to any given input to the real-time object detector running in an edge system. It is a black-box attack by adversarial transferability since it can be employed directly upon receiving a benign input to the object detector at runtime by only applying the pretrained perturbation. This attack can fool the object detector at an almost zero online attack time cost.

3 EXPERIMENTAL EVALUATION

Extensive experiments are conducted using two popular benchmark datasets: PASCAL VOC [2] and MS COCO [6] on four state-of-the-art object detectors from two popular families of detection algorithms: "YOLOv3-D" and "YOLOv3-M" are the YOLOv3 [12] models with a Darknet53 backbone and a MobileNetV1 backbone respectively. "SSD300" and "SSD512" are the SSD [8] models with an input resolution of (300×300) and (512×512) respectively. The VOC 2007+2012 dataset has 16,551 training images and 4,952 testing images. The COCO 2014 dataset has 117,264 training images and 5,000 testing images. We report the results on the entire test set. The mean average precision (mAP) of each dataset and victim detector is presented in the 3rd column of Table 1. All experiments use the default configurations from each detector without any fine-tuning of the hyperparameters in each setting. We produce adversarial perturbations in L_∞ norm with a maximum distortion $\epsilon = 0.031$, a step size $\alpha = 0.008$, and the number of iterations $T = 10$. For universal attacks, 12,800 images from the training set are extract to form $\mathcal{D}$ with a maximum distortion $\epsilon = 0.031$, a learning rate $\alpha = 0.0001$, and the number of training epochs $Q = 50$. All attacks were conducted on NVIDIA RTX 2080 SUPER (8 GB) GPU with Intel i7-9700K (3.60GHz) CPU and 32 GB RAM on Ubuntu 18.04.

3.1 Quantitative Analysis

Table 1 compares the mAP of each dataset and victim detector given benign examples (the 3rd column) and adversarial examples (the 4th-8th columns). The four attacks are TOG-vanishing, TOG-fabrication, TOG-mislabeling with ML and LL targets, and TOG-universal. Compared to the benign mAP, all four attacks drastically reduce the mAP of the victim detector. For instance, TOG-vanishing attacks on VOC and COCO break down the detection capability of the three YOLOv3 detectors: YOLOv3-D (VOC), YOLOv3-M (VOC) and YOLOv3-D (COCO), by reducing their mAP from 83.43%, 72.51% and 54.89% to 0.31%, 0.58%, and 0.51% respectively. Also, the TOG-mislabeling (LL) attacks collapse the mAP of all cases to less than 2%. Due to the space limit, we only report the comparison of the four victim detectors with respect to four TOG attacks on the VOC dataset and the YOLOv3 detector with a Darknet53 backbone (YOLOv3-D) on the COCO dataset.

It is worth noting that the above adversarial vulnerabilities are not limited to the detection capability for just a few classes but equally detrimental to any class supported by the victim detector. Figure 4 shows the average precision (AP) of all VOC classes on

