Certified defences can hurt generalisation

Piersilvio De Bartolomeis[1], Jacob Clarysse[1], Fanny Yang[1], and Amartya Sanyal[2]

[1]Department of Computer Science, ETH Zürich
[2]ETH AI Center

Abstract

In recent years, much work has been devoted to designing certified defences for neural networks, i.e., methods for learning neural networks that are provably robust to certain adversarial perturbations. Due to the non-convexity of the problem, dominant approaches in this area rely on convex approximations, which are inherently loose. In this paper, we question the effectiveness of such approaches for realistic computer vision tasks. First, we provide extensive empirical evidence to show that certified defences suffer not only worse accuracy but also worse robustness and fairness than empirical defences. We hypothesise that the reason for why certified defences suffer in generalisation is (i) the large number of relaxed non-convex constraints and (ii) the strong alignment between the adversarial perturbations and the "signal" direction. We provide a combination of theoretical and experimental evidence to support these hypotheses.

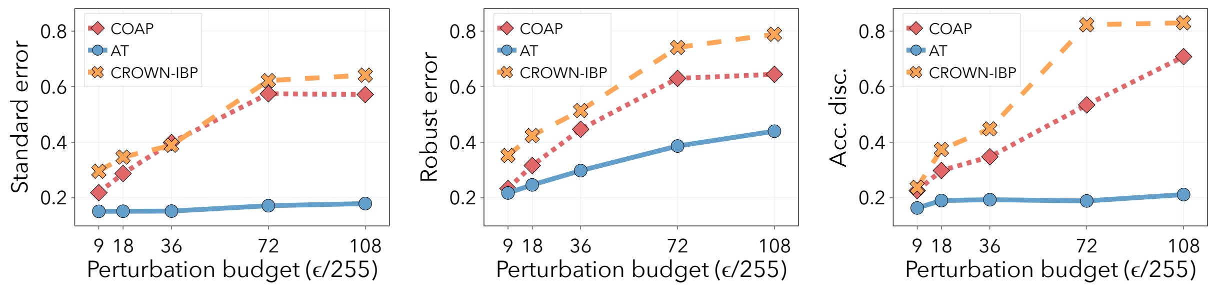
1 Introduction

Several works have shown the existence of adversarial examples: imperceptible perturbations to the input can fool state-of-the-art classifiers [2, 36]. Consequently, robustness to adversarial examples has become a crucial design goal for machine learning models. In real-world scenarios, robustness against many different types of input perturbations may be desired depending on the domain of application. Therefore, to build robust models, we must first define a threat model for the adversary. In this paper, we consider the well-studied ℓ_p-ball threat model, where $\mathcal{B}_\epsilon := \{\delta : \|\delta\|_p \leq \epsilon\}$ represents the set of allowed perturbations for some ℓ_p-ball with radius ϵ centred around the origin.

Once a threat model is defined, we can formalise the problem of building models that are robust to adversarial examples. For any distribution $\mathcal{D}$, neural network model $f_\theta : \mathbb{R}^d \to \mathbb{R}^k$ parameterised by the weights $\theta \in \mathbb{R}^p$, and loss function L, our goal is to solve the following robust optimisation problem:

$$\min_\theta \mathbf{R}_\epsilon(\theta) := \mathbb{E}_{(x,y)\sim\mathcal{D}} \left[\max_{\delta \in \mathcal{B}_\epsilon} L(f_\theta(x+\delta), y) \right] \tag{1}$$

We call $\mathbf{R}_\epsilon(\theta)$ the robust error when L is the 0-1 loss function. In practice, as the distribution $\mathcal{D}$ is unknown, we minimise the empirical robust error on a finite dataset D sampled from $\mathcal{D}$. Further, in the case of neural networks, the inner-maximisation is a non-convex optimisation problem and prohibitively hard to solve from a computational perspective [19, 38]. Instead, two efficient techniques are widely used to overcome the computational barrier: *empirical* defences that provide a lower bound on the solution and *certified* defences that provide an upper bound.



(a) Standard error (b) Robust error (c) Unfairness

Figure 1: Results for ℓ_2-ball perturbations on the CIFAR-10 test set. We compare ResNet architectures trained using state-of-the-art certified defenses CROWN-IBP [46, 42] and COAP [40, 39] against the most popular empirical defense to date AT [15, 27]. In Figures 1a, 1b and 1c we plot respectively standard error, robust error and accuracy discrepancy as the perturbation budget increases.

Adversarial training (AT) [15, 27] is one of the few empirical defences that has stood the test of time. AT minimises the worst-case empirical loss in Equation (1) by approximately solving the inner-maximisation problem with first-order optimisation methods. However, despite its simplicity and computational efficiency, AT does not provide any robustness guarantees, which are essential in many safety-critical domains.

To address this limitation, there has been significant interest in designing certified defences, i.e., methods for learning neural networks that are *provably* robust to ℓ_p-ball perturbations on the training data. Many recent works have proposed to solve a convex relaxation of the inner-maximisation problem by relaxing the non-convex ReLU constraint sets with convex ones [39, 31, 10, 46]. Despite all of these progresses, certified defences based on convex relaxations suffer from an inherent flaw: the upper bound they provide on the robust error is far from being tight. In this paper, we argue that the fundamental looseness of convex relaxations hinders the practical effectiveness of current certified defences. In particular, as shown in Figure 1, certified defences suffer significantly worse accuracy, robustness, and fairness on the test data compared to adversarial training. Our contributions are as follows:

- In Section 2, we provide extensive experimental evidence on real-world vision datasets that current certified defences hurt accuracy, robustness, and fairness across a range of ℓ_2-ball perturbations.
- In Section 3, we hypothesise that certified defences hurt generalisation because of (i) the large number of relaxed non-convex constraints and (ii) the strong alignment between the adversarial perturbations and the ”signal” direction. We provide experimental evidence on synthetic datasets to support these hypotheses.
- In Section 4, we investigate hypothesis (ii) from a theoretical perspective. First, we introduce a new threat model consisting of perturbations that are perfectly aligned with the signal direction. Then, we extend the formulation of certified defences to this setting. Finally, we prove, for a simplified network in high-dimensions, that certified defences yield higher robust risk than empirical defences when the adversarial perturbation perfectly aligns with the signal direction.

1.1 Related work

We discuss here how our work relates to phenomena that have been observed in the literature.

Looseness of convex relaxations The most relevant prior work investigating the looseness of convex relaxations is Salman et al. [33]. They show that convex relaxations that approximate each ReLU output separately suffer an inherent tightness barrier. In particular, even the optimal convex relaxation cannot obtain tight bounds on the robust error. Subsequently, Singh et al. [35] show that this barrier can be overcome by approximating multiple ReLUs outputs jointly. However, such approaches cannot be integrated into a training procedure due to their computational inefficiency. For this reason, this class of convex relaxations is not studied here. Instead, this paper investigates how the tightness barrier affects generalisation and fairness for convex relaxations that approximate each ReLU output separately.

Randomised smoothing A popular alternative to certified defences based on convex relaxation is randomised smoothing [6, 26, 25]. The key idea behind this technique is to transform an arbitrary classifier $f_\theta : \mathbb{R}^d \to \mathbb{R}^k$ into a "smoothed" classifier $g_\theta : \mathbb{R}^d \to \mathbb{R}^k$. In particular, for a given data point $x \in \mathbb{R}^d$ and variance σ^2, the smoothed classifier prediction $g_\theta(x)$ is defined as the most probable prediction by f_θ of the random variable $\mathcal{N}(x, \sigma^2 I_d)$. Despite its popularity, randomised smoothing still suffers from several limitations. For example, Mohapatra et al. [29] investigate the side-effects of randomised smoothing and show that it significantly hurts the disparity in class-wise accuracy. Further, several works have exposed an accuracy-robustness tradeoff related to the smoothness of the classifier [43, 3, 22].

2 Real-world: certified defences hurt generalisation and fairness

In this section, we show that certified defences hurt standard error, robust error, and fairness on three real-world computer vision datasets: Tiny ImageNet [23], CIFAR-10 [21] and MNIST [24]. Among certified defences based on convex relaxations, we consider the convex outer adversarial polytope (COAP) [40, 39], which achieves state-of-the-art certified robustness under ℓ_2-ball perturbations. Additionally, we consider CROWN-IBP [45, 42], which combines the tight convex relaxation CROWN [45] with interval bound propagation (IBP) [16, 28] and achieves state-of-the-art certified robustness under ℓ_∞-ball perturbations. As for empirical defences, we consider adversarial training (AT) [15, 27], which is one of the most popular and effective defences to date.

Models and robust evaluation We consider the ℓ_2-ball perturbations threat model and use the strongest version of AutoAttack (AA+) [7] to reliably evaluate the robust error. For CIFAR-10, we train a residual network (ResNet) and for MNIST we train a convolutional neural network (CNN). Both architectures were introduced in Wong et al. [40] as standard benchmarks for certified defences. For Tiny ImageNet, we train a WideResNet along the same lines of Xu et al. [42]. We refer the reader to Appendix D.3 for complete experimental details.

Certified defences hurt standard and robust error Several studies have shown that adversarial training may lead to an increase in standard error when compared with standard training [32, 37, 44]. We observe the same phenomenon to a much higher degree in certified defences. In particular, our experimental results show that certified defences not only suffer worse standard error but also worse robust error than adversarial training. First, we observe on both MNIST and CIFAR-10 in Figures 2a and 2b, respectively, that for increasing perturbation budget,

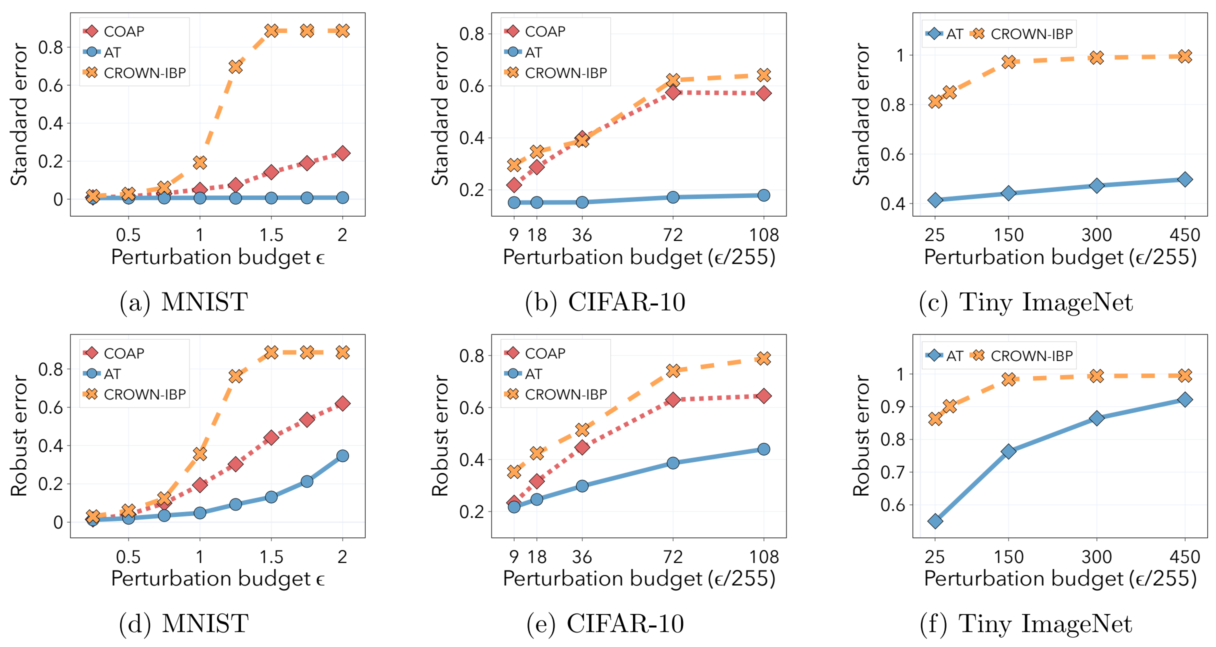


Figure 2: Results for ℓ_2-ball perturbations on MNIST, CIFAR-10 and Tiny ImageNet test sets. In Figures 2a to 2c we plot the standard error as the perturbation budget increases for MNIST, CIFAR-10 and Tiny ImageNet, respectively. In Figures 2d to 2f we plot the robust error as the perturbation budget increases for MNIST, CIFAR-10 and Tiny ImageNet, respectively.

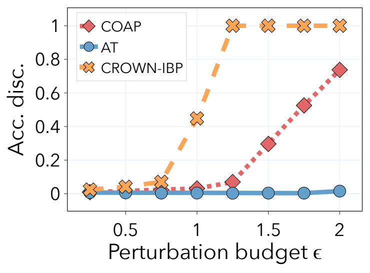
the standard error gap between certified (CROWN-IBP, COAP) and empirical defences (AT) increases. Specifically, the gap reaches almost 90% for CROWN-IBP on MNIST when $\epsilon = 1.5$. Secondly, we observe that the robust error gap increases with increasing perturbation budget for both MNIST and CIFAR-10 in Figures 2d and 2e, respectively. In particular, the gap reaches almost 40% for the largest perturbation budgets. Additionally, we observe a significant standard and robust error gap between AT and CROWN-IBP for Tiny ImageNet in Figures 2c and 2f.

