ADVERSARIAL DEFENSE STRATEGIES FOR OBJECT DETECTION IN AUTONOMOUS VEHICLE PERCEPTION SYSTEMS

Manas Dixit, Nitish Poojari & Sharva Khandagale
Miinnesota Robotics Institute
University of Minnesota, Twin Cities
Minnneapolis, MN 55455, USA
{dixit084,pooja011,khand137}@umn.edu

ABSTRACT

Convolutional Neural Networks (CNNs) have achieved state-of-the-art performance in a wide range of computer vision tasks, including image classification, object detection, and segmentation, owing to advances in training techniques and architectural innovations. However, their susceptibility to adversarial attacks raises serious concerns about their reliability, particularly in safety-critical applications such as autonomous driving. While adversarial training has emerged as a promising defense, its effectiveness is typically limited to the types of attacks seen during training. Moreover, conventional training pipelines terminate learning after convergence, lacking the adaptability to counter evolving adversarial threats encountered during real-world deployment. This project addresses these limitations in the context of object detection for autonomous vehicle perception systems, focusing on CNN-based detectors such as YOLOv4. We propose a self-supervised continual learning pipeline that enables the detector to adapt and improve during deployment by leveraging adversarial inputs in real time. Our input dataset consists of both clean and adversarial examples generated using Fast Gradient Sign Method (FGSM) and Projected Gradient Descent (PGD). We incorporate the MagNet detector to identify adversarial samples during inference, which are then used for training the target YOLOv4 via self-supervised learning mechanism. This mechanism allows the YOLOv4 model to incrementally enhance its robustness through continuous self-supervised updates. Through this adaptive training pipeline, we demonstrate improved resilience of YOLOv4 to adversarial perturbations, achieving a measurable increase in mean Average Precision (mAP) under FGSM and PGD attacks—from x% to y%.

1 SIGNIFICANCE AND MOTIVATION

As the world makes significant strides towards autonomy, autonomous vehicles have become a household name and hence the importance of ensuring robustness in their operation. This project is particularly relevant in the context of the ongoing evolution of mobility, offering an opportunity to contribute to the safety and reliability of autonomous systems. Furthermore, by addressing vulnerabilities in visual systems and their defenses, this project extends beyond the realm of autonomous vehicles, with potential applications in fields such as medical imaging and other AI-driven technologies. Since the focus of this project is to explore more complex defenses for object detection networks, it provided us with the opportunity to work with advanced approaches to improve the object detector network's robustness to adversarial attacks and allowed us to come up with a unique adversarial robustness pipeline that extends traditional adversarial training into a dynamic, real-time learning setting.

2 Previous Work

Previous research on adversarial attacks in deep learning has significantly advanced our understanding of neural networks' vulnerabilities and provided potential avenues for optimizing their robustness. Goodfellow et al. Goodfellow et al. (2015) argued that the primary cause of adversarial vulnerability lies in the linearity of neural networks in high-dimensional spaces, introducing the Fast Gradient Sign Method (FGSM) for crafting adversarial examples and laying the groundwork for adversarial training as a form of regularization. Expanding on this, Xie et al. Xie et al. (2017) proposed the Dense Adversary Generation (DAG) algorithm to craft perturbations for more complex tasks like object detection and semantic segmentation, demonstrating that adversarial examples can transfer across models and tasks. Lu et al. Lu et al. (2017) and Choi & Tian Choi & Tian (2022) extended these findings to object detectors such as YOLO and Faster R-CNN, showing that physical adversarial examples can fool real-world detectors and emphasizing the need to incorporate objectness into attack strategies.

Defense mechanisms such as Defense-GAN Samangouei et al. (2018) and auxiliary detection subnetworks Metzen et al. (2017) use generative modeling or secondary classifiers to mitigate adversarial effects, while Carlini & Wagner Carlini & Wagner (2017) evaluated and defeated several defenses via stronger optimization-based attacks. More recently, Stutz et al. Stutz et al. (2020) demonstrated that conventional adversarial training "does not generalize to unseen threat models," motivating approaches that continually expose networks to new perturbations. Complementing this, Chen et al. Tsai et al. (2023) argue that training-time defenses incur high computational cost and still fail to generalize, advocating *test-time detection and repair* pipelines. These insights have motivated us to come up with a run-time training pipeline ensuring adaptability to evolving attacks by continuously training itself, thereby closing the generalization gap identified in prior work. Collectively, these studies show that adversarial examples not only expose fundamental blind spots in deep networks but also serve as a tool for optimizing robustness through adversarial training, detection-and-repair frameworks, gradient smoothing, and generative data projection, ultimately guiding the design of more resilient neural architectures.

