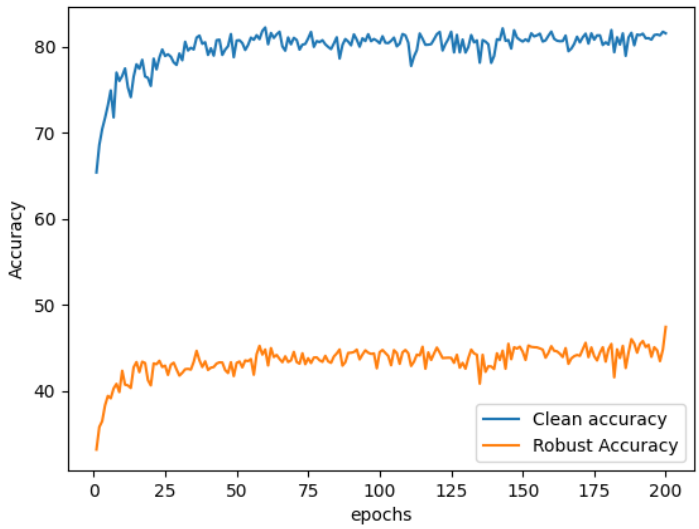
CS 291A - HW2 Report
Umang Garg, 6787683

PART- 1

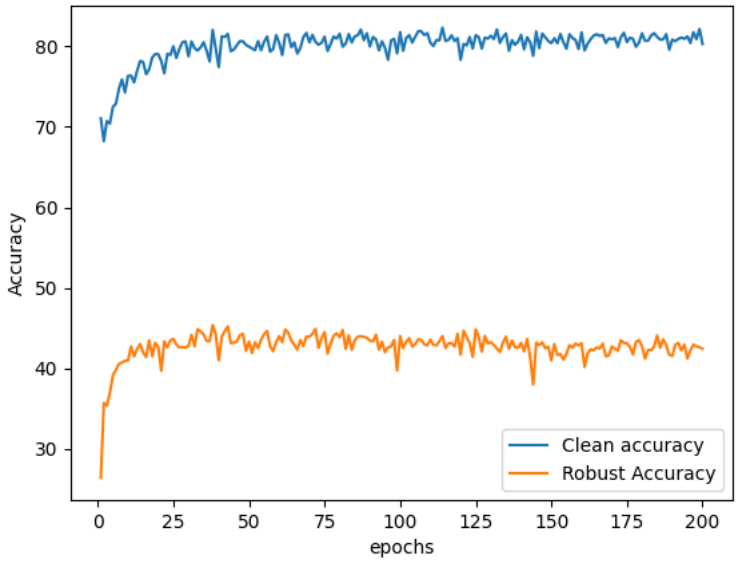
(Note: accuracy curves were reported on the final batch of the validation set. Optimizer: SGD with a learning rate of 0.05 and momentum of 0.9 was used for experiments)

Q-1: Adversarial Training validation set accuracies (clean and robust) with every epoch. Training hyperparameters: 10-step $linf$ PGD attack with eps of 8/255 to configure the attack. attack learning rate of PGD was set to 2/255.



Clean Accuracy: 81%
Robust Accuracy: 47%

Q-2. Fast Adversarial Training. 1-step PGD attack training with uniform random delta initialization with the epsilon bound, and alpha being updated to 1.25 of the standard case.



Clean Accuracy: 80%, Robust Accuracy (PGD-10 step attack used): 40%

Q-3.

Adversarial Training Results

50-step with 2/255 attack step size, 8/255-tolerant untargeted PGD attack with CE loss:

(a) Clean Test Accuracy % = 76.7699966430664
(b) Robust Test Accuracy % = 25.59000015258789

50-step with 2/255 attack step size, 8/255-tolerant untargeted PGD attack with CW loss:

(a) Clean Accuracy: 76.7699966430664%
(b) Robust Accuracy: 49.63%

Fast Adverdarial Training Results

50-step with 2/255 attack step size, 8/255-tolerant untargeted PGD attack with CE loss:

(a) Clean Test Accuracy % = 74.37000274658203
(b) Robust Test Accuracy % = 23.959999084472656

50-step with 2/255 attack step size, 8/255-tolerant untargeted PGD attack with CW loss:

(a) Clean Test Accuracy % = 74.37000274658203
(b) Robust Test Accuracy % = 45.560001373291016

Q-4: **Pre-trained model Robustness metrics** on the previously mentioned metrics are below :

50-step with 2/255 attack step size, 8/255-tolerant untargeted PGD attack with CE loss:

(a) Clean Test Accuracy % = 82.08999633789062
(b) Robust Test Accuracy % = 52.06999969482422

50-step with 2/255 attack step size, 8/255-tolerant untargeted PGD attack with CW loss:

(a) Clean Accuracy: 82%
(b) Robust Accuracy: 68%

PART-2

Training methodology: semisupervised learning with TRADES algorithm from the paper: link

300 epochs - Training Clean accuracy: 80.93% and Robust accuracy: 52.11%

This paper was chosen as it is hallmarked as the 2019 Neurips Adversarial Challenge winner with similarly reported metrics as above.