Certified defences hurt fairness We now show that certified defences (CROWN-IBP, COAP) suffer significantly worse fairness than empirical defences (AT). Let $\mathbf{R}(\theta)$ be the standard error of the classifier f_θ and $\mathbf{R}^k(\theta)$ the standard error conditioned on the class label k. The degree of unfairness is measured as follows:

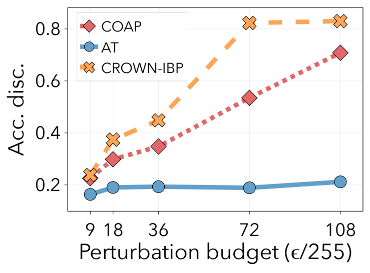
$$\frac{\max_k \mathbf{R}^k(\theta) - \mathbf{R}(\theta)}{1 - \mathbf{R}(\theta)} \tag{2}$$

Using the terminology in Sanyal et al. [34], we refer to this metric as *accuracy discrepancy*. Nevertheless, we expect our results to translate to other related fairness metrics as well [9, 11, 17, 18]. Additionally, we consider the discrepancy in robust accuracy, as it was observed in Xu et al. [41] that adversarial defences may induce a large discrepancy of robustness among different classes. We refer to this metric as *robust accuracy discrepancy* and it consists in replacing the standard error with the robust error in Equation (2).

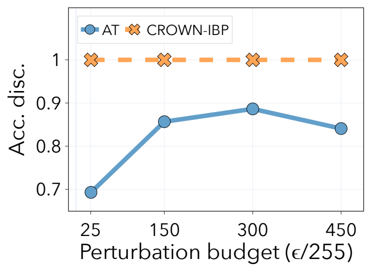
For MNIST, we observe in Figures 3a and 3d that COAP and CROWN-IBP have a significant discrepancy for both standard and robust accuracy. For large perturbations, these methods reach 100% discrepancy, indicating that their accuracy on the worst class can be as low as 0%. By contrast, AT preserves fairness for both standard and robust accuracy much better. In



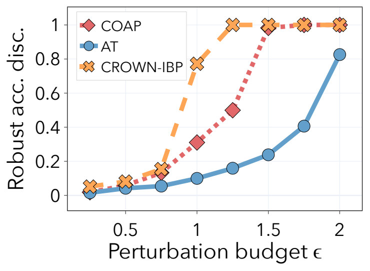
(a) MNIST



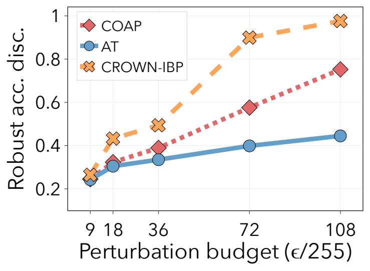
(b) CIFAR-10



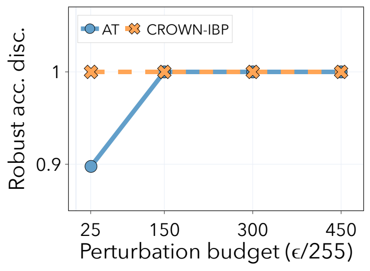
(c) Tiny ImageNet



(d) MNIST



(e) CIFAR-10



(f) Tiny ImageNet

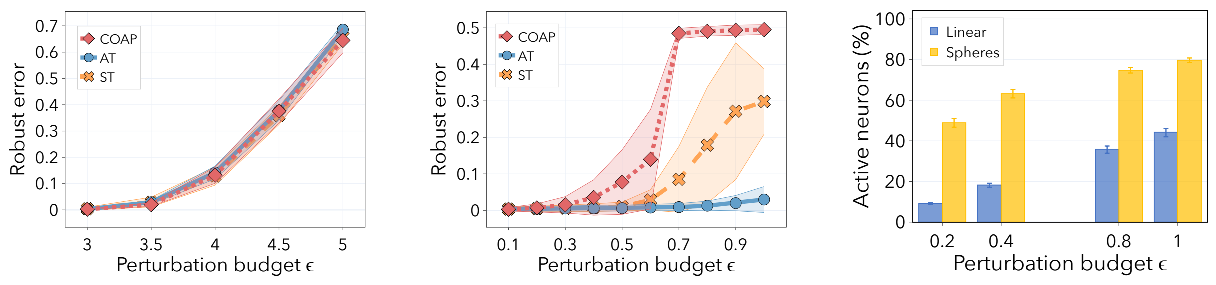
Figure 3: Results for ℓ_2-ball perturbations on MNIST, CIFAR-10 and Tiny ImageNet test sets. In Figures 3a to 3c we plot the accuracy discrepancy as the perturbation budget increases for MNIST, CIFAR-10 and Tiny ImageNet, respectively. In Figures 3d to 3f we plot the robust accuracy discrepancy as the perturbation budget increases for MNIST, CIFAR-10 and Tiny ImageNet, respectively.

particular, the discrepancy for standard accuracy is always less than 2% for all perturbation budgets considered. For CIFAR-10, we observe in Figure 3b that AT maintains a constant accuracy discrepancy around 20% for all perturbation budgets considered, whereas for certified defences it steadily increases with the perturbation budget, reaching above 80%. Similarly, we observe high robust accuracy discrepancy in Figure 3e for certified defences. For Tiny ImageNet, we observe in Figure 3c that AT has significantly better accuracy discrepancy. However, robust accuracy discrepancy is maximal for both empirical and certified defences, as shown in Figure 3f.

3 Intuition: relaxed constraints and signal perturbations

In this section, we hypothesise that certified defences hurt robust and standard generalisation because of (i) the large number of relaxed non-convex constraints and (ii) the strong alignment between the adversarial perturbations and the signal direction. We investigate these hypotheses on more controlled settings. In particular, we consider two synthetic data distributions: a linearly separable distribution as in Clarysse et al. [5], which is similar to the distributions studied in Nagarajan and Kolter [30], Tsipras et al. [37], and the concentric spheres distribution studied in Gilmer et al. [14], Nagarajan and Kolter [30].

Data and threat models Similarly to the previous section, we focus on ℓ_2-ball perturbations of radius ϵ. As for distributions, we consider the linearly separable distribution where first, the label $y \in \{+1, -1\}$ is drawn with equal probability. Then, for some $\gamma > 0$, the covariate vector is $x = [\gamma \operatorname{sgn}(y); \tilde{x}]$, where $\tilde{x} \in \mathbb{R}^{d-1}$ is a random vector drawn from a standard normal distribution $\tilde{x} \sim \mathcal{N}\left(0, \sigma^2 I_{d-1}\right)$ and $[;]$ represents concatenation. We sample the concentric spheres dataset as follows; for $0 < R_1 < R_{-1}$, we first draw a binary label $y \in \{+1, -1\}$ with



(a) Linearly separable (b) Concentric spheres (c) COAP active neurons

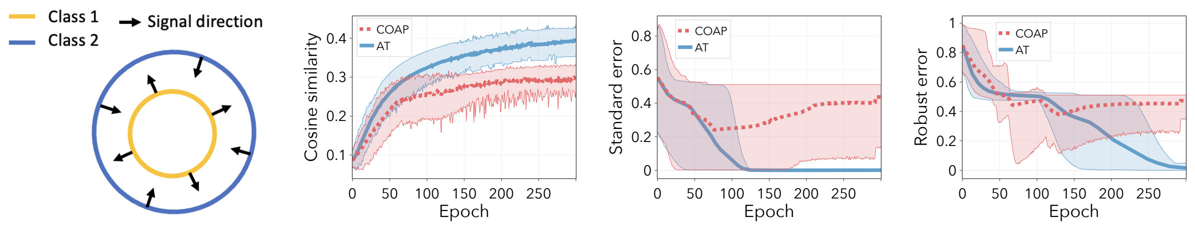
Figure 4: We report mean and standard deviation over 15 seeds. In Figures 4a and 4b we plot the robust error for standard training (ST), adversarial training (AT) and convex outer adversarial polytope (COAP), when training on the linearly separable and concentric spheres distributions respectively. In Figure 4c, we plot the percentage of neurons in the activation set for the linearly separable and concentric spheres distribution respectively. See Appendix D.2 for complete experimental details.

equal probability and then the covariate vector $x \in R^d$ is distributed uniformly on the sphere of radius R_y. Observe that achieving a low test error on the concentric spheres distribution requires a non-linear classifier.

Results In Figures 4a and 4b, we plot the robust error of standard training (ST), adversarial training (AT), and convex outer adversarial polytope (COAP) on the linear and concentric spheres distributions respectively. We see that in contrast to the linear setting, COAP has a much higher robust error on the concentric spheres distribution than AT and ST, where the gap increases with increasing perturbation budget ϵ. The intuition as to why COAP has a much higher robust error than AT on the concentric spheres distribution is two-fold. First, COAP relaxes the non-convex ReLU constraints for all neurons that activate within the perturbation set, i.e., there exists a perturbation $\delta \in \mathcal{B}_\epsilon$ for which the input to the neuron crosses 0. Hence, the larger the percentage of relaxed neurons, the worse the approximation. This is formally captured by Theorem 4.1 in Section 4. Secondly, the ℓ_2-ball perturbations are significantly aligned with the signal direction, meaning that they effectively reduce the information about the label in the data. We will prove that applying an approximation in this direction yields poor generalisation in Theorem 4.2 for the linearly separable distribution.

COAP relaxes many constraints on the concentric spheres In Figure 4c we empirically show that COAP convexly approximates a large number of constraints when trained on the concentric spheres distribution. In particular, we plot the percentage of active neurons on the concentric spheres and linear distributions as the perturbation budget increases. Note that the percentage is much higher for the concentric spheres than for the linearly separable distribution and increases with perturbation budget ϵ. Indeed, the complex spherical decision boundary requires much more active neurons compared to the linear decision boundary which only needs 1 active neuron.

ℓ_2-ball perturbations align with the signal direction We empirically show that ℓ_2-ball perturbations align with the signal direction on the concentric spheres distribution. Note that for a point x drawn from the concentric spheres distribution, the signal direction is given by $y\frac{x}{\|x\|_2}$ (see Figure 5a for a 2D visualization). In Figure 5b, we plot the cosine distance between the ℓ_2-ball perturbations computed on the training set, and the signal direction. Comparing Figures 5b to 5d, we see that during the early stages of training, the ℓ_2-ball perturbations are not



(a) Signal direction (b) Signal alignment (c) Standard error (d) Robust error

Figure 5: We report mean and standard deviation over 15 seeds. In Figure 5a we plot a 2-dimensional realisation of the adversarial spheres dataset, the black arrow illustrates the signal direction. In Figure 5b we plot the cosine similarity between ℓ_2 norm-bounded perturbations on the training data (average) and the signal directed vector. In Figures 5c and 5d we plot standard and robust error for adversarial training (AT) and convex outer adversarial polytope (COAP). We observe that when cosine similarity is high, the gap in standard and robust error between COAP and AT increases. Hence, especially approximations in the signal direction can hurt standard and robust generalisation.

aligned with the signal direction and the robust and standard errors for COAP are similar to AT. However, after some epochs, when the perturbations start to align with the signal direction, both the robust and standard error gaps between COAP and AT increase. This provides evidence that, as training progresses, ℓ_2-ball perturbations become significantly aligned with the signal direction and the generalisation gap worsens.

4 Theoretical results for signal-directed perturbations

In this section, we further investigate our hypothesis that certified defences hurt generalisation when the adversarial perturbations are aligned with the signal direction. First, we introduce a new threat model consisting of perturbations that are perfectly aligned with the signal direction. Then, we extend the convex outer adversarial polytope (COAP) [39] to this setting and prove for a simple neural network that, in high dimensions, certified defences (COAP) yield higher robust error than empirical defences (AT) for large perturbation budgets. Further, we corroborate our theoretical results with experimental evidence on synthetic data.

Data and threat model For the rest of this section, we consider the linearly separable distribution described in Section 3. As for the threat model, we consider signal-directed perturbations that efficiently concentrate their budget on the signal in the input. Since the signal direction corresponds to the first component of the input, we define the set of allowed perturbations as:

$$\mathcal{B}_\epsilon(x) = \{z_1 = x + e_1\beta \mid |\beta| \leq \epsilon\} \tag{3}$$

where e_1 is the standard basis vector of the first coordinate.

4.1 Certified defences for signal-directed perturbations

We now formulate the convex outer adversarial polytope (COAP) [39] for adversaries that concentrate all their budget along the signal direction in the input. Our derivation can be seen as an extension of Wong and Kolter [39], Erdemir et al. [12].