3 Goals

This project aims to deepen our understanding of how adversarial vulnerabilities can impact the accuracy and reliability of object detection models like YOLO, especially in high-stakes environments such as autonomous driving. Initially, our objective was to evaluate YOLO's robustness against popular adversarial attacks such as FGSM and PGD, and implement defense mechanisms for enabling an object detection model immune to a wide range of adversarial attacks. Building on that foundation, and informed by recent literature—such as Carlini & Wagner (2017) and Xie et al. (2017)—our goals evolved to explore more dynamic and intelligent forms of defense.

Specifically, we now aim to enhance the benchmark YOLOv4 model by incorporating a self-supervised learning-based defense framework. This defense strategy focuses on enabling the model to autonomously detect when it is being exposed to adversarial inputs in run-time. Upon detection, the model will flag and store these samples for further adversarial training, incorporating an element of experiential learning. Through this adaptive mechanism, we intend to bolster the security and reliability of object detection systems, evolving them into a more robust, trustworthy and up-to date object detector over the time.

4 Approach

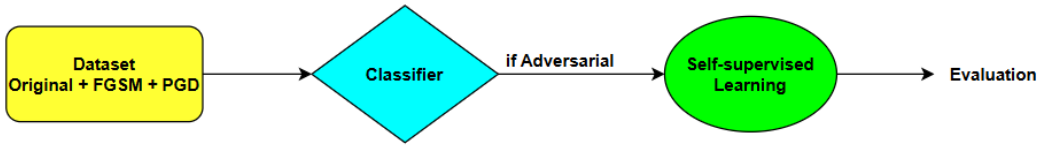


Figure 1: Idea

4.1 Dataset Creation: Adversarial Sample Generation

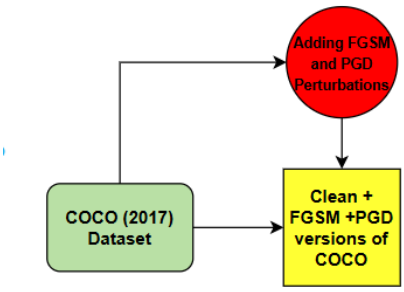


Figure 2: Generation of FGSM + PGD

Figure 3 demonstrates the steps followed to create the input dataset that contains clean samples imported from COCO 2017 dataset and the adversarial samples generated using gradient-based adversarial attacks such as, FGSM and PGD implemented on the clean samples. Following are details of the implemented attacks:

- **Fast Gradient Sign Method (FGSM)**
  - A single-step perturbation of magnitude $\varepsilon = \frac{8}{255}$
  - Perturbations were crafted by maximizing the objectness confidence loss with respect to the input pixels.
- **Projected Gradient Descent (PGD)**
  - An iterative variant of FGSM with step size $\alpha = \frac{8}{255}$ over 10 steps.
  - After each update, the perturbed image was projected back onto the ℓ_∞-ball of radius $\varepsilon = \frac{8}{255}$ to ensure the total perturbation remained bounded.

In both cases, we preserved and duplicated the original annotation files alongside the perturbed images, ensuring dataset consistency and enabling a fair, direct comparison of detector performance on clean versus adversarial inputs.

4.2 Design and Deployment of an Adversarial Detector / Classifier

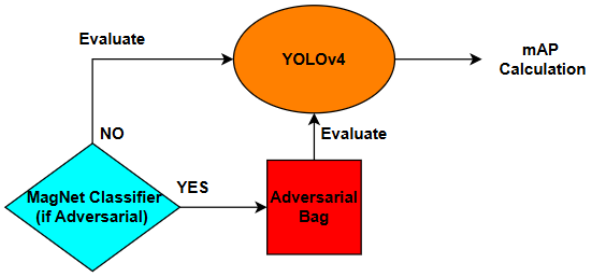


Figure 3: Classification of Adversarial Samples

The MagNetDetector is a convolutional neural network architecture working to function as a detector for adversarial inputs, concentrating on identifying disturbances caused by attacks like the Fast Gradient Sign Method (FGSM). The MagNetDetector identifies an adversarial sample by learning the statistical and structural distinctions between clean and adversarial samples. The architecture comprises a chain of convolutional layers—beginning with 3 input channels (RGB) and advancing through 16, 32, and 64 filters—each succeeded by batch normalization and ReLU activations to