Additional data used: this pickle file containing unlabeled data was used from this link to provide the model with additional information to figure out the semantics of the initially labeled data. This significantly boosted the robust accuracy, as proposed in the paper.

Hyperparameters used: eps: 8, alpha: 2, beta: 6, epochs: 300.
Optimizer used: Stochastic gradient descent.
Learning schedule: 0<epochs< 200: lr = 0.05, momentum = 0.9
200 < epochs < 300: lr = 0.01, momentum = 0.5

How TRADES algorithm work?

At its core, the algorithm still minimizes the natural loss, consequently increasing the accuracy of the model, but at the same time, it introduces a regularization term, which induces a shift of the decision boundaries away from the training data points.

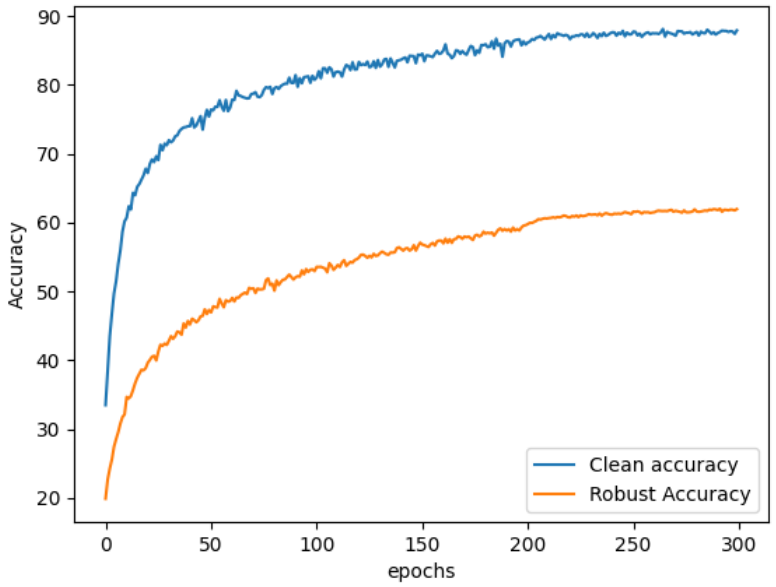
TRADES minimizes a regularized surrogate loss L(.,.) (e.g., the cross-entropy loss) for adversarial training:

$$\min_f \mathbb{E}\left\{\mathcal{L}(f(X),Y) + \beta \max_{X' \in \mathbb{B}(X,\epsilon)} \mathcal{L}(f(X), f(X'))\right\}.$$

The first term encourages the natural error to be optimized by minimizing the "difference" between f(X) and Y, while the second regularization term encourages the output to be smooth, that is, it pushes the decision boundary of the classifier away from the sample instances via minimizing the "difference" (K-L loss is used this case) between the prediction of natural example f(X) and that of adversarial example f(X′). The tuning parameter β plays a critical role in balancing the importance of natural and robust errors.

More data improves accuracy, and also robustness. Cheap data without labels can be found and used in a self-training semi-supervised fashion to obtain pseudo-labels. This in turn is added to the initial CIFAR-10 dataset and used for training the new classifier which would be more robust.
These two techniques were combined and used for custom adversarial training.

TRAINING CURVE



Q-1. 50-step with 2/255 attack step size, 8/255-tolerant untargeted PGD attack with CE loss:

(a) Clean Test Accuracy % = 79.94000244140625

(b) Robust Test Accuracy % = 51.380001068115234

50-step with 2/255 attack step size, 8/255-tolerant untargeted PGD attack with CW loss:

(a) Clean Test Accuracy % = 79.94000244140625
(b) Robust Test Accuracy % = 60.81999969482422

Q-2: Autoattack results on the custom model:
initial accuracy: 80.93%
robust accuracy after APGD-CE: 52.64% (total time 254.3 s)
robust accuracy after APGD-T: 48.86% (total time 2199.2 s)
robust accuracy after FAB-T: 48.86% (total time 7455.9 s)
robust accuracy after SQUARE: 48.86% (total time 13924.9 s)
max Linf perturbation: 0.03137, nan in tensor: 0, max: 1.00000, min: 0.00000
robust accuracy: 48.86%

Potential Improvements:

1. A better learning schedule can be designed for the same learning mechanism to instill better learning of the data.
2. Triplet loss can be used ad a regularizer term to further improve the learning. Triplet loss tries to make training examples of the same class come closer while pushing the examples of different classes farther. Hence the decision boundaries become more pronounced and would improve robustness.

Provided files:

eval_Autoattack.py: Corresponds to standard Adversarial training
eval_Autoattack_Fast_AT.py: Corresponds to Fast Adversarial training
Advanced_AT.py: Corresponds to custom training method used in q2
Gdrive link to trained models and other files for reference

Referred Papers and code:

1. Opportunities and Challenges in Deep Learning Adversarial Robustness: A Survey [link]
2. TRADES [link]
3. Unlabeled data improves adversarial robustness [link] [repo]