Linearity Grafting: Relaxed Neuron Pruning Helps Certifiable Robustness

Tianlong Chen [*1] **Huan Zhang** [*2] **Zhenyu Zhang** [1] **Shiyu Chang** [3]
Sijia Liu [4 5] **Pin-Yu Chen** [5 6] **Zhangyang Wang** [1]

Abstract

Certifiable robustness is a highly desirable property for adopting deep neural networks (DNNs) in safety-critical scenarios, but often demands tedious computations to establish. The main hurdle lies in the massive amount of non-linearity in large DNNs. To trade off the DNN expressiveness (which calls for more non-linearity) and robustness certification scalability (which prefers more linearity), we propose a novel solution to strategically manipulate neurons, by *"grafting"* appropriate levels of linearity. The core of our proposal is to first linearize insignificant ReLU neurons, to eliminate the non-linear components that are both redundant for DNN performance and harmful to its certification. We then optimize the associated slopes and intercepts of the replaced linear activations for restoring model performance while maintaining certifiability. Hence, typical neuron pruning could be viewed as a *special case* of grafting a linear function of the fixed zero slopes and intercept, that might overly restrict the network flexibility and sacrifice its performance. Extensive experiments on multiple datasets and network backbones show that our ***linearity grafting*** can (1) effectively tighten certified bounds; (2) achieve competitive certifiable robustness *without certified robust training* (i.e., over 30% improvements on CIFAR-10 models); and (3) scale up complete verification to large adversarially trained models with 17M parameters. Codes are available at `https://github.com/VITA-Group/Linearity-Grafting`.

[1]University of Texas at Austin [2]Carnegie Mellon University [3]University of California, Santa Barbara [4]Michigan State University [5]MIT-IBM Watson AI Lab [6]IBM Research. Corresponding to: Zhangyang Wang <atlaswang@utexas.edu>. [*]Equal contribution.

Proceedings of the 39^{th} International Conference on Machine Learning, Baltimore, Maryland, USA, PMLR 162, 2022. Copyright 2022 by the author(s).

1. Introduction

Despite the prevailing successes of deep neural networks (DNNs), they remain vulnerable to adversarial examples (Szegedy et al., 2013). Therefore, certifying whether a DNN is provably robust under all bounded input perturbations are crucial for adopting DNNs in high-stake and safety-critical applications. To obtain non-trivial certified robust accuracy, certified adversarial defense (Raghunathan et al., 2018a; Wong & Kolter, 2018; Mirman et al., 2018; Gowal et al., 2018; Zhang et al., 2019b) which minimize a certifiable loss during training, and robustness verification methods (Gehr et al., 2018; Dvijotham et al., 2018; Zhang et al., 2018; Wang et al., 2021) which compute tight certified bounds for post-training networks serve as the most effective two approaches.

However, several pain points persist for these two mechanisms: (i) As DNNs grow larger (e.g., dozens of layers), (complete) verification can be extremely time-consuming and computationally expensive due to the massive non-linearity in activation functions like ReLU. For ReLU networks under input perturbations, the ReLU neurons whose inputs may cross zero (referred to as "unstable" neurons) are often the key factor to determine the difficulty of verification. Current solutions like linear relaxations (Zhang et al., 2018; Singh et al., 2019b), semidefinite relaxations (Raghunathan et al., 2018b) or Branch and Bound (BaB) (Bunel et al., 2017; Wang et al., 2021) suffer from either trivially loose bounds or exponentially increased complexity for large-scale networks. (ii) Many certified robust training methods (Mirman et al., 2018; Gowal et al., 2018; Wong et al., 2018; Balunovic & Vechev, 2019) tend to reduce non-linearity ("unstable" ReLU neurons) to improve certifiable robustness. For example, many certified defense approaches tend increase the number of "inactive" neurons (ReLU neurons with negative inputs and zero output) for tightening bounds (Shi et al., 2021). This often leads to reduced standard accuracy, and in order to stably train a network with certified defense, a much longer training schedule is often required: e.g., Xu et al. (2020a) used $2,000$ epochs and Zhang et al. (2019b) used $3,200$ epochs for training a convolutional neural network on CIFAR-10 to state-of-the-art certified performance, much longer than conventionally adversarial training ($100 \sim 200$ epochs) (Madry et al., 2018).

To address these pain points, this paper proposes a brand new alternative, *linearity grafting*, to remarkably alleviate the burden of certification, by strategically operating neurons to control an appropriate level of non-linearity on a pretrained network. Linearity grafting is inspired by neuron pruning, which strategically removes a large portion of neurons in a network while maintaining its performance. However since we aim to enhance certifiability rather than reducing network size or computation, we can *relax the typical pruning paradigm* by exploiting the fact that certification is easy for linear neurons. Specifically, linearity grafting first treats a pre-trained model using adversarial training as the starting point which has good empirical robustness but usually undergoes poor certifiable robustness. Then, it replaces unstable yet insignificant ReLU neurons with a linear activation function, balancing the certification scalability and network expressiveness. Lastly, the slope and intercept are further tuned to maintain competitive classification performance thanks to its enhanced representation power compared to inactive (completely pruned) ReLU neurons. Note that vanilla neuron pruning is a special case of our proposal, where it grafts a fixed linear function of zero to unstable neurons, yet often ends up with worse performance due to over-restriction. Our contributions can be summarized as follows:

- We pioneer a thorough investigation to reveal that introducing appropriate linearity benefits certifiable robustness in multiple aspects: (1) substantially shrinking the bounds and reducing the number of unstable ReLU neurons; (2) improving certification scalability and enabling complete verification on large-scale networks; (3) obtaining competitive certified accuracy in a more efficient manner without explicitly certified training.
- We propose a new algorithm, *grafting*, for trimming down the unnecessary non-linearity for networks under verification. It linearizes the unstable yet insignificant neurons by replacing the ReLU with a linear activation function, whose slope and intercept are subsequently optimized to achieve better model performance.
- Extensive experiments across diverse network architectures on MNIST, SVHN, and CIFAR-10, validate our proposal by demonstrating the superior certified accuracies (up to 82.30% improvements) *without certified robust training*. Furthermore, our grafted large-scale networks with $\geq$ 17M parameters reach competitive certified accuracy (32.30% at $\epsilon = 8/255$) without using certified defense training.

2. Related Work

Network Verification. Neural network verification aims to formally prove or disprove desired properties of a neural network, and it becomes increasingly important to safety-crucial scenarios (Katz et al., 2017; 2019). Given sufficient time, a *complete* verifier gives a definite "yes/no" answer; on the other hand, an incomplete verifier solves a relaxed problem and may produce "don't know". Complete verification generally produces tighter bounds at the expense of more computations, compared to its incomplete counterpart.

Early complete verifiers are limited to very small-scale problems and relied on costly solvers like MILP (Tjeng et al., 2017; Dutta et al., 2017) or SMT (Katz et al., 2017; Ehlers, 2017; Huang et al., 2017). On the other hand, branch and bound (BaB) based complete verifiers use an incomplete verifier as a sub-procedure and conduct branching on ReLU neurons (Bunel et al., 2017; Wang et al., 2018b; Lu & Kumar, 2019; Botoeva et al., 2020) or model input (Wang et al., 2018c; Rubies-Royo et al., 2019; Anderson et al., 2019; Bunel et al., 2017). Recently, a series of effective BaB verifiers are proposed to use efficient iterative solvers or bound propagation methods on GPUs as the alternative for LP solvers. For example, BaDNB (De Palma et al., 2021) designs a new filtered smart branching (FSB) and combines it with Lagrangian decomposition (Bunel et al., 2020) to enhance verification performance. Xu et al. (2020b) use an optimizable bound propagation method (α-CROWN) as a massively paralleled incomplete solver on GPUs and an LP solver for completeness checking. β-CROWN (Wang et al., 2021) completely eliminate the dependency on an LP solver with optimizable constraints, greatly boosting verification performance and efficiency.

An incomplete verifier is usually weaker but faster than complete verifiers. They typically rely on duality (Dvijotham et al., 2018; Wong & Kolter, 2018), linear relaxations (Wang et al., 2018b; Zhang et al., 2018; Salman et al., 2019) or semidefinite relaxations (Raghunathan et al., 2018b; Dathathri et al., 2020). Cheap incomplete verifiers can be applied at training time (Wong & Kolter, 2018; Wang et al., 2018a; Mirman et al., 2018; Wong et al., 2018; Gowal et al., 2018; Balunovic & Vechev, 2019; Shi et al., 2021) for certified defense mechanisms. Our work achieves certified robustness without relying on a certified defense, but with the help of a more powerful complete verifier on an adversarially trained model with post-training linearity grafting.

