Verification-friendly Networks: the Case for Parametric ReLUs

Francesco Leofante [*1] **Patrick Henriksen** [*1] **Alessio Lomuscio** [1]

Abstract

It has increasingly been recognised that verification can contribute to the validation and debugging of neural networks before deployment, particularly in safety-critical areas. Despite considerable progress, present techniques still do not scale to large architectures used in many applications. In this paper we show that substantial gains can be obtained by employing Parametric ReLU activation functions in lieu of plain ReLU functions. We give training procedures that produce networks which achieve one order of magnitude gain in verification overheads and 30-100% fewer timeouts with a SoA Symbolic Interval Propagation-based verification toolkit, while not compromising the resulting accuracy. Furthermore, we show that adversarial training combined with our approach improves certified robustness up to 36% compared to adversarial training performed on baseline ReLU networks.

1. Introduction

Deep Neural Networks (DNNs) are significantly contributing to several innovations in AI, ranging from computer vision and speech recognition to natural language processing and beyond (Goodfellow et al., 2016). As a result, DNNs are increasingly considered for a wider set of applications, including safety-critical ones (Kouvaros et al., 2021; Fremont et al., 2020). However, using neural networks in safety-critical systems requires them to undergo verification and validation efforts in ways that are similar to common practice in software deployment in safety-critical systems.

The area of Verification of Neural Networks (VNN) is concerned with the development of methods for verifying the robustness of DNNs against input perturbations. While the key verification problem is NP-complete in general (Katz et al., 2017), several scalable verification frameworks have been proposed in recent years that can handle a wide variety of architectures and properties (Bak et al., 2020; Balunovic et al., 2019; Botoeva et al., 2020; Bunel et al., 2020; Cheng et al., 2017; Dvijotham et al., 2018; Henriksen & Lomuscio, 2021; Henriksen et al., 2021; Katz et al., 2019; Singh et al., 2019; Tjeng et al., 2019; Tjandraatmadja et al., 2020; Tran et al., 2020; Wang et al., 2018b; 2021). Special attention has been devoted to the verification of DNNs using ReLU activations, given their prominence in Deep Learning. In this context, approaches based on Symbolic Interval Propagation (SIP) (Wang et al., 2018b; Singh et al., 2019; Wang et al., 2021; Henriksen & Lomuscio, 2021) are among the top performing, as witnessed in the VNN Competition 2021 (VNN-COMP21) (Bak et al., 2021).

Given a DNN and an input to be analysed, SIP-based ver-

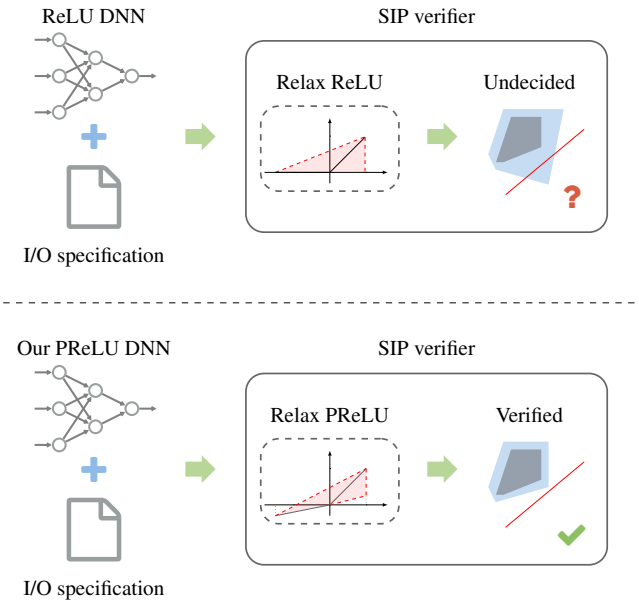

Figure 1: Overview of the proposed contribution. **Top**: Relaxing ReLU functions produces large over-approximations (blue) of the exact output-reachable set (grey) of a DNN; this may yield inconclusive verification results. **Bottom**: We combine PReLUs with dedicated regularisers to train networks that can be relaxed more succinctly, thus improving the performance of verification.

[*]Equal contribution [1]Department of Computing, Imperial College London, London, United Kingdom. Correspondence to: Francesco Leofante <f.leofante@imperial.ac.uk>.

1st Workshop on Formal Verification of Machine Learning, Baltimore, Maryland, USA. Colocated with ICML 2022.

ifiers typically operate as follows. First, the network activation functions are abstracted into more computationally tractable functions. This operation is commonly realised via *linear relaxations* that approximate non-linear behaviours by means of linear models (Ehlers, 2017; Wang et al., 2018b; Singh et al., 2019). Then, Symbolic Interval Propagation is used to compute an over-approximation of the output-reachable space generated by the DNN for the input under analysis. The method guarantees by construction that if the over-approximated computation satisfies the property in question, so does the original model. However, due to over-approximation it is possible that no conclusion can be reached on the query (Figure 1, top), introducing the need for further refinement steps.

