AI²: Safety and Robustness Certification of Neural Networks with Abstract Interpretation

Timon Gehr  Matthew Mirman  Dana Drachsler-Cohen  Petar Tsankov  Swarat Chaudhuri  Martin Vechev

I. INTRODUCTION

Adversarial examples can be especially problematic when safety-critical systems rely on neural networks. For instance, it has been shown that attacks can be executed physically (e.g., [5], [12]) and against neural networks accessible only as a black box (e.g., [7], [18], [20]). To mitigate these issues, recent research has focused on reasoning about neural network robustness, and in particular on local robustness. Local robustness (or robustness, for short) requires that all samples in the neighborhood of a given input are classified with the same label [15]. Many works have focused on designing defenses that increase robustness by using modified procedures for training (e.g., [7], [8], [14], [15], [19]). Others have developed approaches that can show non-robustness by underapproximating neural network behaviors [1] or methods that decide robustness of small fully connected feedforward networks [10]. However, no existing sound analyzer handles convolutional networks, one of the most popular architectures.

The main challenge facing sound analysis of neural networks is scaling to large classifiers while maintaining a precision that suffices to prove useful properties. The analyzer must consider all possible outputs of the network over a prohibitively large set of inputs, processed by a vast number of intermediate neurons.

To avoid this state space explosion, current methods (e.g., [9], [10], [16]) symbolically encode the network as a logical formula and then check robustness properties with a constraint solver. However, such solutions do not scale to larger (e.g., convolutional) networks, which usually involve many intermediate computations.

II. CONTRIBUTIONS

The key insight of our work is to address this challenge by leveraging the classic framework of abstract interpretation (e.g., [2], [3]), a theory which dictates how to obtain sound, computable, and precise finite approximations of potentially infinite sets of behaviors. Concretely, we leverage numerical abstract domains – a particularly good match, as AI systems tend to heavily manipulate numerical quantities. By showing how to apply abstract interpretation to reason about AI safety, we enable one to leverage decades of research in numerical abstract domains (e.g., [17]). With abstract interpretation, a neural network computation is overapproximated using an abstract domain. An abstract domain consists of logical formulas that capture certain shapes (e.g., zonotopes, a restricted form of polyhedra).

Based on this insight, we developed a system called AI² (Abstract Interpretation for Artificial Intelligence). AI² is the first scalable analyzer that handles common network layer types, including fully connected and convolutional layers with rectified linear unit activations (ReLU) and max pooling layers.

Given a neural network and an input specification, abstract interpretation computes an abstract output, which is an over-approximation of all possible concrete outputs. This enables AI² to verify safety properties such as robustness directly on the abstract output.

We evaluated AI² on important tasks such as verifying robustness and comparing neural network defenses.

Our main contributions are:

- AI², an end-to-end analyzer, extensively evaluated on feed-forward and convolutional networks (computing with 53 000 neurons), far exceeding capabilities of current systems.
- An application of AI² to evaluate provable robustness of neural network defenses.

III. NEURAL NETWORKS

We give a short introduction to neural networks. This introduction covers the type of neural network that can be analyzed by the presented version of the AI² system.

a) Layers: Layers are functions. Neural networks are often organized as a sequence of layers, such that the neural network is their composition.

b) Activation Functions: Typically, neural network layers perform a linear transformation followed by a nonlinear activation function applied individually to all components of the input. We focus on the commonly-used ReLU activation: ReLU(x) = \max(x, 0).

c) Fully-connected Layer: A fully-connected layer FC_{W,b}: \mathbb{R}^m \rightarrow \mathbb{R}^n is parameterized by a weight matrix \( W \in \mathbb{R}^{n \times m} \) and a bias vector \( b \in \mathbb{R}^n \) and is defined as \( FC_{W,b}(x) = \text{ReLU}(W \cdot x + b) \).

d) Convolutional Layer: A convolutional layer Conv_{W,b} : \mathbb{R}^{m \times q \times r} \rightarrow \mathbb{R}^{m-p+1 \times n-q+1 \times t} is parameterized by weights \( W \in \mathbb{R}^{p \times q \times r} \) and a bias \( b \in \mathbb{R}^t \). A convolutional layer computes \( t \) convolutions of the input with different filters, before it applies the ReLU activation function.
e) Max Pooling layer: The max pooling layer MaxPool\(_{p,q}: \mathbb{R}^{n \times m} \rightarrow \mathbb{R}^{n/p \times m/q}\) partitions the input into \(p \times q\) subrectangles and replaces each of them by their maximum.

