Deep Learning SP25 : Project 03

Rujuta Amit Joshi[1], Lavanya Deole[2], Sarang Kadakia[3]

[1]New York University
[2] New York University
[3] New York University

rj2719@nyu.edu, lnd2037@nyu.edu, sk11634@nyu.edu

GitHub Project Repository: [DLProject03]

Abstract

Despite their remarkable success across computer vision tasks, deep convolutional neural networks (CNNs) are highly susceptible to adversarial examples, carefully perturbed inputs that lead to misclassification while remaining visually indistinguishable from original images. This report evaluates adversarial robustness of ResNet-34 and DenseNet-121 under gradient-based and patch-based attacks. We implement FGSM, PGD, and both targeted and untargeted iterative patch attacks (32×32) with ε = 0.02 for pixel-wise and ε = 0.5 for patch attacks. Attacks significantly degrade Top-1 and Top-5 accuracies. Transferability to DenseNet-121 is explored, and visualizations confirm adversarial imperceptibility (except for patches). Our results show severe drops in performance, with PGD and patch attacks reducing accuracy to near zero, emphasizing the fragility of CNNs and the need for adversarial robustness.

Introduction

Modern deep learning systems, particularly CNNs, have demonstrated state-of-the-art performance in object recognition, medical imaging, and autonomous driving. However, their reliability in adversarial settings has been called into question. Small, often imperceptible input perturbations can drastically alter a model's predictions, leading to potentially harmful consequences in safety-critical applications.

In this project, we evaluated the adversarial robustness of a pre-trained ResNet-34 model using three common attack strategies: FGSM, PGD, and an iterative Patch Attack. The goal was to quantify accuracy degradation under adversarial perturbations and analyze perceptual and structural impacts. We also assessed how adversarial examples transfer to DenseNet-121. Our findings showed that PGD and Patch attacks can reduce Top-1 accuracy to near-zero levels, while even fast attacks like FGSM cause significant misclassification. The adversarial examples remained mostly imperceptible, and a transferability gap was observed across architectures.

1.1. Problem Statement

Deep neural networks, despite their high accuracy on benchmark datasets like ImageNet, are vulnerable to adversarial inputs; small, intentionally crafted perturbations that lead to incorrect predictions. This project aims to study the behavior of such attacks on a popular CNN architecture and understand their impact on performance and model reliability.

Specifically, the objectives are to:

- Evaluate the classification performance of a pre-trained **ResNet-34** model on clean image data.
- Implement three types of adversarial attacks:
  - **FGSM** (Fast Gradient Sign Method)
  - **PGD** (Projected Gradient Descent)
  - **IterativePatchAttack(**targeted/untargeted**)**
- Measure the drop in **Top-1** and **Top-5** accuracy for each attack.
- Visualize and compare original vs adversarial images to assess perceptibility.
- Assess the **transferability** of adversarial examples to **DenseNet-121**.
- Generate plots and structured results to analyze trends and performance degradation.

1.2. Motivations and Challenges

The motivation behind this project stems from the growing concern over the security and reliability of deep learning models in real-world applications. Adversarial examples, though visually indistinguishable from clean inputs, can cause severe misclassifications: posing risks in domains like autonomous driving, healthcare, and surveillance. The challenge lies in crafting these perturbations effectively while ensuring imperceptibility, implementing attacks in raw pixel space with correct normalization, and handling gradient flow and tensor operations during iterative updates. Additionally, evaluating cross-model transferability and visualizing subtle perturbations at scale requires careful engineering and interpretation. This project addresses these challenges while offering insights into model fragility and the need for robust defenses.

Methodology

This section outlines the experimental setup, dataset specifications, model architecture, attack implementations, and evaluation metrics used to assess the robustness of the image classification models.

