# Adversarial Attention Perturbations for Large Object Detection Transformers

Supplementary Material

## 1. Reproducibility Statement

We make the following efforts to enhance the reproducibility of our results.

- For AFOG's implementation a link to an anonymous downloadable source repository is included in our abstract. The source includes links for all datasets and models used in our experiments.
- Our experiment details are given in Section 4, containing selected hyperparameters and hardware specifications.
- We also show example images under benign and AFOG attack scenarios throughout the paper and appendix.

## 2. Additional Experimental Setup

### 2.1. Model Details

In our experiments to validate the effectiveness of AFOG attacking vision transformer models for object detection, we select twelve transformers of varying model sizes, ranging from Detection Transformer (DETR) [4], a lightweight 40 million parameter model, to EVA [11] with more than one billion parameters. We use the Detrex framework [30], built on top of Detectron2 [35], and DINO [38] to standardize our model implementations. Both DINO and Detrex adapt numerous general-purpose transformer models to object detection. Further, to demonstrate that AFOG also works beyond Detrex, we choose some transformer models such as DETR and Swin from their original repositories associated with the original papers instead of the versions provided in Detrex. In this section of the Supplementary Material, we first provide a brief overview of each of the twelve transformer models.

**Detection Transformer (DETR)**[4] adapts the transformer architecture for object detection framed as a set prediction task. It implements a transformer encoder-decoder architecture and a global set loss with bipartite matching. We explore DETR with ResNet-50 and ResNet-101 backbones [12]. DETR is the foundation of many other detection transformers [2, 39, 38], making it a compelling target for AFOG. Original code adapted for our implementation is available at https://github.com/facebookresearch/detr

**Deformable-DETR**[39] improves upon DETR's attention mechanism by confining self-attention to a small region around a given point. This matches DETR's performance in significantly reduced training time and improves performance on small object detection. We use the version trained by Detrex with a ResNet-50 backbone. It is pretrained on ImageNet-1k and then trained on COCO 2017. Attacking Deformable-DETR demonstrates that AFOG is effective against its unique self-attention mechanism. Original code adapted for our implementation is available at https://github.com/fundamentalvision/Deformable-DETR

**DINO** [38] advances DETR with improved denoising anchor boxes. DINO supports numerous backbones, making it ideal for attacking many types of models with AFOG. We choose to attack DINO to assess the viability of AFOG against a variety of backbones and to investigate AFOG's performance under robust denoising. Original DINO code adapted for our implementation is available at https://github.com/IDEA-Research/DINO

**AlignDETR** [2] addresses an alignment issue in DETR that incorrectly matches correct predictions with the wrong ground-truths during training. The authors remedy this problem with an IoU-aware binary cross entropy (BCE) loss, mixed-matching, and a sample weighting method that reduces the effects of unimportant samples. We attack AlignDETR to explore AFOG's performance against this alternative loss functon approach. Our implementation uses a ResNet-50 backbone, and it is trained on COCO 2017. Original code adapted for our implementation is available at https://github.com/IDEA-Research/detrex/tree/main/projects/align_detr

**ViTDet** [15] adapts the original vision transformer (ViT) to object detection. We use the version trained by Detrex, which uses a plain ViT as the backbone for ViTDet with masked auto encoder and ImageNet-1k pretraining. ViTDet is further used as a backbone for DINO, and the combination architecture is trained on COCO 2017. Like DINO and DETR, ViT is a widely popular model that has been adapted for numerous uses. We choose ViTDet for our experiments to investigate AFOG's potential applicability to all ViT-based models. Original code adapted for our implementation is available at https://github.com/IDEA-Research/detrex/tree/main/projects/dino

**ConvNeXt-Large-384** [22] ConvNeXt is a modernized pure convolutional neural network that is designed to compete with transformers on object detection and semantic segmentation tasks. We use ConvNeXt-Large-384 pretrained by Detrex on Imagenet-22k. ConvNeXt is used as a backbone with the DINO trans-

former, and both are trained on COCO 2017. Attacking ConvNeXt demonstrates that AFOG is effective against modern CNNs as well as transformers. Original code adapted for our implementation is available at https://github.com/IDEA-Research/detrex/tree/main/projects/dino

