Homework 3

CS 521: Fall 2025

Due on Friday, October 17, 2025 11:59 PM Central

Homework Policy: You are allowed to collaborate with your classmates, but report this in your submission. Even if you work on problems together, each student must write up their solution individually in their own words in their submission. This CS 473 course page on academic integrity is a handy reference: `https://courses.engr.illinois.edu/cs473/sp2023/integrity.html`. We have ZERO tolerance for breaches of academic integrity, such as, but not restricted to, plagiarism, use of LLMs except for when stated explicitly, etc.
You can take upto 3 days extra in total across the semester without explanation. Once you exhaust the three-day late submission allowance, any further late homework submissions will result in a 25% penalty per each extra day. 5 minutes late in submitting the homework will be counted as 1 day late. If you have exceptional personal circumstances that will prevent you from submitting the assignment on time, write to the course staff as soon as possible. Unless explicitly stated, use of LLMs for solving homework or cheating from others is not allowed – we will detect it! There will be an oral **exam** if we detect violations, during which you will be asked to answer a randomly selected question from the homework. If you are unable to show sufficient knowledge, then you will get zero for the full homework.
Please format your submission as a PDF (file naming convention: *CS521_⟨netid⟩_hw2.pdf*) and upload it on Gradescope by the deadline. Try to start every problem on a new page, to accurately map pages to questions on Gradescope.
Programming Problems: Read these guidelines carefully! Describe your solution at a high-level (if applicable). Write only small snippets of code, if at all. DO NOT paste tall walls of code into your submission! Upload your solution files to GitHub publicly and provide a link to the relevant files as part of your solution.

Points: Problem 1: 25 + 10 = 35 points, Problem 2: 25 + 10 = 35 points, Problem 3: 7 + 8 + 8 + 5 = 30 points. Total: 100 points

Problem 1 (Adversarial Training)**.** In this question, you are asked to implement adversarial training algorithms with PGD.

(a) (25 points) Using the code you had written for HW1, perform (PGD-based) adversarial training. Report the standard accuracy, robust accuracy, and any other observations with **different epsilon values** by filling in the blanks of the provided code.

(b) (10 points) Try to attack the model you obtained after adversarial training using PGD and compare the effectiveness (based on the lowering of accuracy) with that of the original model. How effective is the model against an **untargeted** FGSM attack (one step of PGD)? Record your observations and try to explain them. Try **different epsilon values** against the adversarially trained models and report your observations.

Hints. Adversarial training is nothing but adding adversarial examples during training. You can generate the adversarial examples in the designated space provided in the notebook. The attack is the PGD attack you implement in the previous problem.

Problem 2 (Interval Bound Propagation, Programming)**.** In this problem, you will implement interval bound propagation (IBP) training [GDS+19] for a simple neural network.

Network Description. Implement a fully connected neural network consisting of 3 layers (each 'layer' here is a linear layer followed by a ReLU), each of size 50 neurons. Use cross-entropy loss and train on the MNIST dataset. You can use the dataloaders from previous assignments (`https://github.com/enyijiang/CS521FA25HW/blob/main/hw2/mnist_interval_analysis.ipynb`).

(a) Implement the IBP ibp training procedure on your network. Use the training tricks used by the paper — gradual reduction of the weighing factor κ in Equation 12 of the paper from 1 to 0.5 (see hints below) and gradual increase in the robustness radius ϵ_{train} as the training progress, starting from 0 to target $\epsilon_{train} = 0.1$. Report the standard accuracy and robust accuracy (wrt PGD attack) of your network. Also report the training time and contrast it with that of standard training.

(b) Use your box verification implementation from HW-2 and report the verified accuracy (number of test images for which the network is certified robust in an L_∞ radius of ϵ_{test} ball), where ϵ_{test} can take 10 values, evenly between 0.01 and 0.1. Analyze images of some adversarial examples with perturbations within different ϵ_{test} if your network is not certified to be robust for some ϵ_{test}.

Solution Requirements. You should present your solution for this in the form of a Jupyter notebook. We recommend using Google Colab since we can interact with your solution easily, but you can also just upload the notebook to your GitHub repo.

Hints. IBP procedure modifies the training loss function by having another *robustness loss* term. The overall loss looks like as follows, where $CE(.,.)$ is the cross-entropy loss and z_K are the logits at the last layer K:

$$L_{IBP} = \kappa \cdot CE(z_K, y_{true}) + (1 - \kappa) \cdot CE(\hat{z}_K(\epsilon_{train}), y_{true})$$

The last linear layer is typically absorbed in the robustness specification, as demonstrated in Equation 9 of the paper. You need to basically redefine the loss function in your training process, including the training tricks in the paper for your IBP training. You can refer to `https://github.com/Zinoex/bound_propagation/blob/main/examples/fashion_mnist.py` for the implementation.

Problem 3 (Certified Training for Multiple Norms)**.** We have discussed various certified training techniques in the lectures. CURE [JCS24] is a unified certified training framework for robustness against multiple perturbations. In this problem, we ask you to read the aforementioned paper and write a critical, conference-style review for it, including the following parts:

1. *Summary*: Summarize the paper in your own words.
2. *Strengths* (atleast 3): Elaborate on the strengths you see in the paper.
3. *Weaknesses* (atleast 3): Mention the weaknesses of this paper and suggest possible fixes for the same.
4. *Extensions* (atleast 1): Although such a section is not included in conference reviews, mention possible extensions of this work in the context of contemporary research. For example, given the evolution of multi-norm research for adversarial robustness, how could we potentially adapt them to the certified robustness field?

Bonus Research Question (25 points). We discussed the first technique for certified UAP training [XS24] in the lectures. In this problem, we ask you to brainstorm and improve upon the technique so as to achieve higher standard and worst-case UAP accuracies on the MNIST dataset. You can train on the architecture used for the other problems in the homework, or come up with another novel architecture (e.g., L_∞ UAP net). In this open-ended research problem, we will reward upto 15 points for a well-described idea with pseudocode and upto 25 points for the idea, pseudocode, and implementation. You should provide valid justifications for each idea/component of your procedure.

References

[GDS+19] Sven Gowal, Krishnamurthy Dvijotham, Robert Stanforth, Rudy Bunel, Chongli Qin, Jonathan Uesato, Relja Arandjelovic, Timothy Mann, and Pushmeet Kohli. On the effectiveness of interval bound propagation for training verifiably robust models, 2019.

[JCS24] Enyi Jiang, David S Cheung, and Gagandeep Singh. Towards universal certified robustness with multi-norm training. *arXiv preprint arXiv:2410.03000*, 2024.

[XS24] Changming Xu and Gagandeep Singh. Cross-input certified training for universal perturbations, 2024.