INF575: Final project (An Abstract Domain for Certifying Neural Networks)

Quentin LAO

Introduction

Artificial neural networks (NNs) are skyrocketing nowadays and are being used increasingly in various fields for their capacity to predict well characteristics of outputs they may have never seen before. The issues of NN validation are becoming ever more important. Does a NN do what one expected? For instance, FAA (the agency of the United States Department of Transportation) developed an Airborne Collision-Avoidance System (ACAS Xu) for drones which is implemented with 45 NNs: if an intruder is nearby on the left, will the advisory always respond "Strong right"? Another issue of NN validation would be defining to what extent NNs are sensitive to tiny variations. For example, if one takes a Convolutional Neural Network (CNN) image classifier, will the model be able to still guess correctly when shifting all the pixels by at most ε (that is called an adversarial attack)?
Actually, verifying NNs is a difficult problem. They are composed of thousands of neurons that were trained and are hard to interpret for humans. They act like a black box: one gives input and the NN returns back mysterious output without knowing what happened. In this report, I will be presenting DeepPoly[1], a new method for certification of NNs and more precisely of Deep Neural Networks (DNNs).

1 Main challenges

1.1 Crucial Criteria

Ideally, a verifier has to aim to respect the following crucial criteria :

- Soundness: if the property is violated, the verifier has to notify it was violated.
- Completeness: if the verifier notifies the property holds, it actually holds.
- Scalability: the capacity to adapt to large NNs such DNNs.

In practice, researchers try to balance these criteria for instance by making sure that at least two of them are satisfied and by trying to improve the third one.

1.2 Main challenges

The specificity of NNs is their non-linear activation functions which render the problem non-convex. Indeed, if activation functions in a neural network are removed, it would become a mere giant linear regression model where linear programming (LP) problems could be resolved. The most popular activation function is Rectified Linear Unit (ReLU) and most of the research papers treat this activation function above the others. NN verification is experimentally beyond the reach of LP solvers or basic Satisfiability Modulo Theories (SMT) solvers.
The main goal would be building a certifier that takes a program (for instance any NN) and a property, and returns a validation flag if the latter property is always verified and a counter-example if it is not. Rice's theorem says that such a general verifier is not possible even less with the scalability criterion. Thus, in the past years, researchers attempted to explore new methods that could prove whether a specific property of interest only is verified. For instance, some researchers dealt with SMT problems solving like Reluplex[2]. With successive back variable assignments, Reluplex can solve

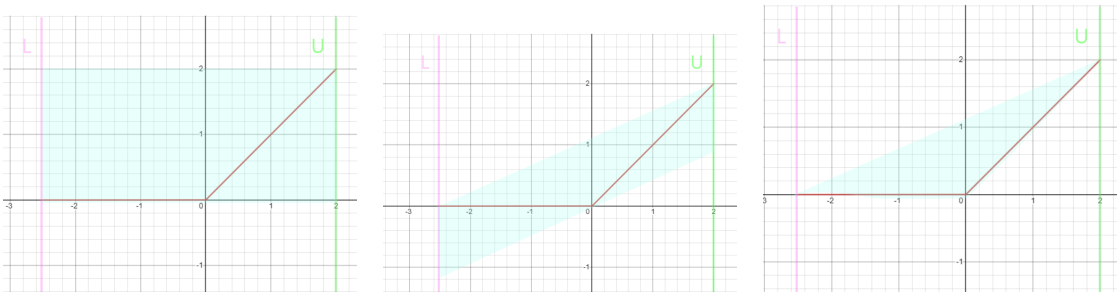

Figure 1: How Box relaxation, Zonotope and Triangle abstraction deal with ReLU

existing problems of type "Is it possible that an input $x_0 \in [a, b]$ produces an output $x_n \in [c, d]$?". However, Reluplex is not efficient to prove the robustness of NNs, another type of verification problem. Indeed, Reluplex can only prove robustness properties on small networks. Abstract interpretation, more scalable, has been recently used to certify the robustness of NNs[3]. For instance, adversarial attacks occur when inputs are shifted by at most ε due to a perturbation. This is the case when pixels in an image are slightly brightened. Another transformation would be rotating the image.
Let's focus on the problem of robustness in this report. As it is impossible to compute all the concrete input values, the main challenge for DeepPoly is to approximate the output state regarding the criteria of **soundness**; that is DeepPoly abstraction set should contain the concrete values. In addition to soundness, the main challenge facing by DeepPoly is **scaling** to large and deep classifiers/regressors while maintaining a precision that suffices to prove useful properties.

