Evaluating and Enhancing Adversarial Attacks on Deep Neural Network Image Classifiers

Raunak Choudhary, Sharayu Rasal
New York University
rc5553@nyu.edu, srr10019@nyu.edu
GitHub Repository: https://github.com/raunak-choudhary/DeepModel-AdversarialAttacks-ResNet-DenseNet.git

Abstract

This project investigates the vulnerability of deep neural networks to adversarial attacks, targeting a pre-trained ResNet-34 image classifier on an ImageNet-1K subset. We implement and evaluate four adversarial attack methods: Fast Gradient Sign Method (FGSM), Projected Gradient Descent (PGD), Momentum Iterative FGSM (MI-FGSM), and a spatially constrained Patch Attack. Empirical results show varying efficacy; iterative attacks like PGD and MI-FGSM are most potent in white-box settings, reducing ResNet-34's Top-1 accuracy to 0.00% and 0.20%, respectively. FGSM (3.40% Top-1 accuracy) is a computationally efficient baseline. Patch attacks, modifying only a 32x32 pixel region, significantly impact performance (12.00% Top-1 accuracy) and show considerable transferability to an unseen DenseNet-121 model. This study highlights adversarial threats and the need for robust defense mechanisms in deep learning.

Overview

Deep neural networks' proliferation highlights their vulnerabilities, especially to adversarial attacks. This project evaluates prominent adversarial strategies against production-grade image classifiers, focusing on "jailbreaking deep models." Our primary target is ResNet-34 (pre-trained on ImageNet-1K), with DenseNet-121 used for transferability tests. Adversarial attacks use crafted, often imperceptible, perturbations to cause misclassifications and degrade model performance [1]. Understanding these attack vectors is crucial for resilient AI.

Our approach implements and analyzes four diverse attack methodologies:

- **Fast Gradient Sign Method (FGSM):** A foundational, single-step attack perturbing images along the loss gradient.
- **Projected Gradient Descent (PGD):** An iterative attack with multiple small steps, projecting perturbations within an ϵ-ball.
- **Momentum Iterative FGSM (MI-FGSM):** Enhances iterative methods with momentum for stable updates and better transferability.
- **Patch Attack:** A localized attack confining perturbations to a small image patch (e.g., 32x32), simulating plausible physical attacks.

These attacks were tested on 500 ImageNet-1K images (100 classes). Effectiveness was measured by Top-1/Top-5 accuracy reduction on ResNet-34. Adversarial examples were also tested on DenseNet-121 for transferability. Findings show iterative methods (PGD, MI-FGSM) are highly effective on the source model, while patch attacks show notable transferability.

Methodology

This section covers data acquisition/preprocessing, model architectures, adversarial attack design, perturbation generation, evaluation, and hyperparameter rationale.

Data Processing and Preprocessing

Evaluation used an ImageNet-1K subset (500 images, 100 classes) [6]. Input images, assumed to be 224x224 pixels, were preprocessed using a PyTorch `preprocess_transform` pipeline involving:

- **Tensor Conversion:** PIL Images to PyTorch tensors (via `transforms.ToTensor()`).
- **Normalization:** Pixel values to ImageNet statistics (mean: $[0.485, \dots]$, std: $[0.229, \dots]$) using `transforms.Normalize()`.

Model Architecture

Two pre-trained CNNs from TorchVision were used:

- **ResNet-34 (Target Model):** A 34-layer Residual Network [4], loaded with `IMAGENET1K_V1` weights.
- **DenseNet-121 (Transfer Model):** A 121-layer Densely Connected Network [5], also with `IMAGENET1K_V1` weights, for transferability tests.

Both models operated in evaluation mode (`model.eval()`) for deterministic outputs during all attack and evaluation phases.

Adversarial Attack Design

Four distinct adversarial attack strategies were implemented, each leveraging different principles to generate adversarial examples:

Fast Gradient Sign Method (FGSM): FGSM [1] is an efficient, single-step L_∞ attack. In our implementation, after enabling gradients on the input image, the

`nn.CrossEntropyLoss` L is computed. Its gradient $\nabla_x L$ (accessed via the `.grad.data` attribute after `loss.backward()`) guides the perturbation. The image x is updated by $x_{adv} = x + \epsilon \cdot \text{sign}(\nabla_x L(\theta, x, y))$, using the L_∞ perturbation budget ϵ. Resultant adversarial images are clamped to the valid [0,1] pixel range using `torch.clamp` to ensure image validity.