Pruning. Pruning (LeCun et al., 1990; Han et al., 2015a) as one of the effective compression approaches eliminates the redundancy in over-parameterized neural networks, achieving storage and computational savings with nearly unimpaired performance. According to the granularity of removed components, pruning can be roughly categorized into two groups: (1) Unstructured pruning, which zeroes out insignificant parameters based on some heuristics like weight magnitude (Han et al., 2015a;b), gradient (Molchanov et al., 2019), or hessian (LeCun et al., 1990) statistics. It usually leads to competitive performance but is hard to be accelerated due to irregular sparsities. (2) Structural pruning, which discards sub-structures (e.g., channels (Liu et al., 2017; Zhou et al., 2016; He et al., 2017) or layers (Wang et al., 2018e; Wu et al., 2018; Zhang et al., 2019a)) based on optimized importance scores or heuristics. It produces hardware-friendly subnetworks at the cost of moderate performance degradation. Normally, structural pruning also brings the reduction of intermediate activation (or neuron) dimension. Dhillon et al. (2018); Ye et al. (2020) particularly devote themselves to trimming down the number of neurons for empirical robustness and network acceleration.

Pruning and Robustness. On one hand, Gui et al. (2019); Ye et al. (2019); Sehwag et al. (2019); Jordao & Pedrini (2021); Fu et al. (2021) pursue empirical robust and efficient subnetworks that can be deployed in security-critical yet resource-limited platforms. Sehwag et al. (2020) also consider the trade-off between certified robustness and efficiency by learning sparse weights. On the other hand, Wang et al. (2018d); Gao et al. (2017); Dhillon et al. (2018) treat model compression as an empirical defense mechanism. Xiao et al. (2018) for the first time utilize *weight* sparsity to speed up the certification of multi-layer perceptron (MLP) with LP or MILP verifiers. However, sparse weights do not necessarily result in zero or stable activation, especially for convolutional neural networks. Han et al. (2021) conduct certified defense against adversarial patches by removing related superficial neurons.

Our work is fundamentally different from existing literature by recognizing **two important facts**: *(i) linearity plays a central role in the difficulty of robustness certification; and (ii) sparsity is only a special case of linearity*. Our proposed grafting focuses on linearizing unstable and insignificant neurons. Thus, it directly reduces ReLU instability, obtains tight verification bound, and in the meanwhile ensures sufficient flexibility for restoring model performance. As a result, we improve robustness verification in terms of certified robust accuracy, speed, and scalability.

3. Preliminaries

3.1. The Neural Network Verification Problem

Let $x \in \mathbb{R}^{d_0}$ be the input sample of a L-layer deep neural network $f : \mathbb{R}^{d_0} \to \mathbb{R}$, and f is parameterized by weights $\mathbf{W}^{(i)} \in \mathbb{R}^{d_i \times d_{i-1}}$ and biases $\mathbf{b}^{(i)} \in \mathbb{R}^{d_i}$, $i \in \{1, \cdots, L\}$, respectively. Then, we have $f(x) = z^{(L)}(x)$, $z^{(i)}(x) = \mathbf{W}^{(i)}\hat{z}^{(i-1)}(x)+\mathbf{b}^{(i)}$, $\hat{z}^{(i)}(x) = \mathcal{A}(z^{(i)}(x))$, and $\hat{z}^{(0)}(x) = x$, where $\mathcal{A}$ is activation function, $z_j^{(i)}$ and $\hat{z}_j^{(i)}$ represents the pre-activation and post-activation values of the j-the neuron in the i-th layer. The neural network verification can be formulated as:

$$\begin{aligned}&\min f(x) := z^{(L)}(x)\\ &\mathbf{s.t.}\ z^{(i)} = \mathbf{W}^{(i)}\hat{z}^{(i-1)} + \mathbf{b}^{(i)}, \hat{z}^{(i)} = \mathcal{A}(z^{(i)})\\ &\hat{z}^{(0)} = x,\ x \in \mathcal{C}\end{aligned} \tag{1}$$

where the set $\mathcal{C}$ denotes the allowed input region, and the goal is to find the lower bound of $f(x)$ given $x \in \mathcal{C}$. Without loss of generality, $\mathcal{C}$ can be an ℓ_∞ ball with a radius of ϵ around a data example x_0, i.e., $\mathcal{C} = \{x| \|x - x_0\|_\infty \le \epsilon\}$. We consider the canonical specification $f(x) > 0$: if we can prove that $f(x) > 0, \forall x \in \mathcal{C}$, we say $f(x)$ is verified. Other "specifications" like the margin between logits in practical settings can be often seen as an output layer of networks, merged into $f(x)$ during verification, and turned back to the canonical specification (Wang et al., 2021).

Unstable neurons. The most challenging part of verification is the non-linearity of activation functions $\mathcal{A}$. Specifically, given $\mathcal{A}(z_j^{(i)}) := \mathrm{ReLU}(z_j^{(i)}) = \max(0, z_j^{(i)})$, we denote its intermediate layer bounds as $\mathbf{l}_j^{(i)} \le z_j^{(i)} \le \mathbf{u}_j^{(i)}$, which bound the input of this specific ReLU neurons given $x \in \mathcal{C}$. Given intermediate layer bounds, each ReLU neuron (i.e., activation value) can be grouped into two categories: (1) if $\mathbf{l}_j^{(i)} \ge 0$ or $\mathbf{u}_j^{(i)} \le 0$, ReLU lies in linear active $(\hat{z}_j^{(i)} = z_j^{(i)})$ or inactive $(\hat{z}_j^{(i)} = 0)$ regions ; (2) if $\mathbf{l}_j^{(i)} \le 0 \le \mathbf{u}_j^{(i)}$, we call this ReLU neuron as an *unstable neuron*, which usually places obstacles to certification.

3.2. Branch and Bound based Complete Verifier

A large portion of complete verifiers (Bunel et al., 2017; Henriksen & Lomuscio, 2020; Wang et al., 2021) adopt the Branch and Bound (BaB) method. They first divide the domain of the verification problem $\mathcal{C}$ into two subdomains $\mathcal{C}_1 = \{x \in \mathcal{C}, z_j^{(i)} \ge 0\}$ and $\mathcal{C}_2 = \{x \in \mathcal{C}, z_j^{(i)} \le 0\}$, according to an unstable ReLU neuron $z_j^{(i)}$ that now becomes linear in each subdomain. Then, cheap incomplete verifiers are applied to estimate the *lower bound* of the objective (1). for each subdomain under some relaxations. If subdomain $\mathcal{C}_i$'s lower bound is greater than 0, $\mathcal{C}_i$ is verified; otherwise, another unstable ReLU neuron will be further split over domain $\mathcal{C}_i$, until all subdomains are verified. Therefore, the amount of non-linearity or unstable neurons directly determines the computation and memory complexity of complete verifiers. When each subdomain is solved using an incomplete verifier, the tightness of the lower bound also heavily depends on the number of unstable ReLU neurons - too many unstable neurons may lead to vacuous bounds and zero verified accuracy. Thus, the goal of our grafting linearity is to scale up verification by reducing unstable ReLU neurons that are insignificant for model clean performance.

4. Grafting for Certified Verification

This section presents the detailed processes of linearity grafting: (i) starting from an empirical robust DNN; (ii) identifying and linearizing the unstable & insignificant neurons; (iii) further optimizing the parameters in grafted linear acti-

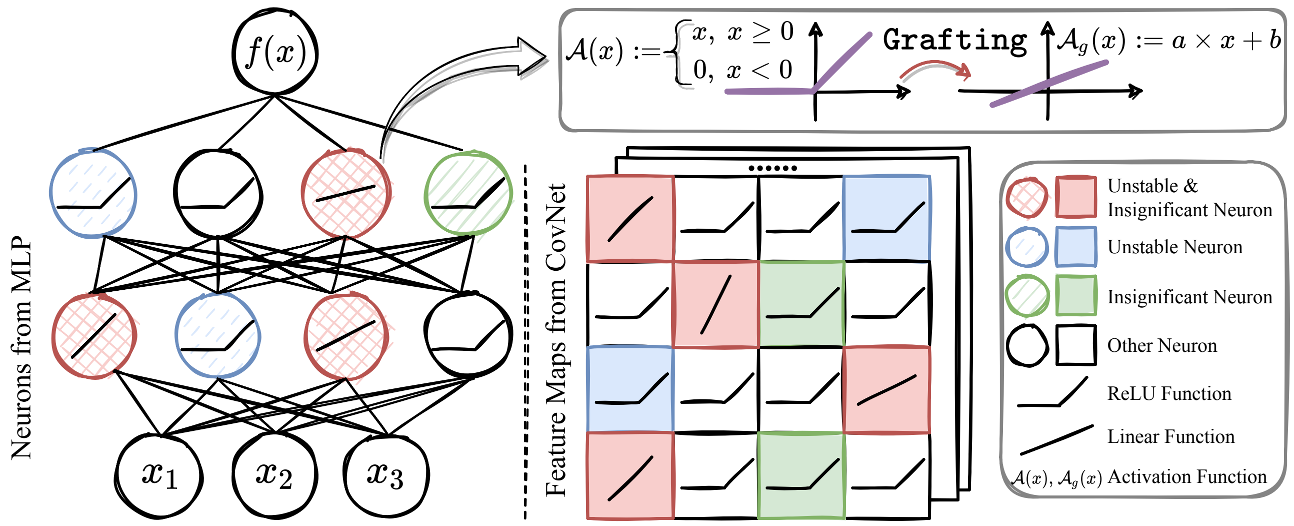
$f(x)$
$\mathcal{A}(x) := \begin{cases} x, & x \geq 0 \\ 0, & x < 0 \end{cases}$
Grafting
$\mathcal{A}_g(x) := a \times x + b$
Neurons from MLP
Feature Maps from CovNet
x_1
x_2
x_3
Unstable & Insignificant Neuron
Unstable Neuron
Insignificant Neuron
Other Neuron
ReLU Function
Linear Function
$\mathcal{A}(x)$, $\mathcal{A}_g(x)$ Activation Function

