CIS 350: AI for Cybersecurity

Week 10 Activity: Craft Adversarial Examples to Fool a Classifier

CTFd Capture-the-Flag Challenge (Hard)

Activity Overview

In this capture-the-flag challenge you will act as an authorized red-team tester attacking a handwritten-digit classifier. A bank uses this model to read the amount on scanned checks, and its security team added a rule: flag any prediction the model is less than 90% confident about. Your mission is to add a tiny, near-invisible perturbation that makes the model read the WRONG digit while staying above the 90% confidence bar, so it sails past the alarm. You will do all of this in Google Colab and submit the captured flag through the CTFd course server.

Ethics reminder: This is a sandboxed, authorized exercise. The model, data, and 'bank' are simulated for teaching. Only ever probe systems you have explicit permission to test.

Key Idea in One Line

An adversarial example is an input with a tiny added change (controlled by a dial called epsilon, or eps) that flips a model's prediction. Small eps is invisible but may not fool the model; large eps fools it reliably but becomes visible. High confidence does NOT mean the answer is correct.

Warm-Up (5 min): Predict Before You Attack

Before you touch any code, write down your gut prediction, then compare with a partner (think-pair-share). You will check these guesses against your real results in Tasks 1-3.

P1: How much noise do you think it takes to fool the model - a little or a lot? Circle one: LITTLE / A LOT.
P2: If the model is 95% confident in its answer, how likely is that answer to be correct? Circle one: DEFINITELY / PROBABLY / NOT NECESSARILY.
P3: In one sentence, name one way the bank could try to catch a tampered image.

Keep these predictions. Most people guess wrong on P1 and P2 - that surprise is the whole point of this challenge.

Setup Instructions (Google Colab)

1. Open the CTFd course server: ctfd.io (link posted on Canvas).
2. Open the 'Adversarial Examples' challenge and download the starter notebook.
3. Open the notebook in Google Colab (File -> Open notebook -> Upload). No local install needed.
4. Run the top cells. They load a pre-trained digit model and one test image, and show its prediction.
5. Skim the NIST Adversarial ML guidance at csrc.nist.gov/publications for background.

Resource: CTFd course server (ctfd.io) plus NIST Adversarial ML guidance (csrc.nist.gov/publications)

Starter perturbation pattern you will reuse in every task:

```
adv_image = image + eps * sign(gradient)
prediction, confidence = model.predict(adv_image)
```

Task 1 (Warm-Up): Add Noise and Watch the Trade-Off

Goal: see the imperceptibility trade-off with your own eyes. Using the starter line above, generate an adversarial image at eps = 1.0, 2.0, 3.0, and 4.0. For each value record the predicted digit and the model's confidence, and note when the image first looks visibly changed to you.

Hint: loop over the eps values, call model.predict each time, and display the image. Do not try to fully solve the challenge yet - just observe.

Paste your code below:

Fill in your results table (write in your observed values):

eps	Predicted digit	Confidence	Visibly changed? (Y/N)
1.0			
2.0			
3.0			
4.0			

Short explanation: In one or two sentences, describe what happened to the prediction and the image quality as eps increased.

Task 2 (Core): Find the Smallest Flip

Goal: find the SMALLEST eps that flips the prediction to a wrong digit. Sweep eps in small steps (for example, 0.5 increments) starting from a small value, and stop at the first eps where the predicted digit no longer matches the original.

Lab 8 connection: In Lab 8 the flip typically happens around eps = 3.5. Search near there and confirm whether your model behaves the same way.

Hint: use a loop; break as soon as predicted_label != original_label, then report that eps.

Paste your code below:

Result: The smallest eps that flips the prediction is: _____. Original digit: _____ -> New (wrong) digit: _____.

Short explanation: How close was your smallest flipping eps to the Lab 8 value of about 3.5? Why might it differ slightly (for example, a different starting image or digit)?

Task 3 (Hard): Beat the Confidence Alarm and Capture the Flag

Goal: defeat the 'confidence > 0.9' detector. Starting from your Task 2 eps, keep increasing eps and watch the model's confidence in the WRONG label climb. Find an eps where the prediction is wrong AND the confidence is above 0.9 - meaning the detector would let it through. When you succeed, the notebook prints a flag.

Hint: raise eps past the Task 2 flip point in steps and log the wrong-label confidence each time. The flag unlocks once wrong-label confidence passes 0.9.

Paste your code below:

Result: eps used: _____ Wrong digit predicted: _____ Confidence: _____ (must be > 0.9).

Captured flag (paste exactly):

Success Criteria and Flag Format

You have succeeded when: (1) Task 1 shows a completed table, (2) Task 2 reports the smallest flipping eps (near 3.5), and (3) Task 3 produces a wrong prediction with confidence above 0.9 and a flag. The flag has the format **CIS350{...}**. Paste the exact flag into the CTFd challenge; if CTFd marks it correct, your exploit worked.

Reflection

Answer each question in 2-4 sentences. These are worth the Understanding points, so explain your reasoning.

R1: In your own words, WHY does adding a tiny perturbation flip the model's prediction? Use the idea of a decision boundary in your answer.

R2: Why does the 'flag if confidence > 0.9' detector FAIL against your Task 3 attack? What wrong assumption does that detector make?

R3 (the defense - required): If you were the bank's security engineer, what is at least ONE real defense against this attack (for example, adversarial training, input preprocessing, or ensembles)? Explain how it would help AND why it still would not be a complete guarantee (the arms race).

Grading Rubric (20 points)

Criterion	Description	Points
Completion	All three tasks attempted; Colab notebook submitted	8
Technical Execution	Exploit works: smallest flip found and flag captured	7
Understanding	Reflection explains WHY it works and a real DEFENSE	5

Total: 20 points

Scoring: Completion (8) + Technical Execution (7) + Understanding (5) = 20 points

How to Submit

1. Complete Tasks 1-3 in your Colab notebook.
2. Paste the captured flag into the CTFd 'Adversarial Examples' challenge to confirm it is correct.
3. Answer all three reflection questions in this worksheet.
4. Submit your notebook (.ipynb) and this worksheet PDF on Canvas by [DUE DATE].

Reminder: this is a sandboxed, authorized exercise. Questions? Post in the course forum or attend office hours.