**Swin-L** [21] is a hierarchical transformer designed for computer vision. Swin uses shifting windows to boost efficiency and limit the computational cost of its self-attention mechanism. This efficiency also makes it viable at a range of model sizes. We use Swin-Large-384-4-Scale from Detrex, pretrained on ImageNet-22k and trained on COCO 2017. We choose to attack Swin in our experiments to investigate AFOG's efficacy against its shifting window mechanism and because it is a popular backbone for a variety of object detection models. Original code adapted for our implementation is available at https://github.com/IDEA-Research/DINO

**InternImage-Large** [32] is a CNN-based foundation model that leverages deformable convolution instead of sparse kernels. This reduces the inductive bias of traditional CNNs and enables InternImage to learn larger-scale patterns from larger datasets. We use InternImage as a backbone for the DINO transformer. The pairings are pretrained on Imagenet-22k and trained on COCO 2017. We choose InternImage-Large for our experiments to explore AFOG's applicability to deformable convolution. A larger version of InternImage is also currently one of the strongest models on the COCO object detection leaderboard [9]. We use InternImage-Large, a deeper model with more parameters than vanilla InternImage, because of its stronger benign performance. Original code adapted for our implementation is available at https://github.com/IDEA-Research/detrex/tree/main/projects/dino

**FocalNet** [37] replaces self-attention with a focal modulation mechanism that describes interactions of vision tokens. This mechanism is comprised of three parts: hierarchical contextualization, gated aggregation, and element-wise modulation. Using FocalNet in our experiments demonstrates AFOG's viability against this focal modulation mechanism, which may have different weaknesses than the traditional self-attention inside a transformer. We adapt FocalNet-Large-4scale, pretrained on ImageNet-22k for object detection with the DINO transformer. This combination is trained on COCO 2017. Original code adapted for our implementation is available at https://github.com/IDEA-Research/detrex/tree/main/projects/dino

**EVA** [11] is a large vision foundation model. It is based on a classical ViT that is pre-trained on masked vision-text features. It demonstrates strong performance in various transfer learning tasks. We adapt EVA for object detection with DINO and Detrex. The Detrex version of EVA is an EVA-01 model trained on COCO 2017 with additional large-scale jittering (LSJ) augmentation. We select EVA for our experiments to investigate AFOG's performance against a large foundation model with extensive pre-training. Original code adapted for our implementation is available at https://github.com/IDEA-Research/detrex/tree/main/projects/dino_eva

**DETA** [26] reintroduces IoU-based assignment and non-maximum suppression (NMS) to the classic DETR architecture. The authors show that this is superior to the bipartite matching used in the original DETR. Our version of DETA is implemented via Detrex and uses a Swin-Large-384 backbone. It is pretrained on ImageNet-1k and trained on COCO 2017. We include DETA with a Swin backbone in our experiments to investigate AFOG's performance against IoU-based assignment and NMS, and also to explore the Swin backbone with another architecture in addition to DINO. Original code adapted for our implementation is available at https://github.com/IDEA-Research/detrex/tree/main/projects/dino_eva

In addition to detection transformers, we also evaluate AFOG attacking conventional CNN-based object detection models, represented by Faster-R-CNN, SSD and YOLOv3.

**Faster R-CNN** (FRCNN) [29] is a two-phase CNN-based object detector that uses a region proposal network to share convolutional features with the detection network. It is the standard victim object detector in many other attacks [34, 8, 36, 16] making it an ideal standard of comparison. We use a version with a ResNet-50 backbone, trained on Pascal VOC 2007. Original code adapted for our implementation is available at https://github.com/chenyuntc/simple-faster-rcnn-pytorch

**YOLOv3** [28] is a single-phase CNN-based object detector that predicts existence of objects, their bounding boxes and their class labels in a single pass. YOLOv3 is a version of YOLO family of algorithms with improved performance, and we used the pre-trained YOLOv3 on Pascal VOC 2007. YOLOv3 is another popular model among other attacks, making it ideal for comparison. Original code adapted for our implementation is available at https://github.com/qqwweee/keras-yolo3

