RTAI 2023 Project: DeepPoly Verifier

Robin Staab, robin.staab@inf.ethz.ch
Yuhao Mao, yuhao.mao@inf.ethz.ch

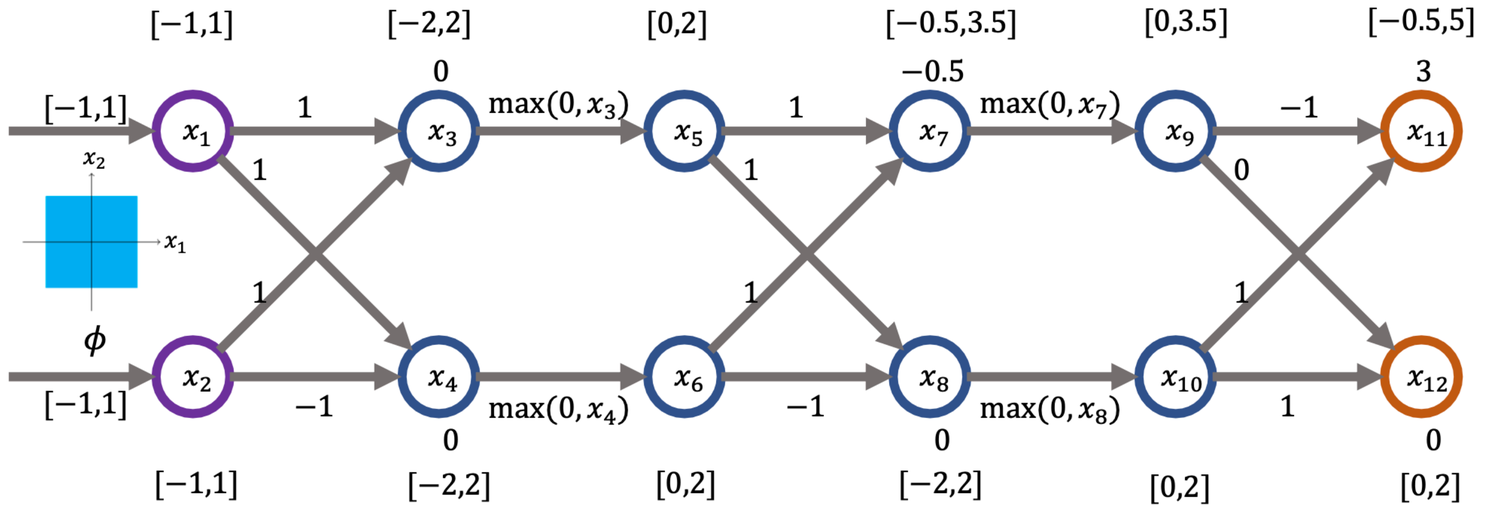
Why This Project?

- Verification of neural networks is considered key to achieving robust AI.
- Deterministic, convex-relaxation based verifications are efficient and practically also a key component of certified training.
- DeepPoly [1] empirically achieves the best approximation precision, compared to other common relaxations [2].

What to Expect

- Design of sound, as precise as possible DeepPoly verifiers for leaky ReLU.
- Implementation (PyTorch) of DeepPoly for convolution, fully connected and leaky ReLU layers.
- Hands-on practice with neural network verification.

Concept of DeepPoly



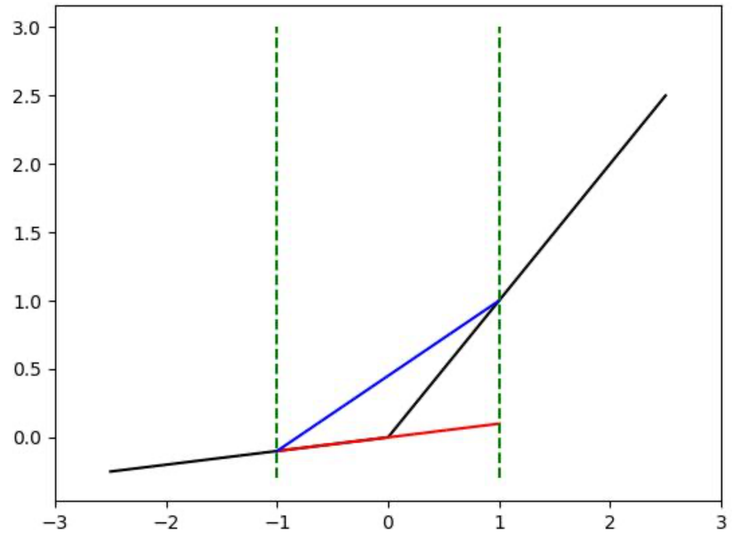
ψ: we want to prove that $x_{11} > x_{12}$ for all values of x_1, x_2 in the input set

