Beyond PhantomSponges
Enhancing Sponge Attack on Object Detection Models

Schoof, Coen; Koffas, Stefanos; Conti, Mauro; Picek, Stjepan

DOI
10.1145/3649403.3656485

Publication date
2024

Document Version
Final published version

Published in
WiseML 2024 - Proceedings of the 2024 ACM Workshop on Wireless Security and Machine Learning

Beyond PhantomSponges: Enhancing Sponge Attack on Object Detection Models

Coen Schoof
Radboud University
Nijmegen, the Netherlands
coen.schoof@ru.nl

Stefanos Koffas
Delft University of Technology
Delft, the Netherlands
s.koffas@tudelft.nl

Mauro Conti
University of Padua
Padua, Italy
mauro.conti@unipd.it

Stjepan Picek
Radboud University
Nijmegen, the Netherlands
Delft University of Technology
Delft, the Netherlands
stjepan.picek@ru.nl

ABSTRACT

Given today's ongoing deployment of deep learning models, ensuring their security against adversarial attacks has become paramount. This paper introduces an enhanced version of the PhantomSponges attack by Shapira et al. The attack exploits the non-maximum suppression (NMS) algorithm in YOLO object detection (OD) models without compromising OD, substantially increasing inference time. Our enhancement focuses on improving the attack's impact on YOLOv5 models by modifying its bounding box area loss term, aiming to directly decrease the intersection over union and, thus, exacerbate the computational load on NMS. Through a parameter study using the Berkeley Deep Drive dataset, we evaluate the enhanced attack's efficacy against various sizes of YOLOv5, demonstrating, under certain circumstances, an improved capability to increase NMS time with a minimal loss in OD accuracy. Furthermore, we propose a novel defense that dynamically resizes input images to mitigate the attack's effectiveness, showcasing a substantial restoration in inference speed and OD accuracy. Our findings show that the enhanced attack could result in a 550% increase in NMS time on the YOLOv5 small configuration. Moreover, our defense's results show a substantial decrease of 90.18% in NMS execution time when applied to an attacked YOLOv5 large model.

CCS CONCEPTS

• **Computing methodologies → Neural networks**; • **Security and privacy → Systems security**.

KEYWORDS

Object Detection, Adversarial Machine Learning, Sponge Attacks

ACM Reference Format:
Coen Schoof, Stefanos Koffas, Mauro Conti, and Stjepan Picek. 2024. Beyond PhantomSponges: Enhancing Sponge Attack on Object Detection Models. In *Proceedings of the 2024 ACM Workshop on Wireless Security and Machine Learning (WiseML '24), May 31, 2024, Seoul, Republic of Korea.* ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/3649403.3656485

WiseML '24, May 31, 2024, Seoul, Republic of Korea
© 2024 Copyright held by the owner/author(s).
ACM ISBN 979-8-4007-0602-8/24/05.
https://doi.org/10.1145/3649403.3656485

1 INTRODUCTION

The rapid adoption of deep learning models has led research towards adversarial attacks that impact their confidentiality, integrity, and availability [6]. Notably, recent findings have revealed vulnerabilities in object detection (OD) deep learning models, specifically in terms of their integrity [2, 5, 13, 15].

Although initially, the focus was primarily on attacks compromising the integrity of OD models, the discovery of attacks targeting model availability has introduced a new dimension of vulnerabilities. Commonly, OD models consist of a pipeline containing preprocessing, candidate filtering based on confidence scores, and applying non-maximum suppression (NMS) [4]. PhantomSponges [10] attacks this process by exploiting the worst-case complexity of the NMS algorithm often contained within You Only Look Once (YOLO) OD model pipelines. PhantomSponges creates a universal adversarial perturbation (UAP) that can be superimposed on input data. Consequently, when feeding this perturbed data through the OD model, the duration of the NMS algorithm increases, thereby slowing down the model's inference time. This work of Shapira et al. [10] marked a seminal point as it is the first to target the availability of an OD model. In practice, targeting the availability could have implications for real-world applications that rely on timely OD, such as autonomous driving systems and surveillance, where delays in OD can have severe consequences. Their work not only demonstrated a vulnerability in a popular OD model but also provided an opportunity for further investigation into the resilience of OD models against such attacks. Research on enhancing the PhantomSponges attack or developing targeted defenses against such efficiency-impacting adversarial tactics remains limited. Our work advances our understanding of how such attacks work and how to defend against them. Our contributions are:

- We introduce an enhanced version of the PhantomSponges attack [10] that is up to 550% more effective than the original attack.
- We propose a defense strategy that can mitigate the impact posed by the enhanced PhantomSponges attack, yielding up to a 90.18% decrease in NMS's execution time.

2 BACKGROUND

2.1 Object Detection

OD is a computer vision task that identifies and locates objects within images. The process typically involves several key steps: pre-processing input images, feature extraction, object proposal, classification, and post-processing. Here, we focus on the YOLO model, popular for its efficiency and accuracy in real-time OD tasks [9].

YOLO Object Detection Pipeline. The YOLO framework simplifies the traditional OD pipeline into a single neural network. This network divides the input image into a grid, and each grid cell is responsible for predicting objects that center within it. The main steps in the YOLO pipeline include:

- **Pre-processing:** Input images are resized to match the network's expected input dimensions.
- **Prediction:** The neural network predicts bounding boxes and their class probabilities simultaneously. Each grid cell predicts multiple bounding boxes, their confidence scores (reflecting the accuracy of the box), and class probabilities.
- **Non-maximum Suppression:** To reduce redundancy and discard improbable bounding boxes, NMS is applied. This step selects the most probable bounding box when multiple boxes predict the same object.