**SSD-300** [20] is also a single-phase CNN-based object detector, which divides an image into default regions at varying scales and aspect ratios and predicts object existence, bounding box and class label with confidence scores for each bounding box. High-scoring boxes are also adjusted to better fit predicted objects. We use a pretrained SSD-300 on Pascal VOC 2007, which takes $300\times300$ inputs. Original code adapted for our im-

plementation is available at https://github.com/pierluigiferrari/ssd_keras

### 2.2. Hyperparameter Tuning

We introduce an attention learning rate hyperparameter (see Equation 6) that controls the sensitivity of AFOG's adversarial attention maps to gradient updates. Figure 8 shows the performance of AFOG with varying attention learning rates compared to AFOG without attention (attention learning rate zero) for three detection transformers. We observe that an attention learning rate of 0.1 shows the strongest performance across all three models, and so we use this value for all models and all experiments.

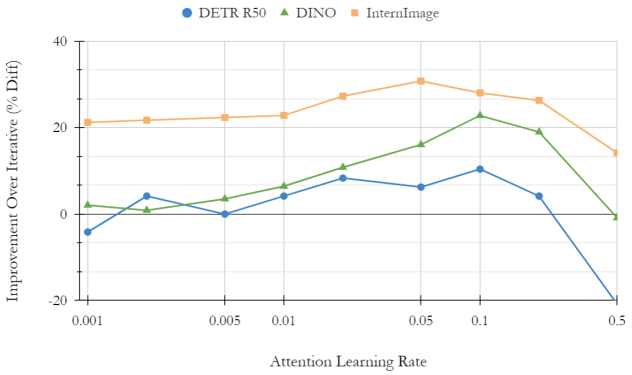


Figure 8. AFOG improvement over iterative for varying values of attention learning rate across three detection transformers.

## 3. AFOG Variants and Pseudocode

In addition to the pseudocode for AFOG provided in our main paper, in this section, we also provide the pseudocode for AFOG-V in Algorithm 2 and AFOG-F in Algorithm 3.

The key difference between AFOG and AFOG-V is the replacement of $O_x$ with a set of zero predictions $\varnothing$ instead of forward propagating image $\mathcal{O}_x \leftarrow f_D(x;\vartheta)$. AFOG-V also aims to minimize $\mathcal{L}_{AFOG}$ according to its slightly different formulation. Algorithm 2 provides a sketch of the pseudo code.

AFOG-F changes AFOG by modifying benign predictions $\mathcal{O}_f$ to change all confidence scores to 1.0. This means every benign prediction by the model, regardless of quality, is used as a ground truth during attack iteration. AFOG-F aims to minimize $\mathcal{L}_{AFOG}$, learning perturbations that increase the likelihood of inducing these spurious low-quality predictions. This is reflected in Algorithm 3.

## 4. Adversarial Attacks: Visualization

In this section, we provide additional visualization of AFOG, AFOG-V, and AFOG-F attacks.

**Algorithm 2** AFOG-V attack

**Require:** Victim image $x \in \mathcal{D}$, test-set $\mathcal{D}$, Victim pre-trained model $f_D(\vartheta)$, Perturbation step size $\alpha_P$, Attention step size $\alpha_A$, Num. iters $T$, Max pert. $\epsilon$.

Initialize $\mathcal{O}_x \leftarrow \varnothing$
Initialize attention map $A_0 \leftarrow 1$;
Initialize perturbation $P_0 \leftarrow Random(-\epsilon, \epsilon)$;
Initialize step variable $k \leftarrow 1$;
**while** $k \leq T$ **do**
  Attack image $x_{adv\,k} \leftarrow \prod_{x,\epsilon}(x + A_k \odot P_k)$;
  Forward propagate $x_{adv}$ through $f_D(\vartheta, x_{adv})$;
  Compute bbox-loss $\mathcal{L}_{bbox}(x_{adv}, \mathcal{O}_x; \vartheta)$;
  Compute cls-loss $\mathcal{L}_{cls}(x_{adv}, \mathcal{O}_x; \vartheta)$;
  $\mathcal{L}_{AFOG}(x_{adv}, \mathcal{O}_x; \vartheta) = -$bbox-loss $-$ cls-loss;
  Calculate losses with respect to $A_k$ and $P_k$: $\mathcal{L}_A(x_{adv}, \mathcal{O}_x; \vartheta), \mathcal{L}_P(x_{adv}, \mathcal{O}_x; \vartheta)$;
  Normalize attention loss $\mathcal{L}_A \leftarrow$ Norm$(\mathcal{L}_A)$;
  Take sign of perturbation loss $\mathcal{L}_P \leftarrow$ Sign$(\mathcal{L}_P)$;
  $A_{k+1} \leftarrow A_k - \alpha_A \mathcal{L}_A$;
  $P_{k+1} \leftarrow P_k - \alpha_P \mathcal{L}_P$;
  $k \leftarrow k + 1$;
**end while**
$x_{adv\,k+1} \leftarrow \prod_{x,\epsilon}(x + A_{k+1} \odot P_{k+1})$;
**return** $x_{adv}$

