

# Article Review: Explaining and Harnessing Adversarial Examples From Classification to Object Detection

Adonis Jamal  
CentraleSupélec, ENS Paris-Saclay  
adonis.jamal@student-cs.fr

Jean-Vincent Martini  
CentraleSupélec, ENS Paris-Saclay  
jean-vincent.martini@student-cs.fr

## Abstract

*This paper presents a detailed study of the work “Explaining and Harnessing Adversarial Examples” by Goodfellow et al. [? ], analyzing the origin of adversarial vulnerability in neural networks and the Fast Gradient Sign Method (FGSM). We further extend this analysis to object detection, demonstrating the transferability of adversarial perturbations to modern lightweight detectors and evaluating adversarial training strategies.*

## 1. Introduction

This report presents a study of the article “Explaining and Harnessing Adversarial Examples” written by Ian J. Goodfellow, Jonathon Shlens and Christian Szegedy [? ]. It serves as the final assessment for the Deep Learning course within the MVA Master’s program and CentraleSupélec.

In the examined article, the authors investigate the phenomenon of adversarial examples, inputs constructed by applying precise, worst-case perturbations to dataset elements, which cause the model to output incorrect predictions with high confidence. Crucially, the authors argue that the primary vulnerability of neural networks stems not from non-linearity or overfitting, as previously hypothesized in the literature, but from their linear behavior in high-dimensional spaces. Leveraging this insight, they introduce the Fast Gradient Sign Method (FGSM), a computationally efficient, single-shot attack mechanism. While the article primarily validates this method on classification tasks and demonstrates its utility in adversarial training, the underlying principles are robust and broadly generalizable to complex tasks such as object detection and segmentation. This work is pioneering in the field of computer vision security, establishing fundamental methodologies for both generating attacks and developing defenses. The remainder of this report is organized as follows: we first define the problem of adversarial perturbations and detail the mechanics of the FGSM; then, we present the results, incorporating our specific contribu-

tion by applying the method to object detection; and finally, we discuss the limitations of these approaches.

## 2. Theoretical Framework

### 2.1. Mathematical Definition of Adversarial Examples

The authors propose that the vulnerability of machine learning models to adversarial examples comes primarily from their linear behavior in high-dimensional spaces. Consider a linear model with a weight vector  $w$  and an input  $x$ . The model’s activation is given by the dot product  $w^\top x$ . An adversarial example  $\tilde{x}$  is constructed by adding a perturbation  $\eta$  to the original input  $x$ :

$$\tilde{x} = x + \eta \quad (1)$$

The perturbation is constrained such that it remains imperceptible or within the sensor precision limit. Formally, this is expressed as the following constraint:

$$\|\eta\|_\infty < \varepsilon \quad (2)$$

where  $\varepsilon$  is a sufficiently small constant. Thus, the activation of the model on the perturbed input becomes:

$$w^\top \tilde{x} = w^\top x + w^\top \eta \quad (3)$$

To maximize the deviation in activation subject to the constraint 2, the authors assign  $\eta = \text{sign}(w)$ . If the weight vector  $w$  has  $n$  dimensions and the average magnitude of an element is  $m$ , the activation will then grow by  $\varepsilon mn$ . This demonstrates that in high-dimensional spaces (large  $n$ ), many small changes can accumulate to produce a massive change in the output, sufficient to cause errors in the outputs of the model.

### 2.2. The Fast Gradient Sign Method

The authors generalize the linear perspective to neural networks, hypothesizing that they are “too linear” to resist linear adversarial perturbations.

In the case of a classification task, as considered by the paper, let  $\theta$  be the parameters of the model,  $x$  be the input, and  $y$  be the target label. The cost function used to train the network is denoted by  $J(\theta, x, y)$ . The authors linearize the cost function around the current value of  $x$  such that  $J(\theta, x + \eta, y) \approx J(\theta, x, y) + \eta^\top \nabla_x J(\theta, x, y)$ . To generate an adversarial example, the goal is to maximize the cost  $J(\theta, x + \eta, y)$  subject to  $\|\eta\|_\infty \leq \varepsilon$ . Similarly to the previous section, this is achieved by setting  $\eta$  in the direction of the gradient sign:

$$\eta = \varepsilon \cdot \text{sign}(\nabla_x J(\theta, x, y)) \quad (4)$$

This formulation is referred to as the Fast Gradient Sign Method. It allows for the efficient generation of adversarial examples using backpropagation to compute the gradient.