Vulnerability to Adversarial Examples. Despite their effectiveness, YOLO and similar OD models are susceptible to adversarial examples. Adversarial examples are inputs to a machine learning model that an attacker has intentionally designed to cause the model to make a mistake. In the context of OD, such inputs can lead to missed detections or misclassifications. Recent studies [2, 5, 13, 15] have demonstrated the susceptibility of models like YOLO to such attacks. For instance, Thys et al. [13] discovered that placing adversarial patches in images could render people invisible to YOLO detectors.

2.2 Non-maximum Suppression

NMS aims to discard redundant bounding boxes. It does so by discarding bounding boxes that overlap with one another beyond a predefined threshold T_{IoU}. The intersection over union (IoU) determines the overlap. As shown in Algorithm 1, first, all bounding boxes are sorted on their confidence scores, which are a product of F as shown in Equation (1). Next, for each bounding box b_i, we discard all other bounding boxes b_j if their IoU exceeds the threshold T_{IoU}. This process repeats until no more bounding boxes can be discarded. Intuitively, this leaves the bounding boxes that are sufficiently apart from one another and have the highest confidence score.

Algorithm 1 Non-maximum Suppression

```
 1: Start with a list of boxes $B$, and confidence scores $S$.
 2: Define a threshold $T_{IoU}$.
 3: Sort $B$ by $S$ in decreasing order.
 4: for each box $b_i$ in $B$ do
 5:     for each subsequent box $b_j$ in $B$ do
 6:         if IoU($b_i$, $b_j$) > $T_{IoU}$ then
 7:             Remove $b_j$ from $B$.
 8:         end if
 9:     end for
10: end for
11: Return the remaining boxes in $B$.
```

2.3 Adversarial Machine Learning

One of the attacks first introduced in AI is the evasion attack [12], modifying the model's input at inference to generate incorrect results, targeting the model's integrity. Later, backdoor attacks were introduced where an adversary inserts a secret functionality into a trained model that can be activated during inference, mostly affecting the model's integrity [3]. Untargeted poisoning attacks were the first to target the model's availability by decreasing the model's performance and causing a denial of service. However, recently, a new category of attacks targeting the model's availability has been introduced: sponge attacks [1, 7, 8, 10, 11]. Sponge attacks compromise the model's availability by increasing the latency or energy consumption required during inference. Sponge attacks can be executed either during training (e.g., sponge poisoning [1]), at inference time (e.g., sponge examples [11]), or by altering the weights of a trained model [7].

2.4 PhantomSponges Attack

Among the attacks on OD models, the PhantomSponges attack, introduced in [10], stands out for its novel approach to increasing YOLO's inference time. YOLO outputs a 3D tensor for every image containing the coordinate offsets of bounding boxes, the objectness score, and the class score. The objectness score refers to the model's confidence that the bounding box contains an object, whereas the class score refers to the model's confidence that a given class is contained within the bounding box. The PhantomSponges attack is focused on, among other versions, YOLOv5 [10].

For the PhantomSponges attack, the authors exploited the NMS algorithm to impact the target model's availability. The attack generates UAPs that, when superimposed onto input images, significantly slow down the OD process without substantially affecting detection accuracy. The perturbed images achieve this by tricking YOLO into generating "phantom" bounding boxes, which overload the NMS algorithm and thus increase the model's inference time. This attack reveals a previously unexplored vulnerability in the efficiency of OD models. Before the NMS stage, the predicted candidates are filtered based on a predefined threshold T_{conf}. Specifically, this is done as follows:

$$F = \{c_{obj\ score} \cdot max\{c_{class\ score\ i}\}_{i=0}^{N_c} > T_{conf} | c \in C\}. \quad (1)$$

The objectness score of each bounding box ($c_{obj\ score}$) is multiplied by its highest class score ($max\{c_{class\ score\ i}\}_{i=0}^{N_c}$). In turn, these new scores must exceed a predefined threshold T_{conf} to proceed to the NMS stage of YOLO, where redundant bounding boxes are removed. In line with the original model and the work by Shapira et al., we used $T_{conf} = 0.25$ [10, 14].

2.4.1 NMS's weakness. As previously mentioned, the NMS algorithm iterates over all bounding boxes b, ordered by confidence score. For each bounding box b, the algorithm calculates its overlap with every other bounding box. As a result, the runtime complexity of NMS is factorial ($O(b!)$), where b is the number of bounding boxes. However, for each iteration, some bounding boxes are discarded, decreasing the runtime complexity as the algorithm progresses. The attack aims to reduce the number of bounding boxes discarded for every bounding box b. With a successful attack, the NMS algorithm would take more time to complete. Thus, the YOLO's inference time would also be increased.

2.4.2 Creation of the UAP. We create the UAP (δ) by starting from a black image and iteratively optimizing its pixel values using projected gradient descent (PGD) with the L_2 norm. With PGD, we create a perturbed image x' (x is the original image that could be any image in the dataset) so that it incurs the lowest loss. To be able to obtain stealthy perturbations, we constrain the domain of x' to S, where $S = \{x' : \|x - x'\|_p < \epsilon\}$. In short, we constrain the perturbation (the difference between x and x') to ϵ, where the difference is expressed in a p-norm (p) of choice (in our case L_2). The aforementioned loss consists of three terms: the max-objects loss, the bounding box area loss, and the IoU loss.

Max-objects loss: This loss term increases the number of candidate bounding boxes passed to the NMS algorithm. It does so by decreasing the number of bounding boxes discarded by F (defined in Equation (1)) as follows:

$$\ell_{single\ conf}(c') = T_{conf} - c'_{obj\ score} \cdot max\{c'_{class\ score\ i}\}_{i=0}^{N'_c}. \quad (2)$$

$C' = \{c'_1, c'_2, \dots c'_N\}$ is the set of bounding boxes produced using the perturbed image. The second term in Equation (2) is the output from the filter F for the perturbed image's bounding boxes. Thus, with this loss function, we aim to maximize the number of bounding boxes whose output from F exceeds T_{conf}, making them eligible to be processed by the NMS algorithm.