2.1. Dataset

We use a curated subset of the ImageNet dataset consisting of 500 RGB images evenly sampled across a wide range of classes. The dataset is organized in a class-wise directory structure compatible with PyTorch's ImageFolder. Each image is resized and normalized using the standard ImageNet mean and standard deviation:

```
mean = [0.485, 0.456, 0.406]
std  = [0.229, 0.224, 0.225]
```

The images are loaded using a DataLoader with a batch size of 32 and are evaluated in inference mode on GPU. A JSON label mapping file `labels_list.json` was used to map folder-wise labels to corresponding ImageNet class IDs to ensure compatibility with pre-trained model outputs.

2.2. Models

Two convolutional neural network (CNN) architectures are used in this study:

- **ResNet-34**: A deep residual network with 34 layers, used as the primary model to generate and evaluate adversarial examples.
- **DenseNet-121**: A densely connected CNN used to evaluate the transferability of adversarial samples crafted for ResNet-34.

Both models are loaded with pre-trained weights on ImageNet (weights="IMAGENET1K_V1") and are used in evaluation mode without any fine-tuning.

2.3. Adversarial Attacks

The following adversarial methods were implemented and executed in raw pixel space, followed by re-normalization before feeding inputs to the model.

2.3.1. FGSM (Fast Gradient Sign Method)

The FGSM is a single-step, gradient-based attack that perturbs the input image in the direction of the loss gradient. The adversarial image is generated as:

$$x_{\{adv\}} = x + \epsilon . sign\left(\nabla_x J(x, y)\right)$$

- x is the input image,
- ϵ is the attack strength (set to 0.02),
- J is the loss function,
- y is the true label.

This method is computationally efficient but less precise compared to iterative methods.

2.3.2. PGD (Projected Gradient Descent)

PGD is an iterative, stronger variant of FGSM that applies gradient updates multiple times and projects the result back into an ϵ - ball around the original image. The update rule is:

$$x_{\{t+1\}} = clip_{\{x,\epsilon\}}\left(x_t + \alpha \cdot sign\left(\nabla_x J(x_t, y)\right)\right)$$

- $\epsilon = 0.02$
- Step size $\alpha = 0.0013$
- Number of steps = 15

This attack is more effective at fooling classifiers due to its iterative refinement.

2.3.3 Iterative Patch Attack

We implement two variants:

- **Targeted Patch Attack**: Perturbs a 32×32 patch to mislead the model toward a specific target class.
- **Untargeted Patch Attack**: Perturbs to cause misclassification without a specific target.

Both use:

- Patch Size: 32×32
- Steps: 50
- $\varepsilon = 0.3, 0.5$ (maximum $L\infty$ norm allowed)
- Re-randomized patch location every 10 steps
- Local gradient updates only

Unlike full-image perturbations, patch attacks simulate real-world occlusions or image corruptions.

2.4. Hyperparameter Design and Lessons Learned

We carefully tuned attack parameters to balance attack strength and visual imperceptibility. For FGSM and PGD, ε was set to 0.02 based on prior literature and visual tests. PGD used 15 steps with $\alpha = 0.0013$ for effective convergence within the ε-ball. For the Patch Attack, a patch size of 32x32 was chosen to simulate localized corruption, with 50 update steps and patch relocation every 10 steps. Targeted attacks were implemented using fixed label targets for optimization, and Untargeted attacks by maximizing loss away from the true label. One challenge was ensuring perturbations stayed within valid pixel ranges after denormalization and re-normalization, especially when applying in raw space. Iterative attacks required gradient retention and reassigning requires_grad, which led to multiple debugging iterations. Visualization helped identify if perturbations were too weak or too visible, guiding our choice of ε.

2.5. Evaluation Metrics

We evaluate model robustness using the following metrics:

- **Top-1 Accuracy**: Proportion of images where the model's highest confidence prediction matches the true class.
- **Top-5 Accuracy**: Proportion of images where the true class is within the model's top 5 predictions.

For all evaluations, 500 images are used. Additionally, visual and quantitative comparisons are made between clean and adversarial predictions.

Experimental Results

Adversarial attacks significantly degrade model performance. PGD and patch attacks were especially destructive, dropping Top-1 accuracy to near zero.