### 2.3. Adversarial Training Formulation

Adversarial training is a regularization technique that involves training the model on a mixture of clean and adversarial examples, effectively minimizing the worst-case error when the data is perturbed by an adversarial sample.

To understand the mechanics of this training, we first look at logistic regression, which offers the simplest case where the FGSM yields the exact worst-case perturbation. For a logistic regression model trained to recognize labels  $y \in \{-1, 1\}$ , standard training minimizes the expected cost  $\mathbb{E}_{x, y \sim p_{data}} \zeta(-y(w^\top x + b))$ , where  $\zeta(z) = \log(1 + \exp(z))$ . The adversarial training version modifies this to minimize the worst-case error within the ball of radius  $\varepsilon$ , resulting in the following objective function:

$$\mathbb{E}_{x, y \sim p_{data}} \zeta(y(\varepsilon \|\mathbf{w}\|_1 - \mathbf{w}^\top \mathbf{x} - b)) \quad (5)$$

This formulation resembles  $L_1$  regularization but adds the penalty inside the activation function rather than subtracting it from the cost, meaning the penalty effectively disappears if the model's prediction is confident enough.

For general deep networks, where exact worst-case optimization is intractable, the authors propose a mixed objective function that combines the standard cost on clean examples with the cost on adversarial examples generated via FGSM. The modified objective function  $\tilde{J}$  is defined as:

$$\begin{aligned} \tilde{J}(\theta, x, y) = & \alpha J(\theta, x, y) \\ & + (1 - \alpha) J(\theta, x + \varepsilon \cdot \text{sign}(\nabla_x J(\theta, x, y))) \end{aligned}$$

where  $\alpha$  is a hyperparameter that balances the importance of clean and adversarial accuracy. In the article's experiments, a value of  $\alpha = 0.5$  was used, and this method was shown to act as an effective regularizer.

## 3. Results In Classification

### 3.1. Vulnerability of Standard Architectures

In the article, the authors applied FGSM to several standard models to demonstrate the universality of the vulnerability. On the ImageNet dataset, the authors showed that applying the method to "GoogLeNet" could cause the model to misclassify a panda as a gibbon with 99.3% confidence, despite the perturbation being indistinguishable to the human eye.

Quantitative results on the MNIST dataset revealed that this vulnerability extends to both deep and shallow models. In fact, a shallow softmax classifier exhibited an error rate of 99.9% on adversarial examples with  $\varepsilon = 0.25$  while a deep maxout network misclassified 89.4% of adversarial examples with high confidence (97.6%).

In contrast, models with non-linear properties, specifically Radial Basis Function (RBF) networks, demonstrated natural immunity. While a shallow RBF network had a high error rate (55.4%) on adversarial examples, its confidence on these mistaken examples was extremely low (1.2%), effectively allowing the model to reject inputs it did not understand.

### 3.2. Efficacy of Adversarial Training

The authors implemented the adversarial objective function defined in section 2.3 to train a deep maxout network. This regularization technique yielded significant improvements in robustness.

The error rate on adversarial examples generated via FGSM dropped from 89.4% in the original model to 17.9% in the adversarially trained model. The method also acted as an effective regularizer for clean data. The adversarially trained model achieved a test set error of 0.78%, which was the best result reported on the permutation invariant version of MNIST at the time of publication. As we can see in figure 1, the weights of the adversarially trained model became more localized and interpretable compared to the naively trained model.

The authors compared this approach to other regularization forms, finding that simple  $L_1$  weight decay was insufficient. Coefficients small enough to permit successful training conferred no regularization benefit, while larger coefficients caused the model to get stuck with high training error. Similarly, adding random noise (zero mean and covariance) to inputs proved inefficient because the expected dot product between a weight vector and a random noise vector is zero.