The approach sketched above offers benefits in terms of scalability; but its precision, and hence its ability to solve verification queries, is directly impacted by the over-approximation error introduced by linear relaxations. Considerable efforts have been put into designing more precise approximations; recent developments include (Batten et al., 2021; Fazlyab et al., 2020; Raghunathan et al., 2018). While these advances cannot match the scalability of SIP on linear relaxations, and SIP does remain the most performing verification technique, even SIP cannot scale to the large models used in applications. Scalability thus remains one of the key issues to be solved in DNN verification.

This paper. In this paper we show that careful design choices for the DNN can lead to at least *one order of magnitude* speed-up in SIP-based verifiability. Most importantly we show that these gains need not impact the learning abilities of the classifier, nor their resulting accuracy and robustness. Summing up, we make the following contributions: **(i)** we consider DNNs using ReLU, Leaky ReLU and Parametric ReLU activations. We study the over-approximation induced by their linear relaxations and its impact on DNN verifiability, both theoretically and empirically **(ii)** we identify Parametric ReLU as a promising candidate to generate verification-friendly DNNs and propose regularisation techniques to further improve verifiability **(iii)** we evaluate the proposed learning pipeline and compare the experimental results obtained against ReLU stability (Xiao et al., 2019) and network pruning (Guidotti et al., 2020) approaches, showing up to 20x speed-up with 38% fewer timeouts for larger networks; **(iv)** we conclude by illustrating how the method can be further integrated with adversarial training to improve certified robustness of our networks up to 36% compared to similarly trained ReLU architectures, while keeping accuracy and verifiability high.

2. Background

The Verification Problem. Formal verification of neural networks considers establishing whether some constraints on the network's output are satisfied for all inputs satisfying some input constraints.

Definition 2.1 (Verification Problem)**.** Given a neural network $N : \mathbb{R}^m \to \mathbb{R}^n$, a set of input constraints $\psi_{\boldsymbol{x}}$, and a set of output constraints $\psi_{\boldsymbol{y}}$, the verification problem is to establish whether $N(\boldsymbol{x})$ satisfies $\psi_{\boldsymbol{y}}$ for all $\boldsymbol{x}$ satisfying $\psi_{\boldsymbol{x}}$.

Verification has been shown to be NP-complete for piecewise linear feed forward neural networks (Katz et al., 2017), which are the main focus of this work. Widely adopted scalable approaches to the verification problems typically proceed in three steps: *(i)* they create a linear relaxation of the network *(ii)* they attempt to solve the verification problem wrt to the relaxation and *(iii)* if the problem cannot be solved due to the coarseness of the relaxation, they refine the relaxation and repeat from *(i)*[1]. Refinement is often implemented via a Branch-and-Bound (BaB) procedure (see e.g. (Wang et al., 2021; Henriksen & Lomuscio, 2021; 2020; Bunel et al., 2020)). For piecewise-linear networks such procedures may be complete (i.e. the correct solution can always be determined in a finite number of steps). However, since the worst-case complexity of BaB is exponential in the number of nodes, reducing the number of refinement steps is crucial. As such, minimising the overestimation introduced by the relaxations is essential for efficient resolution.

3. Verification-friendly Networks via PReLUs

Relaxations play an important role in enabling the verification of large DNN models. When evaluating the advantages of a relaxation, two key criteria need to be taken into account: the *precision* of a relaxation, i.e., how closely it approximates the original activation, and the *computational effort* required to reason on the relaxed model.

Very precise relaxations have been proposed, e.g., (Batten et al., 2021; Fazlyab et al., 2020; Raghunathan et al., 2018); however, verifying such relaxations typically requires expensive procedures which hardly scale to large models. On the contrary, linear relaxations schemes, such as the triangle (Singh et al., 2019) and the parallel (Wang et al., 2018b) relaxations, can be used to solve the verification problem for large, high-dimensional DNNs (Henriksen et al., 2021). Unfortunately the reduced computational burden of linear relaxations comes at a price: if the relaxations are too coarse, verification results may be inconclusive.

We show that slight changes in the DNN activation functions may result in considerable gains from a verification perspective without any impact on learning. We validate this hypothesis by analysing the case of Parametric ReLUs (PReLUs). PReLUs have been shown to advantageous in terms of accuracy over plain ReLUs at negligible extra computational

[1] More details about each step can be found in the appendix.

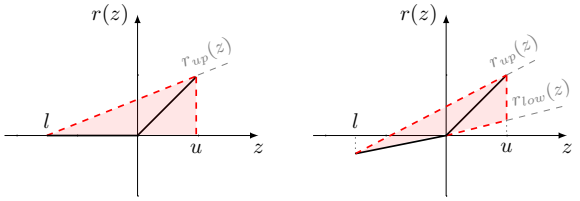

Figure 2: The overestimation induced by PReLUs (2b) is smaller than the corresponding area from ReLUs (2a)

cost (Xu et al., 2015). Crucially, we observe that verifying PReLUs may result in smaller over-approximations via linear relaxations. In the following, we formulate our results in the context of triangle relaxtions; other commonly used linear relaxation schemes are addressed in the Appendix.

Proposition 3.1. *Consider a piecewise linear function defined as*

$$y = \begin{cases} \alpha \cdot x & \text{if } x < 0 \\ x & \text{if } x \geq 0 \end{cases}$$

for $\alpha \in [0, 1]$. *The over-approximation area induced by the triangle relaxation on this function decreases with* $(1 - \alpha)$ *for increasing* α.