enhance and stabilize feature representations. Max pooling layers are utilized following each convolutional block to reduce the spatial dimensions, enabling the network to concentrate on abstract features. An adaptive average pooling layer compresses each feature map into a single value, guaranteeing a consistent output size irrespective of the original image dimensions. The 64-dimensional feature vector goes through a dropout layer to avoid overfitting in training, and into a fully connected linear layer that produces a single logit. A sigmoid activation is utilized during the forward pass, generating a probability score ranging from 0 to 1 that indicates the model's confidence about the input being adversarial.

The MagNetDetector model was developed using a supervised learning approach aimed at differentiating between clean and adversarial images. The training procedure was conducted on a system equipped with a GPU (A100). The model was fine-tuned using the AdamW optimizer, which merges the adaptive learning rate features of Adam with weight decay regularization to reduce overfitting. A binary cross-entropy loss function (BCELoss) was employed to evaluate the difference between the predicted probabilities and the actual binary labels, where a label of 0 represents a clean image and a label of 1 signifies an adversarial sample. In each epoch, the model received training on groups of input images along with their respective labels. The forecasts were generated via a forward pass, the loss was determined, and gradients were backpropagated to adjust the model parameters as needed.

After each training epoch, the model's performance was evaluated using AUROC on a validation set to track its ability to distinguish clean from adversarial inputs. The best-performing model (highest AUROC) was saved. A conservative detection threshold was set at the 99th percentile of clean validation scores to cap the false positive rate at 1

In a production pipeline, the trained `MagNetDetector` is inserted as a preprocessing filter ahead of YOLOv4. Incoming image batches are first passed through MagNet, which flags any inputs with high "adversarialness" scores. All detected adversarial samples are automatically saved to a designated "adversarial bag" for further retraining. Once the first batch has been processed and adversarial examples have been quarantined, the remaining clean images are forwarded to YOLOv4 for standard object detection. This approach ensures that YOLOv4 operates only on verified clean data during the run-time, improving overall system reliability and enabling the continuous collection of adversarial instances for further training.

4.3 Implementation of Self-Supervised Learning

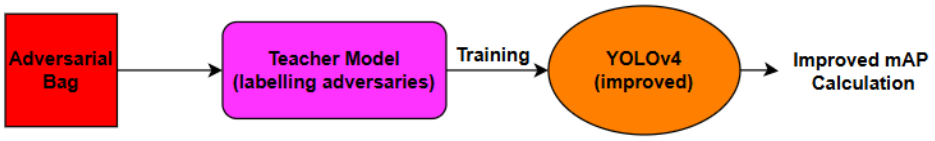


Figure 4: Self-Supervised Learning

Adversarial samples flagged by MagNet detector, stored in an "adversarial bag," lack ground truth labels. To label them without human intervention, we use a high-performing teacher model-YOLOv8-which performs inference on these inputs to generate pseudo-labels. The resulting labeled dataset is then used to retrain the YOLOv4 model, allowing it to incrementally improve its performance against evolving adversarial threats.

The training strategy for YOLOv8 in our pipeline is centered on its role as a teacher model for generating high-quality pseudo-labels on unlabeled adversarial inputs. YOLOv8-large, a pretrained object detector, was fine-tuned on a curated subset of the COCO 2017 dataset containing clean, FGSM-, and PGD-perturbed versions of images from 12 autonomous-driving-relevant categories. To align with YOLO's format, the COCO annotations were converted and remapped to sequential class indices. The dataset was split into training, validation, and test sets in a 60-20-20 ratio and organized under a YOLO-compatible folder structure with a corresponding dataset.yaml file. Data augmentations—including random horizontal flips, contrast shifts, and geometric transformations—were applied to improve generalization. The model was trained for 50 epochs with a batch size of 32 and

image size of 640×640 using the Ultralytics training framework. Care was taken to discard samples with missing bounding boxes post-augmentation. This robust training strategy enables YOLOv8 to accurately infer labels for perturbed images, making it a reliable pseudo-labeling component within our self-supervised learning pipeline.