Over the set of bounding boxes that F discarded ($C' \setminus F(C')$), we calculate the sum of single bounding box losses $\ell_{single\ conf}(c')$, for all $c' \in C'$. We do this as we want the loss to lower when more bounding boxes are being passed to the NMS stage:

$$\ell_{max\ objects} = \frac{1}{|C'|} \cdot \sum_{c' \in C' \setminus F(C')} \ell_{single\ conf}(c'). \quad (3)$$

Bounding box area loss: As mentioned earlier, the attack increases the time needed for the NMS algorithm to process all bounding boxes. The bounding box area loss term aims to decrease the IoU by optimizing bounding boxes to be as small as possible, thereby making the IoU of two bounding boxes less likely to exceed T_{IoU}. Specifically, the loss is the combined areas of all bounding boxes, averaged over $F(C')$ as follows:

$$\ell_{bbox\ area} = \frac{1}{|F(C')|} \cdot \sum_{c' \in F(C')} \ell_{single\ area}(c'). \quad (4)$$

IoU loss: To preserve the original detections, this loss term aims to maximize the IoU of the post-NMS bounding boxes produced given the perturbed image and the post-NMS bounding boxes produced given a clean image. To adjust the function to minimize the loss, we calculate:

$$\ell_{single\ IoU}(c) = 1 - Max\ IoU(c). \quad (5)$$

The final loss function is as follows:

$$\ell_{total} = \lambda_1 \cdot \ell_{max\ objects} + \lambda_2 \cdot \ell_{bbox\ area} + \lambda_3 \cdot \ell_{max\ IoU}. \quad (6)$$

3 METHODOLOGY

3.1 Threat Model

Identically to [10], the attacker aims to disrupt an OD system by increasing NMS's execution time, leading to slower model inference without affecting its detection accuracy. To this end, the adversary needs access to a subset of the training data and a copy of the target OD model to generate the UAP. The adversary should be able to craft the UAP by using the local copy of the model and the subset of the training data and apply it to the model's input images.

3.2 Enhanced PhantomSponges Attack

To enhance the original PhantomSponges attack, we replaced the bounding box area loss with a loss that aims to decrease the IoU of a given bounding box and every other bounding box instead of just decreasing the bounding box area. Specifically, the loss was determined by calculating the average IoU of all unique combinations of bounding boxes for a given image. In this way, for n bounding boxes, the output of the IoU function would contain n^2 entries. Preliminary experiments showed that the output of the IoU function could exceed the available working memory. Therefore, we decided that if the memory for a given image were exceeded, the original box area loss would be determined. Although this enhanced bounding box area loss term increases training time, it should not be detrimental to the usability of the enhanced PhantomSponges, as the UAP is only generated once and generalizes to the dataset to be used.

3.3 Defense against Enhanced Attack

The defense resizes the y-axis of the input images using a uniformly random scalar within a predetermined domain γ. We apply the resizing after the perturbation is added to the input image. This mitigates the effectiveness of the attack. We chose a random value within a predefined domain instead of a static value so that the attacker would not be able to ascertain the scalar value and train a new perturbation based on the new y-axis size of the image.

4 EXPERIMENTAL SETUP

In our experiments, we used four types of the YOLOv5 OD model, i.e., nano, small, medium, and large [14]. These types describe the complexity of the models in terms of parameter count. The UAPs were trained using the Berkeley Deep-Drive dataset [16]. Averages of the following metrics were calculated for evaluation:

- $|F(C')|$: the number of candidates passed to the NMS stage.
- t: the end-to-end pipeline's total processing time (ms).
- t_{NMS}: the processing time of the NMS stage (ms).
- Recall: the percentage of original objects detected in the perturbed image.

Similar to the original PhantomSponges paper, we evaluated 30 iterations for each image and calculated its average. We conduct a parameter study based on the following variables: ϵ, λ_1, λ_2, λ_3, model size, presence/absence of original or enhanced bounding box area loss term, and γ.

Moreover, we ran our experiments on an Ubuntu 22.04.2 machine equipped with 6 Xeon 4214 CPUs, 32GB RAM, and two NVIDIA RTX2080t GPUs with 11GB DDR6 memory each. The code for the neural networks is developed with PyTorch 2.1. Our code is made available on GitLab.[1]

5 RESULTS

In Tables 1, 3, 5 and 7, we show the results for our experiments using the original attack (baseline). In Tables 2, 4, 6 and 8, we show the results for the enhanced attack for the nano, small, medium, and large model versions, respectively.

