A EXPERIMENT DETAILS

Verification experiments. In these experiments, we used pre-trained MNIST and CIFAR-10 image classifiers provided by (Weng et al., 2018). These are MLP networks with ReLU activation functions trained in a standard vanilla manner. The input for these models are normalized to fit in the range $[-0.5, 0.5]$. The experiments were done on a personal laptop with a 4 core Intel Core i7-8550U CPU. All verifications were done under the threat model where the class with the second-highest logit is a target.

Probabilistic training experiments. In these experiments, we trained the small architecture in table 1 on MNIST using IBP and using PROVEN-IBP. The input for these models were normalized to fit in $[0, 1]$. In order to compare the two models, they share training parameters whenever possible. We used batch sizes of 512 for 100 epochs, or 11.8k steps. The learning rate was fixed at 5×10^{-4} and we used the AdamW optimizer. We have a warm-up period for the first 5 epochs in which the loss is the standard cross-entropy loss. After this, the next 45 epochs use the corresponding loss regularization terms and ramp-up the parameters (ϵ, κ for IBP and ϵ, κ, β for PROVEN-IBP) over these 45 epochs. We use $\epsilon_{\text{train}} = 0.4$ and $\epsilon_{\text{test}} = 0.3$. Furthermore, κ goes from 1 to 0.5 and for PROVEN-IBP, β goes from 1 to 0. Rather than a linear schedule, we use an exponential-growth schedule which allows for a more gradual change of parameters. In particular, for a schedule which starts at epoch u, ends at epoch v, and changes a parameter from a to b, we set that parameter to

$$(b-a)\left(\frac{\exp\left(\frac{r(t-u)}{v-u}\right)-1}{\exp(r)-1}\right)+a$$

during epoch t for hyperparameter $r = 1.5$. PROVEN-IBP uses Q_{train} fixed at 1×10^{-4} while $Q_{\text{test}} = 1 \times 10^{-2}$. We also set the linear relaxations on ReLU so that the bounds are always at least as tight as IBP by using the zero function as the lower bound when the interval crosses both negative and positive values. As noted in the main text, these experiments were done on a NVIDIA Tesla V100 GPU.

Sparsity experiments. For these experiments, we train several standard and robust classifiers with the architectures listed in table 1. We approximately follow the same parameters as (Gowal et al., 2019), rounding to use whole epochs when necessary. We opt to use their 350-epoch CIFAR training, although we do not use any of their data augmentation. Notably, we clip the model gradients at a low value of 0.1 as we found large losses during the beginning of the ramp-up periods. This was done on a NVIDIA Tesla V100 GPU. To evaluate the linear bounds A_L, A_U, we use **I-PROVEN** with $Q = 1 \times 10^{-2}, \epsilon = 0.3$ for MNIST and $Q = 1 \times 10^{-2}, \epsilon = 8/255$ for CIFAR-10.

Table 1: Model architectures used in (Gowal et al., 2019). All layers are followed by ReLU activations except the last fully connected output layer which is omitted. CONV $k :: w \times h :: s$ denotes a 2D convolutional layer with k filters of size $w \times h$ and stride s while FC n corresponds to a fully connected layer with n outputs.

small	medium	large
CONV 16 :: 4×4 :: 2	CONV 32 :: 3×3 :: 1	CONV 64 :: 3×3 :: 1
CONV 32 :: 4×4 :: 1	CONV 32 :: 4×4 :: 2	CONV 64 :: 3×3 :: 1
FC 100	CONV 64 :: 3×3 :: 1	CONV 128 :: 3×3 :: 2
	CONV 64 :: 4×4 :: 2	CONV 128 :: 3×3 :: 1
	FC 512	CONV 128 :: 3×3 :: 1
	FC 512	FC 512

B Additional results

We compared I-PROVEN against the same Monte Carlo method used in Section 4.3 of the paper. We report largest ϵ such that the $B_\infty(x, \epsilon)$ uniform distribution which is correctly classified with probability at least $1 - Q$. We perform 20 steps of binary search on ϵ for both I-PROVEN and the Monte Carlo method, and we evaluate samples in large batches to leverage the parallelizability of the sampling. We average our results over 10 inputs and report the runtimes in seconds below as well. Note that the original image values are in [-0.5, 0.5] and we do not truncate them. The first 3 models are for MNIST and the last two are for CIFAR-10. We consider the margin between the best class and second-best class. This experiment was done on a NVIDIA Tesla V100 GPU.