### 3.3. Generalization of Adversarial Examples

A key finding of the paper is transferability, i.e. that adversarial examples generalize across different models and architectures. An example generated for one model is often misclassified by another, even if they have different archi-

tures or training sets. The authors refute the hypothesis that adversarial examples exist in fine, isolated "pockets" of the input space. Instead, they demonstrate that these examples occupy broad, contiguous subspaces defined by the direction of the gradient.

This generalization occurs because different models trained on similar data often learn similar linear functions [? ]. Consequently, a perturbation aligned with the weight vector of one model will likely align with the weight vectors of others. Experiments showed that adversarial examples generated for a deep maxout network were misclassified by a shallow softmax network 84.6% of the time.

### 3.4. Discussion and Limitations

While adversarial training significantly reduced vulnerability, it did not eliminate it. The adversarially trained maxout network still suffered a 17.9% error rate on adversarial examples. Furthermore, when the model did misclassify an adversarial example, it remained highly confident, with an average confidence of 81.4% on mistakes. This indicates that while the model creates a larger margin of safety, it does not fundamentally solve the "blind spot" issue where the model fails to recognize it is leaving the data manifold.

The authors argue that models can only be trained to resist adversarial perturbation if they have the capacity to act as universal approximators. Because the final layer of most neural networks is a linear-softmax or linear-sigmoid layer, it lacks the capacity to represent functions that are constant near training points but distinct at different class locations. This creates a bottleneck where applying adversarial perturbations to the final hidden layer leads to underfitting.

The paper highlights a fundamental tension between ease of optimization and robustness. Models designed to be linear (ReLUs, Maxout, LSTMs) are easy to train using gradient-based optimization but are inherently vulnerable to linear perturbations. Conversely, non-linear models like RBF networks are naturally resistant but difficult to train and lack the generalization capabilities of linear units. The authors conclude that the very properties that make deep networks efficient to train are the root cause of their susceptibility to adversarial attacks.

## 4. Our Contribution: Results in Object Detection

### 4.1. Attacking Images in Object Detection

Object detection is a significantly more complex task than classification. In addition to just assigning a class to images, object detectors often have to handle multiple objects in a single image, determining both their respective positions in bounding boxes and their respective classes. To apply FGSM to our images, we were able to use PyTorch's

*requires\_grad* function to compute the gradient of our detector on the target images.

Due to computational constraints, we focus on lightweight, one-stage detectors, which predict object bounding boxes and class probabilities in a single evaluation pass. In our experiments, adversarial samples were generated targeting the Ultralytics YOLO11n (YOLO11 in "nano" version) model [? ]. Additionally, we utilized PyTorch's SSDlite MobileNetV3 to assess the transferability of our attacks across different architectures.

Both detectors were trained on the COCO (Common Objects in Context) dataset, a comprehensive benchmark comprising 80 object categories and approximately 1.5 million instances [? ]. The adversarial evaluation was performed using the test subset of the VOC2007 dataset. VOC2007 is a widely used standard in object detection, containing nearly 5,000 images with 20 object classes, all of which are a subset of the COCO categories [? ].

To quantify detector degradation under attack, we employ the standard mean Average Precision (mAP) metric. The mAP represents the mean of Average Precision (AP) values, calculated by integrating the Precision-Recall curves for each class. Precision measures the ratio of true positives to total detections, while Recall measures the fraction of ground truth objects successfully detected. A prediction is considered a true positive if its Intersection over Union (IoU) with the ground truth exceeds a specific threshold. Following standard protocols [? ? ? ], we fix this threshold at 0.5. Finally, as adversarial perturbations must remain imperceptible, attack efficacy is evaluated based on the mAP drop relative to the perturbation magnitude  $\epsilon$ .

Figure 2 illustrates the variation in predicted bounding boxes for a single image across different perturbation magnitudes  $\epsilon$ . As the perturbation increases, the detector's performance visibly declines. However, for  $\epsilon$  between 0.08 and 0.10, the noise becomes perceptually obvious, rendering the attack unsuitable for practical scenarios requiring imperceptibility. Beyond general performance degradation, adversarial attacks in object detection manifest in diverse failure modes. Figure 3 presents three distinct outcomes: fabrication, where the model detects non-existent objects; mislabeling, where the wrong class is predicted; and vanishing, where valid objects are erased from predictions. While FGSM produces all these outcomes as a "random outcome" attack, it is worth noting that other methods are designed to specialize in specific error types depending on the target application [? ? ? ].