Table 1: Results of the original attack on YOLOv5 nano

		γ	$\lambda_2 = 10$	$\lambda_2 = 20$
			Total time , NMS time , $\lvert F(C')\rvert$, Recall	
	Clean		9.8,0.9,85,1.0	
	λ_1, λ_3	[0.65, 0.8)	10.0,0.9,106,0.814	12.5,1.1,90,0.849
	=	[0.8, 0.95)	9.6,0.9,372,0.833	11.3,1.1,89,0.905
	0.6, 0.4	No resize	9.8,1.5,3489,0.76	10.9,1.1,82,0.982
		[1.05, 1.2)	9.5,1.0,1152,0.721	11.1,1.1,75,0.805
ϵ		[1.2, 1.35)	9.5,1.0,1069,0.603	11.0,1.0,65,0.672
=	λ_1, λ_3	[0.65, 0.8)	12.2,1.1,94,0.792	12.1,1.1,95,0.781
30	=	[0.8, 0.95)	11.5,1.1,195,0.792	11.5,1.1,254,0.793
	0.7, 0.35	No resize	12.1,2.1,4990,0.682	11.9,2.0,4526,0.68
		[1.05, 1.2)	11.3,1.2,976,0.702	11.3,1.2,1024,0.686
		[1.2, 1.35)	11.2,1.1,776,0.585	11.2,1.1,833,0.568
	λ_1, λ_3	[0.65, 0.8)	11.2,1.1,130,0.763	11.3,1.1,88,0.839
	=	[0.8, 0.95)	11.3,1.1,556,0.759	11.4,1.1,86,0.887
	0.6, 0.4	No resize	11.5,1.7,3763,0.691	11.0,1.1,79,0.979
		[1.05, 1.2)	11.1,1.2,1101,0.673	11.1,1.1,71,0.799
ϵ		[1.2, 1.35)	11.0,1.2,919,0.546	11.0,1.0,60,0.653
=	λ_1, λ_3	[0.65, 0.8)	11.1,1.0,80,0.658	11.2,1.1,85,0.679
70	=	[0.8, 0.95)	11.4,1.1,223,0.628	11.4,1.1,192,0.65
	0.7, 0.3	No resize	13.6,3.8,8825,0.429	13.6,3.7,8740,0.457
		[1.05, 1.2)	11.2,1.2,888,0.524	11.1,1.2,723,0.555
		[1.2, 1.35)	11.1,1.1,578,0.43	11.1,1.1,324,0.468

Table 2: Results of the enhanced attack on YOLOv5 nano

		γ	$\lambda_2 = 10$	$\lambda_2 = 20$
			Total time , NMS time , $\lvert F(C')\rvert$, Recall	
	Clean		11.0,1.1,83,1.0	
	λ_1, λ_3	[0.65, 0.8)	10.6,0.9,100,0.792	12.1,1.1,109,0.805
	=	[0.8, 0.95)	9.7,0.9,230,0.796	11.5,1.1,257,0.809
	0.6, 0.4	No resize	10.1,1.7,4194,0.746	11.6,1.8,3949,0.773
		[1.05, 1.2)	9.6,1.0,736,0.695	11.2,1.2,886,0.705
ϵ		[1.2, 1.35)	9.5,0.9,472,0.584	11.1,1.1,583,0.582
=	λ_1, λ_3	[0.65, 0.8)	12.1,1.1,102,0.799	12.0,1.1,104,0.791
30	=	[0.8, 0.95)	11.4,1.1,227,0.795	11.4,1.1,294,0.805
	0.7, 0.3	No resize	11.9,2.0,4568,0.726	11.9,2.1,4543,0.741
		[1.05, 1.2)	11.2,1.2,791,0.688	11.2,1.2,926,0.69
		[1.2, 1.35)	11.2,1.1,593,0.584	11.0,1.1,715,0.575
	λ_1, λ_3	[0.65, 0.8)	11.3,1.1,91,0.691	11.2,1.1,84,0.67
	=	[0.8, 0.95)	11.5,1.1,223,0.695	11.4,1.1,183,0.657
	0.6, 0.4	No resize	12.9,3.0,7315,0.567	12.9,3.1,7646,0.555
		[1.05, 1.2)	11.2,1.1,562,0.574	11.2,1.1,525,0.541
ϵ		[1.2, 1.35)	11.1,1.1,282,0.475	11.1,1.1,278,0.454
=	λ_1, λ_3	[0.65, 0.8)	11.3,1.1,83,0.678	11.2,1.1,81,0.667
70	=	[0.8, 0.95)	11.5,1.1,166,0.66	11.4,1.1,182,0.65
	0.7, 0.3	No resize	13.6,3.6,8615,0.55	13.6,3.8,9021,0.511
		[1.05, 1.2)	11.3,1.2,621,0.552	11.2,1.2,580,0.544
		[1.2, 1.35)	11.1,1.1,260,0.449	11.1,1.1,372,0.448

[1]https://gitlab.science.ru.nl/cschoof/PhantomSponges

Table 3: Results of the original attack on YOLOv5 small

		γ	$\lambda_2 = 10$	$\lambda_2 = 20$
			Total time , NMS time , $\lvert F(C')\rvert$, Recall	
	Clean		9.4,0.8,103,1.0	
	λ_1, λ_3	[0.65, 0.8)	9.6,0.9,138,0.846	9.6,0.9,123,0.855
	=	[0.8, 0.95)	9.8,1.0,780,0.849	9.7,1.0,814,0.842
	0.6, 0.4	No resize	11.1,2.2,5850,0.618	11.4,2.6,6549,0.543
		[1.05, 1.2)	10.0,1.1,2411,0.711	10.1,1.2,2782,0.671
ϵ		[1.2, 1.35)	9.6,1.0,1589,0.619	9.6,1.0,1790,0.593
=	λ_1, λ_3	[0.65, 0.8)	9.5,0.9,121,0.842	9.7,0.9,128,0.83
30	=	[0.8, 0.95)	9.6,0.9,461,0.853	9.8,1.0,882,0.814
	0.7, 0.3	No resize	10.7,1.9,5201,0.592	11.7,2.9,7100,0.53
		[1.05, 1.2)	9.9,1.0,1849,0.73	10.3,1.3,3090,0.623
		[1.2, 1.35)	9.5,0.9,1140,0.622	9.8,1.1,2272,0.574
	λ_1, λ_3	[0.65, 0.8)	9.4,0.9,150,0.809	9.5,0.9,116,0.902
	=	[0.8, 0.95)	9.6,1.0,1027,0.799	9.6,0.9,114,0.921
	0.6, 0.4	No resize	11.6,2.8,7291,0.522	9.5,0.8,107,0.988
		[1.05, 1.2)	9.9,1.2,2805,0.666	9.6,0.8,98,0.85
ϵ		[1.2, 1.35)	9.4,1.0,1791,0.595	9.3,0.8,84,0.731
=	λ_1, λ_3	[0.65, 0.8)	9.5,0.9,163,0.728	9.7,0.9,129,0.709
70	=	[0.8, 0.95)	9.8,1.0,1435,0.702	9.8,1.0,1061,0.691
	0.7, 0.3	No resize	16.4,7.5,11972,0.403	14.7,5.8,11638,0.317
		[1.05, 1.2)	10.4,1.6,4404,0.527	10.4,1.5,3938,0.519
		[1.2, 1.35)	9.9,1.3,3262,0.485	9.9,1.2,2881,0.466