**Algorithm 3** AFOG-F attack

**Require:** Victim image $x \in \mathcal{D}$, test-set $\mathcal{D}$, Victim pre-trained model $f_D(\vartheta)$, Perturbation step size $\alpha_P$, Attention step size $\alpha_A$, Num iters $T$, Max pert $\epsilon$.

Initialize $\mathcal{O}_f \leftarrow f_D(x;\vartheta)$
Initialize attention map $A_0 \leftarrow 1$;
Initialize perturbation $P_0 \leftarrow Random(-\epsilon, \epsilon)$;
Initialize step variable $k \leftarrow 1$;
**while** $k \leq T$ **do**
  Attack image $x_{adv\,k} \leftarrow \prod_{x,\epsilon}(x + A_k \odot P_k)$;
  Forward propagate $x_{adv}$ through $f_D(\vartheta, x_{adv})$;
  Compute bbox-loss $\mathcal{L}_{bbox}(x_{adv}, \mathcal{O}_x; \vartheta)$;
  Compute cls-loss $\mathcal{L}_{cls}(x_{adv}, \mathcal{O}_x; \vartheta)$;
  $\mathcal{L}_{AFOG}(x_{adv}, \mathcal{O}_x; \vartheta) = -$bbox-loss $-$ cls-loss;
  Calculate losses with respect to $A_k$ and $P_k$: $\mathcal{L}_A(x_{adv}, \mathcal{O}_x; \vartheta), \mathcal{L}_P(x_{adv}, \mathcal{O}_x; \vartheta)$;
  Normalize attention loss $\mathcal{L}_A \leftarrow$ Norm$(\mathcal{L}_A)$;
  Take sign of perturbation loss $\mathcal{L}_P \leftarrow$ Sign$(\mathcal{L}_P)$;
  $A_{k+1} \leftarrow A_k - \alpha_A \mathcal{L}_A$; $P_{k+1} \leftarrow P_k - \alpha_P \mathcal{L}_P$;
  $k \leftarrow k + 1$;
**end while**
$x_{adv\,k+1} \leftarrow \prod_{x,\epsilon}(x + A_{k+1} \odot P_{k+1})$;
**return** $x_{adv}$