The proof of Proposition 3.1 can be found in the Appendix.

Corollary 3.2. *The over-approximation area resulting from the triangle relaxation of a PReLU is smaller than that obtained for ReLU, given the same pre-activation bounds.*

Corollary 3.2 shows that replacing ReLUs with PReLUs may generally result in smaller relaxations *at neuron level*. In the following we introduce a lightweight regulariser that promotes this property across the whole network.

Training PReLUs for easier verification. PReLUs have the potential to generate DNNs that more amenable to SIP-based verification. We have established that increasing the coefficient of a PReLU may generally result in smaller over-approximations when linearly relaxed. However, standard training procedures typically adjust these based on a different criterion, i.e., classification accuracy, and do not necessarily result in easier-to-verify networks.

Co-design principles can be applied to ensure that the full potential of PReLUs is realised, both in terms of learning and verification. Since Proposition 3.1 suggests that higher leakage coefficients for PReLUs may generally be beneficial for verification, we augment standard training losses to account for this secondary objective as $\mathcal{L} = \mathcal{L}_{train} - \beta \cdot \mathcal{L}_2(\alpha_{PReLU})$, where $\mathcal{L}_{train}$ is any standard training loss, while the second term is an $\mathcal{L}_2$ loss applied only to the parameters of the PReLU to encourage steeper slopes for negative values. The new loss term is weighted by a coefficient β to make sure that accuracy is not compromised when training for verifiability. The added loss term requires that the slopes of the PReLU are clipped to $[0, 1]$ after each training iteration to avoid cases where the slope becomes negative or the PReLU becomes concave.

The effect of this simple modification is that DNNs are now trained to maximise their accuracy, while making sure that the resulting DNN is also amenable to SIP-based verification Notably, we achieve this result at negligible extra computational cost, which marks a clear difference between our proposal and previous work in the field. We also note that the proposed modification is orthogonal to, and can be seamlessly integrated with, other commonly used regularisation techniques. Crucially, we will show in Section 4 that our proposal can be used in conjunction with standard techniques for adversarial training, ultimately resulting in DNNs that are accurate, robust and easier to verify.

4. Experimental Analysis

We designed three sets of experiments to show that: **(i)** PReLUs do indeed result in improved verifiability when compared to other rectified linear activations; **(ii)** our approach is competitive with, and often outperforms, SoA approaches for training verification-friendly DNNs; and **(iii)** it can be seamlessly integrated with adversarial training techniques to improve robustness of the resulting networks. We discuss our approach in the context of verification friendly learning and robust training in Section 5.

In the following experiments we used VERINET (Henriksen & Lomuscio, 2020; 2021), a leading and publicly available SIP-based verifier (Bak et al., 2021). We ran a total of 27000 verification queries on a range of networks trained on 3 datasets: MNIST (LeCun, 1998), CIFAR-10 (Krizhevsky et al., 2014), and a reduced version of the Tiny ImageNet (Lee & Yang, 2015) trained on 50 classes. The size and architecture of the DNNs used for these experiments are in line with or exceed standard evaluations in the verification literature. In terms of size, the MNIST and CIFAR-10 networks are comparable to those used in VNN-COMP21; the Tiny ImageNet instead has approximately twice as many activation nodes as the largest architectures used in VNN-COMP21. We tuned hyper-parameters such that the test-set accuracy of the resulting DNNs is comparable to that of standard ReLU networks. Notice, however, that our choice to use architectures in line with the verification literature limits the accuracy attainable by these networks. Further details on the experimental setup are provided in the Appendix.

To allow for a quantitative comparison of verification gains produced by each method considered, we calculate speedup by comparing the N fastest verified cases for each network.

Table 1: Evaluation of rectified linear activations.