Figure 1. The overall illustration of our proposed **`linearity grafting`**, which linearizes the piece-wise activation function of unstable and insignificant neurons with a linear function (i.e., $a \times x + b$). (*Left*) and (*Right*) show demos of applying grafting to MLP and ConvNet, where unstable & insignificant neurons (e.g., red cycles or blocks) are our main focus.

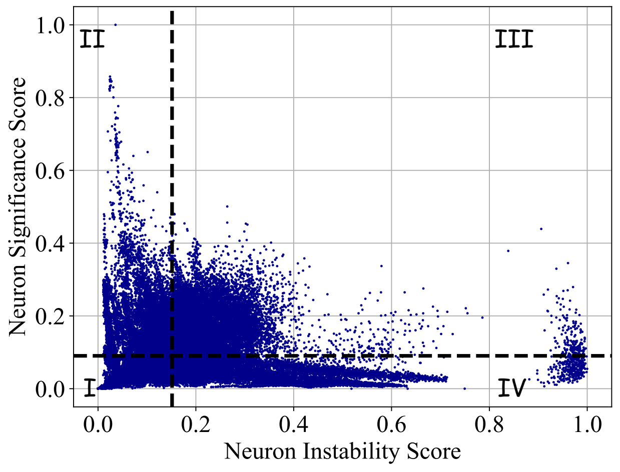

Figure 2. Normalized neurons' significance scores ($r_s^{(i)}$) over instability scores ($r_u^{(i)}$). Significance scores are calculated based on the absolute magnitude of activation gradients, and instability scores are computed by a SOTA verifier, α,β-CROWN. Intuitively, regions (I, II, III, IV) indicate (*insignificant and stable, significant and stable, significant and unstable, insignificant and unstable*) neurons.

vation functions; (iv) performing complete verification on grafted NNs. We detail each step below.

▷ **Robustify a DNN using adversarial training as the starting point.** We start from an empirical robust model from cheap adversarial defenses, which might not be certifiable especially when the model is large. We aim to improve its certifiable robustness without explicit certified defense robust training, which is usually slow and can hurt model performance. Specifically, a fast adversarial training approach (Andriushchenko & Flammarion, 2020) with gradient alignment regularization is adopted in our case, whose training loss is:

$$\mathcal{L} := \mathcal{L}_{\text{FAT}} + \lambda \times \mathcal{R}_{\text{GA}}, \tag{2}$$

where $\mathcal{L}_{\text{FAT}}$ is the loss of fast adversarial training (Wong et al., 2020) with GradAlign regularizer $\mathcal{R}_{\text{GA}}$; λ controls the portion of regularization, which is 0.2 in our case following the default choice in Andriushchenko & Flammarion (2020).

▷ **Identify insignificant and unstable neurons.** Given a trained DNN, our goal is to reduce the ReLU instability (i.e., unstable neurons), gain certifiable robustness, and preserve sound generalization. To locate qualified neuron candidates, we design the selection pipeline as below:

❶ Rank all neurons according to the number of images for which it is unstable, normalize the rank and treat it as the neuron instability score $r_u^{(i)} \in [0, 1]$, where i is the neuron index.

❷ Compute the importance of each neuron via certain heuristics or optimized scores (e.g., the magnitude of activation's gradient), rank and normalize them to obtain the neuron significance score $r_s^{(i)} \in [0, 1]$, where i is the neuron index.

❸ Based on some criteria like $\text{argmax}_i \gamma \times r_u^{(i)} - r_s^{(i)}$ to identify the insignificant and unstable neurons, where γ adjusts the portion of instability scores. If $\gamma \to 0$ or ∞, it chooses neurons purely on significant or instability scores. In our case, we decay γ from 2 to 0, along with the total number of grafted neurons. For example, to graft 20% neurons, we select every 5% neurons with $\gamma = 2 \to 1.5 \to 1 \to 0$, respectively.

Figure 2 demonstrates the distributions of $r_u^{(i)}$ and $r_s^{(i)}$, which can be roughly divided into four regions (I, II, III,

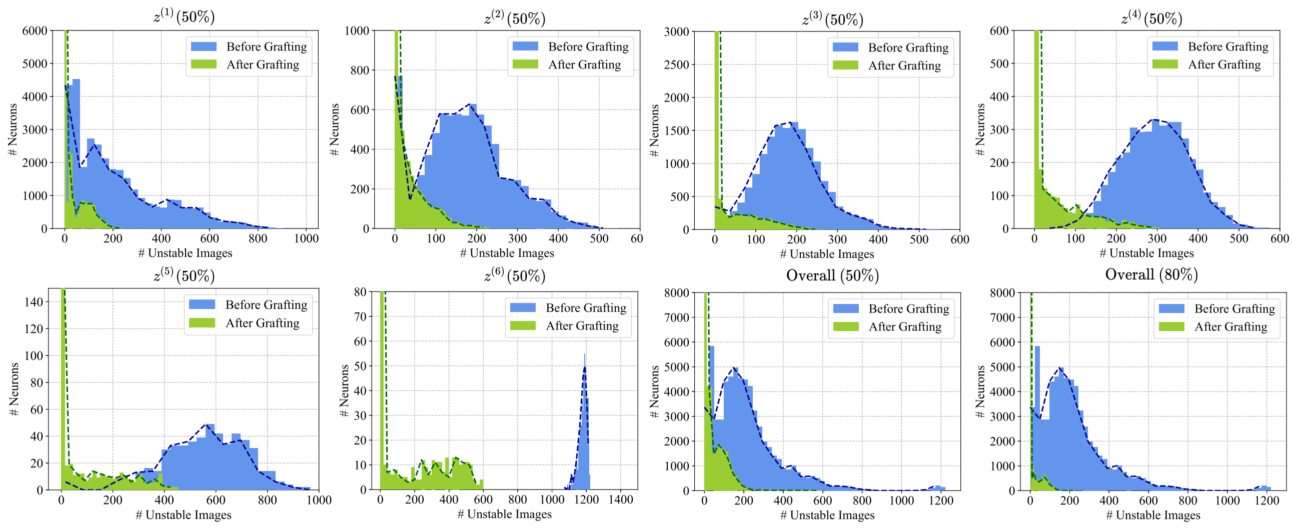

Figure 3. Layer-wise ($z^{(i)}$) and overall unstable neuron distribution of the 7-layer ConvBig on CIFAR-10, before and after performing grafting on 50% or 80% neurons. On these figures, a bar located at m unstable images (x-axis) with a height of n neurons (y-axis) means that n neurons are unstable for m images in the test set.

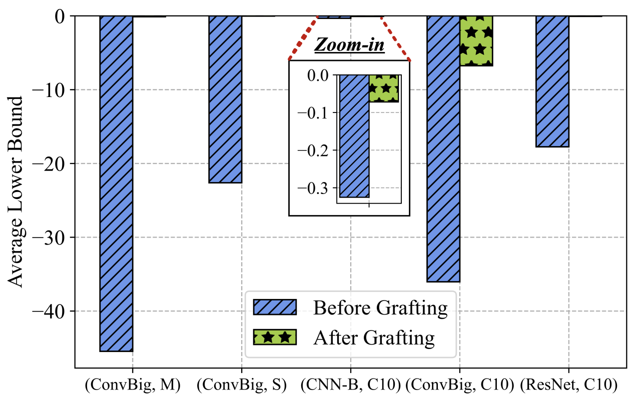

Figure 4. Average lower bounds after BaB branching of verification (Wang et al., 2021) for models before and after grafting 50% neurons on 5 models. Bounds are produced by α,β-CROWN. A very negative lower bound often means the bound is vacuous due to the large amounts of unstable neurons.

IV), and each region has a quarter of neurons. Region IV is our main focus, implying insignificant and unstable neurons. An investigation of different heuristics for computing importance scores is provided in Sec. 5.3.