We quantify the impact of FGSM on detector performance in Figure 4 (a), which plots the mAP against the perturbation magnitude  $\epsilon$  for the YOLO11n-generated samples. As anticipated, performance exhibits an inverse correlation with perturbation intensity. To assess attack transferability, we evaluate these samples on the SSDlite de-

tector. Given the difference in baseline performance between the two models, Figure 4 (b) presents the normalized performance (defined as the ratio of adversarial mAP to clean mAP). This comparison confirms that while the FGSM attack successfully transfers to SSDlite, the performance drop is less precipitous than on YOLO11n. This attenuation is expected, as the adversarial gradients were optimized specifically for the YOLO11n architecture.

## 4.2. Our Attempt at Adversarial Training

The adversarial training strategy employed for our object detection application is more streamlined than the theoretical framework discussed in Section 2.3. Specifically, we fine-tuned the YOLO11n detector using perturbed images to enhance robustness against attacks, following methodologies seen in the literature [? ? ].

The training utilized adversarial samples from the VOC2007 trainval subset [? ], as described in the previous section. We performed a comparative analysis of the baseline YOLO11n detector, a version fine-tuned on clean images, and variants fine-tuned on adversarial examples generated with varying perturbation magnitudes  $\varepsilon$ . To ensure a fair comparison, training configurations remained consistent across all runs, utilizing a maximum of 30 epochs, a learning rate of 0.001, and a batch size of 32.

Figure 5 (a) illustrates the performance of YOLO11n variants following adversarial training across various perturbation magnitudes. A fundamental trade-off is observed: as the detectors' robustness to attacks increases, their efficacy on clean images declines. Preserving high performance on unperturbed data is critical, as these images constitute the predominant input in real-world scenarios. Consequently, balancing standard accuracy with adversarial robustness is essential for effective adversarial training. To quantify this impact, Figure 5 (b) details the performance of each detector specifically on clean images.

Fine-tuning on clean VOC training images enhanced both the detector's performance on the clean test set and its robustness against moderate perturbations ( $\varepsilon < 0.10$ ). Resistance to these attacks is particularly critical, as they remain imperceptible to the human eye. This improvement is likely attributed to the reduction in the domain gap between the original pre-training dataset COCO and the target dataset VOC. While this benefit is not observed at larger perturbation magnitudes, performance in that regime degrades to such an extent that meaningful estimation becomes unreliable. As previously discussed, increasing the perturbation magnitude during fine-tuning enhances adversarial robustness but comes at the expense of accuracy on clean images. Our experiments reveal that fine-tuning with a low magnitude ( $\varepsilon=0.02$ ) provides negligible defense against most attacks. Conversely, using a high magnitude ( $\varepsilon=0.20$ ) causes performance on clean images to drop be-

low acceptable thresholds. Therefore, we identify the range of  $\varepsilon \in [0.08, 0.10]$  as the optimal operating point in our case, offering the most appropriate balance between standard accuracy and adversarial robustness.

## 4.3. About the Imperceptibility Constraint

In section 2.1, we introduced the imperceptibility constraint as  $\|\eta\|_\infty < \varepsilon$ , with a  $L_\infty$  norm. However, as the primary constraint of adversarial attacks is imperceptibility, the definition must rely on the human eye perception. This raises the question of whether the  $L_\infty$  norm is an adequate estimation for human perception. Figure 2 displays an image and its adversarial counterparts across various  $\varepsilon$  values. While quantifying the precise sensitivity of the human eye is complex, the relationship seems non-linear: a numerical increment at a low magnitude (e.g., from 0.02 to 0.04) does not seem equivalent to the same increment at a higher magnitude (e.g., from 0.08 to 0.10). We propose to introduce alternative similarity measures designed to better estimate visible disturbance and evaluate their correlation with the evolution of  $\varepsilon$ .

