Notes on AI2

Daniel S. Fava

May 23, 2018

1 Background and motivation

The AI2 paper is *not* about learning *per se*. Instead, it is about analyzing neural networks that have already been learned. The goal is to be able to prove that a neural network is robust to *adversarial examples.*

Let i be an image, n be a neural network, and $l = n(i)$ be the label attributed to i by n. An adversarial example is an image i' created by performing small perturbations on i with the objective of "tricking" the neural network into mislabeling i'. Robustness against adversarial attacks involves a notion of continuity: Small perturbations to the input image should not cause large perturbations to the resulting label. In other words:

Problem statement *Consider the tuple (n, i, l, P) where n is a feed-forward neural network, i is an image, $l = n(i)$ the label given to i by n, and P is a function from image to set of images where $P(i)$ be the set of images obtained by making perturbations to i. Informally, the goal is to prove that, if $n(i) = l$, then for all $i' \in P(i)$, $n(i') = l'$ where the "distance" between l and l' is small.*

2 Intermediate representation

The paper restricts itself to *feed-forward* neural networks. These are directed acyclic graphs. There are no feedback loops. The restriction to feed-forward neural networks is important. It allows us to map a neural network into a simple programming model called *conditional affine transformation* functions (CAT functions). This programming model has no loops and only bounded recursion. Removing unbounded loops and recursion significantly simplify the analysis.

Section 2 of the paper explains how one can translate a feed-forward neural network into CAT functions. The analysis via abstract interpretation is then performed on the CAT functions representing the neural network.

3 Analysis

To argue for robustness, we must not only consider how a network (more precisely, the CAT function representation of the network) classifies an image. Instead, we must ask the question of how the network classifies images in the vicinity of the image under analysis.

Let's say a neural network is a function $n : I \to L$ from image to label. Let $P(S)$ be the power-set of a set S. We can define a neural network $N : P(I) \to P(L)$ that acts not on an image, but on a set of images $P(I)$, mapping them to a set of labels $P(L)$. The behavior of N can be derived from n:

$$N(S) = \{n(s) | \text{ for all } s \in S\}$$

We have lifted the definition of neural network by going from n to N. So far, however, we have only swept the dust under the rug; meaning, to speak about the behavior of N, we still need to evaluate $n(s)$ for all $s \in S$. When it comes to proving robustness against adversarial attacks, the size of S is prohibitively large.

Here is where abstract interpretation comes into play. We abstract a set of images S into a single mathematical object A and we analyze the behavior of the neural network on A. But we need to think about what it means to operate on this abstract image. Since this abstract object is no longer an image or a set of images, we cannot apply n or N to it. Instead, we need to formulate a corresponding "abstract" neural network, say N', that operates on abstract images. This is discussed on sections 3 and 4 of the paper.

4 Abstract interpretation

The paper looks into three abstract domains: box, zonotopes, and polyhedron. Let $\alpha_x(i)$ be the abstract version of image i in the box domain. What this means is, as opposed to an image i, computing $\alpha_x(i)$ gives us a set of inequalities that define a bounding box around the image i. This set of inequalities is a finite representation (a finite set of inequalities) for a potentially infinite number of images (all the images that fall inside the box). Similarly, let $\alpha_z(i)$ be a zonotope around an image i and $\alpha_p(i)$ a polyhedron.

Note 1 *We put curly brackets around i because abstraction functions are usually defined from a set of concrete elements as opposed to from a single element. In the discussion so far, we spoken of a single image, so, we use brackets to turn the image into a set.*

Abstract domains have abstract transformers associated with them. For example, if an abstract version of multiplication by scalar is defined for the box domain, we can then multiply a box by a scalar and stay within the abstract domain.

In order to be able to implement an abstract neural network with ReLU, the abstract domain needs to support the mathematical operations used in affine

transformations as well as the mathematical operations required by the ReLU activation function.

Note 2 *Affine transformations include shear. Shear turns, for example, a square into a parallelogram without right angles. If we are in the box domain, the abstract transformers need to take shear into account by re-boxing resulting geometric object in order to remain in the box domain. Note also that affine transformations are naturally handled by the polyhedron domain; for example, shearing of a polyhedron results in another polyhedron.*

5 Soundness

We informally sketch how one can show that operating at the abstract level *makes sense,* meaning, that it is safe to derive conclusions about the concrete domain from an analysis of the abstract domain.

Let T be an operation in the concrete domain and $T^{\#}$ be the corresponding operation in the abstract domain. Recall that α is the abstraction function and that $\alpha(S)$ is an element of the abstract domain which over approximates S. We can show that an abstraction is sound by showing that, if we apply T to S and we apply $T^{\#}$ to $\alpha(S)$, then the resulting object in the abstract domain remains an over approximation of the resulting objects of the concrete domain. If that is the case, we say that the abstract domain *simulates* the concrete domain.

References

[Gehr et al.,] Gehr, T., Mirman, M., Drachsler-Cohen, D., Tsankov, P., Chaudhuri, S., and Vechev, M. Ai2: Safety and robustness certification of neural networks with abstract interpretation.