▷ **Linearize and tune the grafted activation functions.** As the last step of linearity grafting, we linearize the selected neurons (i.e., red cycles or blocks in Figure 1) by replacing the non-linear activation function (e.g., ReLU $\mathcal{A}(x)$) with a parameterized linear function $\mathcal{A}_g(x) := a \times x + b$, where a and b are trainable slope and intercept respectively. Note that each neuron has its own associated a and b. Since the exorbitant non-linearity (or unstable ReLU neurons) causes burdensome computation and limits the scalability of verification, our grafting directly discards redundant non-linearity and injects learnable linearity, encouraging DNN to be more amenable for verification. Furthermore, contrary to typical certified defense where the inactive neurons may dominate in the network (Shi et al., 2021), the grafted neurons are never forced to be inactive and can be optimized towards better model performance.

▷ **Robustness verification with a complete verifier.** After linearity grafting, the robustness of our model is verified using a complete verifier. Unlike cheap verification methods used in certified defense such as IBP (Gowal et al., 2018; Mirman et al., 2018) where the models have to be trained to adapt to a very weak certification method which may impose strong constraints on model expressiveness, a complete verifier is much more powerful and allows the model to be more flexible for a better trade-off between verified accuracy and standard accuracy. We choose the a state-of-the-art (SOTA) complete verifier, α,β-CROWN (Zhang et al., 2018; Xu et al., 2020a; Wang et al., 2021), as our default certification method, with added support on a customized activation function that can be selected from either a ReLU function (neurons not grafted) or a linear function (grafted neurons). We find that SOTA complete verifiers can produce quite competitive results when an sufficient number of unstable neurons are grafted, as we will show in Section 5.

5. Experiments

Datasets and architectures. Our experiments are conducted on three representative datasets in adversarial robustness and verification literature, MNIST (Deng, 2012), SVHN (Netzer et al., 2011) and CIFAR-10 (Krizhevsky & Hinton, 2009). For MNSIT and SVHN, we consider a 7-layer convolutional neural network (ConvBig) from Mirman et al. (2018); Singh et al. (2019b). For CIFAR-10, we adopt

Table 1. **Unstable neuron ratio (UNR $\%$), verified accuracy (VA $\%$), standard accuracy (SA $\%$), PGD-100 robust accuracy (RA $\%$), and average time (s)** of FAT trained models w./w.o. grafting on MNIST, SVHN, and CIFAR-10. α,β-CROWN, a SOTA complete verifier is used to compute VA. The target ℓ_∞ norm perturbation is $\epsilon = \frac{2}{255}$ except for MNIST. "OOM" indicates that DNNs have too many unstable neurons and the verifier is unable to load it with 48 GB GPU memory, leading to "∞" verification time and a null VA ("-").

FAT ($\epsilon = \frac{2}{255}$)	(ConvBig, MNIST w. $\epsilon = 0.1$)					(ConvBig, SVHN)					(CNN-B, CIFAR-10)				
	UNR	VA	SA	RA	Time	UNR	VA	SA	RA	Time	UNR	VA	SA	RA	Time
Baseline	31.27	0.10	99.29	97.14	262.11	10.78	16.70	89.71	75.74	218.49	15.85	37.40	79.95	62.23	127.50
SAP (Dhillon et al., 2018) (50%)	7.38	4.20	99.22	96.34	292.94	5.65	25.90	89.85	76.03	195.87	6.27	47.30	75.10	58.01	58.98
GAP[†] (Ye et al., 2020) (50%)	17.29	3.50	99.19	96.46	295.21	6.14	26.20	90.09	77.28	195.78	10.22	42.50	79.05	61.81	103.03
Hydra[‡] (Sehwag et al., 2020) (50%)	15.39	12.70	98.90	95.22	269.71	5.04	26.60	81.28	62.92	172.98	6.28	44.40	72.99	55.55	59.99
Random Grafting (50%)	17.16	12.00	98.93	95.38	273.94	6.13	37.40	87.37	73.27	150.23	9.07	42.50	75.02	57.19	83.25
Grafting (50%)	5.85	82.30	98.68	92.73	40.21	3.11	57.80	78.75	63.90	16.68	5.36	50.40	74.08	58.76	39.32
Grafting (30%)	10.43	59.40	99.13	95.24	137.40	5.45	56.80	80.71	66.05	31.76	7.15	49.00	77.10	60.87	64.80
Grafting (80%)	4.04	82.40	98.63	92.71	39.64	1.63	58.70	78.56	63.91	12.93	1.87	44.40	61.20	48.34	15.25

FAT ($\epsilon = \frac{2}{255}$)	(ResNet-4B, CIFAR-10)					(ConvBig, CIFAR-10)					(ConvHuge, CIFAR-10)				
	UNR	VA	SA	RA	Time	UNR	VA	SA	RA	Time	UNR	VA	SA	RA	Time
Baseline	19.94	0.80	76.69	60.14	45.56	17.75	1.30	84.90	68.10	121.61	OOM	-	90.68	73.57	∞
SAP (Dhillon et al., 2018) (50%)	6.18	21.70	49.03	38.30	137.77	5.54	25.80	65.08	50.45	156.28	8.52	2.00	80.29	60.29	181.06
GAP[†] (Ye et al., 2020) (50%)	13.67	5.10	68.42	53.43	239.14	10.97	1.10	81.91	64.50	190.42	7.43	1.00	86.38	67.91	111.77
Hydra[‡] (Sehwag et al., 2020) (50%)	9.52	15.10	42.01	31.27	162.34	11.10	1.10	67.97	47.77	297.19	9.88	1.00	70.68	48.81	291.00
Random Grafting (50%)	13.59	7.40	69.56	52.53	267.74	12.23	3.90	79.33	60.92	285.71	11.34	1.00	84.47	64.76	206.97
Grafting (50%)	6.03	38.10	60.13	46.12	42.83	4.32	39.12	62.23	47.73	42.80	4.41	28.30	62.62	49.37	155.78
Grafting (30%)	12.89	24.50	63.71	49.16	153.69	10.30	27.30	71.97	54.97	159.74	OOM	-	90.19	72.34	∞
Grafting (80%)	2.91	39.70	57.64	44.61	25.16	1.89	41.00	55.20	44.27	10.87	0.17	32.30	40.80	33.43	4.06

† The heuristic of activation gradient magnitude (Ye et al., 2020) is utilized to guide activation pruning.
‡ Based on the official implementation of Sehwag et al. (2020), we extend the original sparse mask learning to activation sparsification.

four network architectures, including a 4-layer CNN (CNN-B) (Dathathri et al., 2020), a 11-layer ResNet-4B (Bak et al., 2021), ConvBig (Mirman et al., 2018), and a wider 7-layer CNN (ConvHuge) with 17.2M parameters. Note that it is usually impractical to verify a purely adversarially trained model with over 10M parameters, and our linearity grafting procedure enables complete verification for these models.

Training details. For fast adversarial training (Wong et al., 2020), we adopt the effective GradAlign regularization (Andriushchenko & Flammarion, 2020) with a coefficient of 0.2, for all 200 training epochs. The learning rate starts from 0.1 and decays by ten times at epochs 100 and 150, while the batch size is 128. We use an SGD optimizer with 0.9 momentum and 5×10^{-4} weight decay. During the fine-tuning of grafted networks, an initial learning rate of 0.01 is used for trainable slopes and intercept (a, b) of grafted neurons, and 0.001 for original model parameters. And the learning rate decays with a cosine annealing schedule of 100 training epochs. Other configurations are the same as the corresponding original pre-trained networks.

Evaluation metrics. We evaluate our methods on the official test sets with five classical metrics: (1) unstable neuron ratio (**UNR** $\%$) is number of unstable neurons divided by total number of neurons; (2) verified accuracy (**VA** $\%$) is the percentage of verifiably robust test images; (3) standard accuracy (**SA** $\%$); (4) robust accuracy (**RA** $\%$) is the percentage of robust test images under empirical attacks; and (5) average verification time per sample (**Time**). RA is calculated on perturbed testing samples generated by PGD-100 (Madry et al., 2019) with 100 restarts. VA, Time, and UNR are computed via the current SOTA complete verifier α,β-CROWN with a time-out of 300 seconds and other parameters leaving at the default. If the time of complete verification for an image exceeds 300s, its certification fails. SA, RA are evaluated on the full test set. VA is computed on the first $1,000$ images due to its high computational cost, following the setup in Singh et al. (2019a); Müller et al. (2021); Tjandraatmadja et al. (2020); Wang et al. (2021). The reported verification time excludes examples that are classified incorrectly or attacked successfully.