Table 2: **I-PROVEN** versus original PROVEN versus simple Monte Carlo method on MNIST classifier with 2 layers, 1024 neurons each on the probabilistic robustness certificate

Q	0.0001	0.01	0.05	0.25	0.5	0.95
PROVEN (Weng et al., 2019)	0.0292	0.0300	0.0304	0.0309	0.0313	0.0319
Time (s)	0.4847	0.4789	0.4813	0.4811	0.4834	0.4822
I-PROVEN	0.0757	0.0888	0.0953	0.1034	0.1076	0.1120
Time (s)	0.9136	0.4886	0.4897	0.4912	0.4950	0.4899
Monte Carlo	0.3691	0.4977	0.5543	0.6843	0.6958	0.7858
Time (s)	233.7793	4.8198	2.6030	1.6715	1.5603	1.5039

Table 3: **I-PROVEN** versus original PROVEN versus simple Monte Carlo method on MNIST classifier with 3 layers, 1024 neurons each on the probabilistic robustness certificate

Q	0.0001	0.01	0.05	0.25	0.5	0.95
PROVEN (Weng et al., 2019)	0.0223	0.0226	0.0227	0.0228	0.0229	0.0231
Time (s)	0.8991	0.8993	0.9047	0.8983	0.8977	0.9007
I-PROVEN	0.0640	0.0745	0.0797	0.0860	0.0893	0.0926
Time (s)	1.0111	0.9104	0.9204	0.9102	0.9017	0.9136
Monte Carlo	0.4480	0.5745	0.6431	0.7780	0.8295	0.8923
Time (s)	524.8032	7.7765	3.3641	1.9709	1.7936	1.7367

Table 4: **I-PROVEN** versus original PROVEN versus simple Monte Carlo method on MNIST classifier with 4 layers, 1024 neurons each on the probabilistic robustness certificate

Q	0.0001	0.01	0.05	0.25	0.5	0.95
PROVEN (Weng et al., 2019)	0.0083	0.0084	0.0084	0.0084	0.0084	0.0084
Time (s)	1.5498	1.5483	1.5521	1.5522	1.5541	1.5459
I-PROVEN	0.0299	0.0347	0.0370	0.0398	0.0413	0.0428
Time (s)	1.6641	1.5747	1.5774	1.5781	1.5739	1.5745
Monte Carlo	0.3619	0.4884	0.5566	0.6647	0.7369	0.8141
Time (s)	810.0939	10.5965	3.7936	1.9063	1.7350	1.5721

In addition, we tested these three methods for Q from 10^{-6} to 10^{-2}. We had to use a very small MNIST model in this experiment because of the prohibitive runtime of the sampling method.

For reference, the adversarial L_∞ certified radius found by CROWN on this small MNIST model was 0.0145.

Table 5: **I-PROVEN** versus original PROVEN versus simple Monte Carlo method on CIFAR-10 classifier with 5 layers, 2048 neurons each on the probabilistic robustness certificate

Q	0.0001	0.01	0.05	0.25	0.5	0.95
PROVEN (Weng et al., 2019)	0.0018	0.0018	0.0018	0.0018	0.0018	0.0018
Time (s)	14.0501	14.0448	14.0455	14.0449	14.0349	14.0342
I-PROVEN	0.0123	0.0141	0.0150	0.0161	0.0166	0.0171
Time (s)	14.3249	14.2359	14.2281	14.2094	14.2036	14.2218
Monte Carlo	0.1477	0.2213	0.2734	0.3350	0.4423	0.4756
Time (s)	5013.3533	61.4112	20.1818	10.4928	9.2565	8.4926

Table 6: **I-PROVEN** versus original PROVEN versus simple Monte Carlo method on CIFAR-10 classifier with 7 layers, 1024 neurons each on the probabilistic robustness certificate

