CS771 Fall 2025

Homework Assignment 2

cliu768@wisc.edu

October 19, 2025

Acknowledgment of AI Assistance

This assignment was completed with the assistance of OpenAI's ChatGPT (GPT-5). The tool was used throughout the process to brainstorm ideas, clarify relevant concepts, explain code logic, and check grammar. All final decisions regarding content, analysis, and conclusions were made by the author.

3.3 Adversarial Samples

Implementation of PGD and the results (accuracy drop and adversarial samples).

For each step PGDAttack::perturb enables gradients on the current adversarial tensor, runs a forward pass to get logits, and on the first iteration computes and fixes the least-likely class as the target; it then forms a targeted loss toward that fixed target, backpropagates to obtain the gradient with respect to the adversarial input, and updates the adversarial example by subtracting the step size times the sign of the gradient (which moves the prediction toward the chosen target); after the update it projects the perturbation back into the allowed $[-\epsilon, +\epsilon]$ range around the original input using elementwise clamping, replaces the adversarial tensor with the projected (detached) result to avoid accumulating the computation graph, and disables gradients until the next iteration, so the loop repeats these steps for the specified number of PGD iterations while keeping all operations in the model's input (normalized) space and maintaining the perturbation constraint.

Type	Acc@1	Acc@5
without attack	43.920	73.090
with attack	0.260	5.120

Can you see the difference between the original images and the adversarial samples?

At a glance the adversarial samples look basically the same as the originals — you won't notice large, semantic changes like objects moved or colors

Figure 1: adv samples

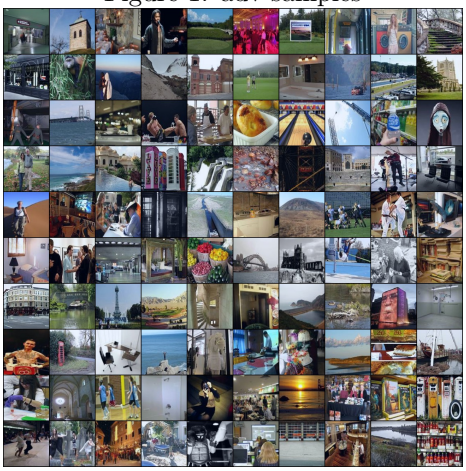

Figure 2: org samples

wildly shifted. If you look closely or amplify the difference, there are faint high-frequency perturbations: tiny speckled noise, slight local contrast/color shifts or edge shimmering. Those perturbations are often sub-pixel and spatially distributed so they're easy to miss in the collage but become visible if you subtract the images and scale the residual, or if you zoom very close.

What if you increase the number of iterations and reduce the error bound (ϵ)?

Increasing the number of PGD iterations while reducing the ℓ_∞ bound (ϵ) is a standard way to make attacks *stronger* but less visually obvious: many small, carefully directed steps let the attack find a more effective adversarial direction inside a tighter ball, so we often obtain higher attack success while keeping per-pixel change tiny.