The first relevant metric to consider is the Structural Similarity Index Measure (SSIM), a perception-based method for estimating image similarity. Unlike simple pixel-wise distance metrics, SSIM accounts for structural dependency, the principle that spatially adjacent pixels exhibit strong inter-dependencies, while also incorporating perceptual factors such as luminance and contrast masking [? ? ]. The second metric employed is Learned Perceptual Image Patch Similarity (LPIPS). LPIPS is explicitly designed to align with human visual perception by leveraging deep learning representations. It operates on the premise that internal activations within networks trained for high-level classification correlate strongly with human perceptual judgments. LPIPS computes the distance between images within a feature space using a pre-trained neural network (AlexNet in our experiments) [? ].

Figure 6 illustrates the evolution of the  $L_\infty$ , SSIM, and LPIPS metrics as a function of  $\varepsilon$ . Specifically, Figure 6 (a) depicts the average values, while Figure 6 (b) displays the worst-case values, which are the primary interest when establishing safety constraints. As expected, Figure 6 (b) confirms a direct linear correlation between  $\varepsilon$  and the measured  $L_\infty$  norm. The perceptual metrics, SSIM and LPIPS, exhibit non linear behavior, especially for the lower perturbation values ( $\varepsilon < 0.10$ ), as hypothesized. While incorporating these perceptual measures into the attack optimization process could present a mathematical challenge, they seem to provide a more relevant constraint for real-world scenarios.

#### 4.4. Discussion and Limitations

As detailed in the section 4.1, our experimental scope is constrained by hardware limitations, which restricts our evaluation. Regarding object detectors, we were only able to deploy lightweight variants of one-stage models where larger or more advanced architectures might have demonstrated increased robustness. Two-stage detectors such as R-CNNs, which first generate region proposals and subsequently classify them, are also known to be more resilient to adversarial perturbations, prioritizing accuracy over inference speed [? ].

The attack method FGSM now represents a baseline approach in the adversarial machine learning literature. While computationally efficient and simple to implement, contemporary attack methods are typically more sophisticated, often relying on multi-step optimization procedures and exploiting specific vulnerabilities in detector components [? ? ]. Moreover, recent work places strong emphasis on transferability, aiming to produce perturbations that remain effective across a wide range of detector architectures and training pipelines [? ? ? ]. In realistic adversarial scenarios, attackers generally lack access to the defender’s model architecture, weights, training data, or configuration. Defenders, on the other hand, must anticipate attacks originating from diverse threat models and detector types, and therefore must carefully select the attack algorithm used for adversarial training, if such training is implemented, to ensure maximal robustness.

Furthermore, although our study focuses on attacks aimed at remaining imperceptible to the human eye, many adversarial strategies in object detection rely on patch-based attacks. In such approaches, the adversarial effect is concentrated within a localized region of the image. These patches are typically not designed to be visually subtle but they are instead intended to be easily deployable in real-world settings due to their physical, sticker-like shape [? ? ].

Regarding our adversarial training protocol, while the approach proved effective, it would be valuable to examine the influence of the number of images for the fine-tuning process, as well as the impact of mixing clean and perturbed samples during training. It is also important to note that alternative adversarial training frameworks exist, some of which incorporate additional neural networks as discussed in [? ], and that a variety of other defense methods have been proposed in the literature [? ? ? ].

#### 5. Conclusion

In this report, we reviewed the article ”Explaining and Harnessing Adversarial Examples” [? ], which shows that adversarial examples arise from the linear behavior of high-dimensional neural networks, and studied FGSM as both an

attack mechanism and a tool to perform adversarial training.

We extended this investigation to object detection, demonstrating that lightweight one-stage detectors remain highly sensitive to small bounded perturbations, which can induce object disappearance, misclassifications, or false detections. Despite the increased complexity of detection tasks, adversarial examples remained transferable between models. Adversarial training improved robustness but at the expected cost of reduced clean-image performance, with a limited perturbation range offering the best trade-off under our computational constraints.

Finally, by comparing  $L_\infty$  constraints with perceptual metrics such as SSIM and LPIPS, we observed that imperceptibility does not scale linearly with  $\epsilon$ , suggesting that those distances might be more suitable to express the applied perturbations.

