TML Assignment – 3

Team number – 39

Team token - 09596680

Name1: Soham Ritesh Tolwala

Matriculation number: 7076028

Email id1: soto00002@stud.uni-saarland.de

Name2: Laxmiraman Dnyaneshwar Gudewar

Matriculation number: 7076312

Email id2: lagu00003@stud.uni-saarland.de

Introduction:

The goal of Assignment 3 was to develop a robust image classifier capable of defending against adversarial attacks. Specifically, we focused on the CIFAR-10 dataset and evaluated model robustness under two prominent threat models: Fast Gradient Sign Method (FGSM) and Projected Gradient Descent (PGD). The task demanded not just high clean accuracy but also resilience against crafted adversarial perturbations.

To meet this objective, we designed a training pipeline incorporating adversarial training techniques, progressive attack intensification, and careful data preprocessing. The assignment tested our understanding of model generalization under adversarial stress and emphasized the importance of seemingly subtle but critical implementation details.

Setup:

FILENAME / PATH	PURPOSE
`colab_training.ipynb`	Main notebook used to experiment with and train the final PGD-based model.

`FREE.ipynb`	Early attempt to implement FREE adversarial training. Dropped due to poor clean accuracy.
`PGD.ipynb`	Initial base implementation of PGD attack logic. Later ported to Colab.
`example_assignment_3.py`	Official submission script with assertions. Used for submitting final model.
`try_script.py`	Minimal version of submission script, used for raw state_dict submission bypassing assertion checks.
`models/resnet_wrapper.py`	Custom ResNet18 wrapper used in early training setups. Later replaced with vanilla ResNet.
`data/Train.pt`	Provided training dataset in serialized format.
`saved_models/`	Directory containing all trained .pt models (both successful and rejected).
`sanity_check.py`	Script used to verify model compatibility with evaluation server (input range, accuracy, etc.).

Approaches tried:

1. Baseline PGD Adversarial Training

- Approach: Trained a ResNet18 model from scratch using PGD-generated adversarial examples.
- Key Details:
    - PGD attack with gradually increasing steps (1 → 7) across epochs.
    - 100% adversarial training at the start.
- Outcome: Very slow convergence. Clean accuracy plateaued below 40%, indicating over-regularization and loss of generalization.

2. PGD + Clean Mix (70:30)

- Approach: Mixed clean and PGD adversarial samples in 70:30 ratio during training.
- Rationale: Prevent excessive degradation of clean accuracy while retaining robustness.

- Outcome: Clean accuracy improved significantly (~78%) while maintaining decent PGD robustness (~35%).

3. Gradual PGD Ramp-up

- Approach: Increased number of PGD iterations over epochs, starting with 1 and going up to 7.
- Rationale: Smooth regularization allows the model to learn robust features without being overwhelmed in early stages.
- Outcome: Helped convergence. Performance steadily improved across epochs.

4. FREE Adversarial Training (Attempted)

- Approach: Tried implementing FREE adversarial training to reduce per-batch compute.
- Outcome: Implementation was functional, but final model accuracy was too low to meet submission thresholds. Approach was dropped.

5. Custom ResNet Wrapper (Initial Phase)

- Approach: Wrapped torchvision.models.resnet18 in a custom class for easier experimentation.
- Issue: Submission server required vanilla ResNet18 architecture; wrapper-based models were rejected due to incompatible keys in state_dict.

6. Submission Compatibility Adjustments

- Final model was trained with vanilla torchvision.models.resnet18.
- Correct normalization (to_tensor() instead of ToTensor()) was applied in DataWrapper to ensure input consistency with server expectations.

7. 80:20 Clean:Adv Composition with Adam + MultiStepLR

- Approach: Adjusted training to 80% clean and 20% PGD adversarial data. Optimizer: Adam; Scheduler: MultiStepLR.
- Outcome: Clean accuracy on local validation seemed decent, but server submission failed due to low clean accuracy. Sanity check revealed poor generalization.

8. PGD + FGSM Joint Training (CosineAnnealing + SGD)

- Approach: Combined PGD and FGSM adversarial examples during training. Optimizer changed to SGD, with a CosineAnnealing scheduler. Randomized smoothing with light Gaussian noise ($\sigma = 0.05$) was added.

- Outcome: Very poor adversarial robustness. Final results — Clean: **61.87%**, FGSM: **22.76%**, PGD: **5.63%**. Dropped due to clear overfitting to clean and failure against attacks.

9. 75:25 Clean:Adv Composition

- Approach: Final attempt using a 75% clean and 25% PGD mixture. Hyperparameters: PGD(ε = 8/255, α = 2/255, K = 7), optimizer: Adam, scheduler: MultiStepLR.
- Outcome: Failed again with "clean accuracy too low" error upon submission. Locally observed ~55% clean accuracy. The change in composition did not significantly boost robustness.

Challenges:

Challenge	Description	Resolution
Model Clean Accuracy Too Low (Submission Error)	Even after reaching ~80% clean accuracy locally, the server threw an error rejecting the model.	Realized the server feeds images in [0, 255] format. Used sanity_check.py to confirm and added a x/255 normalization layer.
Wrong Normalization Layer	Used transforms.ToTensor() inside a wrapper class, which led to subtle inconsistencies.	Switched to transforms.functional.to_tensor() based on a peer's suggestion. This fixed the mismatch with the server input.
Wrapper Model Incompatibility	Using a custom ResNetWrapper broke submission due to mismatched state_dict keys during loading.	Re-trained using plain torchvision.models.resnet18 and replaced wrapper with standard model before saving.
FREE Adversarial Training Fails	Tried implementing FREE adversarial training for faster learning but the model failed to converge.	Dropped FREE. Focused on PGD-based adversarial training with clean+adv mixture (70:30) instead.

Results:

These values are from the best model submitted using 70:30 Clean:Adv training with gradual PGD ramp-up, Adam optimizer, and MultiStepLR scheduler. Other attempted strategies (like 75:25, joint PGD+FGSM, etc.) failed to surpass this.

Final Submission Results (Leaderboard)

Metric	Accuracy
Clean Accuracy	52.43%
FGSM Accuracy	38.27%
PGD Accuracy	35.83%

Conclusion:

This assignment offered practical exposure to adversarial robustness techniques in image classification. Despite several technical hurdles—including model incompatibility, incorrect normalization, and submission rejections—we successfully developed a PGD-trained ResNet-18 model that achieved competitive performance across clean and adversarial benchmarks. A 70:30 clean-to-adversarial training ratio, along with correct preprocessing and gradual PGD ramp-up, proved effective in optimizing both robustness and accuracy. Alternative strategies, such as randomized smoothing and joint FGSM-PGD training, were explored but did not yield significant improvements.