IV. ABSTRACT INTERPRETATION

Given a function \(f: \mathbb{R}^m \rightarrow \mathbb{R}^n\), a set of inputs \(X \subseteq \mathbb{R}^m\), and a property \(C \subseteq \mathbb{R}^n\), the goal of abstract interpretation is to determine whether the property holds, that is, whether for any input \(x\) in \(X\), the output \(f(x)\) is in \(C\).

a) Abstract Interpretation: Abstract interpretation uses abstract transformers which operate on abstract domains \(\mathcal{A}^n\), whose elements be represented and processed on a finite computer. Each element \(a \in \mathcal{A}^n\) describes a set \(\gamma(a) \subseteq \mathbb{R}^n\) of possible concrete values. An abstract transformer \(T^\mathcal{P}_f: \mathcal{A}^n \rightarrow \mathcal{A}^n\) propagates such a set through a function \(f: \mathbb{R}^m \rightarrow \mathbb{R}^n\).

In the end, the property can be verified directly on the abstract output. This abstract output may represent a strict superset of the set of possible concrete outputs.

Accordingly, abstract interpretation is sound, but incomplete: A property proved using abstract interpretation holds, but a particular abstract domain will normally not be able to prove all true properties.

Our approach can be instantiated with different abstract domains. For our purposes, an abstract domain has to support the following operations:

- A bottom element \(\bot\), representing the empty set.
- Meet (intersect) an abstract element with a conjunction of linear constraints.
- Join (union) between two abstract elements.
- Apply an affine transformation to an abstract element.

Those operations do not need to be precise, but they need to be sound: the resulting abstract element must represent a superset of the set of concrete results obtained when applying the concrete version of the operation to the sets represented by the input abstract elements.

V. EXAMPLE ABSTRACT DOMAINS

In this section, we provide an incomplete set of example abstract domains.

a) Box Domain: Abstract interpretation with the box domain corresponds to evaluation using interval arithmetic.

b) Zonotope Domain [6]: A zonotope \(z \in Z^N\) is either \(\bot\), or it can be represented as a matrix \(M^{n \times m}\) for some \(m \in \mathbb{N}\), and a center \(c \in \mathbb{R}^n\). In this case, it in turn represents the set \(\gamma(z) = \{M \cdot \epsilon + c \mid \epsilon \in [0, 1]^m\}\). The zonotope domain supports an exact affine transformer, while meet and join necessarily produce a sound approximation, because an exact result may not be representable as a zonotope.

c) Polyhedra Domain [4]: The polyhedra domain consists of abstract elements that represent a convex polyhedron and are in turn represented as a set of linear constraints over the input variables. The polyhedra domain supports an exact affine transformer as well as an exact meet, while join produces a convex hull, the best possible sound overapproximation.

d) Bounded Powerset Domains (e.g. Zonotope\(N\)): The bounded powerset domain is parameterized with a base domain. An element in the bounded powerset domain is a set from the base domain with bounded size. Such abstract elements represent the union of the sets represented by their elements. For example, elements of Zonotope\(N\) consist of \(N\) elements of Zonotope.

VI. ABSTRACT TRANSFORMERS FOR NEURAL NETWORKS

As abstract interpretation is composable, it suffices to define abstract transformers for each of the types of layers.

a) Fully-connected Layer, Convolutional Layer: ReLU applied to the \(i\)-th component of the abstract element \(a \in \mathcal{A}^n\) can be represented as \(g_i(a \cap (x_i < 0)) \cup (a \cap (x_i \geq 0))\), where \(g_i: \mathbb{R}^n \rightarrow \mathbb{R}^n\) is an affine transformation that assigns 0 to the \(i\)-th component of the input. Both the fully-connected and the convolutional layer are particular affine transformations followed by componentwise ReLU, therefore we can obtain abstract transformers for them by composing ReLU abstract transformers for all components with an affine transformer.

b) Max Pooling Layer: The abstract transformer for the max pooling layer operates on one subrectangle at a time. For one such subrectangle, it uses meets with linear constraints to express a case distinction with one case for each possible location of the maximal element within the subrectangle. For each case, it then uses the affine transformer to extract that particular element. The results from all cases are joined, and the abstract transformers for all subrectangles are composed together.

VII. RESULTS

We have experimentally evaluated \(\text{AI}^2\) on neural networks for two well-known classification tasks: MNIST [13] and CIFAR-10 [11].

a) Experiments: We were able to show the following results:

- \(\text{AI}^2\) can prove useful robustness properties for convolutional networks with 53 000 neurons and large fully connected feedforward networks with 1 800 neurons.
- \(\text{AI}^2\) benefits from more precise abstract domains: Zonotope enables \(\text{AI}^2\) to prove substantially more properties over Box. Further, Zonotope\(N\), with \(N \geq 2\), can prove stronger robustness properties than Zonotope alone.
- \(\text{AI}^2\) scales better than the SMT-based Reluplex [10]: \(\text{AI}^2\) is able to verify robustness properties on large networks with \(\geq 1200\) neurons within few minutes, while Reluplex takes hours to verify the same properties.

b) Defenses: Additionally, we have evaluated state-of-the-art neural network defenses [7], [14], [19]). Those neural network defenses modify the way that the neural network is trained, to make it more robust (but without formal guarantees). We showed that different defenses produce neural networks that differ significantly in how amenable they are to verification using our \(\text{AI}^2\) system.
REFERENCES