#### Appendix: Figures

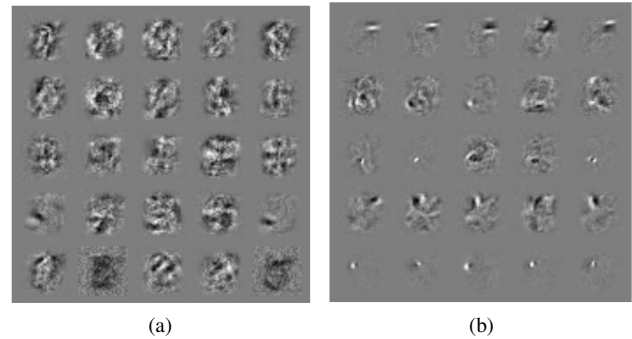


Figure 1. Weight visualizations of maxout networks trained on MNIST. Each row shows the filters for a single maxout unit. a) Naively trained model. b) Model with adversarial training (From [? ]).

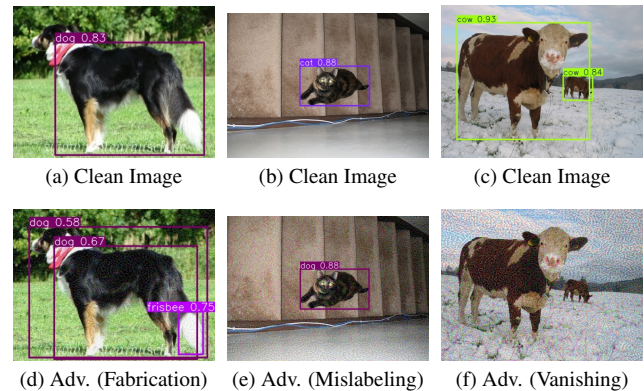


Figure 3. Example of outputs generated by the adversarial attack.

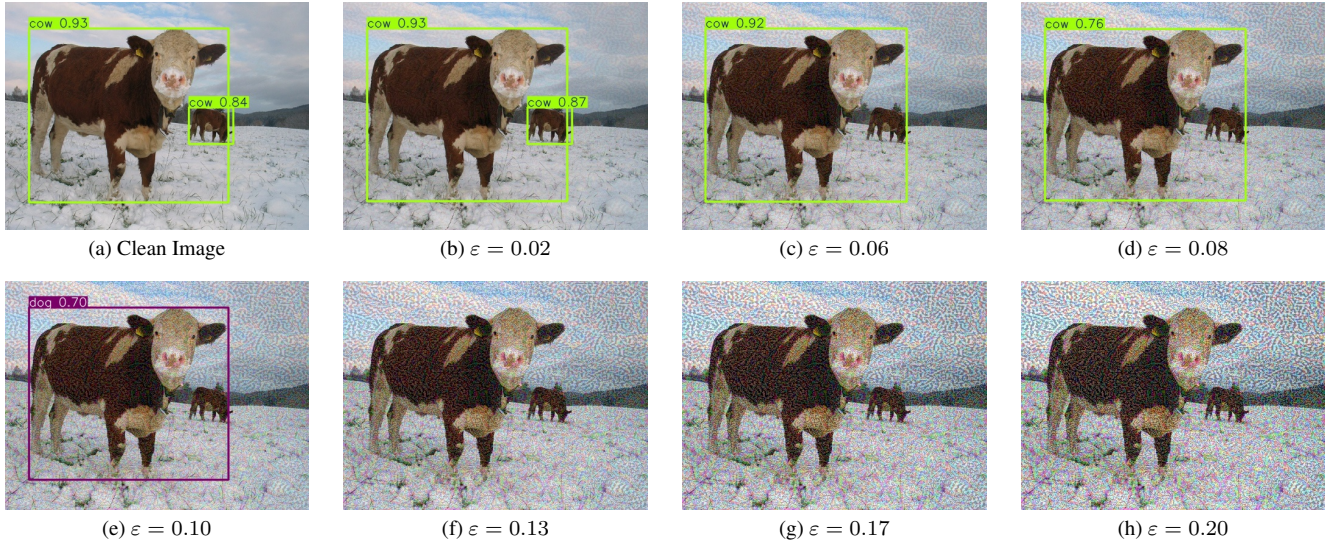
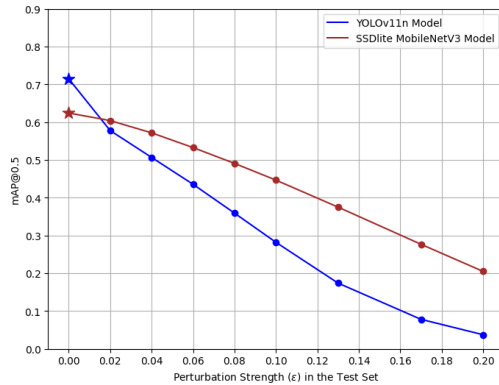
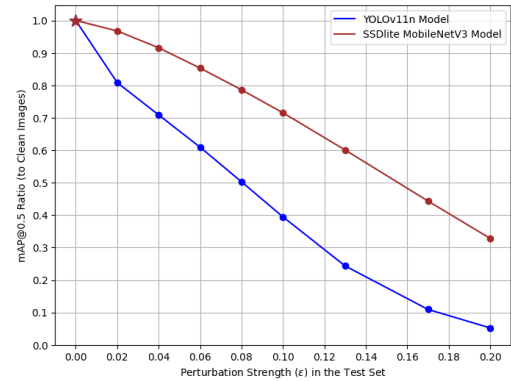