Network architecture For the sake of clarity, we consider a 2-layers feed-forward ReLU network. However, our formulation can easily be extended to multiple layers. We define $f_\theta(x) : \mathbb{R}^d \to \mathbb{R}^2$ as follows:

$$x \xrightarrow{x+\delta} z_1 \xrightarrow{W_1 z_1 + b_1} \hat{z}_2 \xrightarrow{\text{ReLU}(\cdot)} z_2 \xrightarrow{W_2 z_2 + b_2} \hat{z}_3 \tag{4}$$

where $x \in \mathbb{R}^d$, $z_1 \in \mathcal{B}_\epsilon(x)$, W_1 and W_2 are linear operators, and $\theta = \{W_i, b_i\}_{i=1,2}$ is the set of network parameters.

We define the adversarial polytope $\mathcal{Z}_\epsilon(x)$ as the set of all final-layer activations attainable by perturbing x with some $\tilde{x} \in \mathcal{B}_\epsilon(x)$:

$$\mathcal{Z}_\epsilon(x) = \{f_\theta(\tilde{x}) : \tilde{x} \in \mathcal{B}_\epsilon(x)\} \tag{5}$$

The key idea behind COAP [39] is to construct a convex outer bound to this adversarial polytope. Specifically, we relax the ReLU activations $z_2 = \text{ReLU}(\hat{z}_2)$ with their convex envelopes:

$$z_2 \geq 0,\ z_2 \geq \hat{z}_2,\ (u - \ell) z_2 \leq u \hat{z}_2 - u\ell \tag{6}$$

where u and ℓ are respectively the pre-activations $\hat{z}_2$ upper and lower bounds. We assume that these bounds are known and provide a closed form solution to compute them in Appendix A.2. Further, let $\tilde{\mathcal{Z}}_\epsilon(x)$ be the outer bound to the adversarial polytope obtained from relaxing the ReLU constraints. Then, given a data point x with known label y, we can formalise the problem of finding an adversarial example with a linear program as follows:

$$\min_{\hat{z}_3} \, [\hat{z}_3]_y - [\hat{z}_3]_{\bar{y}} = c^\top \hat{z}_3 \qquad \text{s.t. } \hat{z}_3 \in \tilde{\mathcal{Z}}_\epsilon(x) \tag{7}$$

where $\bar{y}$ is the binary negation of y. Note that if we solve this linear program and find that the objective is positive, then we know that no input perturbation within the threat model can misclassify the example. However, solving the linear program in Equation (7) for every example in the dataset is intractable. Therefore, we solve the dual formulation stated in the theorem below, and take a feasible solution, as proposed in Wong and Kolter [39].

Theorem 4.1. *The dual of the linear program* (7) *can be written as*

$$\begin{array}{ll} \max\limits_{\alpha} & \widetilde{J}_\epsilon\left(x, g_\theta(c, \alpha)\right) \\ s.t. & \alpha_j \in [0,1],\ \forall j \end{array}$$

where $\widetilde{J}_\epsilon(x, \nu_1, \nu_2, \nu_3)$ is equal to

$$-\sum_{i=1}^{2} \nu_{i+1}^\top b_i + \sum_{j \in \mathcal{I}} \ell_j [\nu_2]_j^+ - \hat{\nu}_1^\top x - \epsilon \, \|[\hat{\nu}_1]_1\|_1$$

and g_θ is a one-hidden layer neural network given by the equations

$$\begin{array}{l} \nu_3 = -c \\ \hat{\nu}_2 = W_2^\top \nu_3 \\ [\nu_2]_j = 0,\ j \in \mathcal{I}^- \\ [\nu_2]_j = [\hat{\nu}_2]_j,\ j \in \mathcal{I}^+ \\ [\nu_2]_j = \frac{u_j}{u_j - \ell_j} [\hat{\nu}_2]_j^+ - \alpha_j [\hat{\nu}_2]_j^-,\ j \in \mathcal{I} \\ \hat{\nu}_1 = W_1^\top \nu_2 \end{array}$$

where $\mathcal{I}^-$, $\mathcal{I}^+$ and $\mathcal{I}$ denote the sets of activations in the hidden layer where ℓ and u are both negative, both positive or span zero, respectively.

4.2 Approximations along the signal direction hurt generalisation

We are now ready to present our main result. In particular, we will prove that applying an approximation in the signal direction yields poor generalisation for the linearly separable distribution. Below, we outline our setting which consists of a simple one-neuron neural network trained with one step of gradient descent.

One-step gradient descent We study the early phase of neural network optimisation. Under structural assumptions on the data, it has been proved that one gradient step with sufficiently large learning rate can drastically decrease the training loss [4] and extract task-relevant features [13, 8]. A similar setting was also studied recently in Ba et al. [1] for the MSE loss in the high-dimensional asymptotic limit. We focus on the classification setting with binary cross-entropy loss.

One-neuron neural network We consider the hypothesis class to be the set of one-neuron shallow neural networks $f_\theta : \mathbb{R}^d \to \mathbb{R}$, defined by:

$$f_\theta(x) = a \, \mathrm{ReLU}\left(\theta^\top x\right) + b \tag{8}$$

where $x \in \mathbb{R}^d, \theta \in \mathbb{R}^d, a \in \mathbb{R}$, $b \in \mathbb{R}$ and the only trainable parameter is θ_1. Note that as our distribution is linearly separable, our hypothesis class includes the ground truth.

Below we state our main theorem.

Theorem 4.2. *Let $\mathbf{R}_\epsilon$ be the robust risk of f_θ, defined as follows:*

$$\mathbf{R}_\epsilon(\theta) := \mathbb{P}_{(x,y)}\left[\exists z \in \mathcal{B}_\epsilon(x) : y \neq \mathrm{sgn}\left(f_\theta(z)\right)\right] \tag{9}$$

Further, let $\bar{\theta}$ and $\tilde{\theta}$ be the network parameters after one step of gradient descent with respect to AT and COAP objectives. Assume,

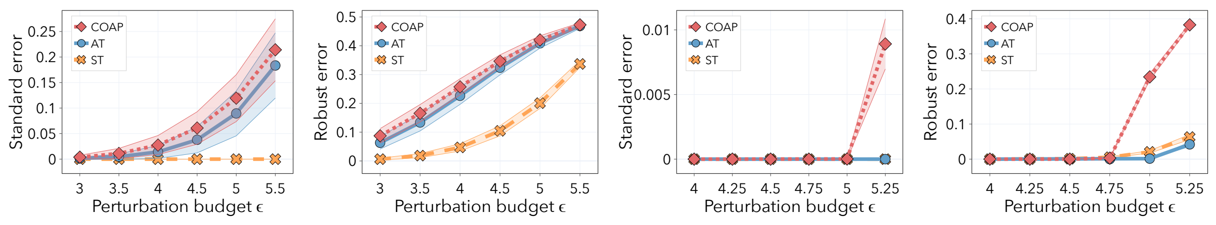
$$\frac{\|\theta_{2:d}\|_2}{\|\theta_1\|_2} > \sqrt{\max\left(\frac{(7\epsilon-\gamma)(\gamma+\epsilon)^4}{4\sigma^2(\gamma^2-10\gamma\epsilon+13\epsilon^2)}, \frac{(\gamma+\epsilon)^3}{12\sigma^2\epsilon}\right)} \quad \text{and} \quad \frac{2}{3}\gamma < \epsilon < \gamma \tag{10}$$

where θ are the network parameters at initialization. Then, COAP yields higher robust risk than AT:

$$\mathbf{R}_\epsilon(\tilde{\theta}) > \mathbf{R}_\epsilon(\bar{\theta}) \tag{11}$$

Theorem 4.2 relies on two main assumptions. The first is an assumption on the data dimensionality and the initialisation of the network parameters θ. For instance, if we initialise the network parameters θ by sampling from a sub-gaussian distribution, then the euclidean norm $\|\theta\|_2$ concentrates around $\sqrt{d}$ with high probability. Hence, the assumption is satisfied when the data dimensionality d is sufficiently high. Further, the second assumption requires that the perturbation budget ϵ is sufficiently close to the separation margin γ. This is consistent with the experimental evidence we presented so far, as the generalisation of certified defences significantly worsen for large perturbation budgets.

Synthetic experiments We corroborate our theory with experimental evidence using a one-hidden layer neural network with 100 neurons. In particular, we investigate the effect of perturbation budget ϵ on generalisation for three different models: standard training (ST), adversarial training (AT) [27, 15] and convex outer adversarial polytope (COAP) [39, 40]. In



(a) Small sample size (b) Small sample size (c) Large sample size (d) Large sample size

Figure 6: We report mean and standard deviation over 15 seeds. In Figure 6a and 6b we plot respectively the standard and robust errors in the small sample size ($n = 50$) regime for standard training (ST), adversarial training (AT) and convex outer adversarial polytope (COAP) as the perturbation budget ϵ increases. In Figure 6c and 6d we plot respectively the standard and robust errors in the large sample size ($n = 10000$) regime for standard training (ST), adversarial training (AT) and convex outer adversarial polytope (COAP) as the perturbation budget ϵ increases. See Appendix D.1 for complete experimental details.

Figure 6, we plot robust and standard errors for both small and large sample size regimes as the perturbation budget ϵ increases. The generalisation gap in the small sample size regime between standard and adversarial training was already observed in Clarysse et al. [5] for linear classifiers. Here, we observe a further generalisation gap between AT and COAP in both small and large sam ple size regimes, which surprisingly worsens in the large sample regime.

5 Conclusions

In this paper, we show that certified defences can hurt accuracy, robustness and fairness for realistic datasets and adversarial perturbations. Further, we develop intuition on synthetic datasets for why certified defences hurt generalisation, combining both theoretical and experimental evidence. We believe that shedding light on the performance gap between empirical and certified defences will not only provide us with a clearer picture of the trade-offs observed in practice but also lead to better approaches for adversarial robustness.

References

[1] Jimmy Ba, Murat A. Erdogdu, Taiji Suzuki, Zhichao Wang, Denny Wu, and Greg Yang. High-dimensional Asymptotics of Feature Learning: How One Gradient Step Improves the Representation, 2022. arXiv:2205.01445.

[2] Battista Biggio, Igino Corona, Davide Maiorca, Blaine Nelson, Nedim Srndic, Pavel Laskov, Giorgio Giacinto, and Fabio Roli. Evasion Attacks against Machine Learning at Test Time. In *Machine Learning and Knowledge Discovery in Databases - European Conference*, 2013.

[3] Avrim Blum, Travis Dick, Naren Manoj, and Hongyang Zhang. Random Smoothing Might be Unable to Certify L\(ınfty\) Robustness for High-Dimensional Images. *Journal of Machine Learning Research*, 2020.

[4] Niladri S. Chatterji, Philip M. Long, and Peter L. Bartlett. When Does Gradient Descent with Logistic Loss Find Interpolating Two-Layer Networks? *Journal of Machine Learning Research*, 2021.

[5] Jacob Clarysse, Julia Hörrmann, and Fanny Yang. Why adversarial training can hurt robust accuracy, 2022. arXiv:2203.02006.

[6] Jeremy M. Cohen, Elan Rosenfeld, and J. Zico Kolter. Certified Adversarial Robustness via Randomized Smoothing. In *Proceedings of the International Conference on Machine Learning*, 2019.

[7] Francesco Croce and Matthias Hein. Reliable evaluation of adversarial robustness with an ensemble of diverse parameter-free attacks. In *Proceedings of the International Conference on Machine Learning*, 2020.

[8] Amit Daniely and Eran Malach. Learning Parities with Neural Networks. In *Advances in Neural Information Processing Systems*, 2020.

[9] John C. Duchi and Hongseok Namkoong. Learning models with uniform performance via distributionally robust optimization. arxiv:1810.08750, 2018.

[10] Krishnamurthy Dvijotham, Robert Stanforth, Sven Gowal, Timothy A. Mann, and Pushmeet Kohli. A Dual Approach to Scalable Verification of Deep Networks. In *Proceedings of the Conference on Uncertainty in Artificial Intelligence*, 2018.

[11] Cynthia Dwork, Moritz Hardt, Toniann Pitassi, Omer Reingold, and Richard S. Zemel. Fairness through awareness. In Shafi Goldwasser, editor, *Innovations in Theoretical Computer Science 2012, Cambridge, MA, USA, January 8-10, 2012*, pages 214–226. ACM, 2012.

[12] Ecenaz Erdemir, Jeffrey Bickford, Luca Melis, and Sergül Aydöre. Adversarial Robustness with Non-uniform Perturbations. In *Advances in Neural Information Processing Systems*, 2021.

[13] Spencer Frei, Niladri S. Chatterji, and Peter L. Bartlett. Random Feature Amplification: Feature Learning and Generalization in Neural Networks, May 2022. arXiv:2202.07626.