Projected Gradient Descent (PGD): PGD [2] is a powerful iterative L_∞ attack. Our `pgd_attack` implementation first initializes the adversarial image by adding a random perturbation uniformly sampled within the ϵ-ball (specifically via `torch.empty_like(image).uniform_(-`ϵ`,` ϵ`)`) to the clean image, and then clamps the result to ensure valid pixel values. Then, for a fixed number of iterations, it applies the gradient ascent step: $x_{t+1} = x_t + \alpha \cdot \text{sign}(\nabla_x L(\theta, x_t, y))$, using gradients derived from the `nn.CrossEntropyLoss`. Critically, after each update, the total perturbation $(x_{t+1} - x_{original})$ is explicitly clipped to remain within the ϵ-neighborhood of the original image, and x_{t+1} is again clamped to the [0,1] range.

Momentum Iterative FGSM (MI-FGSM): MI-FGSM [3], implemented in our code as `mi_fgsm_attack`, enhances iterative attacks by incorporating a `momentum` vector, which is initialized to zeros. In each iteration, the current gradient of the `nn.CrossEntropyLoss` with respect to the input is first normalized by its L_1 norm (using, e.g., `grad / torch.norm(grad, p=1)`). This normalized gradient is then accumulated into the momentum term: $g_t = \mu \cdot g_{t-1} + \frac{\nabla_x L(\theta, x_{t-1}, y)}{\|\nabla_x L(\theta, x_{t-1}, y)\|_1}$, where μ is the `decay_factor`. The image is subsequently updated using $x_t = x_{t-1} + \alpha \cdot \text{sign}(g_t)$. Like PGD, the resulting image is then projected and clamped to stay within the ϵ-ball and maintain valid pixel values.

Patch Attack: Our `advanced_patch_attack` is an L_0-bounded method modifying a 32×32 pixel region (`patch_size`). It performs `num_restarts=5` runs, selecting patch locations from `strategic_locations` (e.g., corners, center) and optimizing patch pixels via Adam (`optim.Adam`). The goal is to maximize a chosen target's logit (from `potential_targets` derived from model outputs/label offsets) against the true label's, using a loss like `-outputs[0,tgt] + outputs[0,true_label_idx]`. A larger local L_∞ budget ϵ (`epsilon_patch`) applies within the patch.

Training Process (Perturbation Generation)

"Training" here refers to generating adversarial perturbations for PGD, MI-FGSM, and Patch attacks, not model weight adjustment (weights are frozen). An adversarial loss (e.g., `nn.CrossEntropyLoss` for PGD/MI-FGSM, or a custom targeting loss for the Patch attack) is optimized with respect to input image pixels x. This involves enabling input gradients (e.g., `image.requires_grad=True`), performing forward/backward passes for $\nabla_x L$, and then updating pixels, subject to perturbation constraints (L_∞ or spatial).

Iterative attacks progressively modify inputs to increase this loss (or meet target objectives for Patch attacks), pushing images towards misclassification. FGSM is a single-step method; its perturbation is calculated once from the initial gradient obtained after `loss.backward()`.

Evaluation and Prediction

Model performance on clean and adversarial data was measured by Top-1 and Top-5 accuracy:

- **Top-1 Accuracy:** The model's highest probability prediction matches the true class label.
- **Top-5 Accuracy:** The true class label is among the model's five highest probability predictions.

Model logits were processed using `torch.topk(outputs, k=5)` to get the top 5 prediction indices. These were compared to ground truth label indices for accuracy calculation, following the logic in our `evaluate_model` function.

