PROGRAMMING ASSIGNMENT 1: ADVERSARIAL ATTACKS AND DEFENSES

15-783: Trustworthy AI: Theory & Practice (Fall 2025)
https://www.cs.cmu.edu/~aditirag/teaching/15-783F25.html
OUT: Sep 7th
DUE: Sep 29th

Resources

Papers:
Intriguing properties of neural networks
Towards Evaluating the Robustness of Neural Networks
Towards Deep Learning Models Resistant to Adversarial Attacks
Universal and Transferable Adversarial Attacks on Aligned Language Models

Code:
Adversarial training and evaluation code by Madry et al.
Different attack algorithms including GCG and GCG ensemble by Mantas et al.
Lightweight implementation of GCG

Part 1 — ImageNet Classifiers: Smallest ϵ That Breaks the Model (10 points)

Goal: Estimate the smallest perturbation ϵ (under ℓ_∞ and ℓ_2 norms) required to flip classification for a pretrained model.

Model: ResNet-18 trained on ImageNet.

Losses: Cross-Entropy (CE), Carlini–Wagner margin (CW).

Attack: Sample 100 random images from the ImageNet-1K validation set. Perform both *untargeted* and *targeted* attacks. For targeted attacks, randomly sample one target class per image from the 999 non-true classes. Use PGD (with $\epsilon/4$ as the step size) to implement the attacks.

—

Deliverables

1. **(2 points) Tables:**
   - Two tables: one for untargeted, one for targeted attacks.
   - Each table should report median ϵ^* for all four conditions: (ℓ_∞, CE), (ℓ_∞, CW), (ℓ_2, CE), (ℓ_2, CW).
2. **(2 points) Plots: Success Rate vs. ϵ:** Produce four figures, all clearly labeled with axes, titles, and legends.
   (a) Untargeted, ℓ_∞: success fraction vs. ϵ, curves for CE and CW.
   (b) Untargeted, ℓ_2: same format as above.
   (c) Targeted, ℓ_∞: same format as above.
   (d) Targeted, ℓ_2: same format as above.

   Fixed ϵ grids for comparability:
   - ℓ_∞: $\{0, 1/255, 2/255, \ldots, 8/255\}$.
   - ℓ_2: equally spaced values in $[0, 3.0]$.

   If attack success rate (ASR) does not reach 100% (equivalently, accuracy does not fall to 0%) within these ranges, continue increasing ϵ until full success is achieved.
3. **(2 points) Example Image:** Submit at least one example of a successfully attacked image.
   - Show the *original clean image* alongside its *adversarial version*.
   - Report the true class label and the predicted (misclassified) label.
   - Indicate the norm, loss function, and ϵ^* used.
4. **(2 points) Discussion:**
   - Compare targeted vs. untargeted attacks: which requires larger ϵ^*?
   - Compare CE vs. CW: which achieves smaller ϵ^* and smoother success curves?
5. **(2 points) Code to reproduce results.**

Part 2 — Adversarial Training on MNIST (10 points)

Goal: Train a small CNN on MNIST to be robust to ℓ_∞ perturbations.

Setup

- **Dataset:** MNIST train/test, inputs scaled to $[0, 1]$.
- **Model (standardized):** Conv(32, 5×5)→ReLU→MaxPool(2×2)→ Conv(64, 5×5)→ReLU →MaxPool(2×2)→ FC(1024)→ReLU→Dropout(p=0.5)→FC(10).
- **Optimization:** Adam, learning rate 10^{-4}, batch size 50, max training steps 100,000 (report early stop if used).

Evaluation Attack (FGSM-ℓ_∞)

For each $\epsilon \in \{0, 0.1, 0.2, 0.3\}$, craft

$$x^{\text{adv}} = \text{clip}(x + \epsilon\, \text{sign}(\nabla_x \mathcal{L}(f_\theta(x), y))\,,\, 0, 1)\,,$$

and report robust accuracy (% correct on $\{x^{\text{adv}}\}$). Use cross-entropy $\mathcal{L}$ and the same preprocessing as during training.