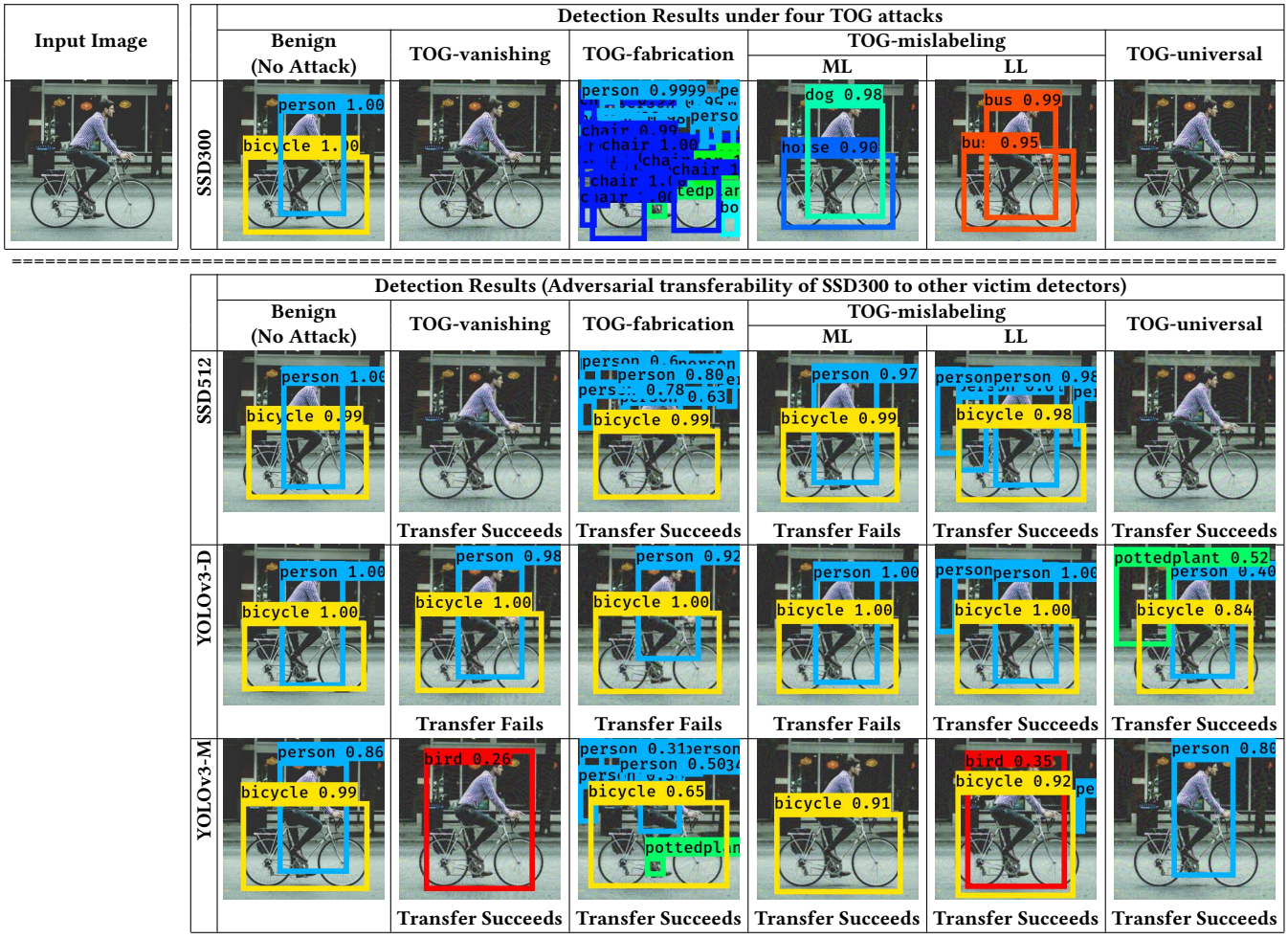


Table 2: Qualitative analysis (1st row) and adversarial transferability of TOG attacks (2nd-4th rows).

YOLOv3-M. Compared to the case with no attack (the benign case with blue curves), all four TOG attacks are shockingly successful in bringing down the APs of the victim detector, and the TOG-vanishing, TOG-fabrication, TOG-mislabeling attacks can drastically reduce the victim detector to very small or close to zero APs.

From the "TOG-universal" column in Table 1 and Figure 4(d), we make two additional observations. First, both show that our universal attacks reduce the mAP of all four detectors significantly with the most noticeable reduction in YOLOv3-D on VOC, with only a low mAP of 14.16%, compared with the high mAP of 83.43% with no attack (benign mAP). Second, it also shows that the TOG-universal attacks are less effective compared to the other three TOG attacks. This is because the other three TOG attacks are generated with per-input optimization, and the TOG-universal attack generates a single universal perturbation through offline training for each victim detector, and then it is applied in real-time to any input sent to the victim detector without per-input fine-tuning optimization. For example, the TOG-universal for VOC on YOLOv3-M generates the universal perturbation offline in 8 hours but can be applied as a black-box attack to the victim detector in only 0.00136 seconds, compared to 0.37 seconds for TOG-vanishing attack online.

3.2 Qualitative Analysis

Given that all four attacks in TOG significantly reduce mAP, we dedicate this subsection to perform qualitative analysis on the intrinsic behavior of each attack and explain how the detection capability of a victim detector is stripped off.