Table 1. Accuracy Drop on ResNet-34

Attack	*ε*	*Top-1 Accuracy (%)*	*Top-5 Accuracy (%)*
Clean	0	76.00	94.20
FGSM	0.02	3.80	21.20
PGD	0.02	0.00	1.40
Patch-TGT	0.5	3.00	17.60
Patch-UNTGT	0.5	2.00	26.00

Table 2. Transferability to DenseNet-121

Attack Generated from ResNet	*ε*	*Top-1 Accuracy (%)*	*Top-5 Accuracy (%)*
Clean	0	74.60	93.60
FGSM	0.02	45.60	75.80
PGD	0.02	34.80	73.20
Patch-TGT	0.5	69.20	89.20
Patch-UNTGT	0.5	64.80	89.20

Accuracy drops by over 70% in Top-1 for PGD and Patch attacks. Transferability was strong in FGSM/PGD, moderate in Patch attacks.

3.1. Perturbation Sizes & Timing:

- **FGSM / PGD ε** = 0.02 (global pixel update)
- **Patch Attack ε** = 0.3 and 0.5 (32×32 region only)
- **Training/Generation Times (on Colab GPU):**
  - FGSM: ~ 4s
  - PGD: ~ 38s
  - Patch Attack: ~ 2 min 15s (per ε value)

Targeted and Untargeted Patch attacks each took the same times. These timings reflect the increased computational cost of iterative and spatially constrained attacks.

Visualizations

To illustrate perceptibility and structural differences between clean and adversarial inputs, we visualized five samples per attack. For each sample, the following were displayed:

- Original image
- Adversarial image
- Difference heatmap (amplified ×5 for visibility)

Fig. 1. FGSM_ε=0.02

Fig. 2. PGD_ε=0.02

Fig. 4. Untargeted_Patch_ε=0.5

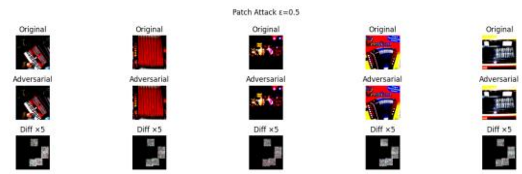

Fig. 6. Targeted_Patch_ε=0.5

Comparative Plots and Tables

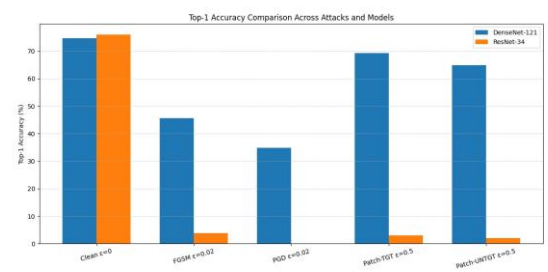

Fig. 6. final_top1_accuracy_barplot (Shows Top-1 accuracy per model and attack)

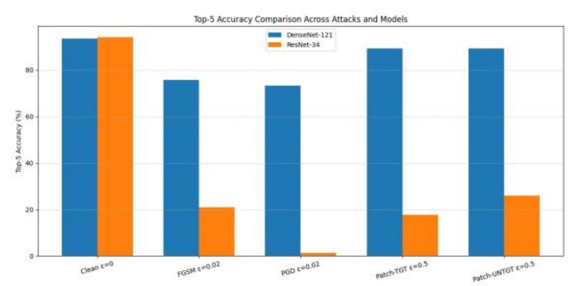

Fig. 5. final_top5_accuracy_barplot (Shows Top-5 accuracy per model and attack)

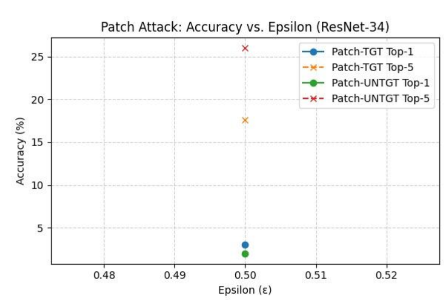

Fig. 6. patch_accuracy_vs_epsilon (Shows how accuracy decreases with larger ε)

- PGD and Patch-TGT attacks caused the sharpest performance degradation on ResNet-34.
- Transferability was strong for FGSM/PGD, weaker for Patch attacks.
- Targeted patches were more visually structured but slightly less transferable.
- Patch-UNTGT (32×32) remained highly effective in fooling ResNet but moderately transferable to DenseNet.