MNIST FC	Accuracy		Robust		Not Robust		Robust/	Timeouts		t_{relu}/t	
512 Act. Nodes	Mean	Sd	Mean	Sd	Mean	Sd	Not Robust	Mean	Sd	Mean	Sd
ReLU	98.3%	0.1%	81.0	3.0	32.8	2.9	2.5	11.2	3.4	1.0	0.0
Leaky ReLU ($\alpha = 0.01$)	**98.4%**	0.0%	81.4	2.3	30.2	5.2	2.7	13.4	4.0	0.8	0.5
Leaky ReLU ($\alpha = 0.20$)	98.3%	0.0%	74.6	2.1	41.4	1.0	1.8	9.0	2.0	4.4	3.1
PReLU ($\beta = 0.00$)	98.3%	0.1%	81.6	2.6	29.6	3.6	**2.8**	13.8	1.2	1.0	1.0
PReLU ($\beta = 0.10$)	98.1%	0.0%	87.2	2.1	37.4	1.9	2.3	0.4	0.5	26.2	11.2
PReLU ($\beta = 0.15$)	98.0%	0.1%	82.6	3.3	42.4	3.3	1.9	**0.0**	0.0	**36.9**	16.1
CIFAR-10 Conv.	Accuracy		Robust		Not Robust		Robust/	Timeouts		t_{relu}/t	
45 056 Act. Nodes	Mean	Sd	Mean	Sd	Mean	Sd	Not Robust	Mean	Sd	Mean	Sd
ReLU	85.7%	0.4%	40.2	0.7	51.8	2.8	0.8	33.0	3.2	1.0	0.0
Leaky ReLU ($\alpha = 0.01$)	86.5%	0.2%	41.6	2.0	50.8	2.3	0.8	32.6	1.4	2.3	2.3
Leaky ReLU ($\alpha = 0.20$)	**88.4%**	0.2%	42.2	1.9	45.8	1.9	0.9	37.0	0.9	1.1	1.8
PReLU ($\beta = 0.00$)	87.5%	0.2%	43.4	1.4	47.0	3.5	0.9	34.6	3.3	2.4	2.8
PReLU ($\beta = 0.10$)	86.1%	0.2%	51.0	1.7	50.2	2.6	**1.0**	23.8	1.9	20.4	5.5
PReLU ($\beta = 0.15$)	84.5%	0.2%	51.8	2.1	54.2	4.8	**1.0**	**19.0**	3.1	**23.9**	6.4
Tiny ImageNet (50)	Accuracy		Robust		Not Robust		Robust/	Timeouts		t_{relu}/t	
106 496 Act. Nodes	Mean	Sd	Mean	Sd	Mean	Sd	Not Robust	Mean	Sd	Mean	Sd
ReLU	46.0%	0.9%	40.0	1.8	58.4	4.4	0.7	26.6	3.3	1.0	0.0
Leaky ReLU ($\alpha = 0.01$)	47.0%	0.4%	36.8	1.5	63.4	1.6	0.6	24.8	1.7	3.1	2.4
Leaky ReLU ($\alpha = 0.20$)	**54.6%**	0.6%	31.4	2.9	54.2	3.8	0.6	39.4	3.1	0.1	0.0
PReLU ($\beta = 0.00$)	51.7%	0.8%	34.0	2.0	58.0	4.4	0.6	33.0	2.9	0.5	0.7
PReLU ($\beta = 0.30$)	46.6%	0.6%	47.2	5.0	62.2	5.6	**0.8**	15.6	2.4	11.9	4.7
PReLU ($\beta = 0.40$)	45.4%	1.0%	49.4	5.4	62.6	4.3	**0.8**	**13.0**	2.3	**20.4**	6.7

Here N is the number of cases solved for the network with the most solved cases among the networks under consideration. For more details, see the appendix.

Evaluating rectified linear activations. To assess the impact that various rectified linear activation functions and training methods have on verifiability, we here compare six network variations: one network with ReLU activation functions, two Leaky ReLU networks and three networks with PReLU activation functions.

The Leaky ReLU networks used leakage coefficient of 0.01 and 0.20. The PReLU networks were trained with three β-values to explore the versatility of the approach. In particular, the first used $\beta = 0$, the second used a β tuned such that the accuracy reached approximately the same level as the ReLU network and the last used a β such that the accuracy was slightly below the ReLU network.

Table 1 shows that Leaky ReLU and standard PReLU ($\beta = 0$) networks generally achieved a slightly higher accuracy than the ReLU network, in line with (Xu et al., 2015). However, note that their performance in terms verifiability was comparable to the ReLU network. However, the PReLU networks with $\beta > 0$ showed significant improvements in verification time and reduced number of timeouts. With the smaller β multiplier ($\beta = 0.1$ for MNIST and CIFAR-10, $\beta = 0.30$ for Tiny ImageNet), the PReLU networks achieved the same accuracy as the ReLU networks; yet they achieved a verification speed-up > 10 and the number of timeouts was on average reduced by $96\%, 28\%$ and 41% for the MNIST, CIFAR and TinyImage networks respectively. Further increasing the β multiplier ($\beta = 0.15$ for MNIST and CIFAR-10, $\beta = 0.40$ for Tiny ImageNet) yielded even more significant speed-ups of > 20 times and $100\%, 42\%$ and 51% fewer timeouts; accuracy, however, was affected in this case. Overall, we observe that the ratio of robust to non-robust cases is either stable or increased, indicating that the added loss term does not affect robustness dramatically.

Table 1 shows that PReLUs do offer important advantages in terms of verifiability. Our comparison with ReLUs and Leaky ReLUs showed that these advantages can be obtained with little to no impact on the accuracy of the resulting networks when compared to the ReLU network.

Comparison with other approaches. We now compare with two other SoA approaches resulting in networks that are easier to verify: training for ReLU stability (Xiao et al., 2019) and weight pruning (Guidotti et al., 2020). For both approaches, we tuned hyper-parameters such that resulting

Table 2: Comparison with SoA on CIFAR-10.