Hyperparameter Choices

Key hyperparameters, corresponding to variables in our Python code, were set for effectiveness and fair comparison:

- **FGSM:** $\epsilon = 0.02$ (`epsilon_fgsm_pgd`).
- **PGD:** $\epsilon = 0.02$ (`epsilon_fgsm_pgd`); $\alpha = 0.005$ (`pgd_alpha`, $\approx \epsilon/4$); Iterations $= 10$ (`pgd_iterations`).
- **MI-FGSM:** $\epsilon = 0.02$ (`epsilon_fgsm_pgd`); Steps $= 10$ (`mi_fgsm_steps`); $\mu = 0.9$ (`mi_fgsm_decay_factor`).
- **Patch Attack:** Size $= 32 \times 32$ (`patch_size`); $\epsilon = 0.8$ (`epsilon_patch`); $\alpha = 0.05$ (`patch_alpha`); Iterations $= 40$ (`patch_iterations`); Restarts $= 5$ (`patch_num_restarts`).

Results

This section presents the detailed outcomes of evaluating adversarial attacks on ResNet-34 and their transferability to DenseNet-121.

ResNet-34 Performance: Baseline and Under Attack

Table 1: ResNet-34 Performance Under Various Attacks & Perturbation Details

Attack Type	ϵ	Top-1 Acc. (%)	Top-5 Acc. (%)	Time (s)
Baseline (Clean Data)	-	77.40	93.00	-
FGSM (Task 2)	0.02	3.40	20.80	3.92
PGD (Task 3)	0.02	0.00	24.60	49.90
MI-FGSM (Task 3)	0.02	0.20	28.60	51.80
Patch Attack (Task 4)	0.8 (patch)	12.00	49.40	925.44

The initial evaluation established the baseline performance of the pre-trained ResNet-34 model on the clean, unperturbed ImageNet subset. ResNet-34 achieved a **Top-1 accuracy of 77.40%** and a **Top-5 accuracy of 93.00%**. These figures represent the model's standard classification capability before any adversarial intervention.

Table 1 summarizes the significant impact of the implemented adversarial attacks on ResNet-34's accuracy. The **FGSM** attack, with a perturbation budget $\epsilon = 0.02$, drastically reduced the Top-1 accuracy from 77.40% to a mere **3.40%**, and Top-5 accuracy from 93.00% to **20.80%**. This substantial drop, achieved with a computationally inexpensive single-step attack, underscores the model's inherent vulnerability to simple gradient-based perturbations. The iterative attacks proved even more devastating. The **PGD** attack, using the same $\epsilon = 0.02$ but applied over 10 iterations, effectively crippled the ResNet-34 model. Its Top-1 accuracy plummeted to an absolute **0.00%**, meaning not a single image from the test set was correctly classified. The Top-5 accuracy also fell sharply to **24.60%**. This demonstrates the power of iterative refinement in finding highly effective adversarial examples. Similarly, **MI-FGSM**, also an iterative method incorporating momentum, achieved a near-total collapse of performance. It resulted in a Top-1 accuracy of only **0.20%** and a Top-5 accuracy of **28.60%**. The momentum term likely helps in navigating the loss landscape to find more potent perturbations.

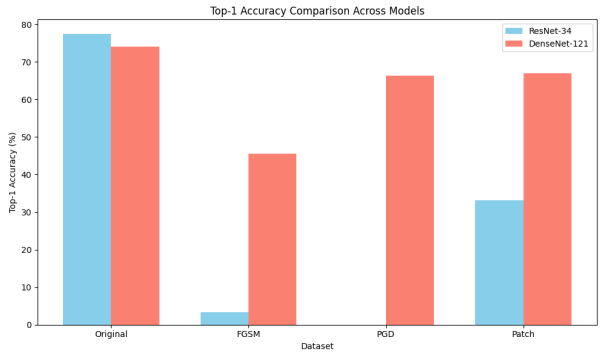

Figure 1: Comparative Top-1 Accuracy for ResNet-34 (source) and DenseNet-121 (transfer) across Original, FGSM, PGD, and Patch adversarial datasets.

The **Patch Attack**, distinct in its approach by modifying only a 32x32 pixel region (albeit with a larger local perturbation budget $\epsilon = 0.8$ for the patch itself), still managed a substantial decline. Top-1 accuracy fell to **12.00%** and Top-5 to **49.40%**. While less effective than PGD or MI-FGSM in this white-box setting, its ability to degrade performance by over 65 percentage points in Top-1 accuracy by altering less than 2.3% of the image pixels (for a 224x224 image) is notable. This attack was, however, the most computationally intensive due to its optimization process involving multiple restarts.