[14] Justin Gilmer, Luke Metz, Fartash Faghri, Samuel S. Schoenholz, Maithra Raghu, Martin Wattenberg, and Ian J. Goodfellow. Adversarial Spheres. 2018. arXiv: 1801.02774.

[15] Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and Harnessing Adversarial Examples. In *Proceedings of the International Conference on Learning Representations*, 2015.

[16] Sven Gowal, Krishnamurthy Dvijotham, Robert Stanforth, Rudy Bunel, Chongli Qin, Jonathan Uesato, Relja Arandjelovic, Timothy Arthur Mann, and Pushmeet Kohli. Scalable Verified Training for Provably Robust Image Classification. In *International Conference on Computer Vision*, 2019.

[17] Moritz Hardt, Eric Price, and Nati Srebro. Equality of opportunity in supervised learning. In *Advances in Neural Information Processing Systems*, 2016.

[18] Úrsula Hébert-Johnson, Michael P. Kim, Omer Reingold, and Guy N. Rothblum. Multicalibration: Calibration for the (computationally-identifiable) masses. In *Proceedings of the International Conference on Machine Learning*, 2018.

[19] Guy Katz, Clark W. Barrett, David L. Dill, Kyle Julian, and Mykel J. Kochenderfer. Reluplex: An Efficient SMT Solver for Verifying Deep Neural Networks. In *Proceedings of the International Conference of Computer Aided Verification*, 2017.

[20] Diederik P. Kingma and Jimmy Ba. Adam: A Method for Stochastic Optimization. In *Proceedings of the International Conference on Learning Representations*, 2015.

[21] Alex Krizhevsky. Learning multiple layers of features from tiny images. *citeseer*, 2009.

[22] Aounon Kumar, Alexander Levine, Tom Goldstein, and Soheil Feizi. Curse of Dimensionality on Randomized Smoothing for Certifiable Robustness. In *Proceedings of the International Conference on Machine Learning*, 2020.

[23] Ya Le and Xuan S. Yang. Tiny imagenet visual recognition challenge. 2015.

[24] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proc. IEEE*, (11), 1998.

[25] Bai Li, Changyou Chen, Wenlin Wang, and Lawrence Carin. Certified Adversarial Robustness with Additive Noise. In *Advances in Neural Information Processing Systems*, 2019.

[26] Mathias Lécuyer, Vaggelis Atlidakis, Roxana Geambasu, Daniel Hsu, and Suman Jana. Certified Robustness to Adversarial Examples with Differential Privacy. In *IEEE Symposium on Security and Privacy*, 2019.

[27] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards Deep Learning Models Resistant to Adversarial Attacks. In *Proceedings of the International Conference on Learning Representations*, 2018.

[28] Matthew Mirman, Timon Gehr, and Martin T. Vechev. Differentiable Abstract Interpretation for Provably Robust Neural Networks. In *Proceedings of the International Conference on Machine Learning*, 2018.

[29] Jeet Mohapatra, Ching-Yun Ko, Lily Weng, Pin-Yu Chen, Sijia Liu, and Luca Daniel. Hidden Cost of Randomized Smoothing. In *Proceedings of the International Conference on Artificial Intelligence and Statistics*, 2021.

[30] Vaishnavh Nagarajan and J. Zico Kolter. Uniform convergence may be unable to explain generalization in deep learning. In *Advances in Neural Information Processing Systems*, 2019.

[31] Aditi Raghunathan, Jacob Steinhardt, and Percy Liang. Certified Defenses against Adversarial Examples. In *Proceedings of the International Conference on Learning Representations*, 2018.

[32] Aditi Raghunathan, Sang Michael Xie, Fanny Yang, John C. Duchi, and Percy Liang. Understanding and Mitigating the Tradeoff between Robustness and Accuracy. In *Proceedings of the International Conference on Machine Learning*, 2020.

[33] Hadi Salman, Greg Yang, Huan Zhang, Cho-Jui Hsieh, and Pengchuan Zhang. A Convex Relaxation Barrier to Tight Robustness Verification of Neural Networks. In *Advances in Neural Information Processing Systems*, 2019.

[34] Amartya Sanyal, Yaxi Hu, and Fanny Yang. How unfair is private learning? In *Proceedings of the Conference on Uncertainty in Artificial Intelligence*, 2022.

[35] Gagandeep Singh, Rupanshu Ganvir, Markus Püschel, and Martin T. Vechev. Beyond the Single Neuron Convex Barrier for Neural Network Certification. In *Advances in Neural Information Processing Systems*, 2019.

[36] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian J. Goodfellow, and Rob Fergus. Intriguing properties of neural networks. In *Proceedings of the International Conference on Learning Representations*, 2014.

[37] Dimitris Tsipras, Shibani Santurkar, Logan Engstrom, Alexander Turner, and Aleksander Madry. Robustness May Be at Odds with Accuracy. In *Proceedings of the International Conference on Learning Representations*, 2019.

[38] Tsui-Wei Weng, Huan Zhang, Hongge Chen, Zhao Song, Cho-Jui Hsieh, Luca Daniel, Duane S. Boning, and Inderjit S. Dhillon. Towards fast computation of certified robustness for relu networks. In *Proceedings of the International Conference on Machine Learning*, 2018.

[39] Eric Wong and J. Zico Kolter. Provable Defenses against Adversarial Examples via the Convex Outer Adversarial Polytope. In *Proceedings of the International Conference on Machine Learning*, 2018.

[40] Eric Wong, Frank R. Schmidt, Jan Hendrik Metzen, and J. Zico Kolter. Scaling provable adversarial defenses. In *Advances in Neural Information Processing Systems*, 2018.

[41] Han Xu, Xiaorui Liu, Yaxin Li, Anil K. Jain, and Jiliang Tang. To be Robust or to be Fair: Towards Fairness in Adversarial Training. In *Proceedings of the International Conference on Machine Learning*, 2021.

[42] Kaidi Xu, Zhouxing Shi, Huan Zhang, Yihan Wang, Kai-Wei Chang, Minlie Huang, Bhavya Kailkhura, Xue Lin, and Cho-Jui Hsieh. Automatic Perturbation Analysis for Scalable Certified Robustness and Beyond. In *Advances in Neural Information Processing Systems*, 2020.

[43] Greg Yang, Tony Duan, J. Edward Hu, Hadi Salman, Ilya P. Razenshteyn, and Jerry Li. Randomized Smoothing of All Shapes and Sizes. In *Proceedings of the International Conference on Machine Learning*, pages 10693–10705, 2020.

[44] Hongyang Zhang, Yaodong Yu, Jiantao Jiao, Eric P. Xing, Laurent El Ghaoui, and Michael I. Jordan. Theoretically Principled Trade-off between Robustness and Accuracy. In *Proceedings of the International Conference on Machine Learning*, 2019.

[45] Huan Zhang, Tsui-Wei Weng, Pin-Yu Chen, Cho-Jui Hsieh, and Luca Daniel. Efficient Neural Network Robustness Certification with General Activation Functions. In *Advances in Neural Information Processing Systems*, 2018.

[46] Huan Zhang, Hongge Chen, Chaowei Xiao, Sven Gowal, Robert Stanforth, Bo Li, Duane S. Boning, and Cho-Jui Hsieh. Towards Stable and Efficient Training of Verifiably Robust Neural Networks. In *Proceedings of the International Conference on Learning Representations*, 2020.

A Certified defences for signal-directed perturbations

A.1 Proof of Theorem 4.1

Theorem 4.1. *The dual of the linear program (7) can be written as*

$$\begin{array}{ll}\max\limits_{\alpha} & \widetilde{J}_\epsilon\left(x, g_\theta(c,\alpha)\right) \\ s.t. & \alpha_j \in [0,1],\ \forall j\end{array}$$

where $\widetilde{J}_\epsilon(x, \nu_1, \nu_2, \nu_3)$ is equal to

$$-\sum_{i=1}^{2} \nu_{i+1}^\top b_i + \sum_{j\in\mathcal{I}} \ell_j [\nu_2]_j^+ - \hat{\nu}_1^\top x - \epsilon \left\| [\hat{\nu}_1]_1 \right\|_1$$

and g_θ is a one-hidden layer neural network given by the equations

$$\begin{array}{l}\nu_3 = -c \\ \hat{\nu_2} = W_2^\top \nu_3 \\ [\nu_2]_j = 0,\ j \in \mathcal{I}^- \\ [\nu_2]_j = [\hat{\nu}_2]_j\,,\ j \in \mathcal{I}^+ \\ [\nu_2]_j = \frac{u_j}{u_j-\ell_j}[\hat{\nu}_2]_j^+ - \alpha_j[\hat{\nu}_2]_j^-\,,\ j \in \mathcal{I} \\ \hat{\nu_1} = W_1^\top \nu_2\end{array}$$

where $\mathcal{I}^-$,$\mathcal{I}^+$ and $\mathcal{I}$ denote the sets of activations in the hidden layer where ℓ and u are both negative, both positive or span zero, respectively.

Proof. Consider a data point x and let $\tilde{x} = x + \delta$ be the adversarial perturbed data point. First, we explicit all the constraints for the linear program defined in (7):

$$\begin{array}{l}\min\limits_{\hat{z}_3} [\hat{z}_3]_y - [\hat{z}_3]_{\bar{y}} = c^\top \hat{z}_3\ , \qquad \text{s.t.} \\ x+\delta \in \mathcal{B}_\epsilon(x) \\ z_1 = x + \delta \\ \hat{z}_2 = W_1 z_1 + b_1 \\ \hat{z}_3 = W_2 z_2 + b_2 \\ [z_2]_j = 0,\ \forall j \in \mathcal{I}^- \\ [z_2]_j = [\hat{z}_2]_j,\ \forall j \in \mathcal{I}^+ \\ [z_2]_j \geq 0,\ \forall j \in \mathcal{I} \\ [z_2]_j \geq [\hat{z}_2]_j,\ \forall j \in \mathcal{I} \\ \left((u_j-\ell_j)\,[z_2]_j\ - u_j[\hat{z}_2]_j\right) \leq -u_j\ell_j,\ \forall j \in \mathcal{I}\end{array}$$

where $\mathcal{I}^-$,$\mathcal{I}^+$ and $\mathcal{I}$ denote the sets of activations in the hidden layer where ℓ and u are both negative, both positive, or span zero respectively. In order to compute the dual of this problem, we associate the following Lagrangian variables with each of the constraints:

$$\begin{array}{r}\hat{z}_2 = W_1 z_1 + b_1 \Rightarrow \nu_2 \\ \hat{z}_3 = W_2 z_2 + b_2 \Rightarrow \nu_3 \\ z_1 = x + \delta \Rightarrow \psi \\ -[z_2]_j \leq 0 \Rightarrow \mu_j,\ \forall j \in \mathcal{I} \\ [\hat{z}_2]_j - [z_2]_j \leq 0 \Rightarrow \tau_j,\ \forall j \in \mathcal{I} \\ \left((u_j-\ell_j)\,[z_2]_j - u_j[\hat{z}_2]_j\right) \leq -u_j\ell_j \Rightarrow \lambda_j,\ \forall j \in \mathcal{I}\end{array}$$

note that we do not define explicit dual variables for $[z_2]_j = 0$ and $[z_2]_j = [\hat{z}_2]_j$ as we can easily eliminate them. We write the Lagrangian as follows:

$$\begin{aligned}
\mathcal{L}(z, \hat{z}, \nu, \delta, \lambda, \tau, \mu, \psi) = &- \left(W_1^\top \nu_2 + \psi\right)^\top z_1 - \sum_{j\in\mathcal{I}} \left(\mu_j + \tau_j - \lambda_j (u_j - \ell_j) + \left[W_2^\top \nu_3\right]_j\right) [z_2]_j \\
&+ \sum_{j\in\mathcal{I}} (\tau_j - \lambda_j u_j + [\nu_2]_j) [\hat{z}_2]_j + (c + \nu_3)^\top \hat{z}_3 - \sum_{i=1}^{2} \nu_{i+1}^\top b_i \\
&+ \sum_{j\in\mathcal{I}} \lambda_j u_j \ell_j + \psi^\top x + \psi^\top \delta + \sum_{j\in\mathcal{I}^-} [\hat{z}_2]_j [\nu_2]_j \\
&+ \sum_{j\in\mathcal{I}^+} [z_2]_j \left([\nu_2]_j - [W_2^\top \nu_3]_j\right) \\
\text{s.t. } & \tilde{x} \in \mathcal{B}_\epsilon(x)
\end{aligned}$$

and we take the infimum w.r.t. $z, \hat{z}, \delta$:

$$\begin{aligned}
\inf_{z,\hat{z},\delta} \mathcal{L}(z, \hat{z}, \nu, \delta, \lambda, \tau, \mu, \psi) = &- \inf_{z_2} \sum_{j\in\mathcal{I}} \left(\mu_j + \tau_j - \lambda_j (u_j - \ell_j) + \left[W_2^\top \nu_3\right]_j\right) [z_2]_j \\
&+ \inf_{\hat{z}_2} \sum_{j\in\mathcal{I}} (\tau_j - \lambda_j u_j + [\nu_2]_j) [\hat{z}_2]_j + \inf_{\hat{z}_3} (c + \nu_3)^\top z_3 - \sum_{i=1}^{2} \nu_{i+1}^\top b_i \\
&+ \sum_{j\in\mathcal{I}} \lambda_j u_j \ell_j + \psi^\top x + \inf_{\tilde{x}\in\mathcal{B}_\epsilon(x)} \psi^\top \delta - \inf_{z_1} \left(W_1^\top \nu_2 + \psi\right)^\top z_1 \\
&+ \inf_{\hat{z}_2} \sum_{j\in\mathcal{I}^-} [\hat{z}_2]_j [\nu_2]_j + \inf_{z_2} \sum_{j\in\mathcal{I}^+} [z_2]_j \left([\nu_2]_j - [W_2^\top \nu_3]_j\right)
\end{aligned}$$