MNIST FC	Accuracy		Robust		Not Robust		Robust/	Timeouts		t_{PReLU}/t	
512 Act. Nodes	Mean	Sd	Mean	Sd	Mean	Sd	Not Robust	Mean	Sd	Mean	Sd
PReLU ($\beta = 0.15$)	98.0%	0.1%	82.6	3.3	42.4	3.3	1.9	**0.0**	0.0	1.00	0.00
ReLU Pruning	**98.4%**	0.1%	78.2	4.6	40.0	2.1	1.9	6.8	2.8	0.06	0.04
ReLU Stability	98.2%	0.1%	103.6	3.2	21.4	3.2	**4.8**	**0.0**	0.0	**2.76**	0.90
ReLU	98.3%	0.1%	81.0	3.0	32.8	2.9	2.5	11.2	3.4	0.03	0.02
CIFAR-10 Conv.	Accuracy		Robust		Not Robust		Robust/	Timeouts		t_{PReLU}/t	
45 056 Act. Nodes	Mean	Sd	Mean	Sd	Mean	Sd	Not Robust	Mean	Sd	Mean	Sd
PReLU ($\beta = 0.15$)	84.5%	0.2%	51.8	2.1	54.2	4.8	**1.0**	**19.0**	3.1	**1.00**	0.00
ReLU Pruning	83.7%	0.4%	36.6	0.5	57.0	3.3	0.6	31.4	3.4	0.06	0.03
ReLU Stability	84.6%	0.8%	43.6	2.2	50.6	3.7	0.9	30.8	2.9	0.06	0.02
ReLU	**85.7%**	0.4%	40.2	0.7	51.8	2.8	0.8	33.0	3.2	0.05	0.01

Table 3: Adversarial augmentation results for CIFAR-10.

MNIST FC	Accuracy		Robust		Not Robust		Robust/	Timeouts		t_{PReLU}/t	
512 Act. Nodes	Mean	Sd	Mean	Sd	Mean	Sd	Not Robust	Mean	Sd	Mean	Sd
PReLU ($\beta = 0.15$)	98.7%	0.0%	112.0	2.5	12.0	2.5	9.3	1.0	0.0	1.00	0.00
ReLU Pruning	**98.9%**	0.0%	96.4	2.4	6.8	1.7	14.2	21.8	0.7	0.04	0.01
ReLU Stability	98.7%	0.1%	116.2	2.9	8.8	2.9	13.2	**0.0**	0.0	**11.68**	2.23
ReLU	**98.9%**	0.0%	82.4	1.6	5.4	2.3	**15.3**	37.2	1.7	0.02	0.01
CIFAR-10 Conv.	Accuracy		Robust		Not Robust		Robust/	Timeouts		t_{PReLU}/t	
45 056 Act. Nodes	Mean	Sd	Mean	Sd	Mean	Sd	Not Robust	Mean	Sd	Mean	Sd
PReLU ($\beta = 0.15$)	84.8%	0.2%	54.8	2.5	52.2	3.3	**1.0**	**18.0**	1.1	**1.00**	0.00
ReLU Pruning	83.9%	0.4%	41.8	2.2	51.0	1.4	0.8	32.2	1.7	0.05	0.03
ReLU Stability	84.8%	0.9%	46.8	3.7	48.6	2.9	**1.0**	29.6	2.1	0.05	0.02
ReLU	**85.7%**	0.3%	42.0	1.4	51.2	1.9	0.8	31.8	3.1	0.05	0.03

accuracy was close to the PReLU Network with $\beta = 0.15$. For these experiments we could use only the MNIST and CIFAR-10 dataset since, to the best of our knowledge, ReLU stability training is not available for ResNet architectures, as used with the Tiny ImageNet dataset.

Training for stability is done using the loss introduced in (Xiao et al., 2019). In order to calculate the bounds used by the ReLU stability loss, we used improved interval propagation for MNIST networks as per (Xiao et al., 2019). For the larger CIFAR-10 networks, however, the same approach could not be used due to heavy memory and computational costs, as already reported in (Xiao et al., 2019). We therefore resorted to the less precise interval propagation as in the original paper (Xiao et al., 2019).

As shown in Table 2, the pruning produces consistent results for both the MNIST and the CIFAR-10 datasets. Pruning yields some improvement over the ReLU networks; however, the PReLU networks achieve a mean speed-up of around 20 times and have $39 - 100\%$ less timeouts than the pruned network. The networks trained with the ReLU stability loss perform extremely well for the small MNIST networks. The networks have a larger speed-up, fewer cases timed out and the robustness increases as indicated by the robust ratio. However, for the larger CIFAR-10 networks where naive interval propagation had to be used, the PReLU networks were verified about 20 times faster with 38% fewer timeouts and with a higher robust ratio. This is as expected, since the naive interval propagation produces less succinct bounds than the improved version and the precision of any interval propagation is reduced for deeper networks.

Overall, the results presented in Table 2 show that our approach produces gains that are competitive with previous methods, and outperform them when larger architectures are considered. As such, these results indicate that our method is successful in generating DNNs that are more amenable to SIP-based verification, only incurring in negligible extra training costs. Additional results on training times and converge rates are reported in the Appendix.

Integration with adversarial augmentation. We now show that the methods proposed here can also be combined

with adversarial training techniques, thus producing networks that are accurate, easy to verify but also robust. To this end, the training procedure used for the previous experiments was augmented with Fast Gradient Sign Method (FGSM) (Goodfellow et al., 2015) attacks.