5.1. The Superiority of Grafting for Verification

In this section, we compare *grafting* with five pruning baseline methods: (i) *Baseline* without neuron pruning or grafting; (ii) *Stochastic Activation Pruning (SAP)* (Dhillon et al., 2018) that removes the activation with the least magnitude; (iii) *Gradient-based Activation Pruning (GAP)* which leverages activation's gradient (Ye et al., 2020) as a heuristic to prune neurons; (iv) *Hydra* (Sehwag et al., 2020) that learns sparse activation masks; (v) *Random Grafting* as a sanity check which randomly selects neurons for grafting linearity, without considering the ranking discussed in Section 4.

Improved verification performance. Experimental results with target perturbation radius $\epsilon = \frac{2}{255}$ and $\epsilon = \frac{8}{255}$ are collected in Table 1, and Table A7 (in Appendix), respectively. Several consistent observations can be drown from these extensive evaluations with four network architectures on MNIST (M), SVHN (S) and CIFAR-10 (C10) datasets:

❶ Compared to the baseline, applying grafting to 50% neurons of {(ConvBig, M), (ConvBig, S), (CNN-B, C10), (ResNet, C10), (ConvBig, C10)} achieves {82.20%, 41.10%, 13.00%, 37.30%, 37.82%} certified accuracy boosts and {25.42%, 7.67%, 10.49%, 13.91%, 13.43%} unstable neuron ratio reductions, where models have never been explicitly trained using a certified defense. Such impressive VA improvements evidence the effectiveness of grafted linearity, which balances the DNN expressiveness and certifiable robustness by controlling the number of unstable neurons. Although it comes with {0.61%, 10.96%, 5.87%, 16.56%, 22.67%} SA drops, grafted NNs still reach a clearly better trade-off, especially for (ConvBig, M / S). Conclusions are also valid in the more challenging case of $\epsilon = \frac{8}{255}$ in Table A7.

❷ Compared to existing activation pruning mechanisms, grafting demonstrates a substantial advantage in terms of reduced UNR, enhanced VA, and overall SA & VA trade-offs. It echos our intuition since grafting learnable linear activation functions is a more general and powerful solution than directly replacing ReLUs by zero functions (i.e., neuron pruning). Among three pruning baselines on CIFAR-10, SAP has the best VA and inferior SA; GAP reaches a superior SA and the worst VA; Hydra achieves an in-between performance.

❸ We enable complete verification for a large-scale ConvHuge model with 17.2M parameters on a single GPU, by grafting appropriate linearity. We notice that only grafting more than 50% neurons can reach satisfied VA like $> 28.30\%$, while it suffers moderate SA drops due to the challenging setting. The major reason for improved certification scalability is the reduced non-linearity that essentially alleviates the computation and memory bottleneck of complete verification.

❹ Comparing grafted NNs with different ratios, from 30% $\rightarrow$ 50%, certified accuracies are further increased with significantly diminished unstable neurons; from 50% $\rightarrow$ 80%, excessive injected linearity starts to harm both generalization and robustness, particularly for CIFAR-10.

❺ Thanks to injected linearity and reduced branching in BaB, most of our grafted NNs largely cut down the complete verification time (by 5.99% $\sim$ 94.08%). Two exceptions are the 30% grafted NN of (ResNet-4B / ConvBig, C10), which take a longer time than baseline networks although they have less unstable neurons. It is because the baseline model (0.80 / 1.30 VA) has vacuous verification bounds for some data points (Figure 4), which triggers an early stop in the verifier to give up early.

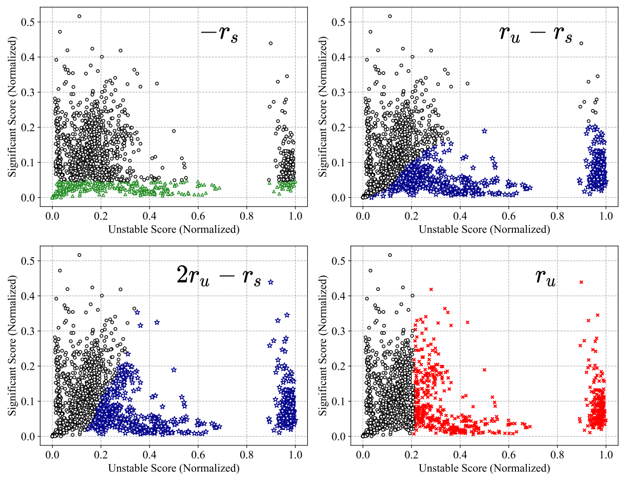

Figure 5. Neuron selections based on diverse picking criteria. △, ★, and ◇ indicate insignificant-only, insignificant-and-unstable, and unstable-only neuron selections respectively. r_u and r_s are defined in Section 4.

Table 2. Ablation on grafting criterion. Unstable neuron ratio (UNR %), VA (%), SA (%), and RA (%) of ConvBig with 50% grafted neurons on CIFAR-10 are reported.

Grafting Criterion	UNR	VA	SA	RA
$-r_s$	10.35	2.10	82.35	64.28
$r_u - r_s$	6.39	14.50	77.88	59.91
$2r_u - r_s$	4.32	38.90	62.15	47.70
r_u	4.13	38.70	59.39	45.77
$\gamma \times r_u - r_s$ (γ linearly decays $2 \rightarrow 0$)	4.32	39.12	62.23	47.73

Less unstable neurons and tighter bounds. Unstable neuron distributions and verification bounds are another two necessary angles to examine certifiable robustness.

Specifically, Figure 3 presents the layer-wise and overall unstable neuron distribution of networks before and after grafting. We find that grafting substantially pushes the distribution towards zero (i.e., the stable status), implying reduced unstable neurons after grafting. It is coherent with the findings in Figure 4 that grafted NNs consistently enjoy much tighter certified bounds (e.g., shrunk by a factor of 5 $\sim$ 744) and are more amenable to verification, compared to their vanilla counterparts. These impressive benefits should be credited to properly inserted linearity.

5.2. Dissecting Grafting

In this section, we perform a comprehensive analysis of each component in linearity grafting and located grafting masks for visualization.

Grafting criterion. Intuitively, the neuron picking criterion plays an essential role in the achievable performance of grafting. As a show case, we study four criteria variants including insignificant-only $(-r_s)$[1], insignificant-and-unstable ($r_u - r_s$ and $2r_u - r_s$), and unstable-only (r_u),

[1]We omit the neuron index i for simplicity.

and display the corresponding selected candidate neurons in Figure 5. We see that the criteria with a larger weighted r_u tends to pick more unstable neurons. Quantitative evaluations are collected in Table 2. The results show that: ❶ grafting neurons solely based on neuron significance scores ($-r_s$) or neuron instability scores (r_u) leads to an inferior trade-off between SA and VA. For example, 82.35% SA and 2.10% VA for the criteria $-r_s$; 59.39% SA and 38.70% VA for the criteria r_u. ❷ The criterion involving both r_u and r_s reach an improved balance between network expressiveness and certifiable robustness. ❸ The linearly decayed γ establishes the superior complete verification accuracy. It tends to bias on neuron significance scores when grafting more neurons, which perseveres a better network expressiveness.

Initialization of grafting. Experiments in Figure 6 demonstrate the effects of initialization for grafting's slope a and intercept b. We found that the grafting performance is more sensitive to its slope especially for $a \in [0.5, 0.8]$; $a \in [0.0, 0.5]$, $b \in [-0.5, 0.5]$ is the safe zone to initialize the grafted linear activation function.

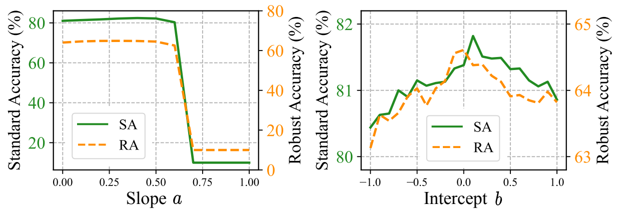

Figure 6. SA (%) and RA (%) over initialization of slope a (*Left*) and intercept b (*Right*) in grafting.

Whether fine-tuning weights in grafting? After replacing ReLU with linear functions, there are two options of the fine-tuning: optimizing $\{a, b\}$ only, or optimizing all $\{a, b, \mathbf{W}\}$. We conduct an ablation study in Table 3, and observe that updating weights together achieves consistently better performance. It is within expectation since FAT fine-tuned weights preserve more empirical robustness (RA), leaving more room for VA improvements.

Table 3. Ablation on whether fine-tuning weights in grafting. Unstable neuron ratio (UNR %), VA (%), SA (%), and RA (%) of ConvBig with 50% grafted neurons on CIFAR-10 are reported.

Settings	UNR	VA	SA	RA
Grafting (50%) w. Tuning Weight	4.32	39.12	62.23	47.73
Grafting (50%) w.o. Tuning Weight	5.24	36.20	57.60	44.33

Visualization of grafting mask and its slope & intercept. As shown in Figure 7, we visualize the obtained grafting mask and learned slopes and intercepts. The grafted neurons mainly scatter in the first few (i.e., $z^{(1)}, z^{(2)}, z^{(3)}$) and the last (i.e., $z^{(6)}$) layers, which appears to have block-wise patterns especially in $z^{(1)}$. The distributions of optimized slope and intercept are centered at $[0.1, 0.3]$ and 0, respectively.