Table 4: Results of the enhanced attack performed on YOLOv5 small

			$\lambda_2 = 10$	$\lambda_2 = 20$
			Total time , NMS time , $\lvert F(C')\rvert$, Recall	
	Clean		9.5,0.8,102,1.0	
	λ_1, λ_3	[0.65, 0.8)	9.4,0.9,140,0.842	9.3,0.9,159,0.832
	=	[0.8, 0.95)	9.6,0.9,581,0.838	9.6,1.0,797,0.841
	0.6, 0.4	No resize	11.4,2.7,6686,0.676	11.1,2.5,6330,0.695
		[1.05, 1.2)	9.8,1.1,2283,0.7	9.9,1.2,2715,0.707
ϵ		[1.2, 1.35)	9.4,1.0,1630,0.596	9.5,1.0,1941,0.607
=	λ_1, λ_3	[0.65, 0.8)	9.4,0.8,116,0.808	9.3,0.9,124,0.8
30	=	[0.8, 0.95)	9.5,0.9,496,0.789	9.4,0.9,784,0.806
	0.7, 0.3	No resize	11.9,3.3,7825,0.643	12.0,3.6,8420,0.618
		[1.05, 1.2)	9.9,1.2,2705,0.66	9.9,1.4,3389,0.654
		[1.2, 1.35)	9.5,1.0,1953,0.563	9.3,1.1,2368,0.572
	λ_1, λ_3	[0.65, 0.8)	9.5,0.9,131,0.803	9.8,0.9,175,0.753
	=	[0.8, 0.95)	9.7,0.9,506,0.806	9.9,1.0,907,0.746
	0.6, 0.4	No resize	10.9,2.1,5807,0.656	14.2,5.2,11109,0.541
		[1.05, 1.2)	9.8,1.0,1763,0.716	10.2,1.2,3024,0.614
ϵ		[1.2, 1.35)	9.5,0.9,926,0.618	9.7,1.0,1767,0.533
=	λ_1, λ_3	[0.65, 0.8)	9.7,0.9,158,0.719	9.6,0.9,137,0.711
70	=	[0.8, 0.95)	9.8,1.0,985,0.714	9.8,1.0,870,0.708
	0.7, 0.3	No resize	15.5,6.6,12485,0.512	16.2,7.2,12478,0.511
		[1.05, 1.2)	10.2,1.3,3276,0.567	10.3,1.4,3500,0.565
		[1.2, 1.35)	9.7,1.1,2135,0.503	9.7,1.0,2103,0.493

5.1 Influence of ϵ on the Attack Effectiveness

Larger ϵ values resulted in a less stealthy but possibly more effective UAP (higher NMS time). This is reflected by our results from experiments using all sizes of YOLOv5, as well as regarding the original and enhanced attack (see Tables 1, 3, 5, 7, 2, 4, 6, and 8). Similar to the original paper, a larger ϵ value generally yields an attack for which the NMS time was higher, but the recall was lower. For example, this phenomenon occurred in Table 5, where, for $\lambda_1 = 0.6, \lambda_2 = 10, \lambda_3 = 0.4$, there was an increase in NMS time (3.2 to 4.1) but a decrease in recall (0.542 to 0.399). We believe the lowered recall to be related to the additional noise introduced by the UAP, complicating the detection of the original objects.

Table 5: Results of the original attack on YOLOv5 medium

		γ	$\lambda_2 = 10$	$\lambda_2 = 20$
			Total time , NMS time , $\lvert F(C')\rvert$, Recall	
	Clean		12.1,0.8,117,1.0	
	λ_1, λ_3	[0.65, 0.8)	12.1,0.9,229,0.87	12.1,0.9,267,0.868
	=	[0.8, 0.95)	12.4,1.1,1561,0.86	12.4,1.1,1583,0.85
	0.6, 0.4	No resize	14.7,3.2,8057,0.542	14.9,3.4,8424,0.472
		[1.05, 1.2)	13.2,1.7,4388,0.682	13.2,1.7,4511,0.645
ϵ		[1.2, 1.35)	12.7,1.5,3823,0.632	12.8,1.6,4188,0.587
=	λ_1, λ_3	[0.65, 0.8)	12.3,0.9,311,0.867	9.7,0.9,128,0.83
30	=	[0.8, 0.95)	12.6,1.2,2061,0.821	9.8,1.0,882,0.814
	0.7, 0.3	No resize	15.7,4.1,9672,0.45	11.7,2.9,7100,0.53
		[1.05, 1.2)	13.7,2.0,5452,0.58	10.3,1.3,3090,0.623
		[1.2, 1.35)	13.2,1.7,4799,0.559	9.8,1.1,2272,0.574
	λ_1, λ_3	[0.65, 0.8)	11.9,0.9,434,0.814	11.7,0.9,468,0.814
	=	[0.8, 0.95)	12.5,1.2,2537,0.758	12.2,1.2,2484,0.761
	0.6, 0.4	No resize	15.4,4.1,9982,0.399	14.6,3.5,8775,0.395
		[1.05, 1.2)	13.4,2.0,5656,0.561	13.0,1.9,5367,0.538
ϵ		[1.2, 1.35)	12.7,1.6,4691,0.535	12.6,1.7,4956,0.502
=	λ_1, λ_3	[0.65, 0.8)	12.4,0.9,462,0.782	12.3,0.9,491,0.778
70	=	[0.8, 0.95)	12.9,1.4,2852,0.717	12.8,1.3,2691,0.714
	0.7, 0.3	No resize	18.9,7.3,13059,0.309	18.6,6.8,12945,0.307
		[1.05, 1.2)	14.4,2.6,6928,0.463	14.3,2.5,6764,0.465
		[1.2, 1.35)	13.6,2.1,5949,0.439	13.6,2.1,5850,0.45