Figure 2. Example of detections provided by the YOLO11n Model of an image with increasing perturbation value  $\varepsilon$ .

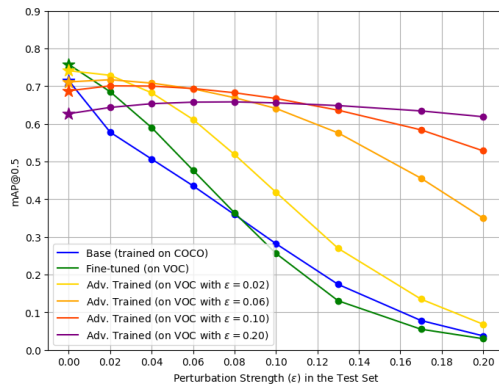


(a)

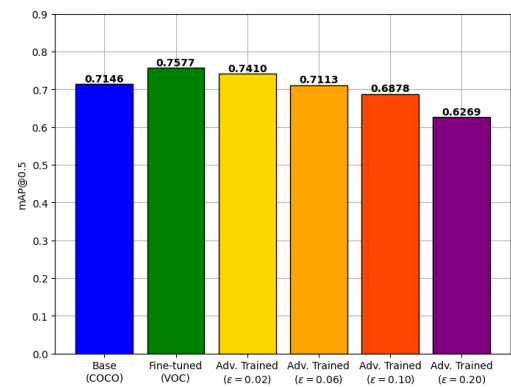


(b)

Figure 4. Comparison of YOLOv11n and SSDlite MobileNetV3 Models under Adversarial Perturbations with metrics a) mAP b) mAP ratio (to its maximum value on Clean Images).



(a) Performance after Adversarial Training



(b) Performance on Clean Images only

Figure 5. Comparison of YOLOv11n Model Performance. (a) illustrates robustness against adversarial attacks, while (b) details the impact on clean image accuracy.

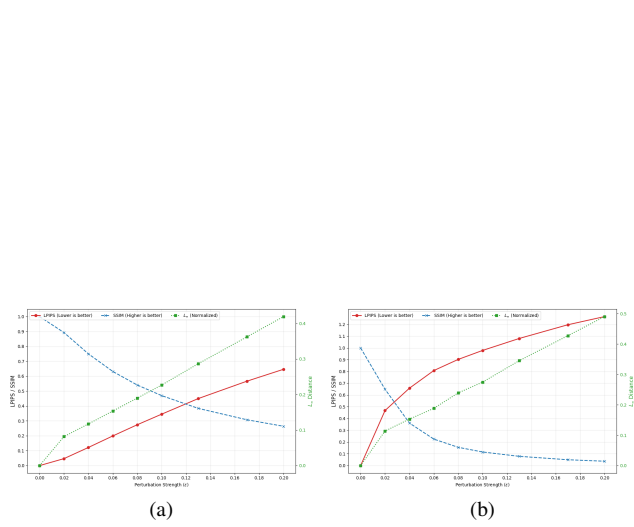


Figure 6. Estimation of the difference between the clean and perturbed images with several distances using (a) the mean value and (b) the worst value of each.