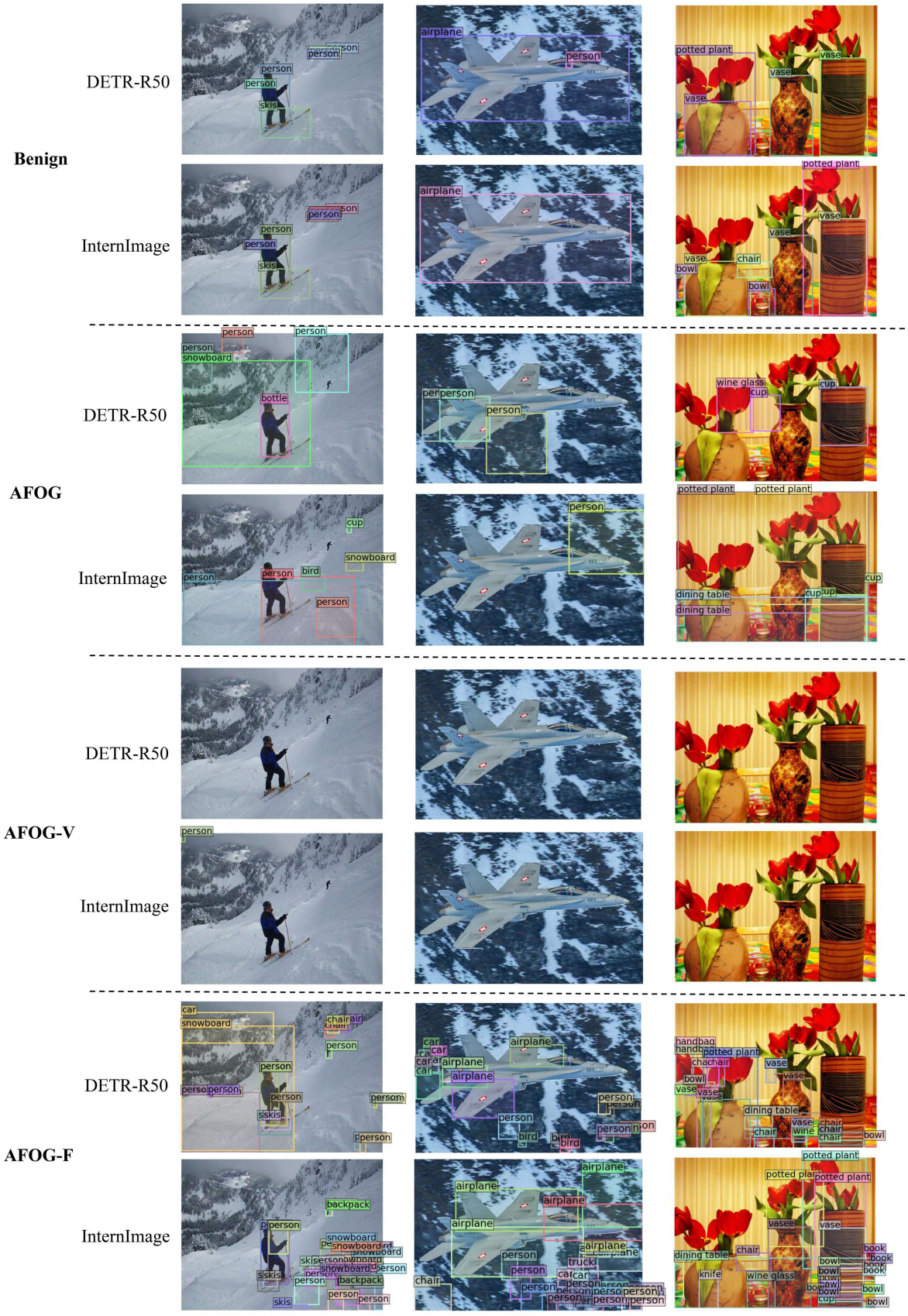


Figure 9. Example images for Benign, AFOG, AFOG-V, and AFOG-F scenarios with victims R50 (DINO) and InternImage (DINO).

### 4.1. AFOG versus AFOG-V and AFOG-F

Figure 9 shows the detection results of the victim transformer detector DETR-R50 under the benign scenario for three examples from COCO testdev in Row 1. In Rows 2-4, we show the AFOG attack, AFOG-V attack and AFOG-F attack on these three example images. In Row 5 we show the detection results of another detection transformer model InternImage (DINO) under the benign scenario for the three example images. Rows 6-8 show the adverse effects of the AFOG attack, AFOG-V attack, and AFOG-F attack on these three example images respectively. We observe that baseline AFOG demonstrates a mixture of generating false positives, obfuscating true positives, and disrupting bounding boxes in benign cases for all three images across both models (Rows 1 and 5). AFOG-V disrupts all detections across both models, with the exception of a small "person" detection in the top left corner of Row 7, Column 1. AFOG-F induces a large quantity of spurious detections, which show different behaviors between the two models. For DETR (Row 4), the false positive detections tend to spread out across each image. For ViTDet (Row 8) the false positive detections tend to cluster in areas without foreground objects. We observe AFOG-F also fails to disrupt some true positive detections, such as the central person in Row 4, Column 1. We conclude that generic AFOG displays the positive aspects of both variants: sufficient vanishing to disrupt true positive detections paired with sufficient fabrication to induce a small number of false positives.