Now, we can compute the infimum for the $\psi^\top \delta$ term:

$$\inf_{\tilde{x}\in\mathcal{B}_\epsilon(x)} \psi^\top \delta = \inf_{\|\beta\|_1 \le \epsilon} \psi_1 \cdot \beta = -\epsilon \cdot \|\psi_1\|_1$$

and since for all the other terms the infimum of a linear function is $-\infty$, except in the special case when it is identically zero, the infimum of $\mathcal{L}(\cdot)$ becomes:

$$\inf_{z,\hat{z},\delta} \mathcal{L}(.) = \begin{cases} -\sum_{i=1}^{2} \nu_{i+1}^\top b_i + \sum_{j\in\mathcal{I}} \lambda_j u_j \ell_j + \psi^\top x - \epsilon \|\psi_1\|_1 & \text{if conditions} \\ -\infty & \text{else} \end{cases}$$

where the conditions to satisfy are:

$$\begin{aligned}
&\nu_3 = -c \\
&W_1^\top \nu_2 = -\psi \\
&[\nu_2]_j = 0, j \in \mathcal{I}_i^- \\
&[\nu_2]_j = \left[W_2^\top \nu_3\right]_j, j \in \mathcal{I}_i^+ \\
&\left.\begin{aligned} (u_j - \ell_j)\lambda_j - \mu_j - \tau_j &= \left[W_2^\top \nu_3\right]_j \\ [\nu_2]_j &= u_j \lambda_j - \tau_j \end{aligned}\right\} j \in \mathcal{I} \\
&\lambda, \tau, \mu \ge 0
\end{aligned}$$

Thus, we can rewrite the dual problem as follows:

$$\begin{aligned}
&\max_{\nu,\psi,\lambda,\tau,\mu} \; -\sum_{i=1}^{2} \nu_{i+1}^\top b_i + \sum_{j\in\mathcal{I}} \lambda_j u_j \ell_j + \psi^\top x - \epsilon \left\|\psi_1\right\|_1 \\
&\text{s.t.} \quad \nu_3 = -c \\
&W_1^\top \nu_2 = -\psi \\
&[\nu_2]_j = 0, j \in \mathcal{I}_i^- \\
&[\nu_2]_j = \left[W_2^\top \nu_3\right]_j, j \in \mathcal{I}_i^+ \\
&\left.\begin{array}{ll} (u_j - \ell_j)\lambda_j - \mu_j - \tau_j & = \left[W_2^\top \nu_3\right]_j \\ {[\nu_2]_j} & = u_j\lambda_j - \tau_j \end{array}\right\} j \in \mathcal{I} \\
&\lambda, \tau, \mu \geq 0
\end{aligned}$$

Note that the dual variable λ corresponds to the upper bounds in the ReLU relaxation, while μ and τ correspond to the lower bounds. By the complementarity property, we know that at the optimal solution, these variables will be zero if the ReLU constraint is non-tight, or non-zero if the ReLU constraint is tight. Since the upper and lower bounds cannot be tight simultaneously, either λ or $\mu+\tau$ must be zero. This means that at the optimal solution to the dual problem we can decompose $[W_2^\top \nu_3]_j$ into positive and negative parts since $(u_j - \ell_j)\lambda_j \geq 0$ and $\tau_j + \mu_j \geq 0$:

$$\begin{aligned}
(u_j - \ell_j)\lambda_j &= [W_2^\top \nu_3]_j^+ \\
\tau_j + \mu_j &= [W_2^\top \nu_3]_j^-
\end{aligned}$$

combining this with the constraint $[\nu_2]_j = u_j\lambda_j - \tau_j$ leads to

$$[\nu_2]_j = \frac{u_j}{u_j - \ell_j}[W_2^\top \nu_3]_j^+ - \alpha_j [W_2^\top \nu_3]_j^-$$

for $j \in \mathcal{I}$ and $0 \leq \alpha_j \leq 1$. Hence, we have that:

$$\lambda_j = \frac{u_j}{u_j - \ell_j}[\hat{\nu}_2]_j^+$$

Now, we denote $\hat{\nu}_1 = -\psi$ to make our notation consistent, and putting all of this together the dual objective becomes:

$$\begin{aligned}
-\sum_{i=1}^{2} \nu_{i+1}^\top b_i + \sum_{j\in\mathcal{I}} \lambda_j u_j \ell_j + \psi^\top x - \epsilon \left\|\psi_1\right\|_1 &= -\sum_{i=1}^{2} \nu_{i+1}^\top b_i + \sum_{j\in\mathcal{I}} \frac{u_j \ell_j}{u_j - \ell_j}[\hat{\nu}_2]_j^+ - \hat{\nu}_1^\top x - \epsilon \left\|[\hat{\nu}_1]_1\right\|_1 \\
&= -\sum_{i=1}^{2} \nu_{i+1}^\top b_i + \sum_{j\in\mathcal{I}} \ell_j [\nu_2]_j^+ - \hat{\nu}_1^\top x - \epsilon \left\|[\hat{\nu}_1]_1\right\|_1
\end{aligned}$$

and the final dual problem:

$$\begin{aligned}
\max_{\nu,\hat{\nu}} \quad & -\textstyle\sum_{i=1}^{2} \nu_{i+1}^\top b_i + \sum_{j\in\mathcal{I}} \ell_j [\nu_2]_j^+ - \hat{\nu}_1^\top x - \epsilon \left\|[\hat{\nu}_1]_1\right\|_1 \\
\text{s.t.} \quad & \nu_3 = -c \\
& \hat{\nu}_2 = W_2^\top \nu_3 \\
& [\nu_2]_j = 0, \; j \in \mathcal{I}^- \\
& [\nu_2]_j = [\hat{\nu}_2]_j, \; j \in \mathcal{I}^+ \\
& [\nu_2]_j = \tfrac{u_j}{u_j - \ell_j}[\hat{\nu}_2]_j^+ - \alpha_j [\hat{\nu}_2]_j^-, \; j \in \mathcal{I} \\
& \hat{\nu}_1 = W_1^\top \nu_2
\end{aligned}$$

□

A.2 Computing upper and lower bounds on the pre-activations

We address here the problem of obtaining the upper and lower bounds u and ℓ for the pre-activations $\hat{z}$. Specifically, the following proposition gives a closed form solution.

Proposition A.1. *Consider the neural network f_θ defined in Equation (4). Let w_1 be the first column of W_1. Then, for a data point x and perturbation budget ϵ, we have the following element-wise bounds on the pre-activation vector $\hat{z}_2$:*

$$\ell \leq \hat{z}_2 \leq u$$

where

$$\ell = W_1 x + b_1 - \epsilon |w_1| \text{ and } u = W_1 x + b_1 + \epsilon |w_1|$$

Proof. Given a data point x and perturbation budget ϵ, let $\tilde{x} = x + \delta$ be the perturbed input to the network. First, we find an upper bound the pre-activations values $\hat{z}_2$:

$$\hat{z}_2 = W_1(x + \delta) + b_1 = W_1 x + b_1 + W_1 \delta$$

In particular, we want to solve the following optimisation problem for each component of the pre-activation vector:

$$u_i = \max_{\tilde{x} \in \mathcal{B}_\epsilon(x)} [\hat{z}_2]_i = [W_1 x]_i + [b_1]_i + \max_{\tilde{x} \in \mathcal{B}_\epsilon(x)} [W_1 \delta]_i$$

where u will be the vector containing element-wise upper bounds. Note that $\delta = \beta e_1$, thus the optimisation problem can be rewritten as:

$$\max_{\tilde{x} \in \mathcal{B}_\epsilon(x)} [W_1 \delta]_i = \max_{\|\beta\|_1 \leq \epsilon} \beta \cdot [w_1]_i = \epsilon \cdot \|[w_1]_i\|_1$$

where w_1 is the first column of W_1. The vector of upper bounds will then be:

$$u = W_1 x + b_1 + \epsilon |w_1|$$

Along the same lines, we can derive the vector of lower bounds ℓ:

$$l = W_1 x + b_1 - \epsilon |w_1|$$

□

B Theoretical results for signal-directed perturbations

In this section, we prove that convex relaxations along the signal direction hurt robust generalisation. We focus on the classification setting with binary cross-entropy loss:

$$L(x, y) = y \log(\boldsymbol{\sigma}(x)) + (1 - y) \log(1 - \boldsymbol{\sigma}(x)) \tag{12}$$

where $\boldsymbol{\sigma}(\cdot)$ is the sigmoid function.

First, in Lemma B.1 we relate the robust error of the classifier f_θ to the signal parameter θ_1. Specifically, we show that robust error monotonically decreases in θ_1.

Lemma B.1. *Let f_θ be the neural network defined in Equation (8) and $\mathcal{B}_\epsilon$ the threat model defined in Equation (3). We define the robust risk $\mathbf{R}_\epsilon$ of f_θ as follows:*

$$\mathbf{R}_\epsilon(\theta) := \mathbb{P}_{(x,y)}\left[\exists z \in \mathcal{B}_\epsilon(x) : y \neq \operatorname{sgn}\left(f_\theta(z)\right)\right]$$

Then, $\mathbf{R}_\epsilon(\theta)$ is monotonically decreasing in θ_1.

Proof. We assume, without loss of generality, that $\theta_1 > 0$, and since a and b are not trainable parameters we must have $a > 0$ and $b < 0$ to solve the classification problem. The robust risk is then equal to:

$$\begin{aligned}\mathbf{R}_\epsilon(\theta) &:= \mathbb{P}_{(x,y)}\left[\exists z \in \mathcal{B}_\epsilon(x) : y \neq \operatorname{sgn}\left(f_\theta(z)\right)\right] \\ &= \frac{1}{2}\left(\mathbb{P}_x\left[a \operatorname{ReLU}\left(\theta^\top x\right) + b < 0 \mid y = 1\right] + \mathbb{P}_x\left[a \operatorname{ReLU}\left(\theta^\top x\right) + b > 0 \mid y = -1\right]\right)\end{aligned}$$

further, we can remove the $\operatorname{ReLU}(\cdot)$ using the fact that $\frac{b}{a} < 0$:

$$\begin{aligned}\mathbf{R}_\epsilon(\theta) &= \frac{1}{2}\left(\mathbb{P}_x\left[\{x : \theta^\top x + \frac{b}{a} < 0 \vee \theta^\top x < 0\} \mid y = 1\right] + \mathbb{P}_x\left[\{x : \theta^\top x + \frac{b}{a} > 0 \wedge \theta^\top x > 0\} \mid y = -1\right]\right) \\ &= \frac{1}{2}\left(\mathbb{P}_x\left[\theta^\top x + \frac{b}{a} < 0 \mid y = 1\right] + \mathbb{P}_x\left[\theta^\top x + \frac{b}{a} > 0 \mid y = -1\right]\right) \\ &= \frac{1}{2}\left(\mathbb{P}_x\left[\sum_{i=2}^{d} x_i\theta_i < -\theta_1(\gamma - \epsilon) - \frac{b}{a}\right] + \mathbb{P}_x\left[\sum_{i=2}^{d} x_i\theta_i > \theta_1(\gamma - \epsilon) - \frac{b}{a}\right]\right)\end{aligned}$$

Recall now that for the linearly separable distribution we have $\sum_{i=2}^{d} x_i\theta_i \sim \mathcal{N}\left(0, \sigma^2 \left\|\theta_{2:d}\right\|^2\right)$. Therefore, we can replace the probabilities with the standard normal CDF Φ:

$$\mathbf{R}_\epsilon(\theta) = \frac{1}{2}\left(\Phi\left(-\frac{(\gamma - \epsilon)\theta_1}{\sigma\left\|\theta_{2:d}\right\|_2} - \frac{b}{a\sigma\left\|\theta_{2:d}\right\|_2}\right) + \Phi\left(-\frac{(\gamma - \epsilon)\theta_1}{\sigma\left\|\theta_{2:d}\right\|_2} + \frac{b}{a\sigma\left\|\theta_{2:d}\right\|_2}\right)\right)$$

hence $\mathbf{R}_\epsilon(\theta)$ is monotonically decreasing in θ_1 and the statement follows. □

B.1 Adversarial training

The basic idea behind adversarial training is to update the network parameters according to the following rule:

$$\theta \leftarrow \theta - \frac{\eta}{|D|} \sum_{(x,y) \in D} \nabla_\theta \max_{x+\delta \in \mathcal{B}_\epsilon(x)} L(f_\theta(x + \delta), y)$$

This is usually done by applying some first-order approximation to the maximisation problem. However, for our simplified network we can analytically compute the gradient. First of all, note that when L is the binary cross-entropy loss function we can rewrite the maximisation problem as follows:

$$\max_{x+\delta \in \mathcal{B}_\epsilon(x)} L(f_\theta(x + \delta), y) = L\left(\operatorname{sgn}(y)\overbrace{\min_{x+\delta \in \mathcal{B}_\epsilon(x)} \operatorname{sgn}(y) f_\theta(x + \delta)}^{:= J_\epsilon(x,y)},\ y\right)$$

In particular, if J_ϵ is strictly positive then no adversarial example exists that fools the network. Further, note that this formulation is closely related to the objective considered in Appendix A.1 for the convex outer adversarial polytope. This will be useful when comparing COAP and AT gradients. Below we provide the gradient of the adversarial training objective w.r.t. the network parameters θ.

