Robust Classifier Against FGSM & PGD Attacks

Team: 15
Github Link: https://github.com/javedakhtar0129/TML25_A3_15

Problem Statement

This assignment focuses on training a robust image classifier that performs well on both clean and adversarial image inputs. Specifically, the classifier is trained to resist attacks created using the Fast Gradient Sign Method (FGSM) and Projected Gradient Descent (PGD). These methods introduce noise in the input images that is not perceptible to the human eye which results in the model making a wrong prediction.

The main challenge is to build a strong model that has very high clean accuracy while performing well in detecting adversarial examples.

Constrains

- The allowed models for this assignment are **ResNet18, ResNet34, or ResNet50** (from torchvision.models).
- The input shape is **(3×32×32)**, and output shape is **(10,)**
- We only have a small perturbed distribution of the original dataset to train our model.
- The model clean accuracy must be above 50% on the private samples to be accepted by the server.
- After uploading on the server, our model will be evaluated against only 30% of the data.

File Structure

- vanilla.ipynb: Main notebook with all code for data loading, training, evaluation, and submission.
- Train.pt: Provided Subset of the model's training data.
- final_high_acc_model.pt: Trained robust classifier.
- README.md: Instructions for usage.
- adversarial_training_progress.png: Training and validation progress plot.

Approach Used

Our approach implements adversarial training to improve model robustness against common attack methods (FGSM and PGD) while maintaining acceptable clean accuracy. We used a carefully balanced training regime that combines clean examples with adversarially perturbed ones to create a model that performs well across different evaluation metrics.

Why use jittering in augmentation?
We observed adversarial perturbations with high-contrast pixels in the dataset. To mitigate this, we included color jittering in the augmentation pipeline. This helped the model become less sensitive to unnatural contrast shifts, effectively regularizing it against adversarial noise patterns during training.

Why set $\varepsilon = 8/255$ and $\alpha = 1/255$?

- The perturbation limit $\varepsilon= 8/255$ defines the maximum allowed distortion. It's a standard bound for adversarial robustness on natural images.
- The step size $\alpha= 1/255$ to avoid overshooting the boundary in PGD and for much fine-grained gradient updates.

Why use a dynamic PGD schedule (PGD_STEPS_TRAIN = 3 → PGD_STEPS_FINAL = 5)?

- Early training uses **fewer PGD steps (3)** for efficiency.
- Gradually increasing to **5 steps** prevents over-regularizing before the model has learned basic structure.

Why use adversarial ratio ADV_TRAIN_RATIO = 0.4?

- 40% of the training loss comes from adversarial samples.
- To ensure clean performance is not sacrificed entirely while still guiding the model toward robustness.

In early epochs, lighter attacks (PGD with 3 steps) and lower adversarial weight (0.4) are used. This helps the model learn the clean data first and then adjust gradually to adversarial perturbations. Clean batches are introduced every 6th iteration to ensure that clean accuracy remains acceptable.

As training progresses, PGD step count is increased to 5, and adversarial batches dominate the learning signal. The model should now generalize across both perturbed and unperturbed samples.

Model
The architecture follows the standard ResNet34 design with modifications:
Input (3×32×32) →
ResNet34 (torchvision.models) →
[Conv2D layers with BatchNorm + ReLU] × N →
GlobalAvgPool →
FC (512) →
FC (10, softmax output)

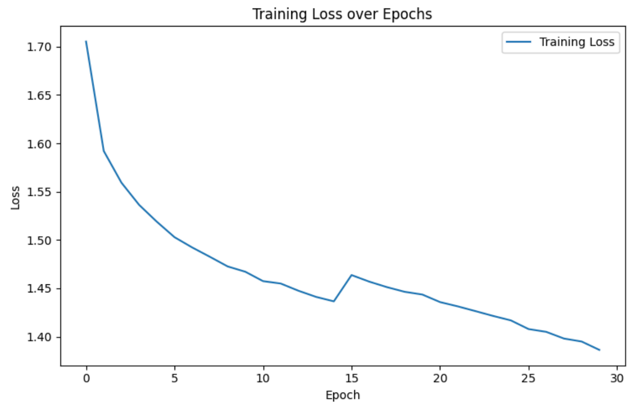
Training Approach

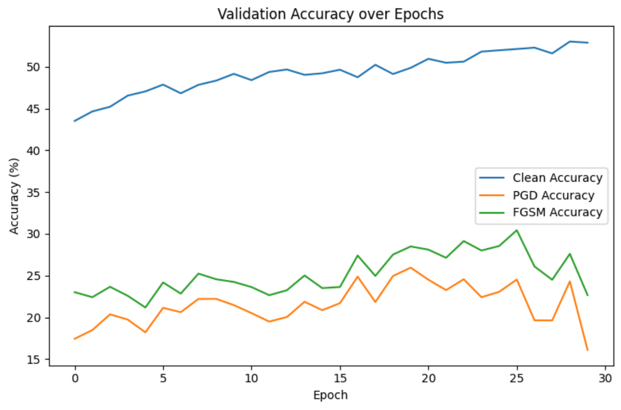
- **Loss Function**: CrossEntropyLoss — balances multi-class classification with softmax.
- **Optimizer**: SGD with learning_rate=0.01, momentum=0.9
- **Weight Decay**: 5e-5 → Regularization via L2 penalty.
- **Scheduler**: ReduceLROnPlateau to halve learning rate if validation accuracy plateaus for clean accuracy.

Training runs for **30 epochs**, with **early stopping after 7 epochs** of no improvement in weighted accuracy.

Validation and Evaluation Models were evaluated on three metrics:
Clean Accuracy: Performance on unmodified examples
FGSM Accuracy: Robustness against single-step attacks
PGD Accuracy: Robustness against more sophisticated multi-step attacks

Result
Our best model achieved 58.7% clean accuracy, 14.63% FGSM accuracy, and 0.27% PGD accuracy, showing the typical trade-off pattern where robustness against stronger attacks is more difficult to achieve.





Limitations and other observations

- There is a robustness accuracy tradeoff while training the model for adversarial examples. Due to these consistent increases in robustness the accuracy for clean images decreases.
- Adversarial training shows more variance in performance across epochs compared to standard training.
- Models consistently performed better against FGSM than PGD, confirming that PGD is a stronger attack.