### 4.2. Visualization of Adverse Effects of AFOG Attack on Different Detection Transformers

In Figures 10-13 we show visualizations of example object detection results under the benign scenario and AFOG attack for five COCO testdev images across all 12 transformers in our experiments. We partition these results into four groups to simplify comparison and highlight trends. Each group shows benign predictions for a set of detectors in the first rows, followed by AFOG-disrupted predictions in the next rows.

Figure 10 compares detection results for FocalNet [37], InternImage [32], DINO-ResNet50 [12], and AlignDETR [2]. We observe that, although these models each have unique architectures, they have nearly identical benign prediction behavior across five example images (Rows 1-4). We further observe that AFOG's learnable attention mechanism probes and exploits the individual weaknesses of each model, inducing unique disrupted behaviors in each (Rows 5-8). Although AFOG's induced malicious behavior can have similar themes, such as vanishing most detections in the first image (Col. 1) or causing a large "Person" class detection in the third image (Col. 3), the details of this behavior change between models. We speculate that this is a result of AFOG's architecture-agnostic adaptability.

Figure 11 compares detection results for DETR [4] with a ResNet-50 backbone and DETR with a ResNet-101 backbone. Rows 1-2 show the benign behavior of the two models. We observe that nearly identical architectures produce similar detections. However, AFOG causes spurious detections that differ between the two models (Rows 3-4). In image two (Col. 2) AFOG induces a large "Couch" prediction in both models by disrupting both class and bounding box losses. Similarly, AFOG induces several small "Cat" detections in image three (Col. 3) for both models. In these cases, AFOG has learned similar perturbations for similar architectures. Images 4 and 5 (Cols. 4-5) exhibit very different behavior under AFOG attack, demonstrating how AFOG's learnable attention is also able to exploit fine-grained differences in victim models as necessary.

Figure 12 compares detection results for VitDet [15], EVA [11], and DETA [26]. We compare AFOG failure modes between these three models, and make two observations. (i) AFOG shows similar weaknesses against some objects across the three models, such as the "Person" in image five (Col. 5) and the "Baseball Bat" in image two (Col. 2). (ii) AFOG exhibits unique failure behavior for other objects, such as the "Couch" object in image four (Col. 4). Against ViTDet (Row 4, Col. 4), AFOG fails to disrupt the "Couch" class label, attacking only the bounding box. Against EVA (Row 5, Col. 4), AFOG fails to disrupt the bounding box and changes "Couch" to a similar "Bed" label. We speculate that AFOG's ability to probe for weak points and adapt to different models may also lead to this kind of unpredictable failure behavior.

Figure 13 compares detection results for the remaining detectors from our experiments: Deformable-DETR [39], ConvNeXt [22], and Swin-L [21]. As with other models, we observe very different detection behavior under AFOG attacks between models. For example, we contrast Deformable-DETR's detections for image three (Row 4, Col. 3) with ConvNeXt's detections for the same image (Row 5, Col. 3), noting that the two have little in common. We also observe that AFOG largely fails to disrupt images two and three against Swin (Row 6, Cols. 2-3), leaving multiple class and bounding box detections intact. We highlight this image in red for emphasis.

### 4.3. Comparison to Other Attacks

We next compare our AFOG attack with four existing representative victim-based detection attacks against FRCNN, a two-stage CNN-based object detector.

Figure 14 provides three examples from the Pascal VOC test set under the benign scenario (Column 1) and compares our AFOG attack (Column 6) with four well-known attacks: TOG [8] in Column 2, UEA [34] in Column 3, RAP [16] in Column 4, and DAG [36] in Column 5. We make two observations. (i) All five attacks are successful in

