Provably Bounding Neural Network Preimages

Suhas Kotha[1*], Christopher Brix[2*], Zico Kolter[1,4],
Krishnamurthy (Dj) Dvijotham[3†], Huan Zhang[1†]

[1]Carnegie Mellon University, Pittsburgh PA, 15213, USA.
[2]RWTH Aachen University, Aachen, 52056, Germany.
[3]Google Research, Brain Team.
[4]Bosch Center for AI.

*Corresponding authors. E-mails: suhask@andrew.cmu.edu; brix@cs.rwth-aachen.de;
†These authors contributed equally to this work.

Abstract

Many safety-critical applications require the computation of preimages of neural networks for a given output. We present INVPROP, an algorithm to compute a convex overapproximation of these preimages that does not rely on the use of LP solvers and can be executed using GPUs. The experimental evaluation demonstrates that some computed overapproximations are over **2500**× tighter and **2.5**× faster than prior work.

1 Introduction

In safety-critical applications, it is often of interest to determine the set of inputs that produces a given set of outputs. Examples include autonomous robots that should reach a target area ("What location must the robot be in, to reach the target in n timesteps?") or out-of-distribution (OOD) detection ("What inputs will cause an OOD-alert?") Since neural networks are not invertible, computing the required exact input set may be computationally infeasible. Instead, we propose a technique to compute a convex hull of the input set in a highly efficient, GPU-supported fashion. Importantly, this separates our approach from state-of-the-art techniques that rely on LP solvers. We demonstrate significant speed and precision improvements over the state-of-the-art [1, 2] using our technique.

2 Problem Statement

For a given network $f : \mathcal{X} \subseteq \mathbb{R}^{\text{in}} \to \mathbb{R}^{\text{out}}$ and an output set $\mathcal{S}_{\text{out}} \subseteq \mathbb{R}^{\text{out}}$, we need to compute a convex overapproximation of $f^{-1}(\mathcal{S}_{\text{out}}) \subseteq \mathcal{X}$. We assume that the output set can be described by linear constraints: $\mathbf{H}f(\boldsymbol{x}) + \mathbf{d} \leq 0$.

The convex hull of $f^{-1}(\mathcal{S}_{\text{out}})$ can be described as $\bigcap_{\boldsymbol{c}\in\mathbb{R}^{\text{in}}}\{\boldsymbol{x} : \boldsymbol{c}^\top \boldsymbol{x} \geq \min_{f(\boldsymbol{x'})\in\mathcal{S}_{\text{out}}} \boldsymbol{c}^\top \boldsymbol{x'}\}$, As an approximation, a finite set of $\boldsymbol{c}$ can be used. Thus, we need to solve several instances of the problem

$$\min_{\boldsymbol{x}} \quad \boldsymbol{c}^\top \boldsymbol{x} \tag{1a}$$

$$\text{s.t.} \quad \boldsymbol{x} \in \mathcal{X}; \quad \mathbf{H}f(\boldsymbol{x}) + \mathbf{d} \leq 0 \tag{1b}$$

3 INVPROP

The constraint $\mathbf{H}f(\boldsymbol{x}) + \mathbf{d} \leq 0$ can be moved into the optimization objective by introducing a dual variable γ. Inverting the order of min and max, we get a lower bound:

$$\max_{\boldsymbol{\gamma}} \min_{\boldsymbol{x}} \quad \boldsymbol{c}^\top \boldsymbol{x} + \boldsymbol{\gamma}^\top (\mathbf{H}f(\boldsymbol{x}) + \mathbf{d})$$
$$\text{s.t.} \quad \boldsymbol{x} \in \mathcal{X}; \quad \boldsymbol{\gamma} \geq 0$$

The inner minimization is only constrained by $\boldsymbol{x} \in \mathcal{X}$, so a lower bound can be efficiently computed by Auto-LiRPA [3]. The result is a relaxed expression with additional parameters $\boldsymbol{\alpha} \in [0, 1]$:

$$\max_{\boldsymbol{\gamma}} \quad \text{AutoLiRPA}(\boldsymbol{\alpha}, \boldsymbol{\gamma})$$
$$\text{s.t.} \quad \boldsymbol{x} \in \mathcal{X}; \quad \boldsymbol{\gamma} \geq 0; \quad 0 \leq \boldsymbol{\alpha} \leq 1$$

We note that each instantiation of $\boldsymbol{\gamma} \geq 0$ yields a valid lower bound. Both λ and α can then be optimized using projected gradient ascent using PyTorch, which iteratively tightens the lower bound of the original problem (1).