Deep Learning

Explaining and Harnessing Adversarial Examples

Adonis JAMAL   Jean-Vincent MARTINI

École Normale Supérieure Paris-Saclay - MVA
CentraleSupélec - MDS

January 6th 2026

Motivation & Problem Statement

The Context:

- Neural Networks are vulnerable to **Adversarial Examples**: inputs with worst-case perturbations causing high-confidence errors [GSS14].
- This is a security risk for safety-critical applications (e.g., autonomous driving).

Project Goals

1. Analyze the seminal work by Goodfellow et al. (2015).
2. Extend the analysis from Classification to Object Detection (YOLO11n).

Theoretical Framework: The Linearity Hypothesis

Why are deep networks vulnerable?

- *Old Hypothesis:* Overfitting or extreme non-linearity.
- *Goodfellow's Hypothesis:* Models are **"too linear"** in high-dimensional spaces.

Consider a linear activation $\boldsymbol{w}^\top \boldsymbol{x}$. With perturbation $\boldsymbol{\eta}$:

$$\boldsymbol{w}^\top(\boldsymbol{x} + \boldsymbol{\eta}) = \boldsymbol{w}^\top \boldsymbol{x} + \boldsymbol{w}^\top \boldsymbol{\eta} \tag{1}$$

If we set $\boldsymbol{\eta} = \varepsilon \cdot \text{sign}(\boldsymbol{w})$:

- The activation grows by εmn (where n is dimensionality).
- Many small changes accumulate to a massive shift in output.

The Fast Gradient Sign Method (FGSM)

FGSM is a single-shot attack designed to maximize loss under an L_∞ constraint.

The Attack Formula

$$\boldsymbol{\eta} = \varepsilon \cdot \text{sign}(\nabla_x J(\boldsymbol{\theta}, \boldsymbol{x}, y)) \tag{2}$$

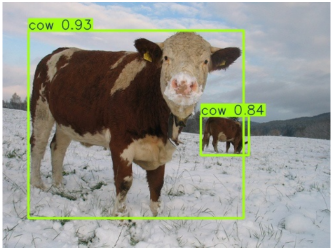
- **Efficient:** Requires only one backpropagation pass.
- **Dual Use:** Used for generating attacks and for adversarial training (regularization).
- **Constraint:** $||\boldsymbol{\eta}||_\infty < \varepsilon$ ensures imperceptibility.

Our Contribution: Object Detection Setup

We extended the study to **One-Stage Detectors**.

Experimental Setup:

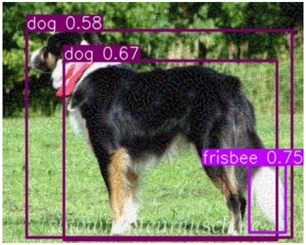
- **Target Model:** YOLO11n (Nano) [JQ24].
- **Transfer Target:** SSDlite MobileNetV3.
- **Dataset:** Trained on COCO, Evaluated on VOC2007.
- **Metric:** Mean Average Precision (mAP) @ IoU=0.5.



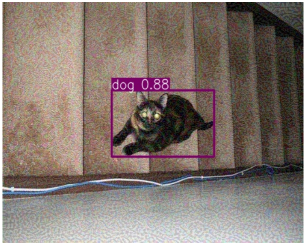
Clean Detection (Confidence 0.70)

Attack Results: Failure Modes

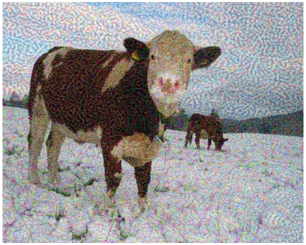
FGSM successfully degrades detection performance, leading to three specific failure modes:



Fabrication: Detecting non-existent objects.



Mislabeling: Predicting wrong classes.

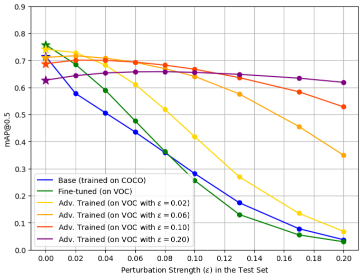


Vanishing: Valid objects erased from prediction.

Transferability: Attacks generated for YOLO11n also degraded SSDlite (Black-box scenario), confirming the universality of the vulnerability.

Adversarial Training: The Trade-off

We fine-tuned YOLO11n on adversarial examples.

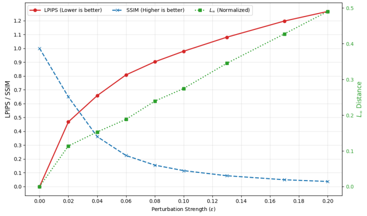


Key Findings:

- **Trade-off:** Increasing ε improves robustness but degrades performance on clean images.
- **Low ε (0.02):** Negligible defense.
- **High ε (0.20):** Destroys clean accuracy.
- **Sweet Spot:** We identified $\varepsilon \in [0.08, 0.10]$ as the optimal balance for this architecture.

Critique of Imperceptibility

Is L_∞ a good proxy for human perception?



- We compared L_∞ against perceptual metrics: **SSIM** and **LPIPS**.
- **Result:** While L_∞ scales linearly with ε, perceptual degradation (LPIPS) is non-linear.
- **Implication:** Future defenses should optimize against perceptual distances rather than simple pixel norms.

Conclusion

1. **Validation:** We confirmed Goodfellow's hypothesis that linearity is the primary cause of adversarial vulnerability.
2. **Generalization:** We showed these vulnerabilities persist in complex Object Detection tasks (YOLO11n).
3. **Defense:** Adversarial Training is effective but introduces a critical trade-off between robustness and standard accuracy.
4. **Future Work:** Investigating patch-based attacks and integrating perceptual metrics (LPIPS) into the loss function.

Bibliography I

[GSS14] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. "Explaining and harnessing adversarial examples". In: *arXiv preprint arXiv:1412.6572* (2014).

[JQ24] Glenn Jocher and Jing Qiu. *Ultralytics YOLO11*. Version 11.0.0. 2024. URL: https://github.com/ultralytics/ultralytics.