Proposition B.1. *Consider the neural network f_θ defined in Equation (8) and the threat model $\mathcal{B}_\epsilon$ defined in Equation (3). Let L be the binary cross-entropy loss function, as defined in Equation (12). Then, we have:*

$$
\begin{aligned}
&\nabla_{\theta_1} \max_{x+\delta \in \mathcal{B}_\epsilon(x)} L(f_\theta(x+\delta), y) \\
&= -\operatorname{sgn}(y)\boldsymbol{\sigma}\left(-J_\epsilon(x,y)\right) \begin{cases} a(x_1 - \epsilon \operatorname{sgn}(\theta_1))\mathbf{1}\{\ell > 0\} & \text{if } a\operatorname{sgn}(y) > 0 \\ a(x_1 + \epsilon \operatorname{sgn}(\theta_1))\mathbf{1}\{u > 0\} & \text{if } a\operatorname{sgn}(y) < 0 \end{cases}
\end{aligned}
$$

where $\ell = \theta^\top x - \epsilon|\theta_1|$ and $u = \theta^\top x + \epsilon|\theta_1|$ are respectively lower and upper bounds on the ReLU *inputs.*

Proof. Given a data point x with known label $y \in \{-1, 1\}$, when L is the binary cross-entropy loss function we have:

$$
\max_{x+\delta \in \mathcal{B}_\epsilon(x)} L(f_\theta(x+\delta), y) = L\left(\operatorname{sgn}(y) \min_{x+\delta \in \mathcal{B}_\epsilon(x)} \operatorname{sgn}(y) f_\theta(x+\delta),\ y\right)
$$

For our simplified network we can analytically compute a closed form solution of the minimisation problem:

$$
\begin{aligned}
J_\epsilon &:= \min_{x+\delta \in \mathcal{B}_\epsilon(x)} \operatorname{sgn}(y)\left(b + a\,\mathrm{ReLU}\left(\theta^\top(x+\delta)\right)\right) \\
&= \begin{cases} \operatorname{sgn}(y)\left(b + a\max(0,\ell)\right) & \text{if } a\operatorname{sgn}(y) > 0 \\ \operatorname{sgn}(y)\left(b + a\max(0,u)\right) & \text{if } a\operatorname{sgn}(y) < 0 \end{cases} \\
\\
&= \begin{cases} \operatorname{sgn}(y)\left(b + a\max(0,\ell)\right) & \text{if } \ a\operatorname{sgn}(y) > 0 \\ \operatorname{sgn}(y)\left(b + a\max(0,u)\right) & \text{if } \ a\operatorname{sgn}(y) < 0 \end{cases}
\end{aligned}
$$

where $\ell = \theta^\top x - \epsilon|\theta_1|$ and $u = \theta^\top x + \epsilon|\theta_1|$ are respectively lower and upper bounds on the pre-activations. Thus, we can compute the gradients for adversarial training w.r.t the signal parameter:

$$
\frac{\partial}{\partial\theta_1} J_\epsilon = \begin{cases} \operatorname{sgn}(y) a(x_1 - \epsilon \operatorname{sgn}(\theta_1))\mathbf{1}\{\ell > 0\} & \text{if } a\operatorname{sgn}(y) > 0 \\ \operatorname{sgn}(y) a(x_1 + \epsilon \operatorname{sgn}(\theta_1))\mathbf{1}\{u > 0\} & \text{if } a\operatorname{sgn}(y) < 0 \end{cases}
$$

and applying the chain-rule we have:

$$
\begin{aligned}
&\frac{\partial}{\partial\theta_1} L\left(\operatorname{sgn}(y) J_\epsilon, y\right) \\
&= \frac{\partial}{\partial J_\epsilon} L\left(\operatorname{sgn}(y) J_\epsilon, y\right) \cdot \frac{\partial}{\partial\theta_1} J_\epsilon \\
&= \operatorname{sgn}(y)\left[\boldsymbol{\sigma}\left(\operatorname{sgn}(y) J_\epsilon\right) - \mathbf{1}\{y = 1\}\right] \cdot \frac{\partial}{\partial\theta_1} J_\epsilon \\
&= -\operatorname{sgn}(y)\boldsymbol{\sigma}\left(-J_\epsilon\right) \begin{cases} a(x_1 - \epsilon \operatorname{sgn}(\theta_1))\mathbf{1}\{\ell > 0\} & \text{if } a\operatorname{sgn}(y) > 0 \\ a(x_1 + \epsilon \operatorname{sgn}(\theta_1))\mathbf{1}\{u > 0\} & \text{if } a\operatorname{sgn}(y) < 0 \end{cases}
\end{aligned}
$$

where in the last equality we use a known property of the sigmoid function, $\boldsymbol{\sigma}(x) = 1 - \boldsymbol{\sigma}(-x)$. $\square$

B.2 Convex outer adversarial polytope

We now consider the dual approximation $\widetilde{J}_\epsilon$ to the optimisation problem in Appendix B.1. Note that, for a binary classification problem, we have $c = \operatorname{sgn}(y)$ and the dual objective in Theorem 4.1 becomes:

$$\widetilde{J}_\epsilon\left(x, g_\theta(c, \alpha)\right) = \widetilde{J}_\epsilon\left(x, y\right) \tag{13}$$

where we set α to the dual feasible solution and for the sake of clarity we omit the dependence on the network parameters θ.

We are particularly interested in the data points for which $J_\epsilon(x, y) \neq \widetilde{J}_\epsilon(x, y)$, i.e., when the certified and adversarial training objectives differ. Below, we provide a necessary and sufficient condition to have a mismatch between the two objectives.

Proposition B.2. *Consider the neural network f_θ defined in Equation (8) and the threat model $\mathcal{B}_\epsilon$ defined in Equation (3). Let L be the binary cross-entropy loss function, as defined in Equation (12). Further, we define $\ell = \theta^\top x - \epsilon|\theta_1|$ and $u = \theta^\top x + \epsilon|\theta_1|$ respectively as lower and upper bounds on the* ReLU *inputs. Let $\mathcal{I}^\star = \{(x, y) : 0 \in [\ell, u] \wedge a \operatorname{sgn}(y) > 0\}$. Then, for data points in $\mathcal{I}^\star$, we have that AT and COAP gradients differ:*

$$\nabla_{\theta_1} J_\epsilon(x, y) \neq \nabla_{\theta_1} \widetilde{J}_\epsilon(x, y) \ \ \forall (x, y) \in \mathcal{I}^\star$$

and COAP gradient is given by:

$$\begin{aligned} &\nabla_{\theta_1} L(\operatorname{sgn}(y) \widetilde{J}_\epsilon(x, y), y) \\ &= -\frac{a \operatorname{sgn}(y) \boldsymbol{\sigma}\left(-\widetilde{J}_\epsilon(x, y)\right)}{2\epsilon} \left(\frac{\ell}{\left\|\theta_1\right\|_1}(x_1 + \epsilon \operatorname{sgn}(\theta_1)) + u \frac{x_1 \left\|\theta_1\right\|_1 - \theta^\top x \operatorname{sgn}(\theta_1)}{\theta_1^2}\right) \end{aligned}$$

Further, for data points that are not in $\mathcal{I}^\star$ we have that AT and COAP gradients are equivalent:

$$\nabla_{\theta_1} J_\epsilon(x, y) = \nabla_{\theta_1} \widetilde{J}_\epsilon(x, y) \ \ \forall (x, y) \notin \mathcal{I}^\star$$

Proof. For the sake of clarity, we report here the definition of COAP objective from appendix A.1.

$$\widetilde{J}_\epsilon(x, y) = -\sum_{i=1}^{2} \nu_{i+1}^\top b_i + \sum_{j \in \mathcal{I}} \ell_j [\nu_2]_j^+ - \hat{\nu}_1^\top x - \epsilon \left\|[\hat{\nu}_1]_1\right\|_1$$

Further, recall that the dual variables ν are given by the following equations:

$$\begin{aligned} &\nu_3 = -c \\ &\hat{\nu}_2 = W_2^\top \nu_3 \\ &[\nu_2]_j = 0, \ j \in \mathcal{I}^- \\ &[\nu_2]_j = [\hat{\nu}_2]_j, \ j \in \mathcal{I}^+ \\ &[\nu_2]_j = \tfrac{u_j}{u_j - \ell_j} [\hat{\nu}_2]_j^+ - \alpha_j [\hat{\nu}_2]_j^-, \ j \in \mathcal{I} \\ &\hat{\nu}_1 = W_1^\top \nu_2 \end{aligned}$$

where $\mathcal{I}^-$, $\mathcal{I}^+$ and $\mathcal{I}$ denote the sets of activations in the hidden layer where ℓ and u are both negative, both positive and span zero, respectively.

First, we consider the case when the neuron is always dead, i.e., $\ell < u < 0$. The dual variables are:

$$\begin{aligned} \nu_3 &= -\operatorname{sgn}(y) \\ \hat{\nu}_2 &= -a \operatorname{sgn}(y) \\ \nu_2 &= 0 \\ \hat{\nu}_1 &= 0 \end{aligned}$$

Hence, AT and COAP objectives are equal in this case:

$$\widetilde{J}_\epsilon = \operatorname{sgn}(y) b = J_\epsilon$$

where the last equality follows from Appendix B.1.

Next, we consider the case when the neuron is always active, i.e., $0 < \ell < u$. The dual variables are:

$$\begin{aligned} \nu_3 &= -\operatorname{sgn}(y) \\ \hat{\nu}_2 &= -a \operatorname{sgn}(y) \\ \nu_2 &= -a \operatorname{sgn}(y) \\ \hat{\nu}_1 &= -a \operatorname{sgn}(y) \cdot \theta \end{aligned}$$

and the dual objective becomes:

$$\begin{aligned} \widetilde{J}_\epsilon &= -\nu_3^\top b - \hat{\nu}_1^\top x - \epsilon \left\| [\hat{\nu}_1]_1 \right\|_1 \\ &= \operatorname{sgn}(y) \left(b + a(\theta^\top x) \right) - \epsilon \| a \operatorname{sgn}(y) \theta_1 \| \\ &= \begin{cases} \operatorname{sgn}(y) \left(b + a\ell \right) & \text{if } a \operatorname{sgn}(y) > 0 \\ \operatorname{sgn}(y) \left(b + au \right) & \text{if } a \operatorname{sgn}(y) < 0 \end{cases} \\ &= J_\epsilon \end{aligned}$$

where the last equality follows from the fact that $0 < \ell < u$.

Finally, we consider the case when the neuron is in the activation set $\mathcal{I}$, i.e., $\ell < 0 < u$. The dual variables are:

$$\begin{aligned} \nu_3 &= -\operatorname{sgn}(y) \\ \hat{\nu}_2 &= -a \operatorname{sgn}(y) \\ \nu_2 &= -a \operatorname{sgn}(y) \frac{u}{2\epsilon \left\| \theta_1 \right\|_1} \\ \hat{\nu}_1 &= -a \operatorname{sgn}(y) \frac{u}{2\epsilon \left\| \theta_1 \right\|_1} \cdot \theta \end{aligned}$$

Here we have two cases, when $\hat{\nu}_2 > 0$ we can rewrite the dual objective as:

$$\widetilde{J}_\epsilon = \operatorname{sgn}(y) \left(b + au \right) = J_\epsilon$$

and the two objectives coincide.