2 Abstractions models

Let's explore the state of art of abstractions models and see how DeepPoly stands out from these.

2.1 Previous models

Abstract domains consist of shapes that can be expressed as a set of logical constraints. There are a few popular abstract domains: Box, Zonotope and Polyhedra.
Box relaxation only deals with intervals, namely a lower and an upper bound. This abstract domain is computationally cheap and memory friendly but lacks precisions. What we call precisions here is the thickness of the covered area. In figure 1, ReLU curves are plotted with the covered area of each abstraction; notice only the non-trivial case with a negative lower bound and a positive upper bound was illustrated here. Box relaxation is neither exact for affine transformation nor for ReLU whereas Zonotope relaxation is exact for affine transformation but not for ReLU.
Zonotope relaxation is based on the following expression for each neuron: $\hat{x} = a_0 + \sum_{i=1}^{n} a_i \varepsilon_i$ where $\varepsilon_i \in [-1, 1]$. For each ReLU function, an additional uncertain variable ε_i is added to the sum which means that uncertainty grows up as the NN becomes greater! The covered areas (figure 1) of Box Relaxation and Zonotope Relaxation are not comparable, the thickness of the area let us say Zonotope is more precise.
Another abstraction is convex Polyhedra. The optimal convex transformer for ReLU is known as triangle relaxation. The covered area is 2 times smaller than the Zonotope Relaxation (as we can see in figure 1) but is too expensive to work with. Generally speaking, working with Polyhedra is too computationally expensive because of the high number of constraints (figure 1 shows the complexity of Polyhedra). This is why people usually work with restricted Polyhedra; the kind of constraints are limited and thus scale better to larger NNs.

2.2 DeepPoly

The DeepPoly is a new abstract domain that combines floating point Polyhedra with intervals. As DeepPoly is a restricted Polyhedra, it is less precise but more scalable. Figure 1 compares the two models' complexities for each type of transformation. For each neuron x_i, DeepPoly relies on :

Transformer	Polyhedra	DeepPoly
Affine	$O(nm^2)$	$O(w_{max}^2 L)$
ReLU	$O(\exp(n,m))$	$O(1)$

Table 1: Complexity comparison between Polyhedra and DeepPoly
(n : Nb neurons ; m : Nb constraints ; w_{max} : max nb neurons in a layer ; L : Nb layers)

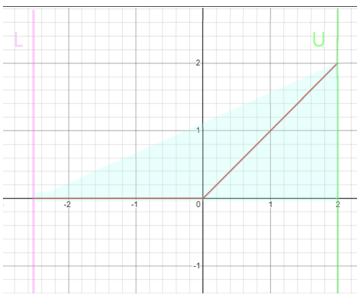

Figure 2: Covered area by DeepPoly

- a lower bound ℓ_i and a upper bound u_i (Interval constraints) : $\ell_i \leq x_i \leq u_i$
- $a_i^{\leq}$ and $a_i^{\geq}$ both of the form $\sum_j w_j x_j + v$ (Relation constraints) : $a_i^{\leq} \leq x_i \leq a_i^{\geq}$.

DeepPoly captures affine transformation precisely as Zonotope does. Moreover, DeepPoly seem to covered less area than Zonotope for a ReLU transformation. In the non-trivial case (a negative lower bound and a positive upper bound), the covered area by DeepPoly is illustrated in figure 2. Even if the DeepPoly's area and Zonotope's area are not comparable, in most of the cases in practice, DeepPoly is more precise. Backsubsitution also contributes to the good performance of DeepPoly. If bounds are too unprecise to verify a property, we can substitute the variables x_j in the relation constraints with the constraints from previous neurons; it is one of the powerful purposes of using symbolic writing.
Thanks to its restricted number of constraints per neuron, DeepPoly is more **scalable** and is **sounded** (the covered area always covers the concrete values). However, it is **incomplete** but is still more precise than Box Relaxation and Zonotope Relaxation. DeepPoly is a most outstanding one regarding precision and scalability.