Table 6: Results of the enhanced attack on YOLOv5 medium

		γ	$\lambda_2 = 10$	$\lambda_2 = 20$
			Total time , NMS time , $\lvert F(C')\rvert$, Recall	
	Clean		12.3,0.8,113,1.0	
	λ_1, λ_3	[0.65, 0.8)	12.1,0.9,338,0.866	12.1,0.9,263,0.88
	=	[0.8, 0.95)	12.5,1.1,1712,0.864	12.4,1.1,1530,0.877
	0.6, 0.4	No resize	15.0,3.4,8407,0.617	15.0,3.5,8547,0.602
		[1.05, 1.2)	13.1,1.6,4262,0.71	13.1,1.6,4186,0.707
ϵ		[1.2, 1.35)	12.6,1.3,3382,0.646	12.6,1.3,3297,0.652
=	λ_1, λ_3	[0.65, 0.8)	12.1,0.9,194,0.87	11.7,0.9,192,0.863
30	=	[0.8, 0.95)	12.3,1.0,1347,0.869	12.0,1.0,1412,0.856
	0.7, 0.3	No resize	15.6,4.2,9467,0.595	15.3,4.2,9907,0.574
		[1.05, 1.2)	13.2,1.6,4333,0.711	12.7,1.6,4450,0.675
		[1.2, 1.35)	12.6,1.3,3474,0.643	12.2,1.3,3346,0.621
	λ_1, λ_3	[0.65, 0.8)	17.6,1.1,494,0.839	17.6,1.1,461,0.785
	=	[0.8, 0.95)	18.3,1.5,2952,0.79	18.2,1.5,2949,0.726
	0.6, 0.4	No resize	22.1,5.3,10876,0.541	24.3,7.5,13302,0.433
		[1.05, 1.2)	19.6,2.7,6598,0.605	19.7,2.8,7325,0.521
ϵ		[1.2, 1.35)	19.3,2.8,5758,0.579	19.4,2.9,5970,0.504
=	λ_1, λ_3	[0.65, 0.8)	12.2,0.9,257,0.792	12.2,0.9,242,0.77
70	=	[0.8, 0.95)	12.6,1.1,1877,0.758	12.5,1.1,2112,0.725
	0.7, 0.3	No resize	19.2,7.5,13173,0.467	20.8,9.3,14013,0.432
		[1.05, 1.2)	13.6,1.9,5655,0.591	13.6,2.1,6010,0.549
		[1.2, 1.35)	12.8,1.4,4287,0.554	12.6,1.4,4266,0.509

5.2 Influence of λ_1 and λ_3 on the Attack Effectiveness

Emphasizing λ_1 meant that, during the UAP's calculation, the max-objects loss term was emphasized more, and the max IoU loss term was emphasized less. This meant that the UAP was trained such that fewer bounding boxes of a perturbed image would be discarded by F, and preserving original objects was less emphasized. In line with the original paper, the attack became more powerful, but the recall was lower when λ_1 was emphasized. This phenomenon occurred for all combinations of attacks and YOLOv5 model sizes. Specifically, $\lambda_1 = 0.6, \lambda_3 = 0.4$ tend to yield superior recall but lower NMS time than $\lambda_1 = 0.7, \lambda_3 = 0.3$.

Table 7: Results of the original attack on YOLOv5 large

		γ	$\lambda_2 = 10$	$\lambda_2 = 20$
			Total time , NMS time , $\lvert F(C')\rvert$, Recall	
	Clean		17.7,1.1,128,1.0	
	λ_1, λ_3	[0.65, 0.8)	17.6,1.1,313,0.843	17.7,1.1,325,0.87
	=	[0.8, 0.95)	18.2,1.5,2491,0.785	18.4,1.5,2562,0.787
	0.6, 0.4	No resize	21.7,4.9,10365,0.416	21.9,4.9,10352,0.384
		[1.05, 1.2)	19.4,2.6,6315,0.55	19.7,2.7,6384,0.537
ϵ		[1.2, 1.35)	19.3,2.8,5551,0.535	19.5,2.8,5486,0.531
=	λ_1, λ_3	[0.65, 0.8)	17.3,1.1,322,0.838	17.5,1.1,364,0.85
30	=	[0.8, 0.95)	18.0,1.5,2704,0.773	18.2,1.6,2777,0.753
	0.7, 0.3	No resize	23.1,6.4,11921,0.37	23.0,6.2,11856,0.32
		[1.05, 1.2)	19.6,3.0,7176,0.507	19.9,3.0,7008,0.484
		[1.2, 1.35)	19.6,3.2,6451,0.469	19.7,3.1,6317,0.469
	λ_1, λ_3	[0.65, 0.8)	17.6,1.1,548,0.808	17.7,1.1,542,0.837
	=	[0.8, 0.95)	18.5,1.7,3311,0.721	18.3,1.6,3118,0.726
	0.6, 0.4	No resize	22.3,5.4,11270,0.342	22.2,5.3,10850,0.325
		[1.05, 1.2)	19.8,2.8,7060,0.488	19.7,2.7,6659,0.478
ϵ		[1.2, 1.35)	19.5,2.9,6042,0.486	19.5,2.9,5859,0.476
=	λ_1, λ_3	[0.65, 0.8)	17.7,1.1,582,0.765	17.6,1.1,515,0.797
70	=	[0.8, 0.95)	18.4,1.7,3669,0.669	18.4,1.7,3528,0.695
	0.7, 0.3	No resize	30.0,13.1,14606,0.289	23.1,6.4,12594,0.291
		[1.05, 1.2)	20.4,3.5,8541,0.423	20.1,3.3,7934,0.441
		[1.2, 1.35)	20.0,3.5,7472,0.421	19.8,3.3,6915,0.449