When $\nu_2 < 0$ we can rewrite the dual objective as:

$$\widetilde{J}_\epsilon = \operatorname{sgn}(y) \left(b + \frac{au\ell}{2\epsilon \left\| \theta_1 \right\|_1} \right) \neq J_\epsilon$$

Hence, the only case when COAP gradient differs from AT gradient is when $\nu_2 < 0$ and the neuron belongs to the activation set $\mathcal{I}$.

We compute the partial derivative w.r.t. the signal parameter θ_1 in this case, by the chain rule we have:

$$\begin{aligned} &\frac{\partial}{\partial \theta_1} L \left(\operatorname{sgn}(y) \cdot \widetilde{J}_\epsilon, y \right) \\ &= \frac{\partial}{\partial \widetilde{J}_\epsilon} L \left(\operatorname{sgn}(y) \cdot \widetilde{J}_\epsilon, y \right) \cdot \frac{\partial}{\partial \theta_1} \widetilde{J}_\epsilon \\ &= \operatorname{sgn}(y) \left[\boldsymbol{\sigma} \left(\operatorname{sgn}(y) \cdot \widetilde{J}_\epsilon \right) - \mathbf{1}\{y = 1\} \right] \cdot \frac{\partial}{\partial \theta_1} \widetilde{J}_\epsilon \\ &= -\frac{a \operatorname{sgn}(y) \boldsymbol{\sigma} \left(-\widetilde{J}_\epsilon \right)}{2\epsilon} \left(\frac{\ell}{\left\| \theta_1 \right\|_1} (x_1 + \epsilon \operatorname{sgn}(\theta_1)) + u \frac{x_1 \left\| \theta_1 \right\|_1 - \theta^\top x \operatorname{sgn}(\theta_1)}{\theta_1^2} \right) \end{aligned}$$

$\square$

B.3 Proof of Theorem 4.2

Theorem 4.2. *Let $\mathbf{R}_\epsilon$ be the robust risk of f_θ, defined as follows:*

$$\mathbf{R}_\epsilon(\theta) := \mathbb{P}_{(x,y)}\left[\exists z \in \mathcal{B}_\epsilon(x) : y \neq \operatorname{sgn}\left(f_\theta(z)\right)\right] \tag{9}$$

Further, let $\bar{\theta}$ and $\tilde{\theta}$ be the network parameters after one step of gradient descent with respect to AT and COAP objectives. Assume,

$$\frac{\|\theta_{2:d}\|_2}{\|\theta_1\|_2} > \sqrt{\max\left(\frac{(7\epsilon-\gamma)(\gamma+\epsilon)^4}{4\sigma^2(\gamma^2-10\gamma\epsilon+13\epsilon^2)}, \frac{(\gamma+\epsilon)^3}{12\sigma^2\epsilon}\right)} \quad \text{and} \quad \frac{2}{3}\gamma < \epsilon < \gamma \tag{10}$$

where θ are the network parameters at initialization. Then, COAP yields higher robust risk than AT:

$$\mathbf{R}_\epsilon(\tilde{\theta}) > \mathbf{R}_\epsilon(\bar{\theta}) \tag{11}$$

Proof. First we assume, without loss of generality, that at initialisation $\theta_1 > 0$, and since a and b are not trainable parameters we must have $a > 0$ and $b < 0$ to include the ground truth in our hypothesis class.

Let J_ϵ be the adversarial training inner maximisation as defined in equation* (B.1). Then, AT solves the following optimisation problem:

$$\min_\theta \mathbb{E}_{(x,y)}\left[L\left(\boldsymbol{\sigma}\left(\operatorname{sgn}(y)J_\epsilon(x,y)\right), y\right)\right]$$

Similarly, let $\widetilde{J}_\epsilon$ be the COAP dual approximation to the inner maximization described in Equation (13). Then, COAP solves the following optimisation problem:

$$\min_\theta \mathbb{E}_{(x,y)}\left[L\left(\boldsymbol{\sigma}\left(\operatorname{sgn}(y)\widetilde{J}_\epsilon\right), y\right)\right]$$

Since we are only training the signal parameter θ_1, after one gradient descent step, we have:

$$\left\|\bar{\theta}_{2:d}\right\|_2 = \left\|\tilde{\theta}_{2:d}\right\|_2$$

Further, from Lemma B.1 we know that AT yields smaller robust risk than COAP if the following holds:

$$\bar{\theta}_1 > \tilde{\theta}_1 \implies \mathbf{R}_\epsilon(\tilde{\theta}) > \mathbf{R}_\epsilon(\bar{\theta})$$

which, after one step of gradient descent, is equivalent to:

$$\mathbb{E}_{(x,y)}\left[\nabla_{\theta_1} L\left(\boldsymbol{\sigma}\left(\operatorname{sgn}(y)J_\epsilon(x,y)\right), y\right)\right] < \mathbb{E}_{(x,y)}\left[\nabla_{\theta_1} L\left(\boldsymbol{\sigma}\left(\operatorname{sgn}(y)\widetilde{J}_\epsilon(x,y)\right), y\right)\right]$$

Now recall from Theorems B.1 and B.2 that the gradients of AT and COAP differ only on the set $\mathcal{I}^\star$. In particular, we have that:

$$(x,y) \notin \mathcal{I}^\star \implies \nabla_{\theta_1} L\left(\boldsymbol{\sigma}\left(\operatorname{sgn}(y)J_\epsilon(x,y)\right), y\right) = \nabla_{\theta_1} L\left(\boldsymbol{\sigma}\left(\operatorname{sgn}(y)\widetilde{J}_\epsilon(x,y)\right), y\right) < 0$$

and

$$(x,y) \in \mathcal{I}^\star \implies 0 = \nabla_{\theta_1} L\left(\boldsymbol{\sigma}\left(\operatorname{sgn}(y)J_\epsilon(x,y)\right), y\right) \neq \nabla_{\theta_1} L\left(\boldsymbol{\sigma}\left(\operatorname{sgn}(y)\widetilde{J}_\epsilon(x,y)\right), y\right)$$

Hence, for our purpose we need to show that:

$$\mathbb{E}_{(x,y)}\left[\nabla_{\theta_1} L\left(\boldsymbol{\sigma}\left(\operatorname{sgn}(y)\widetilde{J}_\epsilon(x,y)\right), y\right) \mid (x,y)\in\mathcal{I}^\star\right] > 0 \tag{14}$$

Our strategy will be to lower-bound the expectation in Equation (14) with some strictly positive quantity. We define

$$Z = \sum_{i=2}^{d} \theta_i x_i$$

and plug-in the gradient computed in Theorem B.2:

$$\begin{aligned}
&\mathbb{E}_{(x,y)}\left[\nabla_{\theta_1} L\left(\operatorname{sgn}(y)\boldsymbol{\sigma}\left(\widetilde{J}_\epsilon(x,y)\right), y\right) \mid (x,y)\in\mathcal{I}^\star\right]\\
&= \mathbb{E}_{(x,y)}\left[\frac{a\boldsymbol{\sigma}\left(-\widetilde{J}_\epsilon(x,y)\right)}{2\epsilon}\left(-\frac{\ell}{\theta_1}(\gamma+\epsilon)+u\frac{\sum_{i=2}^{d} x_i\theta_i}{\theta_1^2}\right) \mid (x,y)\in\mathcal{I}^\star\right]\\
&= \frac{a}{2\theta_1\epsilon}\mathbb{E}_{(x,y)}\left[\boldsymbol{\sigma}\left(-\widetilde{J}_\epsilon(x,y)\right)\left(-\ell(\gamma+\epsilon)+u\frac{Z}{\theta_1}\right) \mid (x,y)\in\mathcal{I}^\star\right]\\
&= \frac{a}{2\theta_1\epsilon}\mathbb{E}_{(x,y)}\left[\boldsymbol{\sigma}\left(-\widetilde{J}_\epsilon(x,y)\right)u\frac{Z}{\theta_1} - \boldsymbol{\sigma}\left(-\widetilde{J}_\epsilon(x,y)\right)\ell(\gamma+\epsilon) \mid (x,y)\in\mathcal{I}^\star\right]
\end{aligned}$$

Now, we observe that Z is always negative on the set $\mathcal{I}^\star$, since we need to satisfy the constraint $\ell < 0 < u$:

$$(x,y)\in\mathcal{I}^\star \implies -\theta_1(\gamma+\epsilon) < \sum_{i=2}^{d}\theta_i x_i < -\theta_1(\gamma-\epsilon) < 0$$

Further, from Appendix B.2 we have:

$$(x,y)\in\mathcal{I}^\star \implies \boldsymbol{\sigma}\left(-\widetilde{J}_\epsilon(x,y)\right) \geq \frac{1}{2}$$

Combining these two observations we can lower-bound the expectation:

$$\begin{aligned}
&\mathbb{E}_{(x,y)}\left[\nabla_{\theta_1} L\left(\operatorname{sgn}(y)\boldsymbol{\sigma}\left(\widetilde{J}_\epsilon(x,y)\right), y\right) \mid (x,y)\in\mathcal{I}^\star\right]\\
&= \frac{a}{2\theta_1\epsilon}\mathbb{E}_{(x,y)}\left[\boldsymbol{\sigma}\left(-\widetilde{J}_\epsilon(x,y)\right)u\frac{Z}{\theta_1} - \boldsymbol{\sigma}\left(-\widetilde{J}_\epsilon(x,y)\right)\ell(\gamma+\epsilon) \mid (x,y)\in\mathcal{I}^\star\right]\\
&\geq \frac{a}{2\theta_1\epsilon}\mathbb{E}_{(x,y)}\left[u\frac{Z}{\theta_1} - \frac{\gamma+\epsilon}{2}\ell \mid (x,y)\in\mathcal{I}^\star\right]
\end{aligned}$$

Now, we need to show that this lower-bound is strictly positive:

$$\mathbb{E}_{(x,y)}\left[u\frac{Z}{\theta_1} - \frac{\gamma+\epsilon}{2}\ell \mid (x,y)\in\mathcal{I}^\star\right] > 0$$

Note that, we can further expand this expression:

$$\begin{aligned}
&\mathbb{E}_{(x,y)}\left[u\frac{Z}{\theta_1} - \frac{\gamma+\epsilon}{2}\ell \mid (x,y)\in\mathcal{I}^\star\right]\\
&= -(\gamma^2-\epsilon^2)\theta_1^2 + (\gamma+\epsilon)\theta_1\mathbb{E}\left[Z \mid (x,y)\in\mathcal{I}^\star\right] + 2\mathbb{E}\left[Z^2 \mid (x,y)\in\mathcal{I}^\star\right]
\end{aligned}$$

Further, $Z \mid (x,y) \in \mathcal{I}^\star$ is distributed as a truncated normal with:

$$\alpha = -\frac{\theta_1(\gamma+\epsilon)}{\sigma \left\|\theta_{2:d}\right\|_2} \text{ and } \beta = -\frac{\theta_1(\gamma-\epsilon)}{\sigma \left\|\theta_{2:d}\right\|_2}$$

Hence, we can plug in the expectations of the truncated normal distribution to obtain the following:

$$\begin{aligned}
&-(\gamma^2-\epsilon^2)\theta_1^2 + \theta_1(\gamma+\epsilon)\mathbb{E}\left[Z \mid (x,y)\in\mathcal{I}^\star\right] + 2\mathbb{E}\left[Z^2 \mid (x,y)\in\mathcal{I}^\star\right] \\
&= -(\gamma^2-\epsilon^2)\theta_1^2 + 2\sigma^2\left\|\theta_{2:d}\right\|_2^2 + \sigma\left\|\theta_{2:d}\right\|_2\theta_1 \frac{(\gamma-3\epsilon)\phi(\beta)-(\gamma+\epsilon)\phi(\alpha)}{\Phi(\beta)-\Phi(\alpha)} \\
&\propto -(\gamma^2-\epsilon^2) + 2\sigma^2 r^2 + \sigma r\frac{(\gamma-3\epsilon)\phi(\beta)-(\gamma+\epsilon)\phi(\alpha)}{\Phi(\beta)-\Phi(\alpha)} \\
&= -f(r)
\end{aligned}$$

where we define $r = \frac{\|\theta_{2:d}\|_2}{\|\theta_1\|_2}$. Now, under our assumptions, from Lemma C.3 we have:

$$f(r) < 0,\ \forall r > \sqrt{\frac{24\gamma^3}{\sigma^2}}$$

which concludes the proof. □

C Auxiliary lemmas

C.1 Upper bound on the exponential function

Lemma C.1. *Let $f:\mathbb{R}\to\mathbb{R}$ be the function defined by $f(x)=\exp(x)$. When $x\le 0$ and n is even we have:*

$$f(x) \le 1 + x + \frac{x^2}{2!} + \cdots + \frac{x^n}{n!}$$

Proof. Let $g:(-\infty,0]\to\mathbb{R}$ be the function defined by

$$g(x) = 1 + x + \frac{x^2}{2!} + \cdots + \frac{x^n}{n!} - \exp(x)$$

Since $g(x)\to\infty$ as $x\to-\infty$, g must attain an absolute minimum somewhere on the interval $(-\infty,0]$. Now, differentiating we have:

- If g has an absolute minimum at 0 , then for all $x, g(x)\ge g(0) = 1-\exp(0)=0$, so we are done.
- If g has an absolute minimum at y for some $y<0$, then $g'(y)=0$. But differentiating,

  $$g'(y) = 1 + y + \frac{y^2}{2!} + \cdots + \frac{y^{n-1}}{(n-1)!} - \exp(y) = g(y) - \frac{y^n}{n!}.$$

  Therefore, for any x,

  $$g(x) \ge g(y) = \frac{y^n}{n!} + g'(y) = \frac{y^n}{n!} > 0,$$

  since n is even.