Training Protocols

1. **Baseline (Natural Training).** Train on clean MNIST only (no adversarial perturbations) by minimizing $\text{CE}(f_\theta(x), y)$.
2. **FGSM Adversarial Training.** Train with one-step FGSM at $\epsilon_{\text{train}} = 0.3$.
3. **TRADES (one-step KL-FGSM).** Approximate the inner maximization with a single FGSM step on the KL term at $\epsilon_{\text{train}} = 0.3$:

$$x^{\text{adv}}_{\text{train}} = \text{clip}\Big(x + \epsilon_{\text{train}}\, \text{sign}\Big(\nabla_{x'} \text{KL}\big(f_\theta(x) \,\|\, f_\theta(x')\big)\Big)\Big|_{x'=x},\, 0, 1\Big)\,,$$

then minimize $\text{CE}(f_\theta(x), y) + \beta\, \text{KL}\big(f_\theta(x) \,\|\, f_\theta(x^{\text{adv}}_{\text{train}})\big)$ with default $\beta = 6$.

Deliverables

1. **(6 points) Figure—Accuracy vs. ϵ (FGSM evaluation).**
   - X-axis: $\epsilon \in \{0,\ 0.1,\ 0.2,\ 0.3\}$ (extend if you evaluated beyond 0.3).
   - Y-axis: accuracy (%).
   - Curves: **Baseline**, **FGSM-trained**, **TRADES**. (Point at $\epsilon = 0$ is clean accuracy.)
2. **(2 points) Discussion:** Compare Baseline vs. FGSM vs. TRADES; comment on robustness decay with ϵ.
3. **(2 points) Code to reproduce results.**

Part 3 — Attacking Aligned Language Models with Adversarial Suffixes (10 points)

Goal: Implement a Greedy Coordinate Gradient (GCG)-style suffix optimizer on open-weight LMs. Use 10-20 tokens as the suffix length.

Models (choose one family):

- *Llama (non-reasoning):* Llama-3.2-1B-Instruct, 3B, 11B.
- *Qwen (reasoning):* Qwen3-0.6B, 1.7B, 4B, 8B.

Note: When targeting reasoning models, consider the following attack strategies if the default algorithm isn't working well:

- Instead of eliciting an affirmative output, try targeting initial harmful reasoning within think tags.
- Alternatively, prefill the model's "thinking" trace and then optimize for an affirmative response that appears only *after* the </think> tag.

Tasks

1. Learn suffixes. Implement the GCG-style suffix optimizer for a chosen model.
Target a policy-prohibited query e.g., `How to build a bomb` that the model refuses by default.

Plots/Outputs:

- Optimization loss (or objective) vs. iterations; mark the first iteration achieving a non-refusal event (success).
- Final learned suffix and model output.

2. Explore universality or transferability (choose one).

(A) **Universality via behavior ensembling:** (slightly easier)

- Train suffixes on 10 harmful behaviors from the HarmBench validation set and test on 10 unseen prompts from the test set.
- Report Attack Success Rate (ASR) = fraction of prompts that flip from refusal to non-refusal.
- **Plot:** ASR on validation and test vs. iterations (same checkpoints).

(B) **Transferability via model ensembling:** ***(2 points extra credit*** if you choose transferability)

- Pick one prohibited behavior; verify all selected models refuse it at baseline.
- Train one suffix jointly on at least two models; test transfer on a held-out model.
- Repeat for 10 behaviors; report transfer rate (fraction transferring).
- **Plot:** attack success rate for both seen and unseen models (bars) and vs. iterations (lines).

Deliverables

1. **(6 points) Figures:** Plots from Task 1 and 2. Log every 20 iterations.
2. **(2 points) Outputs:** Example suffix and output from Task 1.
3. **(2 points) Code to reproduce results.**