Table 8: Results of the enhanced attack on YOLOv5 large

		γ	$\lambda_2 = 10$	$\lambda_2 = 20$
			Total time , NMS time , $\lvert F(C')\rvert$, Recall	
	Clean		17.8,1.1,125,1.0	
	λ_1, λ_3	[0.65, 0.8)	17.5,1.1,306,0.844	17.7,1.1,367,0.866
	=	[0.8, 0.95)	18.1,1.4,2380,0.813	18.2,1.4,2336,0.835
	0.6, 0.4	No resize	22.5,5.7,11231,0.54	22.5,5.5,11153,0.544
		[1.05, 1.2)	19.5,2.6,6448,0.622	19.7,2.7,6639,0.614
ϵ		[1.2, 1.35)	19.4,2.9,5746,0.581	19.6,3.0,6079,0.577
=	λ_1, λ_3	[0.65, 0.8)	17.5,1.1,287,0.831	17.6,1.1,348,0.829
30	=	[0.8, 0.95)	18.2,1.4,2372,0.786	18.2,1.5,2755,0.785
	0.7, 0.3	No resize	24.6,7.7,12365,0.48	24.5,7.7,12695,0.494
		[1.05, 1.2)	19.8,2.9,7087,0.579	19.9,3.0,7492,0.57
		[1.2, 1.35)	19.6,3.1,6256,0.54	19.8,3.4,6904,0.528
	λ_1, λ_3	[0.65, 0.8)	17.6,1.1,494,0.839	17.6,1.1,461,0.785
	=	[0.8, 0.95)	18.3,1.5,2952,0.79	18.2,1.5,2949,0.726
	0.6, 0.4	No resize	22.1,5.3,10876,0.541	24.3,7.5,13302,0.433
		[1.05, 1.2)	19.6,2.7,6598,0.605	19.7,2.8,7325,0.521
ϵ		[1.2, 1.35)	19.3,2.8,5758,0.579	19.4,2.9,5970,0.504
=	λ_1, λ_3	[0.65, 0.8)	17.8,1.1,569,0.758	17.8,1.1,532,0.793
70	=	[0.8, 0.95)	18.8,1.8,3882,0.702	18.6,1.6,3278,0.731
	0.7, 0.3	No resize	28.4,11.2,14061,0.415	24.8,7.6,13134,0.455
		[1.05, 1.2)	20.5,3.3,8393,0.498	20.2,3.0,7650,0.529
		[1.2, 1.35)	19.9,3.1,7147,0.47	19.8,3.0,6570,0.511

5.3 Influence of λ_2 on the Attack Effectiveness

λ_2 determined the emphasis of the bounding box area loss term on the total loss. Again, similar to the original paper, a higher λ_2 resulted in more bounding boxes to be passed to the NMS stage. However, contrary to the findings of Shapira et al. [10], the recall lowered given a higher λ_2.

5.4 Influence of the Model Size

We presumed that higher model complexity would have led to more robustness against the attack, specifically regarding recall. The number of bounding boxes passed to the NMS algorithm overall was higher for larger variants of YOLOv5. Therefore, it could have been more likely for it to identify objects correctly. However, the model's recall was lower for more complex variants. The only difference between model variants were the 'depth_multiple' and 'width_multiple' hyperparameters, where the former determined the number of layers of the model, and the latter determined the number of filters in a layer. A more complex variant of YOLOv5 might, due to a more comprehensive understanding of the image content, have predicted more bounding boxes as they are likely more able to detect nuanced features in the image, possibly also leading to more false positives. Given that many bounding boxes passed to the NMS stage, it could have been more likely that phantom bounding boxes would overlap too much with a real bounding box, causing the real object to be discarded during the NMS stage, explaining the lower recall.

5.5 Original Attack vs. Enhanced Attack

We expected the enhanced bounding box area loss term to yield superior performance compared to the original loss term. This was reflected, to an extent, by our results.

For the nano variant of YOLOv5, the enhanced attack results (Table 2) showed little difference compared to the original attack results (Table 1). We believe this could be explained by nano's low complexity, which possibly made it less affected by the attacks as such models likely can process less nuanced features, therefore being less sensitive to the subtle manipulations introduced by the UAP. In contrast, larger models can capture complex features, making them more vulnerable.

In the context of the YOLOv5 small variant (see Tables 3 and 4), our enhanced attack with $\epsilon = 30$ exhibited a dual impact. On the one hand, it notably increased the NMS time and the count of bounding boxes. Moreover, it led to a concurrent improvement in recall. We posit that this dual impact phenomenon resulted from the increased complexity, enabling YOLOv5 to recognize more complex features in an image.