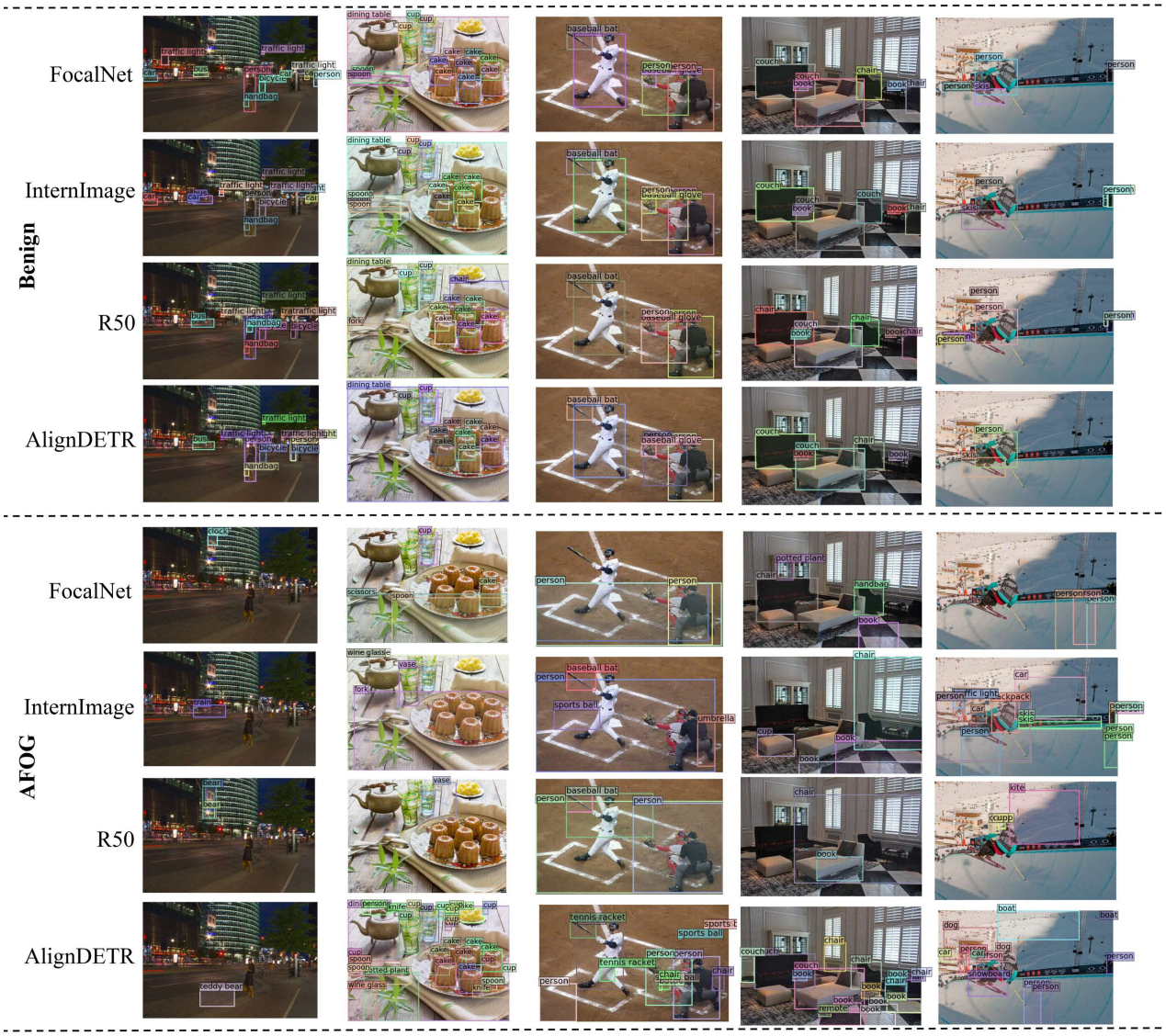


Figure 10. Comparison of Benign and AFOG-disrupted predictions on five example images with ResNet-50, InternImage, FocalNet, and Align-DETR. Despite nearly identical benign behaviors of all four models across five images, AFOG induces different malicious behavior by effectively learning unique vulnerabilities, indicating strong AFOG adaptability.

attacking the example images for which FRCNN can return correct detections under the benign scenario (Column 1). (ii) AFOG displays superior multi-box disruption, whereas other attacks produce misclassification errors with correct bounding boxes.

Figure 15 further compares AFOG, AFOG-V and AFOG-F with TOG on three example images in the Pascal VOC test set. For all three examples, TOG (Row 2) fails to attack the FRCNN detector on the foreground objects (dog, bus, cat) and in contrast, AFOG (Row 3) is successful. AFOG-V (Row 4) shows the adverse effect of AFOG vanishing, and AFOG-F (Row 5) shows the adverse effect of AFOG fabrication.


Figure 11. Comparison of Benign and AFOG-disrupted predictions on five images with two similar architectures, DETR-R50 and DETR-R101. Despite similar architectures, AFOG produces different malicious behavior by probing for unique vulnerabilities in each model.