Q	0.0001	0.01	0.05	0.25	0.5	0.95
PROVEN (Weng et al., 2019)	0.0018	0.0018	0.0018	0.0018	0.0018	0.0018
Time (s)	4.1160	4.1139	4.1163	4.1143	4.0971	4.0928
I-PROVEN	0.0128	0.0147	0.0157	0.0168	0.0174	0.0179
Time (s)	4.4922	4.1673	4.1725	4.1664	4.1689	4.1710
Monte Carlo	0.2157	0.2982	0.3346	0.4751	0.5351	0.7140
Time (s)	1437.8627	18.0755	5.9482	2.3806	1.9541	1.7490

Table 7: **I-PROVEN** versus original PROVEN versus simple Monte Carlo method on MNIST classifier with 3 layers, 20 neurons each on the probabilistic robustness certificate

Q	10^{-6}	10^{-5}	10^{-4}	10^{-3}	10^{-2}
PROVEN (Weng et al., 2019)	0.0202	0.0204	0.0206	0.0208	0.0211
Time (s)	0.6174	0.6185	0.6191	0.6192	0.6177
I-PROVEN	0.0492	0.0526	0.0569	0.0624	0.0699
Time (s)	0.7007	0.6307	0.6356	0.6288	0.6302
Monte Carlo	0.1753	0.1889	0.2071	0.2381	0.2922
Time (s)	16063.6184	1594.6121	161.5650	17.4352	3.4294

C Connection to l_2-norm robustness

I-PROVEN uses the norm of the bounding linear matrices $||A_L||_2, ||A_U||_2$. This is similar to linear relaxation methods when certifying the robustness of a model to adversarial attacks within an l_2-norm ball. This connection is clearest when using **I-PROVEN** to certify Gaussian noise and distributing Q equally so that $q_i = q$ for some constant q. In this case, we have

$$l_{\text{prob}} = A_L x + b_L - \epsilon_{\text{prob}} \operatorname{erf}^{-1}(1-2q)\, ||A_L||_2, \quad u_{\text{prob}} = A_U x + b_U - \epsilon_{\text{prob}} \operatorname{erf}^{-1}(1-2q)\, ||A_U||_2. \tag{1}$$

On the other hand, for adversarial robustness withn an l_2-norm ball, we have

$$l_{\text{strict}} = A_L x + b_L - \epsilon_{\text{strict}} ||A_L||_2, \quad u_{\text{strict}} = A_U x + b_U - \epsilon_{\text{strict}} ||A_U||_2. \tag{2}$$

Thus, if

$$\epsilon_{\text{prob}} \operatorname{erf}^{-1}(1-2q) = \epsilon_{\text{strict}}, \tag{3}$$

the two methods will obtain the same results. In general, however, **I-PROVEN** may not set all q_i's equal. For uniform noise in particular, the use of strict l_∞ bounds when available means that this connection will not always hold. This does suggest that one can make non-trivial statements about the probabilistic robustness of a model given it's adversarial robustness against l_2-norm attacks. This is assuming that the adversarial robustness certificate was obtained using linear relaxation methods.

References

Sven Gowal, Krishnamurthy Dvijotham, Robert Stanforth, Rudy Bunel, Chongli Qin, Jonathan Uesato, Relja Arandjelovic, Timothy Mann, and Pushmeet Kohli. Scalable verified training for provably robust image classification. *ICCV*, 2019. 1

Lily Weng, Huan Zhang, Hongge Chen, Zhao Song, Cho-Jui Hsieh, Luca Daniel, Duane Boning, and Inderjit Dhillon. Towards fast computation of certified robustness for relu networks. In *International Conference on Machine Learning*, pp. 5276–5285. PMLR, 2018. 1

Lily Weng, Pin-Yu Chen, Lam Nguyen, Mark Squillante, Akhilan Boopathy, Ivan Oseledets, and Luca Daniel. Proven: Verifying robustness of neural networks with a probabilistic approach. In *International Conference on Machine Learning*, pp. 6727–6736. PMLR, 2019. 2, 3