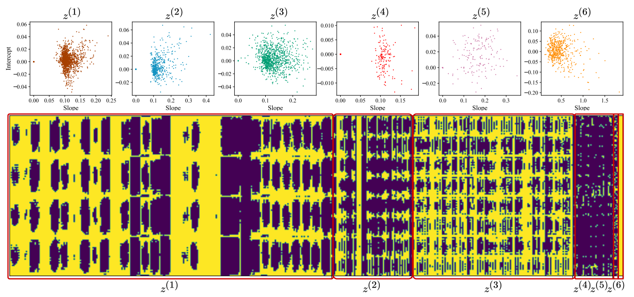

Figure 7. The *Bottom* figure visualizes grafting masks from ConvBig with 50% grafted neurons on CIFAR-10. The bright dots (●) are the grafted neurons and the dark dots (●) stand for vanilla ReLU neurons. The *Upper* figure shows the corresponding learned intercept over slope for each layer's activation, $z^{(1)} \sim z^{(6)}$. Please zoom-in for better readability.

5.3. Additional Studies on Grafting

Combining with existing regularization. Besides manipulating relaxed neurons, there exist several other effective regularizations (Xiao et al., 2018) for improving certifiable robustness: (i) ReLU stability regularizer that enforces ReLU to be always active or inactive given specific input perturbations; (ii) ℓ_1 weight regularization; (iii) small weight pruning. Since they work orthogonally to our proposal, we show results of their combination in Table 4. It suggests that grafting functions complementarily, and is capable of being integrated with previous techniques for further enhanced certifiable robustness.

Table 4. Combining grafting with existing regularizations for achieving better verified accuracy. Unstable neuron ratio (UNR %), VA (%), SA (%), and RA (%) of CNN-B with 50% grafted neurons on CIFAR-10 are reported.

Settings	UNR	VA	SA	RA
Grafting (50%)	5.36	50.40	74.08	58.76
Grafting (50%) + ReLU Stability Reg.	5.71	51.90	75.95	60.77
Grafting (50%) + ℓ_1 Reg.	5.83	51.90	76.31	61.10
Grafting (50%) + Small Weight Pruning	5.88	52.50	76.29	60.85
Grafting (50%) + All	5.73	52.55	76.15	61.08

Different methods to locate insignificant neurons. To determine the unnecessary portion of neurons and compute scores r_s, we can either choose some heuristics like activation magnitude (SAP) (Dhillon et al., 2018) and gradient (GAP) (Ye et al., 2020), or learned importance scores (Hydra) (Sehwag et al., 2020). A comprehensive comparison of these activation pruning algorithms has been carried out in Figure A8, where random pruning and dense networks as two baselines are included. We find that GAP consistently surpasses other approaches at a large range of activation sparsity from 0% to 85%, while the learning-based sparsification shows moderate advantages at the extreme sparsity. Therefore, our grafting pipeline adopts the best-performing "activation gradient" (GAP) (Ye et al., 2020) as a proxy for the neuron's significance.

Grafting 0 versus $a \times x + b$. Pruning (zeroing out) neurons is a spacial case of our grafting, where $\mathcal{A}_g(x) := 0$. From Table 5, we see grafting learnable linear functions obtains superior performance in terms of all evaluation metrics. Note that unlike classical activation pruning methods which only depend on significance scores r_s (Table 1), in the case of grafting zero, we still consider both significance and instability scores for picking candidate neurons in the same way when we graft $a \times x + b$, e.g., $\gamma \times r_u - r_s$.

Table 5. Comparison to grafting zero and gradually grafting. UNR (%), VA (%), SA (%), and RA (%) of CNN-B with 50% grafted neurons on CIFAR-10 are reported.

Settings	UNR	VA	SA	RA
Baseline	15.85	37.40	79.95	62.23
Grafting ($\mathcal{A}_g(x) = 0$)	5.60	48.40	73.19	57.78
Grafting ($\mathcal{A}_g(x) = a \times x + b$)	5.36	50.40	74.08	58.76
Gradually Grafting ($\mathcal{A}_g(x) = a \times x + b$)	5.91	49.00	75.43	58.79

One-shot versus gradually grafting. Although our main results are produced by one-shot grafting, it can be easily extended into a gradual style by assigning the desired amount of linearity to several iterations. Specifically, we adopt the schedule from Zhu & Gupta (2017) and graft a small portion of neurons in each iteration of the first half training epochs (i.e., 100 epochs). Table 5 shows that gradual grafting tends to maintain more standard accuracy by using more unstable ReLU neurons while sacrificing some verified accuracy.

Comparison with classical certified robust training. One of our goals is to achieve satisfying certifiable robustness and avoid time-consuming certified robust training. Supportive results demonstrated in Table 6 again validate our proposed grafting pipeline. It shows that *FAT + Grafting* obtains 2.85% VA, 16.08% SA, 10.14% RA improvements with 92.87% training time savings, compared to the Auto-LiRPA based certified robust training (Xu et al., 2020a) with a target perturbation radius $\epsilon = 2/255$.

Table 6. Comparisons between a representative certified robust training method (Xu et al., 2020a), and our grafting with fast adversarial training (FAT) without explicit certified defense training. Here we report UNR (%), VA (%), SA (%), RA (%), and total training time (hour) of CNN-B w./w.o. 50% grafted neurons on CIFAR-10 are reported.

Settings	UNR	VA	SA	RA	Training Time
Baseline (FAT)	15.85	37.40	79.95	62.23	0.39 h
Certified Robust Training	0.96	47.55	58.00	48.62	16.26 h
FAT + Grafting (50%)	5.36	50.40	74.08	58.76	1.13 h

6. Conclusion

This paper proposes *linearity grafting*, a new relaxed neuron pruning paradigm for improved certifiable robustness and verification scalability. Specifically, it grafts appropriate forms of linearity by replacing the redundant non-linear activation functions. The benefits come from: (1) grafting insignificant and unstable neurons directly reduces the harmful and excessive non-linearity and tightens the verification bounds; (2) the further optimized parameters (slopes and intercepts) in grafted linear neurons help maintain good generalization performance and avoid ill-conditioned activation states (e.g., most ReLU neurons are inactive and fixed at 0). For future works, grafting also motivates us to design verification-friendly architectures with a mixture of linear and non-linear activation functions.

Acknowledgment

Z.W. is in part supported by an NSF SCALE MoDL project (#2133861).

References

Anderson, G., Pailoor, S., Dillig, I., and Chaudhuri, S. Optimization and abstraction: a synergistic approach for analyzing neural network robustness. In *Proceedings of the 40th ACM SIGPLAN Conference on Programming Language Design and Implementation*, pp. 731–744, 2019.

Andriushchenko, M. and Flammarion, N. Understanding and improving fast adversarial training. *arXiv preprint arXiv:2007.02617*, 2020.

Bak, S., Liu, C., and Johnson, T. The second international verification of neural networks competition (vnn-comp 2021): Summary and results. *arXiv preprint arXiv:2109.00498*, 2021.

Balunovic, M. and Vechev, M. Adversarial training and provable defenses: Bridging the gap. In *International Conference on Learning Representations*, 2019.

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

Bunel, R., Turkaslan, I., Torr, P. H., Kohli, P., and Kumar, M. P. A unified view of piecewise linear neural network verification. *arXiv preprint arXiv:1711.00455*, 2017.

Bunel, R., De Palma, A., Desmaison, A., Dvijotham, K., Kohli, P., Torr, P., and Kumar, M. P. Lagrangian decomposition for neural network verification. In *Conference on Uncertainty in Artificial Intelligence*, pp. 370–379. PMLR, 2020.

Dathathri, S., Dvijotham, K., Kurakin, A., Raghunathan, A., Uesato, J., Bunel, R., Shankar, S., Steinhardt, J., Goodfellow, I., Liang, P., et al. Enabling certification of verification-agnostic networks via memory-efficient semidefinite programming. *arXiv preprint arXiv:2010.11645*, 2020.

De Palma, A., Bunel, R., Desmaison, A., Dvijotham, K., Kohli, P., Torr, P. H., and Kumar, M. P. Improved branch and bound for neural network verification via lagrangian decomposition. *arXiv preprint arXiv:2104.06718*, 2021.

Deng, L. The mnist database of handwritten digit images for machine learning research. *IEEE Signal Processing Magazine*, 29(6):141–142, 2012.

Dhillon, G. S., Azizzadenesheli, K., Bernstein, J. D., Kossaifi, J., Khanna, A., Lipton, Z. C., and Anandkumar, A. Stochastic activation pruning for robust adversarial defense. In *International Conference on Learning Representations*, 2018. URL `https://openreview.net/forum?id=H1uR4GZRZ`.