The results, shown in Table 3, are consistent with the observations made above. In particular, PReLU networks combined with adversarial training outperformed all other approaches during verification, except for the MNIST network trained with ReLU stability. Again this is due to the MNIST ReLU stability network being trained with a more precise variant of interval propagation, while a more rudimentary variant was used for larger models to limit the computation overhead. Furthermore, we observe that the adversarially trained PReLU networks exhibit higher certified robustness than their ReLU counterpart (36% increase for MNIST and 30% for CIFAR-10)

In summary, these experiments demonstrate that the method here described can significantly improve the verifiability of networks without sacrificing their robustness, nor accuracy.

5. Related work

Training for verification. Closely related to our work is the training for ReLU stability method proposed in (Xiao et al., 2019). In this approach SIP is used during training to estimate lower and upper bounds for each ReLU node. The bounds are then used to formulate a regulariser that promotes stability of ReLU activations during training. The bounds are calculated via a refined version of SIP for small networks. However, naïve interval propagation is used for larger networks due to memory and computational requirements of interval propagation. Because of this, ReLU stability achieves strong results on smaller architectures in our experiments; however, its benefits diminish when dealing with larger architectures. On the contrary, our approach achieves consistent results across different model sizes.

Pruning has also been considered in (Xiao et al., 2019) and (Guidotti et al., 2020) as a way to ease network verification. Pruning is typically applied by adding a loss during training that encourages weight sparsity (weight pruning) or node sparsity (node pruning). In this work we compared against node pruning as this has a clear parallel to our proposed solution. While (Guidotti et al., 2020) showed that node pruning is particularly effective for verifiers based on MILP and Satisfiability Modulo Theory, in our experiments we could not replicate these gains in the context of SIP based verifiers, which are presently SoA in NN verification.

Certification approaches. A large body of work has considered the topic of certifiable robust training of neural networks. Several approaches use a dedicated loss function to encourage a smaller Lipschitz constant in the networks, thereby increasing ℓ_2 robustness (Leino et al., 2021; Huang et al., 2021; Tsuzuku et al., 2018). Other approaches add a loss term on the bounds calculated by symbolic interval propagation to encourage tighter bounds during training leading to increased ℓ_∞ robustness (Mirman et al., 2018; Wong et al., 2018; Wong & Kolter, 2018; Zhang et al., 2020).

Certifiable robust training typically requires computationally intensive estimations of the Lipschitz constant or SIP bounds at training time. This is in contrast with our objective to ease the task of verification, without requiring significant computational effort. Moreover, robust training in practice often leads to a significantly reduced accuracy ($10-50\%$ for a CIFAR-10 network (Huang et al., 2021)). This is also in clear contrast with our main goal to train verification-friendly networks without sacrificing accuracy. The differences highlighted above are to be expected: certifiable robust training aims at improving the robustness of the networks which is considered to have a trade-off with accuracy (D. Su et al., 2018). Finally, (Lyu et al., 2021) propose a novel parametric activation function to train verifiably robust networks with higher accuracy. While this work has some interesting similarities to the approach here presented, our goal is to improve the verifiability of networks. These two objectives have been shown to be independent as discussed in (Lee et al., 2021; Shi et al., 2021).

6. Discussion and Conclusions

We put forward a novel method to train DNNs such that they are accurate, but also easier to verify than standard ReLU-based architectures. We analysed different activation functions of the ReLU family and discussed the impact that they can have on SIP-based verification methods. We identified PReLUs as the most promising activation to generate verification-friendly networks. This enabled us also to propose a lightweight regularisation method that leverages the principle of co-design to achieve our objective. Our experiments with a SoA SIP verifier showed that DNNs trained by the method here described yield one order of magnitude reduction in verification overheads and 30-100% fewer timeouts compared to previous approaches. Finally, we showed that integration with adversarial training techniques lead to improvements in certified robustness up to 36% compared to ReLU, without sacrificing accuracy and verifiability.

One consideration with our approach is that its applicability is limited to ReLU networks only. This work focuses on ReLUs as these are the most widely studied activation functions in the verification literature. As such, we believe that improving verifiability of these models would maximise the impact of this work and would generate more interest in the verification community. However, this work also motivates further research into how results obtained here could be transferred to a wider class of neural architectures.

References

Bak, S., Tran, H., Hobbs, K., and Johnson, T. Improved geometric path enumeration for verifying relu neural networks. In *Proceedings of the 32nd International Conference on Computer Aided Verification (CAV20)*, volume 12224 of *LNCS*, pp. 66–96. Springer, 2020.

Bak, S., Liu, C., and Johnson, T. The second international verification of neural networks competition (vnncomp 2021): Summary and results. *arXiv preprint arXiv:2103.06624*, 2021.

Balunovic, M., Baader, M., Singh, G., Gehr, T., and Vechev, M. Certifying geometric robustness of neural networks. In *Advances in Neural Information Processing Systems (NeurIPS19)*, pp. 15313–15323. Curran Associates, Inc., 2019.

Batten, B., Kouvaros, P., Lomuscio, A., and Zheng, Y. Efficient neural network verification via layer-based semidefinite relaxations and linear cuts. In *Proceedings of the 30th International Joint Conference on Artificial Intelligence (IJCAI21)*, pp. 2184–2190. ijcai.org, 2021.