4.4 System Integration for Experiential Learning

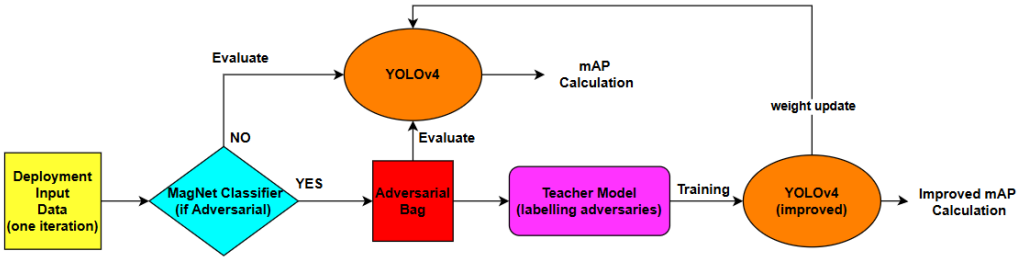


Figure 5: Training of YOLOv4 and Evaluation

The self-supervised learning pipeline connects three core components—classifier, teacher model, and YOLOv4—to enhance robustness against adversarial attacks. Each cycle begins with a batch of images processed by the classifier node, which uses a pretrained MagNet detector to separate clean and adversarial inputs. The adversarial samples from this batch are collected into an adversarial bag, which is then fed in its entirety to the teacher model (YOLOv8-large). The teacher model performs inference on this batch, generating pseudo-labels in the form of bounding boxes and class labels. These labeled adversarial images are subsequently used to train the YOLOv4 target model, improving its ability to handle similar perturbations in future batches. After training, mAP scores are evaluated to monitor progress, and the pipeline repeats with the next incoming batch, enabling ongoing, adaptive learning.

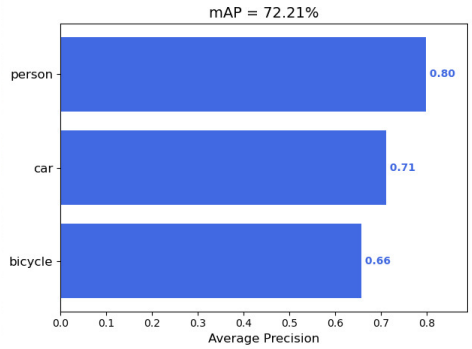
The entire pipeline is implemented in ROS 2 using the `rclpy` library, with each component running as an independent node. The classifier node receives an input batch of images containing both clean and adversarial samples. After processing, it publishes a completion flag (`/classification_completion_flag`) and stores the identified adversarial samples in a designated file, `adv_samples.txt`. The teacher node listens for this flag, loads the adversarial samples from `adv_samples.txt`, and performs inference using the YOLOv8-large model to generate bounding boxes and class labels. Upon completing this annotation step, it publishes a second completion flag (`/SAM_completion_flag`) and saves the pseudo-labeled data in YOLO format into `adv_train.txt` and `adv_val.txt`. The training node monitors this flag, then initiates training of the YOLOv4 model using the newly generated labeled data. Once training is complete, it broadcasts its own completion flag and resets the loop, ensuring the system is prepared for the next incoming batch. This modular ROS 2 setup facilitates seamless inter-node communication and enables a fully automated, batch-wise adversarial learning and model enhancement process. Please find the following link to our Github Repository containing code for the above mentioned implementation.

GitHub Repository

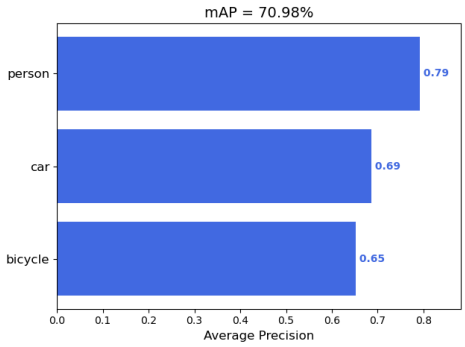
5 Experiments & Results

Two sequential batches of data—each containing a mix of clean and adversarial samples—were fed into the pipeline to evaluate the progressive improvement in the target model's robustness. The first batch included clean images and FGSM-based adversarial samples, while the second batch consisted of clean images combined with PGD-based adversarial samples. The YOLOv4 target model was initialized with `yolo.pth` weights pretrained on the COCO 2017 dataset using only clean samples. This setup allowed us to observe how the model, starting from a clean-data baseline, incrementally adapts and improves its performance across varying adversarial attack types through successive training iterations. Following are the results obtained:

a) Drop in the target model's mAP upon FGSM adversarial attack: 1.23%



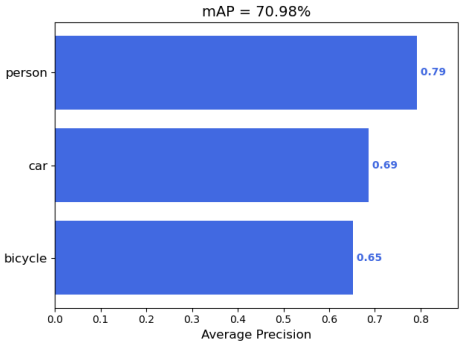
(a) mAP of base YOLOv4 on clean samples



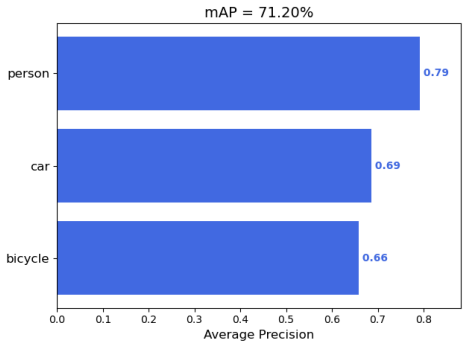
(b) mAP of base YOLOv4 on FGSM samples

Figure 6: mAP drop due to adversarial attack .

b) Model performance improvement after exposure to the first batch comprising clean and FGSM-perturbed samples: 0.22%



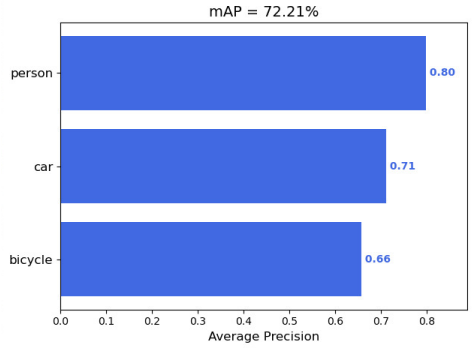
(a) mAP of base YOLOv4 on FGSM samples



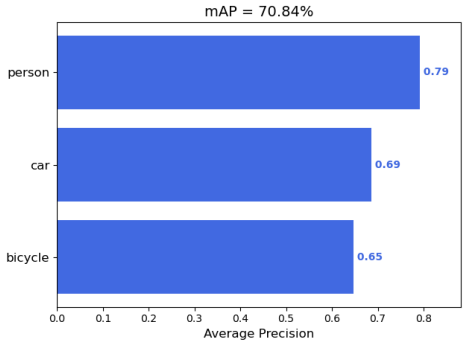
(b) mAP of YOLOv4 after 1st training run on FGSM

Figure 7: Model improvement after 1st run .

c) Drop in the target model's mAP upon PGD adversarial attack: 1.37%



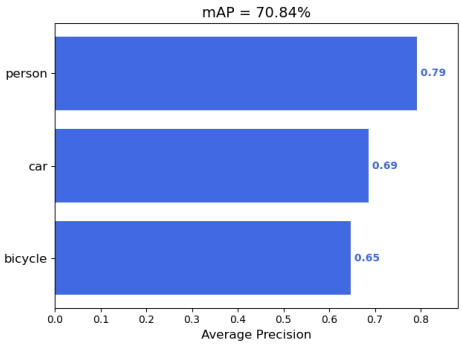
(a) mAP of base YOLOv4 on clean samples



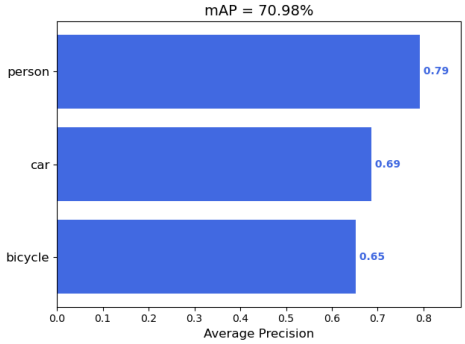
(b) mAP of base YOLOv4 on PGD samples

Figure 8: mAP drop due to adversarial attack .

d) Model performance improvement after exposure to the second batch comprising clean and PGD-perturbed samples: 0.14%



(a) mAP of base YOLOv4 on PGD samples



(b) mAP of YOLOv4 after 2nd training run on PGD

Figure 9: Model improvement after 2nd run.