Dutta, S., Jha, S., Sanakaranarayanan, S., and Tiwari, A. Output range analysis for deep neural networks. *arXiv preprint arXiv:1709.09130*, 2017.

Dvijotham, K., Stanforth, R., Gowal, S., Mann, T. A., and Kohli, P. A dual approach to scalable verification of deep networks. In *UAI*, volume 1, pp. 3, 2018.

Ehlers, R. Formal verification of piece-wise linear feed-forward neural networks. In *International Symposium on Automated Technology for Verification and Analysis*, pp. 269–286. Springer, 2017.

Fu, Y., Yu, Q., Zhang, Y., Wu, S., Ouyang, X., Cox, D. D., and Lin, Y. Drawing robust scratch tickets: Subnetworks with inborn robustness are found within randomly initialized networks. In *Thirty-Fifth Conference on Neural Information Processing Systems*, 2021.

Gao, J., Wang, B., Lin, Z., Xu, W., and Qi, Y. Deepcloak: Masking deep neural network models for robustness against adversarial samples. *arXiv preprint arXiv:1702.06763*, 2017.

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. In *2018 IEEE Symposium on Security and Privacy (SP)*. IEEE, 2018.

Gowal, S., Dvijotham, K., Stanforth, R., Bunel, R., Qin, C., Uesato, J., Arandjelovic, R., Mann, T., and Kohli, P. On the effectiveness of interval bound propagation for training verifiably robust models. *arXiv preprint arXiv:1810.12715*, 2018.

Gui, S., Wang, H. N., Yang, H., Yu, C., Wang, Z., and Liu, J. Model compression with adversarial robustness: A unified optimization framework. *Advances in Neural Information Processing Systems*, 32:1285–1296, 2019.

Han, H., Xu, K., Hu, X., Chen, X., Liang, L., Du, Z., Guo, Q., Wang, Y., and Chen, Y. Scalecert: Scalable certified defense against adversarial patches with sparse superficial layers. *Advances in Neural Information Processing Systems*, 34, 2021.

Han, S., Mao, H., and Dally, W. J. Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding. *arXiv preprint arXiv:1510.00149*, 2015a.

Han, S., Pool, J., Tran, J., and Dally, W. Learning both weights and connections for efficient neural network. In *Advances in neural information processing systems*, pp. 1135–1143, 2015b.

He, Y., Zhang, X., and Sun, J. Channel pruning for accelerating very deep neural networks. In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 1389–1397, 2017.

Henriksen, P. and Lomuscio, A. Efficient neural network verification via adaptive refinement and adversarial search. In *ECAI 2020*, pp. 2513–2520. IOS Press, 2020.

Hooker, S., Courville, A., Clark, G., Dauphin, Y., and Frome, A. What do compressed deep neural networks forget? *arXiv preprint arXiv:1911.05248*, 2019.

Huang, X., Kwiatkowska, M., Wang, S., and Wu, M. Safety verification of deep neural networks. In *International conference on computer aided verification*, pp. 3–29. Springer, 2017.

Jordao, A. and Pedrini, H. On the effect of pruning on adversarial robustness. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 1–11, 2021.

Katz, G., Barrett, C., Dill, D. L., Julian, K., and Kochenderfer, M. J. Reluplex: An efficient smt solver for verifying deep neural networks. In *International Conference on Computer Aided Verification*, pp. 97–117. Springer, 2017.

Katz, G., Huang, D. A., Ibeling, D., Julian, K., Lazarus, C., Lim, R., Shah, P., Thakoor, S., Wu, H., Zeljić, A., et al. The marabou framework for verification and analysis of deep neural networks. In *International Conference on Computer Aided Verification*, pp. 443–452. Springer, 2019.

Krizhevsky, A. and Hinton, G. Learning multiple layers of features from tiny images. *Master's thesis, Department of Computer Science, University of Toronto*, 2009.

LeCun, Y., Denker, J. S., and Solla, S. A. Optimal brain damage. In *Advances in neural information processing systems*, pp. 598–605, 1990.

Liu, Z., Li, J., Shen, Z., Huang, G., Yan, S., and Zhang, C. Learning efficient convolutional networks through network slimming. In *Proceedings of the IEEE international conference on computer vision*, pp. 2736–2744, 2017.

Lu, J. and Kumar, M. P. Neural network branching for neural network verification. *arXiv preprint arXiv:1912.01329*, 2019.

Madry, A., Makelov, A., Schmidt, L., Tsipras, D., and Vladu, A. Towards deep learning models resistant to adversarial attacks. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=rJzIBfZAb.

Madry, A., Makelov, A., Schmidt, L., Tsipras, D., and Vladu, A. Towards deep learning models resistant to adversarial attacks, 2019.

Mirman, M., Gehr, T., and Vechev, M. Differentiable abstract interpretation for provably robust neural networks. In *International Conference on Machine Learning*, pp. 3578–3586. PMLR, 2018.

Molchanov, P., Mallya, A., Tyree, S., Frosio, I., and Kautz, J. Importance estimation for neural network pruning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 11264–11272, 2019.

Müller, M. N., Makarchuk, G., Singh, G., Püschel, M., and Vechev, M. Prima: Precise and general neural network certification via multi-neuron convex relaxations. *arXiv preprint arXiv:2103.03638*, 2021.

Netzer, Y., Wang, T., Coates, A., Bissacco, A., Wu, B., and Ng, A. Y. Reading digits in natural images with unsupervised feature learning. In *NIPS Workshop on Deep Learning and Unsupervised Feature Learning 2011*, 2011. URL http://ufldl.stanford.edu/housenumbers/nips2011_housenumbers.pdf.

Raghunathan, A., Steinhardt, J., and Liang, P. Certified defenses against adversarial examples. In *International Conference on Learning Representations*, 2018a. URL https://openreview.net/forum?id=Bys4ob-Rb.

Raghunathan, A., Steinhardt, J., and Liang, P. S. Semidefinite relaxations for certifying robustness to adversarial examples. In *NeurIPS*, pp. 10877–10887, 2018b.

Rubies-Royo, V., Calandra, R., Stipanovic, D. M., and Tomlin, C. Fast neural network verification via shadow prices. *arXiv preprint arXiv:1902.07247*, 2019.

Salman, H., Yang, G., Zhang, H., Hsieh, C.-J., and Zhang, P. A convex relaxation barrier to tight robustness verification of neural networks. *arXiv preprint arXiv:1902.08722*, 2019.

Sehwag, V., Wang, S., Mittal, P., and Jana, S. Towards compact and robust deep neural networks. *arXiv preprint arXiv:1906.06110*, 2019.

Sehwag, V., Wang, S., Mittal, P., and Jana, S. Hydra: Pruning adversarially robust neural networks. *arXiv preprint arXiv:2002.10509*, 2020.

Shi, Z., Wang, Y., Zhang, H., Yi, J., and Hsieh, C.-J. Fast certified robust training with short warmup. In *ICML 2021 Workshop on Adversarial Machine Learning*, 2021. URL https://openreview.net/forum?id=_jUobmvki51.

Singh, G., Ganvir, R., Püschel, M., and Vechev, M. Beyond the single neuron convex barrier for neural network certification. In Wallach, H., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019a. URL https://proceedings.neurips.cc/paper/2019/file/0a9fdbb17feb6ccb7ec405cfb85222c4-Paper.pdf.

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):1–30, 2019b.

Szegedy, C., Zaremba, W., Sutskever, I., Bruna, J., Erhan, D., Goodfellow, I., and Fergus, R. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*, 2013.

Tjandraatmadja, C., Anderson, R., Huchette, J., Ma, W., Patel, K., and Vielma, J. P. The convex relaxation barrier, revisited: Tightened single-neuron relaxations for neural network verification. *arXiv preprint arXiv:2006.14076*, 2020.

Tjeng, V., Xiao, K., and Tedrake, R. Evaluating robustness of neural networks with mixed integer programming. *arXiv preprint arXiv:1711.07356*, 2017.

Wang, S., Chen, Y., Abdou, A., and Jana, S. Mixtrain: Scalable training of verifiably robust neural networks. *arXiv preprint arXiv:1811.02625*, 2018a.

Wang, S., Pei, K., Whitehouse, J., Yang, J., and Jana, S. Efficient formal safety analysis of neural networks. *arXiv preprint arXiv:1809.08098*, 2018b.

Wang, S., Pei, K., Whitehouse, J., Yang, J., and Jana, S. Formal security analysis of neural networks using symbolic intervals. In *27th {USENIX} Security Symposium ({USENIX} Security 18)*, pp. 1599–1614, 2018c.

Wang, S., Wang, X., Ye, S., Zhao, P., and Lin, X. Defending dnn adversarial attacks with pruning and logits augmentation. In *2018 IEEE Global Conference on Signal and Information Processing (GlobalSIP)*, pp. 1144–1148. IEEE, 2018d.