Figure 12. Comparison of Benign and AFOG-disrupted predictions on five images with ViTDet, EVA, and DETA. Although AFOG shows strong attack performance against various other classes across the five images, it also struggles to disrupt the ”Person” class in the third and the fifth examples.


Figure 13. Comparison of Benign and AFOG-disrupted predictions on five example images with Deformable-DETR, ConvNeXt, and Swin-L. AFOG exhibits widely varying behavior for different models on the same image, such as against image three (Rows 4-6, Col. 3). We also indicate two instances where AFOG failed to disrupt Swin-L's performance, highlighted in red.

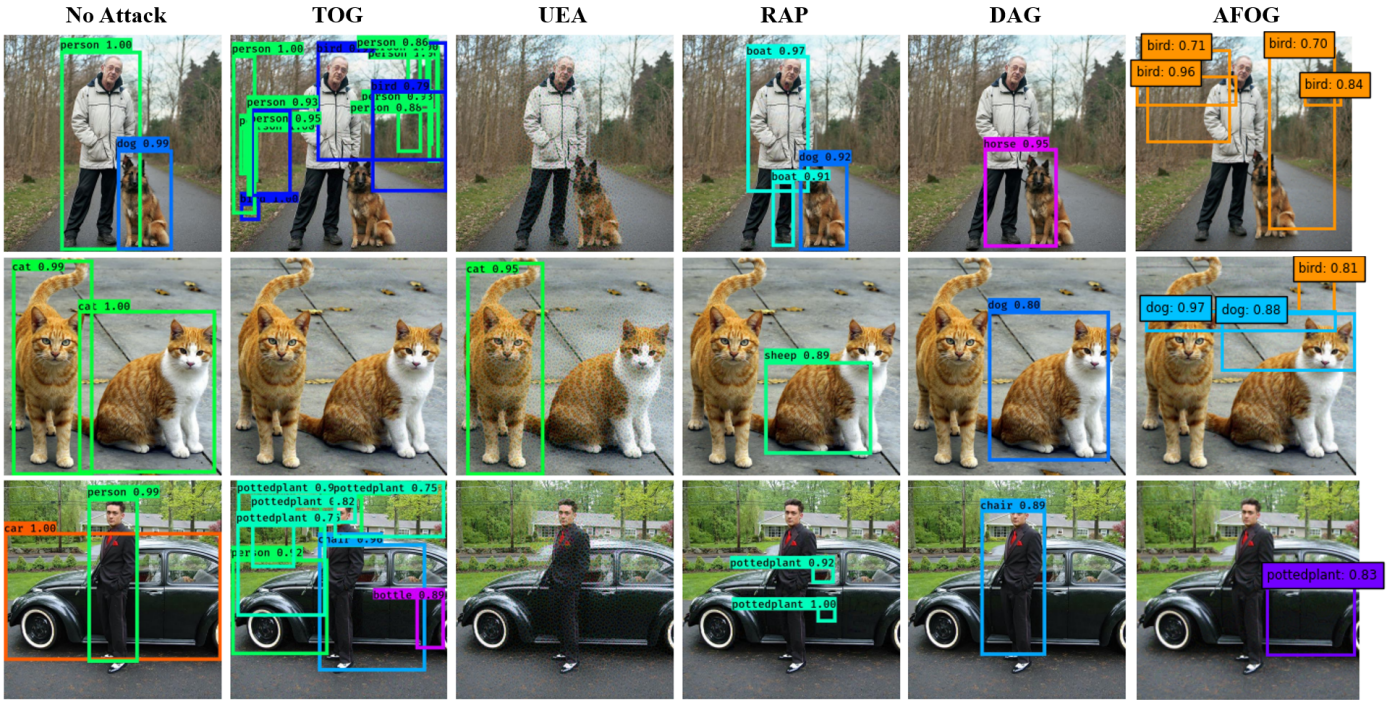


Figure 14. Comparison of three images across Benign, AFOG, TOG, UEA, RAP, and DAG cases. AFOG displays superior class and bounding-box disruption. Figure adapted from [7]


Figure 15. Three example images from the VOC dataset where AFOG outperforms TOG.