DenseNet-121 Performance: Transferability of Attacks

A crucial aspect of adversarial attacks is their transferability—the ability of adversarial examples crafted for one model to fool another, unseen model. This simulates a black-box attack scenario. The baseline performance of DenseNet-121 on the clean dataset was **74.00% Top-1** and **92.60% Top-5 accuracy**, comparable to ResNet-34.

Table 2: DenseNet-121 Performance on Adversarial Datasets Generated for ResNet-34

Adversarial Set Origin (Attack Type)	Top-1 Acc. (%)	Top-5 Acc. (%)
Original Clean Data	74.00	92.60
FGSM Adversarial (from ResNet-34)	45.60	75.20
PGD Adversarial (from ResNet-34)	66.40	91.20
Patch Adversarial (from ResNet-34)	67.00	92.00

Table 2 and Figures 1, 2, and 3 present the transferability results. Adversarial examples generated by **FGSM** against ResNet-34 showed considerable transferability to DenseNet-121. DenseNet-121's Top-1 accuracy on these examples dropped from 74.00% to **45.60%** (a decrease of 28.4 percentage points), and Top-5 accuracy fell to **75.20%**. This indicates that the relatively simple perturbations found by FGSM generalize to some extent across different architectures. In contrast, **PGD** attacks, despite their extreme effectiveness against the source ResNet-34 model (0.00% Top-1 accuracy), transferred less potently. DenseNet-121 maintained a Top-1 accuracy of **66.40%** and Top-5 accuracy of **91.20%** on PGD-perturbed images. This is a significant recovery compared to ResNet-34's performance and suggests that PGD might overfit to the specific gradient landscape of the source model. The **Patch attacks** also demonstrated noteworthy transferability. DenseNet-121's Top-1 accuracy decreased to **67.00%** (from 74.00%) and Top-5 to **92.00%**. While this is a smaller drop than FGSM, the fact that a localized patch can maintain some level of foolery on a different architecture is significant, especially considering its potential for physical deployment.

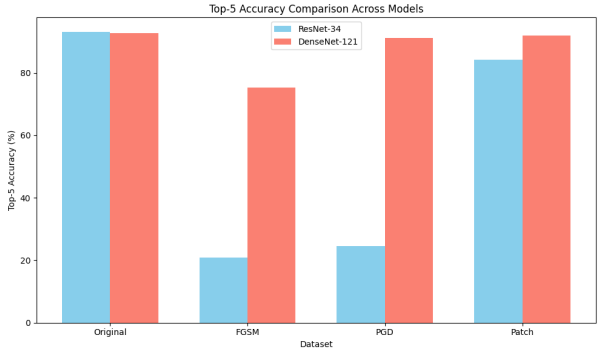

Figure 2: Comparative Top-5 Accuracy for ResNet-34 (source) and DenseNet-121 (transfer) across Original, FGSM, PGD, and Patch adversarial datasets.

As depicted in Figure 1 (Top-1 Accuracy) and Figure 2 (Top-5 Accuracy), while all transferred attacks successfully reduced DenseNet-121's accuracy compared to its baseline, DenseNet-121 generally exhibited greater resilience to these transferred attacks than ResNet-34 did to the direct attacks. Figure 3 (Transferability Rates) more explicitly shows that FGSM achieved a higher transfer rate for Top-1 accuracy degradation. This implies that simpler, less model-specific adversarial patterns generated by FGSM might generalize

better, whereas highly optimized attacks like PGD may tailor perturbations too closely to the source model, limiting their black-box effectiveness. The Patch attack's transferability suggests that highly salient, localized adversarial features can be somewhat model-agnostic.