Given the medium variant, the dual-impact phenomenon tended to reoccur, although overall recall was lower than in the small and nano models. This suggests that a higher model complexity implies lower robustness against the attack. Similar results were given for the large model experiments, increasing the number of bounding boxes generated and lowering the overall recall.

5.6 Influence of γ on the Defense Effectiveness

Overall, the defense strongly mitigated the efficacy of both attacks, in addition to increasing the recall. For $\gamma = [1.05, 1.2)$ and $\gamma = [1.35, 1.5)$, we saw worse results than $\gamma = [0.65, 0.8)$ and $\gamma = [0.8, 0.95)$. In particular, more bounding boxes were being predicted, and processing/NMS time was slightly higher. We believe processing time was higher due to larger images being more resource-intensive. Moreover, we presumed that the NMS time was lower for $\gamma = [0.65, 0.8)$ and $\gamma = [0.8, 0.95)$ due to possible increased IoUs of bounding boxes, making the NMS algorithm discard these bounding boxes. This assumption was in line with the fact that $\gamma = [0.65, 0.8)$ yielded fewer bounding boxes than $\gamma = [0.8, 0.95)$, suggesting that an even smaller image implies even more bounding box overlap.

6 CONCLUSIONS AND FUTURE WORK

In this work, we presented insights into the vulnerabilities of YOLO OD models to an enhanced PhantomSponges attack. The study not only demonstrates the increased efficacy of these attacks but also proposes a novel defense mechanism. However, limitations exist, including the focus on the YOLOv5 model, which may not be generalized to other architectures. Future work should explore the applicability of our findings across different models and real-world scenarios, potentially leading to more robust OD systems.

REFERENCES

[1] Antonio Emanuele Cinà, Ambra Demontis, Battista Biggio, Fabio Roli, and Marcello Pelillo. 2023. Energy-Latency Attacks via Sponge Poisoning. arXiv:2203.08147 [cs.CR]

[2] Kevin Eykholt, Ivan Evtimov, Earlence Fernandes, Bo Li, Amir Rahmati, Chaowei Xiao, Atul Prakash, Tadayoshi Kohno, and Dawn Song. 2018. Robust Physical-World Attacks on Deep Learning Visual Classification. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 1625–1634. https://doi.org/10.1109/CVPR.2018.00175

[3] T. Gu, K. Liu, B. Dolan-Gavitt, and S. Garg. 2019. BadNets: Evaluating Backdooring Attacks on Deep Neural Networks. *IEEE Access* 7 (2019), 47230–47244. https://doi.org/10.1109/ACCESS.2019.2909068

[4] Jan Hosang, Rodrigo Benenson, and Bernt Schiele. 2017. Learning non-maximum suppression. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 4507–4515.

[5] Lifeng Huang, Chengying Gao, Yuyin Zhou, Cihang Xie, Alan Loddon Yuille, Changqing Zou, and Ning Liu. 2019. Universal Physical Camouflage Attacks on Object Detectors. *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (2019), 717–726. https://api.semanticscholar.org/CorpusID:219099302

[6] Ram Shankar Siva Kumar, David O Brien, Kendra Albert, Salomé Viljöen, and Jeffrey Snover. 2019. Failure modes in machine learning systems. *arXiv preprint arXiv:1911.11034* (2019).

[7] Jona te Lintelo, Stefanos Koffas, and Stjepan Picek. 2024. The SpongeNet Attack: Sponge Weight Poisoning of Deep Neural Networks. *arXiv preprint arXiv:2402.06357* (2024).

[8] Souvik Paul and Nicolas Kourtellis. 2023. Sponge ML Model Attacks of Mobile Apps. In *Proceedings of the 24th International Workshop on Mobile Computing Systems and Applications* (Newport Beach, California) *(HotMobile '23)*. Association for Computing Machinery, New York, NY, USA, 139. https://doi.org/10.1145/3572864.3581586

[9] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi. 2016. You Only Look Once: Unified, Real-Time Object Detection. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 779–788.

[10] Avishag Shapira, Alon Zolfi, Luca Demetrio, Battista Biggio, and Asaf Shabtai. 2022. Phantom Sponges: Exploiting Non-Maximum Suppression to Attack Deep Object Detectors. arXiv:2205.13618 [cs.CV]

[11] Ilia Shumailov, Yiren Zhao, Daniel Bates, Nicolas Papernot, Robert Mullins, and Ross Anderson. 2021. Sponge Examples: Energy-Latency Attacks on Neural Networks. In *2021 IEEE European Symposium on Security and Privacy (EuroS&P)*. 212–231. https://doi.org/10.1109/EuroSP51992.2021.00024

[12] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. 2013. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199* (2013).

[13] Simen Thys, Wiebe Van Ranst, and Toon Goedemé. 2019. Fooling Automated Surveillance Cameras: Adversarial Patches to Attack Person Detection. *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)* (2019), 49–55. https://api.semanticscholar.org/CorpusID:121124946

[14] Ultralytics. 2021. YOLOv5: A state-of-the-art real-time object detection system. https://docs.ultralytics.com. Accessed: April 2024.

[15] Zuxuan Wu, Ser-Nam Lim, Larry S. Davis, and Tom Goldstein. 2020. Making an Invisibility Cloak: Real World Adversarial Attacks on Object Detectors. In *Computer Vision – ECCV 2020*, Andrea Vedaldi, Horst Bischof, Thomas Brox, and Jan-Michael Frahm (Eds.). Springer International Publishing, Cham, 1–17.

[16] Fisher Yu, Haofeng Chen, Xin Wang, Wenqi Xian, Yingying Chen, Fangchen Liu, Vashisht Madhavan, and Trevor Darrell. 2020. Bdd100k: A diverse driving dataset for heterogeneous multitask learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2636–2645.