3 DeepSRGR to improve DeepPoly

As written in subsection 1.2, when a verifier tells us a property is not verified, we ideally wish it returns a counter-example. However, due to over-approximations of DeepPoly, the abstraction is unable to give us a counter-example as it may be a *spurious* one. In other words, soundness allows DeepPoly to tell us when a property is satisfied but its incompleteness doesn't give any information when it tells us the property is not. The goal of the tool DeepSRGR[4] is to identify these spurious areas in order to refine the bounds of each neuron and then reduce approximation.
To understand how DeepSRGR algorithm works, let x_1 and x_2 be the two inputs of a NN and let y_1 and y_2 the two outputs. Let's bound $-1 < x_1, x_2 < 1$ et let's say the robustness property we want to verify is $P : y_2 > y_1$. The way the authors process is as follows:

1. Run DeepPoly a first time on the NN to compute the 4 constraints on each neurons (intervals and relations constraints)
2. If the property can be verified, then the algorithm stops there : the property is verified. Otherwise, the property cannot be verified yet; for example, if the lower bound of y_2 is smaller than the upper bound of y_1.
3. The property is not verified. The authors propose to apply linear programming using $\neg P$ as a constraint in addition to the whole relations constraints provided by the previous run of DeepPoly. They want to tighter the bounds of the input neurons as well as some neurons concerned

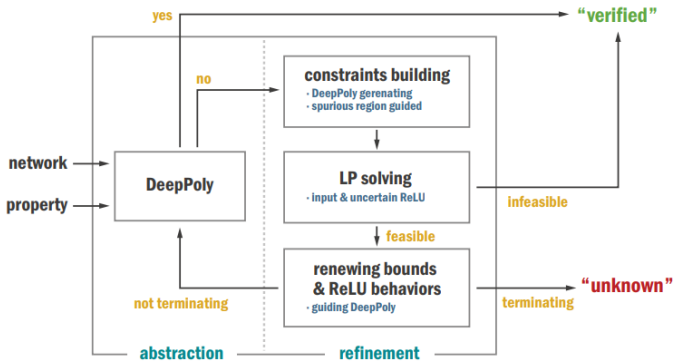

Figure 3: DeepSRGR's process

by uncertain ReLUs (uncertain ReLUs are ReLU transformations where the lower bound was negative and the upper bound positive). They use these neurons as objective functions.

4. Now that the inputs and uncertain hidden neurons have changed bounds, we return at step 1 by rerun DeepPoly with the new bounds.

The above process is actually a loop that repeats once spurious regions are still remaining. We provide an illustration of how DeepSRGR processes in figure 3.
The authors carried out some tests to apply DeepSRGR on robustness verification problems and observed that, by their approach, the bounds can be up to two orders of magnitudes compared to DeepPoly standalone.

Conclusion

To conclude, Neural Network Certifying is a complex issue because of the singularity of NNs: their activation functions and their potential huge size ask researchers to develop new models to tackle their validation. Some models are complete but not scalable like Linear Solvers (LP) and some are scalable but incomplete such as Box and Zonotope even if the minimum requirement is still soundness. We particularly skimmed the state-of-the-art of Abstraction interpretation, espacially used to prove robustness property. These models have the advantage to be scalable. The outstanding abstraction relaxation is DeepPoly. This restricted Polyhedra model is more precise than Box and Zonotope and more scalable than Polyhedra. An interesting way to keep on improving performance is either developing another difference abstraction model or tuning the existing DeepPoly. DeepSRGR is an algorithm that loops and at each iteration refind DeepPoly bounds to increase its precision.

References

[1] Gagandeep Singh, Timon Gehr, Markus Püschel, Martin Vechev 2019. *An Abstract Domain for Certifying Neural Networks*

[2] Guy Katz, Clark W. Barrett, David L. Dill, Kyle Julian, and Mykel J. Kochenderfer. 2017. *Reluplex: An Efficient SMT Solver for Verifying Deep Neural Networks. In Proc. International Conference on Computer Aided Verification (CAV).* 97-117

[3] Matthew Mirman, Timon Gehr, and Martin Vechev. 2018. *Differentiable Abstract Interpretation for Provably Robust Neural Networks. In Proc. International Conference on Machine Learning (ICML).* 3575-3583.

[4] Pengfei Yang, Renjue Li, Jianlin Li, Cheng-Chao Huang, Jingyi Wang, Jun Sun, Bai Xue, and Lijun Zhang, *Improving Neural Network Verification through Spurious Region Guided Refinement*, 2021