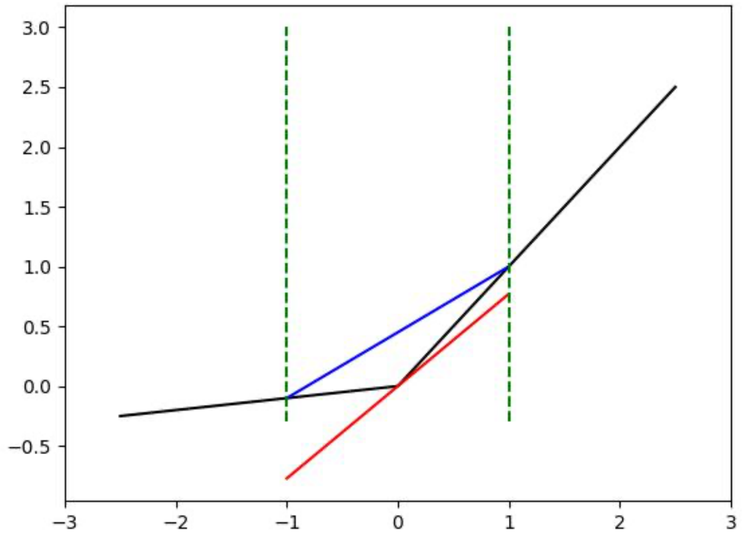
Certification with Box fails as it cannot capture relational information

Plot taken from lecture 4 slide.

Check Exercise 4, Problem 1.b for more details.

Leaky ReLU (neg slope=0.5)





- Best upper bound is unique.
- By default, we expect lower bound slope to be the same as leaky slope.
- You get to decide what is a better slope for the lower bound.

Points

- Implementation of Linear transformer (fully connected and convolutional)
  - Deep Linear Network without convolutional layers. Activation is Identity.
  - Deep Linear Network with convolutional layers. Activation is Identity.
- Implementation of Leaky ReLU transformer
  - ReLU nets w/wo convolutional layers. Activation is ReLU, i.e. slope=0.
  - Leaky ReLU nets w/wo convolutional layers. Activation is leaky ReLU with slope<1.
  - Leaky ReLU nets w/wo convolutional layers. Activation is leaky ReLU with slope>1.
  - Leaky ReLU nets w/wo convolutional layers. Activation is leaky ReLU with mixed slopes (only guaranteed to be non-negative).
- Strategies for slope optimization

Grading

- There are in total 140 test cases.
- You start with 0 points. For each correctly verified case, you get +1 points. For each unsound case (mistakenly return verified), you get -2 points. You do not get any point change if you return unverified. Therefore, it is possible that maximum points are fewer than 140.
- We will make sure always returning verified will be worse than always returning unverified for the final grading.

Evaluation Setup

- We provide a public test set containing the corresponding network and inputs from MNIST and CIFAR-10. The final test set will contain the public test set but also a hidden test set (including hidden test networks and hidden test inputs).
- Any form of adversarial attack is prohibited. You are not allowed to seek adversarial examples during verification.
- The verification should be based on DeepPoly relaxations. It is guaranteed that either DeepPoly with suitable slopes can verify the input, or there exist some adversarial examples for the input.

Evaluation Setup (cont.)

- Example commands for evaluation:
  - `python code/verifier.py --net {net} --spec test_cases/{net}/img{id}_{dataset}_{eps}.txt`
- Your verifier will be executed using 60-second timeout, 8 CPU cores, 20GB memory limit and the Python environment specified in README.
- If extra Python packages are desired, post on Moodle and we will let you know if this is possible. Otherwise, no extra package will be installed in the grading environment.

Preliminary Submission

- Groups can submit their project by the preliminary submission deadline to receive feedback.
- We will run your verifier on 25% test cases which will be used for the final grading and report to you, for each test, the input image, ground truth, output and the runtime of your verifier (no hidden network is included).
- Your preliminary submission results do not affect your final project score.

Deadlines

Project Annoucement	Oct 25
Code Release	Oct 26
Group Registration	Nov 5, 11:59pm
Preliminary Submission (optional)	Dec 3, 11:59pm
Preliminary Feedback	Dec 7
Final Submission	Dec 22, 11:59pm

Group Registration

- Each group must be 2-3 students.
- All group members will get the same points for the project.
- Register your group with following link before Nov 5th: https://forms.gle/ooXhqudsaQerbeN9A
- If you do not have a preferred group but want to take the project, register yourself alone. We will match single-person groups uniformly at random to form a valid group.
- Only one person in each group should fill in the form.

Submission Details

After the group registration deadline, each group is going to receive an invitation to a GitLab repository of name **ddd-rtai-project-2023** where **ddd** here is the group number that will be assigned to you. This repository will contain template code, networks and test cases (content is the same as the zip file released on the course website)

Please ensure you log into https://gitlab.inf.ethz.ch/ at least once before the group registration deadline! Otherwise we might not be able to find you when creating the repository!

Advice

- You can check soundness with adversarial attacks, but do not push the attack into your repository!
- We strongly encourage you to take the preliminary submission to check for issues and get feedback - Final grades will be computed ONLY on your final submission!
- The test cases have a clear difficulty structure; make sure you tested easier features before implementing new functionalities.
- Start leaky ReLU with default slopes; this would verify some inputs. Tune the slope and compare the verifiability gains.

Additional Hints

- ReLU is a special case of Leaky ReLU.
- Note that you will be scored simply on a time limit. In some cases, this makes brute-force approaches a viable option.

References

- [1] DeepPoly paper: https://www.sri.inf.ethz.ch/publications/singh2019domain

- [2] https://www.sri.inf.ethz.ch/publications/jovanovic2022paradox