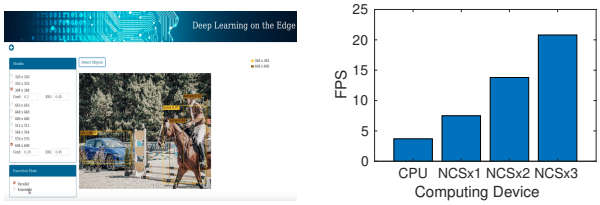
The top part of Table 2 shows a test image (left) of a person riding a bicycle with the detection results made by SSD300 on benign (the "Benign (No Attack)" column) and adversarial examples generated by four attacks (from "TOG-vanishing" to "TOG-universal" columns). Comparing the detection results on the benign example with the five adversarial counterparts (first row) for SSD300, we made a number of interesting observations. (1) The adversarial examples generated by TOG-vanishing and TOG-universal attacks both successfully remove the victim detector's capability in detecting objects (i.e., the person and the bicycle cannot be detected

anymore), even though the TOG-vanishing attack generates its adversarial perturbation tailored for this specific input image, while TOG-universal uses a pretrained universal perturbation. (2) For the TOG-fabrication attack, it fools the victim detector to give a large number of imprecise object detections (bounding boxes), all with high confidence, successfully tricks the victim detector. From the information retrieval perspective, our TOG-vanishing and TOG-universal attacks have a significant impact on the recall of the victim detector (unable to detect any object). In contrast, the TOG-fabrication attack fools the victim detector to have a much lower precision because the detection results contain a larger number of bounding boxes without objects ("false objects" are everywhere). (3) The TOG-mislabeling attacks (ML and LL) aim to disguise its true intent by making the victim detector to detect the same set of bounding boxes as those on benign examples under no attack (camouflage) but mislabel some or all detected objects with incorrect classes as demonstrated in both the "TOG-mislabeling (ML)" and the "TOG-mislabeling (LL)" columns in Table 2. For instance, the person on a bicycle is mislabeled as a dog on a horse with high confidence under the TOG-mislabeling (ML) attack. For the TOG-mislabeling (LL) attack, the two objects are both mislabeled as a bus with at least 95% confidence. Although camouflaged with bounding boxes, TOG-mislabeling attacks successfully bring down the precision of the victim detector, because the detected bounding boxes are associated with wrong labels.

3.3 TOG Attack Transferability

We dedicate this subsection to study the transferability of TOG attacks by generating adversarial examples on SSD300 and sending them to the other three detectors: SSD512, YOLOv3-D, and YOLOv3-M. We study whether the malicious perturbation generated from the attack to one victim detector can be effectively used as the black-box attack to fool the others. Table 2 (the 2nd-4th rows) visualizes the detection results transferring the adversarial examples attacking SSD300 to the other three victim detectors.

First, with no attack, all three detectors can correctly identify the person and the bicycle upon receiving the benign input (the 1st column). Second, the TOG attacks have different degrees of adversarial transferability for different victim detectors under different attacks. Consider the victim detector SSD512, both TOG-vanishing and TOG-universal can perfectly transfer the attack to SSD512 with the same effect (i.e., no object is detected). For TOG-fabrication, we observe that while the number of false objects is not as much as in the SSD300 case, a fairly large number of fake objects are wrongly detected by SSD512. The TOG-mislabeling (LL) attack is transferred to SSD512 but with the object-fabrication effect instead, while the TOG-mislabeling (ML) attack failed to transfer for this example. Now consider YOLOv3-D and YOLOv3-M, the TOG-universal and the TOG-mislabeling (LL) attacks are successful in transferability for both victim detectors but with different attack effects, such as wrong or additional bounding boxes or wrong labels. Also, the attacks from SSD300 can successfully transfer to YOLOv3-M with different attack effects compared to the attack results in SSD300. However, only the universal attack from SSD300 succeeds in transferring, but the other three TOG attacks failed to transfer to YOLOv3-D for this example. Note that with adversarial transferability, the attacks are black-box, generated, and launched without any prior knowledge of the three victim detectors.



(a) A screenshot of the robust object detection system. **(b) The frame per second (FPS) running ensemble detections.**

Figure 5: Robust real-time object detection using Intel NCS2.

4 CONCLUSION

We have presented TOG, a family of targeted adversarial objectness gradient attacks on deep object detection networks executing in edge systems. TOG attacks enable adversaries to generate human-imperceptible perturbations, either by employing adversarial perturbation optimized for each input or by offline training a universal perturbation that is effective on any inputs to the victim detector. We also studied the adversarial transferability from one victim detector to others through black-box attacks. Through experiments on two benchmark datasets and four popular deep object detectors, we show the serious adversarial vulnerabilities of the representative deep object detection networks when deploying in edge systems.