Through these results, we are able to conclude that the experiential learning pipeline is successfully enabling base YOLOv4 model to evolve and improve upon every exposure to adversarial attacks. However, the improvement is limited due to inherent inaccuracies in MagNet classifier and YOLOv8 Teacher model.

6 Future Scope of Work

A key insight from our current pipeline is that its overall performance is tightly coupled with the quality of pseudo-labels generated by the teacher model. This creates a strong dependency on the teacher's robustness to adversarial perturbations. As such, a primary direction for future work is to explore or develop robust teacher models, either generalized across attack types or overfitted to specific adversarial attack families such as FGSM or PGD. By training these models to perform reliably under specific perturbation regimes, we aim to enhance the accuracy of pseudo-labels, leading to more effective training of the target YOLOv4 model.

To support this, another critical enhancement lies in upgrading the adversarial classifier. Rather than simply detecting whether an input is adversarial, we propose designing a multi-class adversarial detector capable of identifying the specific type of attack. This enables dynamic routing of samples to the most appropriate teacher model, effectively transforming the pipeline into a modular, attack-aware training system. Such a framework could emulate a mixture-of-experts paradigm, where each teacher specializes in handling a particular adversarial distribution. Together, these improvements would strengthen the pipeline's adaptability, reduce error propagation, and further automate the process of building robust object detectors for real-world adversarial settings.

7 Point of Interest: Acknowledging a Vital Question

While it's true that powerful teacher models can run inference on adversarial samples—especially when specialized or overfitted for specific attack types—this raises a valid question: why not deploy those teachers directly instead of training a student model at all? This challenge strikes at the core of our pipeline's motivation, but also provides an opportunity to clarify its necessity and long-term value.

Even though these teacher models are effective within their niche, they are not generalists; they perform well only against the specific attack types they were trained or tuned for. In contrast, our pipeline leverages these specialized models as subject matter experts to pseudo-label adversarial inputs, allowing us to distill their robustness into a single unified student model. Over multiple training iterations and exposure to diverse adversarial samples, the student model (YOLOv4) learns

to generalize across attack types. This not only reduces the need for multiple deployed models and complex routing logic, but also opens the possibility for the student to match or even surpass the robustness of its teachers. Thus, the pipeline enables a scalable, efficient, and attack-agnostic solution for real-world deployment.

REFERENCES

Nicholas Carlini and David Wagner. Towards evaluating the robustness of neural networks. In *IEEE Symposium on Security and Privacy (S&P)*, 2017.

Jung Im Choi and Qing Tian. Adversarial attack and defense of yolo detectors in autonomous driving scenarios. In *IEEE Intelligent Vehicles Symposium (IV)*, 2022.

Ian Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. In *International Conference on Learning Representations (ICLR)*, 2015.

Jiajun Lu, Hussein Sibai, and Evan Fabry. Adversarial examples that fool detectors. *arXiv preprint arXiv:1712.02494*, 2017.

Jan Hendrik Metzen, Tim Genewein, Volker Fischer, and Bastian Bischoff. On detecting adversarial perturbations. In *International Conference on Learning Representations (ICLR)*, 2017.

Pouya Samangouei, Maya Kabkab, and Rama Chellappa. Defense-gan: Protecting classifiers against adversarial attacks using generative models. In *International Conference on Learning Representations (ICLR)*, 2018.

David Stutz, Matthias Hein, and Bernt Schiele. Confidence-calibrated adversarial training: Generalizing to unseen attacks. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pp. 9155–9166. PMLR, 2020. URL https://proceedings.mlr.press/v119/stutz20a.html.

Yun-Yun Tsai, Ju-Chin Chao, Albert Wen, Zhaoyuan Yang, Chengzhi Mao, Tapan Shah, and Junfeng Yang. Test-time detection and repair of adversarial samples via masked autoencoder. *arXiv preprint arXiv:2303.12848*, 2023. URL https://arxiv.org/abs/2303.12848.

Cihang Xie, Jianyu Wang, Zhishuai Zhang, Yuyin Zhou, Lingxi Xie, and Alan Yuille. Adversarial examples for semantic segmentation and object detection. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2017.