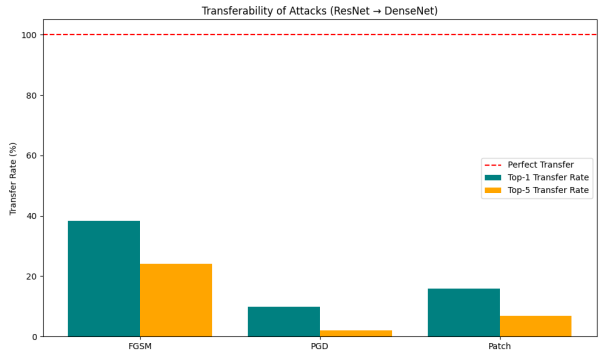

Figure 3: Transferability Rates of FGSM, PGD, and Patch attacks from ResNet-34 to DenseNet-121, illustrating the percentage of accuracy degradation transferred.

Discussion and Lessons Learned

This project offered insights into adversarial attack characteristics and model vulnerabilities, highlighting trade-offs between potency, cost, and transferability.

Attack Potency: Iterative attacks (PGD, MI-FGSM) were highly effective on ResNet-34. PGD reduced Top-1 to zero, showing its white-box power. MI-FGSM performed similarly, confirming momentum's benefit. FGSM, though less potent, caused significant accuracy drops with low computational cost.

L_∞ Perturbations: The L_∞ constraint ($\epsilon = 0.02$) for FGSM, PGD, MI-FGSM meant small, often imperceptible, pixel changes systematically fooled the model, showing sensitivity to high-frequency noise not aligned with human semantic features.

Patch Attacks: Modifying a small 32x32 region with a larger budget ($\epsilon = 0.8$) simulated a plausible physical threat. Despite spatial constraints, it was effective, suggesting vulnerability to localized, high-intensity "distractions." Higher computational cost reflected its complex optimization.

Attack Transferability: FGSM attacks, less powerful on the source, showed better relative transferability to DenseNet-121. PGD, highly optimized for ResNet-34, transferred less effectively, illustrating the "transferability-versus-power" trade-off. Patch attacks showed moderate, consistent transferability.

Computational Costs: FGSM was fast (3.92s). Iterative PGD (49.90s) and MI-FGSM (51.80s) were slower. Patch attacks were most intensive (925.44s).

Model Robustness Variations: DenseNet-121 showed greater robustness to transferred attacks than ResNet-34 to direct attacks, possibly due to architectural differences (dense vs. residual connections) leading to different feature representations.

Limitations & Future Work: Evaluate on full ImageNet and other domains. Explore targeted attacks more systematically. Investigate defenses like adversarial training. Analyze factors governing transferability.

Conclusion

This project systematically evaluated FGSM, PGD, MI-FGSM, and Patch attacks on ResNet-34, assessing transferability to DenseNet-121. Findings confirm deep neural networks' vulnerability. Iterative PGD and MI-FGSM showcased high white-box potency. Patch attacks, though localized, were formidable and transferable, indicating potential in physical scenarios.

Variations in effectiveness and transferability highlight adversarial robustness's complexity. Stronger source model attacks don't always transfer well due to overfitting. Simpler or structurally different attacks (e.g., patches) can sometimes pose a greater black-box threat. Results stress the critical and ongoing need for comprehensive defense strategies and rigorous robustness evaluation. Understanding model failures under diverse adversarial pressures is key to engineering safe, secure AI systems.

References

[1] Goodfellow, I. J., Shlens, J., and Szegedy, C. (2014). Explaining and Harnessing Adversarial Examples. *arXiv preprint arXiv:1412.6572*.

[2] Madry, A., Makelov, A., Schmidt, L., Tsipras, D., and Vladu, A. (2017). Towards Deep Learning Models Resistant to Adversarial Attacks. *arXiv preprint arXiv:1706.06083*.

[3] Dong, Y., Liao, F., Pang, T., Su, H., Zhu, J., Hu, X., and Li, J. (2018). Boosting Adversarial Attacks with Momentum. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

[4] He, K., Zhang, X., Ren, S., and Sun, J. (2016). Deep Residual Learning for Image Recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

[5] Huang, G., Liu, Z., Van Der Maaten, L., and Weinberger, K. Q. (2017). Densely Connected Convolutional Networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

[6] Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., and Fei-Fei, L. (2009). ImageNet: A large-scale hierarchical image database. In *Conference on Computer Vision and Pattern Recognition (CVPR)*.

[7] Anthropic. (2025). Claude 3.7 Sonnet. Technical collaborator for manuscript preparation and code development.