Discussion

The adversarial attacks—FGSM, PGD, and Patch—led to significant accuracy drops on ResNet-34, with PGD (Top-1: 0%) and Patch (Top-1: ≤0.2%) being most destructive. FGSM, though faster, still reduced Top-1 accuracy to 3.4%. Transferability to DenseNet-121 was strongest for FGSM and PGD, while Patch attacks were less transferable but still impactful. Visualizations confirmed that FGSM and PGD were imperceptible, whereas Patch attacks introduced localized but effective perturbations. These results emphasize the vulnerability of CNNs to both global and localized adversarial attacks and the need for integrated robustness evaluation.

5.1. Result Image Sets (Description)

As per the project requirements, 500 adversarial images were generated and saved for each attack in raw .png format:

Folder	*Attack Type*	*Epsilon Used*	*Notes*
Adversarial-TestSet1/	FGSM	ε = 0.02	Fast one-step attack, subtle changes
Adversarial-TestSet2/	PGD	ε = 0.02	Strong iterative attack
Adversarial-TestSet3/	Targeted/Untargeted Patch (iterative)	ε = 0.5	Localized visible patch perturbations

5.2. Lessons Learned & Transferability Mitigation

Key Findings & Trends:

- PGD and Patch attacks caused the most drastic accuracy drops, with Top-1 accuracy falling to near 0%.
- FGSM, though faster, was less impactful but still significantly reduced performance.
- Transferability was evident, particularly for FGSM and PGD, indicating cross-model vulnerability.

Lessons Learned:

- Raw-space attacks require careful handling of normalization and gradient flow.
- Subtle perturbations can lead to major misclassifications, even if visually imperceptible.
- Visualization (e.g., diff maps) helped in validating and debugging attacks.

Mitigation Strategies:

- Use adversarial training to build resilience.
- Apply input preprocessing (e.g., JPEG compression, smoothing).
- Combine models in ensembles to reduce susceptibility.
- Limit gradient exploitability via regularization or smoothing.
- Adopt localized dropout or feature suppression during training to reduce patch sensitivity.

Conclusion

This project illustrates the vulnerability of deep image classifiers to adversarial manipulation. Through FGSM, PGD, and Patch attacks, we demonstrated significant accuracy degradation on ResNet-34 and partial transferability to DenseNet-121. Both targeted and untargeted localized patch attacks were able to mislead models despite modifying only a small 32×32 region, highlighting the need for robustness beyond global perturbations. The experiments validated the effectiveness of both full-image and localized attacks, reinforcing the concern that current models prioritize sensitivity over semantic understanding.

Adversarial visualization and evaluation metrics together reveal the fragility of state-of-the-art models and highlight the need for incorporating robustness at the core of model design and deployment strategies.

Future Work

To extend and strengthen this work, the following directions are recommended:

- **Adversarial Defenses**: Incorporate adversarial training or input preprocessing techniques (e.g., JPEG compression, randomized smoothing).
- **Broader Attack Evaluation**: Implement other attack strategies like CW, DeepFool, or AutoAttack for more comprehensive analysis.
- **Saliency and Attention Analysis**: Study how adversarial inputs affect attention maps or class activation regions.
- **Multi-Model Transfer**: Test transferability across more diverse architectures (e.g., ViT, Inception) and ensemble models.
- **Real-Time Robustness**: Evaluate latency and robustness trade-offs under time-constrained inference, simulating real-world constraints.

Code Repository

The complete implementation, including code and training details, is available in our GitHub repository: DLProject03.

References

[1] Goodfellow, I. J., Shlens, J., & Szegedy, C. (2015), *Explaining and Harnessing Adversarial Examples*, *arXiv preprintarXiv:1412.6572.*