Botoeva, E., Kouvaros, P., Kronqvist, J., Lomuscio, A., and Misener, R. Efficient verification of neural networks via dependency analysis. In *Proceedings of the 34th AAAI Conference on Artificial Intelligence (AAAI20)*, pp. 3291–3299. AAAI Press, 2020.

Bunel, R., Lu, J., Turkaslan, I., Kohli, P., Torr, P., and Mudigonda, P. Branch and bound for piecewise linear neural network verification. *Journal of Machine Learning Research*, 21(42):1–39, 2020.

Cheng, C., Nuhrenberg, G., and Ruess, H. Verification of binarized neural networks. *arXiv preprint arXiv:1710.03107*, 2017.

D. Su, D., Zhang, H., Chen, H., Yi, J., Chen, P.-Y., and Gao, Y. Is robustness the cost of accuracy?–a comprehensive study on the robustness of 18 deep image classification models. In *Proceedings of the 15th European Conference on Computer Vision (ECCV18)*, pp. 631–648, 2018.

Dvijotham, K., Stanforth, R., Gowal, S., Mann, T., and Kohli, P. A dual approach to scalable verification of deep networks. *arXiv preprint arXiv:1803.06567*, 2018.

Ehlers, R. Formal verification of piece-wise linear feed-forward neural networks. In *Proceedings of the 15th International Symposium on Automated Technology for Verification and Analysis (ATVA17)*, volume 10482 of *Lecture Notes in Computer Science*, pp. 269–286. Springer, 2017.

Fazlyab, M., Morari, M., and Pappas, G. J. Safety verification and robustness analysis of neural networks via quadratic constraints and semidefinite programming. *IEEE Transactions on Automatic Control*, 2020. doi: 10.1109/TAC.2020.3046193.

Fremont, D., Chiu, J., Margineantu, D., Osipychev, D., and Seshia, S. Formal analysis and redesign of a neural network-based aircraft taxiing system with verifai. In *Proceedings of the 32nd International Conference on Computer Aided Verification (CAV19)*, volume 12224 of *Lecture Notes in Computer Science*, pp. 122–134. Springer, 2020.

Goodfellow, A., Bengio, Y., and Courville, A. *Deep learning*, volume 1. MIT press Cambridge, 2016.

Goodfellow, I., Shlens, J., and Szegedy, C. Explaining and harnessing adversarial examples. In *Proceedings of the 3rd International Conference on Learning Representations ICLR15*, 2015.

Guidotti, D., Leofante, F., Pulina, L., and Tacchella, A. Verification of neural networks: Enhancing scalability through pruning. In *Proceedings of the 24th European Conference on Artificial Intelligence (ECAI20)*, pp. 2505–2512. IOS Press, 2020.

Henriksen, P. and Lomuscio, A. Efficient neural network verification via adaptive refinement and adversarial search. In *Proceedings of the 24th European Conference on Artificial Intelligence (ECAI20)*, pp. 2513–2520. IOS Press, 2020.

Henriksen, P. and Lomuscio, A. DEEPSPLIT: an efficient splitting method for neural network verification via indirect effect analysis. In *Proceedings of the 30th International Joint Conference on Artificial Intelligence (IJCAI21)*, pp. 2549–2555. ijcai.org, 2021.

Henriksen, P., Hammernik, K., Rueckert, D., and Lomuscio, A. Bias field robustness verification of large neural image classifiers. In *Proceedings of the 32nd British Machine Vision Conference (BMVC21)*, 2021.

Huang, Y., hang, H., Shi, Y., Kolter, J., and Anandkumar, A. Training certifiably robust neural networks with efficient local lipschitz bounds. In *Advances in Neural Information Processing Systems (NeurIPS21)*, 2021.

Katz, G., Barrett, C., Dill, D., Julian, K., and Kochenderfer, M. Reluplex: An efficient SMT solver for verifying deep neural networks. In *Proceedings of the 29th International Conference on Computer Aided Verification (CAV17)*, volume 10426 of *Lecture Notes in Computer Science*, pp. 97–117. Springer, 2017.

Katz, G., Huang, D., Ibeling, D., Julian, K., Lazarus, C., Lim, R., Shah, P., Thakoor, S., Wu, H., Zeljic, A., Dill, D., Kochenderfer, M., and Barrett, C. The marabou framework for verification and analysis of deep neural networks. In *Proceedings of the 31st International Conference on Computer Aided Verification (CAV19)*, pp. 443–452, 2019.

Kingma, D. and Ba, J. Adam: A method for stochastic optimization. In *Proceedings of the 3rd International Conference on Learning Representations (ICLR15)*, 2015.

Kouvaros, P., Kyono, T., Leofante, F., Lomuscio, A., Margineantu, D., Osipychev, D., and Zheng, Y. Formal analysis of neural network-based systems in the aircraft domain. In *Proceedings of the 24th International Symposium on Formal Methods (FM21)*, volume 13047 of *Lecture Notes in Computer Science*, pp. 730–740. Springer, 2021.