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

Wang, X., Yu, F., Dou, Z.-Y., Darrell, T., and Gonzalez, J. E. Skipnet: Learning dynamic routing in convolutional networks. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 409–424, 2018e.

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

Wong, E., Schmidt, F. R., Metzen, J. H., and Kolter, J. Z. Scaling provable adversarial defenses. *arXiv preprint arXiv:1805.12514*, 2018.

Wong, E., Rice, L., and Kolter, J. Z. Fast is better than free: Revisiting adversarial training. *arXiv preprint arXiv:2001.03994*, 2020.

Wu, Z., Nagarajan, T., Kumar, A., Rennie, S., Davis, L. S., Grauman, K., and Feris, R. Blockdrop: Dynamic inference paths in residual networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 8817–8826, 2018.

Xiao, K. Y., Tjeng, V., Shafiullah, N. M., and Madry, A. Training for faster adversarial robustness verification via inducing relu stability. *arXiv preprint arXiv:1809.03008*, 2018.

Xu, K., Shi, Z., Zhang, H., Wang, Y., Chang, K.-W., Huang, M., Kailkhura, B., Lin, X., and Hsieh, C.-J. Automatic perturbation analysis for scalable certified robustness and beyond. *Advances in Neural Information Processing Systems*, 33, 2020a.

Xu, K., Zhang, H., Wang, S., Wang, Y., Jana, S., Lin, X., and Hsieh, C.-J. Fast and complete: Enabling complete neural network verification with rapid and massively parallel incomplete verifiers. *arXiv preprint arXiv:2011.13824*, 2020b.

Ye, S., Xu, K., Liu, S., Cheng, H., Lambrechts, J.-H., Zhang, H., Zhou, A., Ma, K., Wang, Y., and Lin, X. Adversarial robustness vs. model compression, or both? In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 111–120, 2019.

Ye, X., Dai, P., Luo, J., Guo, X., Qi, Y., Yang, J., and Chen, Y. Accelerating cnn training by pruning activation gradients. In *European Conference on Computer Vision*, pp. 322–338. Springer, 2020.

Zhang, C., Bengio, S., and Singer, Y. Are all layers created equal? *arXiv preprint arXiv:1902.01996*, 2019a.

Zhang, H., Weng, T.-W., Chen, P.-Y., Hsieh, C.-J., and Daniel, L. Efficient neural network robustness certification with general activation functions. *arXiv preprint arXiv:1811.00866*, 2018.

Zhang, H., Chen, H., Xiao, C., Gowal, S., Stanforth, R., Li, B., Boning, D., and Hsieh, C.-J. Towards stable and efficient training of verifiably robust neural networks. *arXiv preprint arXiv:1906.06316*, 2019b.

Zhou, H., Alvarez, J. M., and Porikli, F. Less is more: Towards compact cnns. In *European Conference on Computer Vision*, pp. 662–677. Springer, 2016.

Zhu, M. and Gupta, S. To prune, or not to prune: exploring the efficacy of pruning for model compression. *arXiv preprint arXiv:1710.01878*, 2017.

A1. More Experiment Results

Different target perturbation radius ϵ. We validate the effectiveness of our proposed grafting across different target perturbation radius $\epsilon = \frac{8}{255}$, which is usually a more challenging setup. Experiment results are collected in Table A7, which consistently demonstrate the superiority of grafted NN in terms of boosted VA and reduced UNR.

Table A7. **Unstable neuron ratio (UNR %), verified accuracy (VA %), standard accuracy (SA %), PGD-100 robust accuracy (RA %), and average time (s)** of FAT trained models w./w.o. grafting on (CIFAR-10, CNN-B). β-CROWN (Wang et al., 2021) with BaB, the current SOTA complete verifier is used to compute VA. **The target perturbation size is $\epsilon = \frac{8}{255}$.**

FAT ($\epsilon = \frac{8}{255}$)	(CNN-B, CIFAR-10)				
	UNR	VA	SA	RA	Time
Baseline	37.44	0.30	66.47	34.23	300.88
SAP (Dhillon et al., 2018) (50%)	18.20	0.90	63.87	32.66	295.33
GAP† (Ye et al., 2020) (50%)	23.83	0.40	65.91	33.46	300.46
Hydra‡ (Sehwag et al., 2020) (50%)	20.12	0.40	64.38	31.49	283.44
Random Grafting (50%)	20.12	2.00	60.41	31.66	283.44
Grafting (50%)	12.35	4.70	58.87	31.34	257.59
Grafting (30%)	17.47	1.10	64.46	32.83	294.55
Grafting (80%)	4.93	14.90	46.76	27.25	112.11

† The heuristic of activation gradient magnitude (Ye et al., 2020) is utilized to guide activation pruning.
‡ Based on the official implementation of Sehwag et al. (2020), we extend the original sparse mask learning to activation.

Comparison of different neuron significance proxies. As shown in Figure A8, the activate gradient of GAP does the best job of approximating the neuron significance.

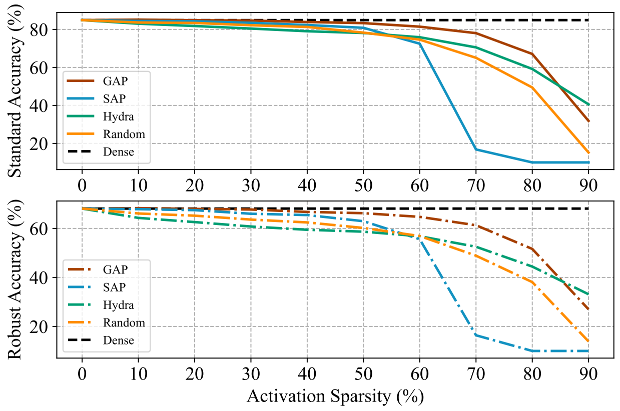

Figure A8. Performance (SA % and RA %) of activation pruning with diverse heuristics or optimized scores, which lies the foundation of locating insignificant neurons.

Corner test cases. We conduct corner case evaluations for our grafted NNs, where pruning identified exemplars (PIE) (Hooker et al., 2019) are adopted as the corner test cases. From Table A8, we observe grafting still obtains substantial VA boosts (+11.71%) on corner cases with a moderate decrease in SA.

Table A8. Corner case study. **Verified accuracy (VA %) and standard accuracy (SA %)** of FAT trained models w./w.o. grafting on (CIFAR-10, CNN-B) are reported.

(C10, CNN-B)	VA (corner cases)	SA (corner cases)	VA (regular testset)	SA (regular testset)
Baseline	9.01%	54.05%	37.40%	79.95%
Grafting (50%)	20.72% (+11.71%)	51.35% (−2.70%)	50.40% (+13.00%)	74.08% (−5.87%)

Investigations on non-ReLU networks. Our idea of replacing highly non-linear activation functions with linear operations to ease verification is general, and can be applied to other non-linear functions as long as the verifier supports them. We demonstrate *preliminary results* on a CNN using **sigmoid** activation function, with improved verified accuracy (VA) after grafting shown in Table A9. It is less improved compared to ReLU networks mostly because current verifiers have limited support for non-ReLU activation.

Table A9. Studies of non-ReLU networks. **Verified accuracy (VA %)** of FAT trained models w./w.o. grafting on (CIFAR-10, Sigmod 3-layer CNN) are reported.

(CIFAR-10, Sigmoid 3-layer CNN)	Baseline	**Grafting** (50%)	GAP (50%)
Verified Accuracy (VA)	38.62%	39.46% (+0.84%)	35.41% (−3.21%)

Linearity accelerates the verification? Our grafted linearity greatly reduced the relaxation and branching needed in verifiers, decreasing the verification complexity exponentially, thus more examples can be verified within a short time. We do not accelerate model computation, and in fact we use an extra binary mask to indicate the grafted neurons which slightly increases computation during training (we measured 0.076s → 0.123s per batch for ConvBig).

Results with the increased verification time. In Figure A9, we plot the number of verified examples with 1,000s verification time. Since grafting reduces verification complexity exponentially, a verifier can quickly verify much more examples than the baseline within a short time.

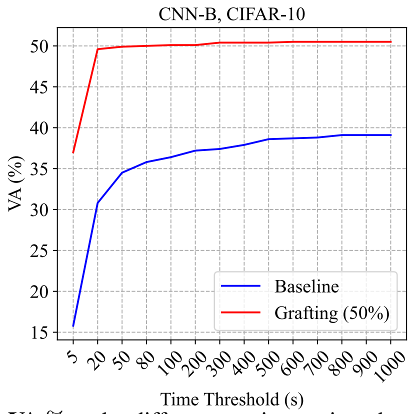

Figure A9. **VA** % under different maximum time thresholds.

Societal impact. We believe our work improves the scalability of certification and therefore has overall positive societal impacts on various safety-crucial applications. However, it may potentially be misused to identify the weakness of DNNs and guide malicious attacks.