From our experiences and experimental study on different victim detectors and on the adversarial transferability, we observe the divergence of attack effects on different detectors. In general, an adversarial example attacking a victim model may not have the same adverse effect when used as a black-box attack based on its transferability. This is because the weak spot of attack transferability may vary from one detector to another trained by using a diverse DNN structure or diverse DNN algorithms as identified in [1, 7]. Our ongoing work is to develop diversity-enhanced ensemble object detection systems that promote strong robustness guarantees for defensibility and resilience against TOG attacks. Our preliminary robustness study with Intel Neural Compute Stick 2 (NCS2) as the AI acceleration module on edge systems shows some encouraging results. Figure 5 shows a robust object detection edge system developed at DiSL, Georgia Institute of Technology, which offers real-time performance in an edge system using an ensemble of multiple object detectors. The alpha release of the open-source software package is accessible on GitHub at https://github.com/git-disl/DLEdge.

ACKNOWLEDGMENTS

This research is partially sponsored by NSF CISE SaTC 1564097 and an IBM faculty award. We also thank Gage Bosgieter and the Intel Artificial Intelligence Developer Program for providing this research with the Intel Neural Compute Sticks. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation or other funding agencies and companies mentioned above.

REFERENCES

[1] Ka-Ho Chow, Wenqi Wei, Yanzhao Wu, and Ling Liu. 2019. Denoising and Verification Cross-Layer Ensemble Against Black-box Adversarial Attacks. In *IEEE International Conference on Big Data*.

[2] Mark Everingham, SM Ali Eslami, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. 2015. The pascal visual object classes challenge: A retrospective. *International Journal of Computer Vision* 111, 1 (2015), 98–136.

[3] Vandit Gajjar, Ayesha Gurnani, and Yash Khandhediya. 2017. Human detection and tracking for video surveillance: A cognitive science approach. In *IEEE International Conference on Computer Vision*.

[4] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. 2015. Explaining and harnessing adversarial examples. In *International Conference on Learning Representations*.

[5] Intel. 2019. *Intel Neural Compute Stick 2.* https://software.intel.com/en-us/neural-compute-stick

[6] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. 2014. Microsoft coco: Common objects in context. In *European Conference on Computer Vision*.

[7] Ling Liu, Wenqi Wei, Ka-Ho Chow, Margaret Loper, Emre Gursoy, Stacey Truex, and Yanzhao Wu. 2019. Deep neural network ensembles against deception: Ensemble diversity, accuracy and robustness. In *IEEE International Conference on Mobile Ad-Hoc and Smart Systems*.

[8] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C Berg. 2016. Ssd: Single shot multibox detector. In *European Conference on Computer Vision*.

[9] Brandon Reagen, Paul Whatmough, Robert Adolf, Saketh Rama, Hyunkwang Lee, Sae Kyu Lee, José Miguel Hernández-Lobato, Gu-Yeon Wei, and David Brooks. 2016. Minerva: Enabling low-power, highly-accurate deep neural network accelerators. In *IEEE International Symposium on Computer Architecture*.

[10] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. 2016. You only look once: Unified, real-time object detection. In *IEEE Conference on Computer Vision and Pattern Recognition*.

[11] Joseph Redmon and Ali Farhadi. 2017. YOLO9000: better, faster, stronger. In *IEEE Conference on Computer Vision and Pattern Recognition*.

[12] Joseph Redmon and Ali Farhadi. 2018. Yolov3: An incremental improvement. *arXiv preprint arXiv:1804.02767* (2018).

[13] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. 2015. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Neural Information Processing Systems*.

[14] Martin Simon, Karl Amende, Andrea Kraus, Jens Honer, Timo Samann, Hauke Kaulbersch, Stefan Milz, and Horst Michael Gross. 2019. Complexer-YOLO: Real-Time 3D Object Detection and Tracking on Semantic Point Clouds. In *IEEE Conference on Computer Vision and Pattern Recognition Workshops*.

[15] Wenqi Wei, Ling Liu, Margaret Loper, Stacey Truex, Lei Yu, Mehmet Emre Gursoy, and Yanzhao Wu. 2018. Adversarial examples in deep learning: Characterization and divergence. *arXiv preprint arXiv:1807.00051* (2018).