Krizhevsky, A., Nair, V., and Hinton, G. The cifar-10 dataset. http://www.cs.toronto.edu/kriz/cifar.html, 2014.

LeCun, Y. The mnist database of handwritten digits. *http://yann. lecun. com/exdb/mnist/*, 1998.

Lee, S., Lee, W., Park, J., and Lee, J. Loss landscape matters: Training certifiably robust models with favorable loss landscape. 2021. https://openreview.net/forum?id=lvXLfNeCQdK.

Lee, Y. and Yang, X. Tiny imagenet visual recognition challenge. 2015. http://cs231n.stanford.edu/reports/2015/pdfs/yle_project.pdf.

Leino, K., Wang, Z., and Fredrikson, M. Globally-robust neural networks. In *Proceedings of the 38th International Conference on Machine Learning (ICML21)*, volume 139 of *Proceedings of Machine Learning Research*, pp. 6212–6222. PMLR, 2021.

Lyu, Z., Guo, M., Wu, T., Xu, G., Zhang, K., and Lin, D. Towards evaluating and training verifiably robust neural networks. In *CVPR'21*, pp. 4308–4317. Computer Vision Foundation / IEEE, 2021.

Mirman, M., Gehr, T., and Vechev, M. Differentiable abstract interpretation for provably robust neural networks. In *Proceedings of the 35th International Conference on Machine Learning (ICML18)*, volume 80 of *Proceedings of Machine Learning Research*, pp. 3575–3583. PMLR, 2018.

Raghunathan, A., Steinhardt, J., and Liang, P. Semidefinite relaxations for certifying robustness to adversarial examples. *arXiv preprint arXiv:1811.01057*, 2018.

Shi, Z., Wang, Y., Zhang, H., Yi, J., and Hsieh, C. Fast certified robust training via better initialization and shorter warmup. In *Advances in Neural Information Processing Systems (NeurIPS21)*, 2021.

Singh, G., Gehr, T., Püschel, M., and Vechev, M. An abstract domain for certifying neural networks. *Proceedings of the ACM on Programming Languages*, 3(POPL):41, 2019.

Tjandraatmadja, C., Anderson, R., Huchette, J., Ma, W., PATEL, K., and Vielma, J. The convex relaxation barrier, revisited: Tightened single-neuron relaxations for neural network verification. *Advances in Neural Information Processing Systems (NeurIPS20)*, 2020.

Tjeng, V., Xiao, K., and Tedrake, R. Evaluating robustness of neural networks with mixed integer programming. In *Proceedings of the 7th International Conference on Learning Representations (ICLR19)*, 2019.

Tran, H., Yang, X., Lopez, D. M., Musau, P., Nguyen, L., Xiang, W., Bak, S., and Johnson, T. NNV: the neural network verification tool for deep neural networks and learning-enabled cyber-physical systems. In *CAV20*, volume 12224 of *Lecture Notes in Computer Science*, pp. 3–17. Springer, 2020.

Tsuzuku, Y., Sato, I., and Sugiyama, M. Lipschitz-margin training: Scalable certification of perturbation invariance for deep neural networks. In *Advances in Neural Information Processing Systems (NeurIPS18)*, pp. 6542–6551, 2018.

Wang, S., Pei, K., Whitehouse, J., Yang, J., and Jana, S. Formal security analysis of neural networks using symbolic intervals. In *Proceedings of the 27th USENIX Security Symposium (USENIX18)*, 2018a.

Wang, S., Pei, K., Whitehouse, J., Yang, J., and Jana, S. Efficient formal safety analysis of neural networks. In *Advances in Neural Information Processing Systems (NeurIPS18)*, pp. 6367–6377, 2018b.

Wang, S., Zhang, H., Xu, K., Lin, X., Jana, S., Hsieh, C., and Kolter, J. Beta-crown: Efficient bound propagation with per-neuron split constraints for complete and incomplete neural network verification. *arXiv preprint arXiv:2103.06624*, 2021.

Wong, E. and Kolter, Z. Provable defenses against adversarial examples via the convex outer adversarial polytope. In *Proceedings of the 35th International Conference on Machine Learning (ICML18)*, pp. 5286–5295. PMLR, 2018.

Wong, E., Schmidt, F., Metzen, J., and Kolter, J. Scaling provable adversarial defenses. In *Advances in Neural Information Processing Systems (NeurIPS18)*, pp. 8410–8419, 2018.

Xiao, K., Tjeng, V., Shafiullah, N., and Madry, A. Training for faster adversarial robustness verification via inducing relu stability. In *Proceedings of the 7th International Conference on Learning Representations (ICLR19)*, pp. 1–20. OpenReview.net, 2019.

Xu, B., Wang, N., Chen, T., and Li, M. Empirical evaluation of rectified activations in convolutional network. *arXiv preprint arXiv:1505.00853*, 2015.

Zhang, H., Chen, H., Xiao, C., Gowal, S., Stanforth, R., Li, B., Boning, D., and Hsieh, C. Towards stable and efficient training of verifiably robust neural networks. In *Proceedings of the 8th International Conference on Learning Representations (ICLR20)*. OpenReview.net, 2020.