□

C.2 Lower bound on the difference of Gaussian CDFs

Lemma C.2. *Let $f : \mathbb{R}^2 \to \mathbb{R}$ be the function defined by $f(x, y) = \Phi(y) - \Phi(x)$. When $x < y < 0$ we have:*

$$\phi(0)\left(y - x + \frac{x^3}{6}\right) \leq \Phi(y) - \Phi(x)$$

where Φ and ϕ are respectively the CDF and PDF of the standard Gaussian distribution.

Proof. First, we want to prove that $\frac{2x}{\sqrt{\pi}}$ is a lower bound for the error function $\text{erf}(x)$ when $x \leq 0$. That is, we want to show that $f(x) \geq 0$ where $f : (-\infty, 0] \to \mathbb{R}$ is the function defined by:

$$f(x) = \text{erf}(x) - \frac{2x}{\sqrt{\pi}}$$

Since f is continuous and $f(x) \to \infty$ as $x \to -\infty$, f must attain an absolute minimimum on the interval $(-\infty, 0]$. Now, differentiating we have:

$$f'(x) = \frac{2}{\sqrt{\pi}}\exp(-x^2) - \frac{2}{\sqrt{\pi}}$$

hence f attains an absolute minimum at 0 and we have $f(x) \geq f(0) = 0$.
Next, we show that $\frac{2}{\sqrt{\pi}}(x - x^3/3)$ is an upper bound for $\text{erf}(x)$ when $x \leq 0$. Let $g : (-\infty, 0] \to \mathbb{R}$ the function defined by:

$$g(x) = \frac{2}{\sqrt{\pi}}(x - x^3/3) - \text{erf}(x)$$

Similarly, since g is continuous and $g(x) \to \infty$ as $x \to -\infty$, g must attain an absolute minimimum on the interval $(-\infty, 0]$. Now, differentiating we have:

$$g'(x) = \frac{2}{\sqrt{\pi}}(1 - x^2 - \exp(-x^2))$$

hence g attains an absolute minimum at 0 and we have $g(x) \geq g(0) = 0$.
Now, since $a < b < 0$ we can use the erf bounds derived above:

$$\begin{aligned}\Phi(b) - \Phi(a) &= \frac{1}{2}\left(\text{erf}(b/\sqrt{2}) - \text{erf}(a/\sqrt{2})\right) \\ &\geq \frac{1}{\sqrt{\pi}}\left(\frac{b}{\sqrt{2}} - \frac{a}{\sqrt{2}} + \frac{a^3}{6\sqrt{2}}\right) \\ &= \phi(0)\left(b - a + \frac{a^3}{6}\right)\end{aligned}$$

which concludes the proof. □

C.3 Upper bound on the ratio of Gaussian PDFs and CDFs

Lemma C.3. *Suppose $f : \mathbb{R} \to \mathbb{R}$ is defined as follows:*

$$f(r) = \gamma^2 - \epsilon^2 - 2\sigma^2 r^2 - \sigma r\frac{(\gamma - 3\epsilon)\phi(\beta) - (\gamma + \epsilon)\phi(\alpha)}{\Phi(\beta) - \Phi(\alpha)}$$

where $\alpha := -\frac{\gamma+\epsilon}{r\sigma}$, $\beta := -\frac{\gamma-\epsilon}{r\sigma}$, Φ and ϕ are respectively the standard Gaussian CDF and PDF.
Assume that:

$$\frac{5 + 2\sqrt{3}}{13}\gamma < \epsilon < \gamma$$

Then, we have:

$$f(r) < 0, \ \ \forall r > \sqrt{\frac{24\gamma^3}{\sigma^2}}$$

Proof. We begin by providing a lower bound on the difference of gaussian cdfs. Applying Lemma C.2 with $x = \alpha$ and $y = \beta$ we have:

$$\Phi(\beta) - \Phi(\alpha) \geq \left(\frac{2\epsilon}{r\sigma} - \frac{(\gamma+\epsilon)^3}{6\sigma^3 r^3}\right)\phi(0), \ \ \alpha < \beta < 0$$

Next, we can upper-bound f:

$$\begin{aligned} f(r) &\leq \gamma^2 - \epsilon^2 - 2\sigma^2 r^2 - \sigma r \frac{(\gamma - 3\epsilon)\phi(\beta) - (\gamma+\epsilon)\phi(\alpha)}{\left(\frac{2\epsilon}{\sigma r} - \frac{(\gamma+\epsilon)^3}{6\sigma^3 r^3}\right)\phi(0)} \\ &\leq \gamma^2 - \epsilon^2 - 2\sigma^2 r^2 - \sigma^2 r^2 \frac{(\gamma - 3\epsilon)\phi(0) - (\gamma+\epsilon)\phi(\alpha)}{\left(2\epsilon - \frac{(\gamma+\epsilon)^3}{6r^2\sigma^2}\right)\phi(0)} \\ &= \gamma^2 - \epsilon^2 - 2\sigma^2 r^2 - \sigma^2 r^2 \frac{(\gamma - 3\epsilon) - (\gamma+\epsilon)\exp(-\alpha^2/2)}{2\epsilon - \frac{(\gamma+\epsilon)^3}{6\sigma^2 r^2}} \end{aligned}$$

Now, we use the upper-bound for the exponential function from Lemma C.1 with $n = 2$:

$$\exp(x) \leq 1 + x - x^2/2, \ \ \forall x \leq 0$$

and substituting it back into our upper-bound for f we get:

$$f(r) \leq \gamma^2 - \epsilon^2 - 2\sigma^2 r^2 - \sigma^2 r^2 \frac{(\gamma - 3\epsilon) - (\gamma+\epsilon)(1 - \frac{(\gamma+\epsilon)^2}{2r^2\sigma^2} + \frac{(\gamma+\epsilon)^4}{8r^4\sigma^4})}{2\epsilon - \frac{(\gamma+\epsilon)^3}{6r^2\sigma^2}}$$

which can be further simplified:

$$\begin{aligned} f(r) &\leq \gamma^2 - \epsilon^2 - 2\sigma^2 r^2 - \sigma^2 r^2 \frac{(\gamma - 3\epsilon) - (\gamma+\epsilon)(1 - \frac{(\gamma+\epsilon)^2}{2r^2\sigma^2} + \frac{(\gamma+\epsilon)^4}{8r^4\sigma^4})}{2\epsilon - \frac{(\gamma+\epsilon)^3}{6r^2\sigma^2}} \\ &= \frac{(\gamma - 7\epsilon)(\gamma+\epsilon)^4 + 4r^2\sigma^2(\gamma+\epsilon)(\gamma^2 - 10\gamma\epsilon + 13\epsilon^2)}{4(\gamma+\epsilon)^3 - 48r^2\sigma^2\epsilon} \\ &= u(r) \end{aligned}$$

and we have that for $\epsilon > \frac{5+2\sqrt{3}}{13}\gamma$ and $r > \sqrt{\max\left(\frac{(7\epsilon-\gamma)(\gamma+\epsilon)^4}{4\sigma^2(\gamma^2-10\gamma\epsilon+13\epsilon^2)}, \frac{(\gamma+\epsilon)^3}{12\sigma^2\epsilon}\right)}$ the upper bound is negative, i.e. $u(r) < 0$. □

D Experimental details

D.1 Synthetic experiments with signal-directed perturbations

Below we provide detailed experimental details to reproduce Figure 6.

Data generation For the linearly separable distribution we set $d = 1000$, $n_{\text{test}} = 10^5$, $\gamma = 6$.

Model and hyper-parameters For all the experiments, we use the one hidden layer architecture defined in Equation (4) with 100 neurons. We use PyTorch SGD optimiser and train all networks for 100 epochs. We sweep over the learning rate $\eta \in \{0.1, 0.01, 0.001\}$ and for each perturbation budget, we choose the one that interpolates the training set and minimises robust error on the test set.

Robust evaluation We perform all the attacks to evaluate robust risk at test-time using exact line search; this is computationally tractable since the attacks are directed along one dimension.

Training paradigms For standard training (ST), wee train the network to minimise the cross-entropy loss. For adversarial training (AT) [27, 15], we train the network to minimise the robust binary cross-entropy loss. At each epoch, we compute an exact adversarial example using line search and update the weights using a gradient with respect to this example. For convex outer adversarial polytope (COAP) [39, 40], at each epoch, we compute upper and lower bounds u and ℓ as described in Theorem A.1. We then train the network to minimise the upper bound on robust error from Theorem 4.1.
Standard training. We train the network to minimise the cross-entropy loss.
Adversarial training [27, 15]. We train the network to minimise the robust binary cross-entropy loss. At each epoch, we compute an exact adversarial example using line search and update the weights using a gradient with respect to this example.
Certified training [39, 40]. At each epoch, we compute upper and lower bounds u and ℓ as described in Proposition A.1. We then train the network to minimize the upper-bound on robust error derived in Theorem 4.1.

D.2 Synthetic experiments with ℓ_2-ball perturbations

Below we provide complete experimental details to reproduce Figures 4 and 5.

Data generation For the spheres dataset, we generate a random $x \in \mathbb{R}^d$ where $\|x\|_2$ is either R_1 or R_{-1} , with equal probability assigned to each norm. We associate with each x a label y such that $y = -1$ if $\|x\|_2 = R_{-1}$ and $y = 1$ if $\|x\|_2 = R_1$. We can sample uniformly from this distribution by sampling $z \sim \mathcal{N}(0, I_d)$ and then setting $x = \frac{z}{\|z\|_2} R_{-1}$ or $x = \frac{z}{\|z\|_2} R_1$. For the linearly separable distribution we set $d = 1000$, $n = 50$, $n_{\text{test}} = 10^5$, $\gamma = 6$. For the concentric spheres distribution we set $d = 100$, $, n = 50$, $n_{\text{test}} = 10^5$, $\gamma_{\min} = 1$ and $\gamma_{\max} = 12$.

Model and hyper-parameters For all the experiments, we use a MLP architecture with $W = 100$ neurons in each hidden layer and $\text{ReLU}(\cdot)$ activation functions. We use PyTorch SGD optimiser with a momentum of 0.95 and train the network for 150 epochs. We sweep over the learning rate $\eta \in \{0.1, 0.01, 0.001\}$ and for each perturbation budget, we choose the one that minimises robust error on the test set and interpolates the training set.

Robust evaluation We consider ℓ_2-ball perturbations. We evaluate robust error at test-time using Auto-PGD [7] with 100 iterations and 5 random restarts. We use both the cross-entropy and difference of logits loss to prevent gradient masking. We use the implementation provided in AutoAttack [7] with minor adjustments to allow for non-image inputs.

Training paradigms For standard training (ST), we train the network to minimise the cross-entropy loss. For adversarial training (AT) [27, 15] we train the network to minimise the robust cross-entropy loss. At each epoch, we search for adversarial examples using Auto-PGD [7] with a budget of 10 steps and 1 random restart. Then, we update the weights using a gradient with respect to the adversarial examples. For convex outer adversarial polytope (COAP) [39, 40]. We train the network to minimise the upper-bound on the robust error. Our implementation is based on the code released by the authors.

D.3 Image experiments

Below we provide complete experimental details to reproduce Figures 2 and 3.

Model architectures For MNIST, we train the CNN architecture with four convolutional layer and two fully connected layers of 512 units introduced in Wong et al. [40]. We report the architectural details in Table 1. For CIFAR-10, we train the residual network (ResNet) with the same structure used in Wong et al. [40]; we use 1 residual block with 16, 16, 32, and 64 filters. For Tiny ImageNet, we train a WideResNet. Following Xu et al. [42] we use 3 wide basic blocks with a widen factor of 10.

CNN
CONV 32 $3 \times 3 + 1$
CONV 32 $4 \times 4 + 2$
CONV 64 $3 \times 3 + 1$
CONV 64 $4 \times 4 + 2$
FC 512
FC 512

Table 1: MNIST model architecture. All layers are followed by ReLU $(\cdot)$ activations. The last fully connected layer is omitted. "CONV k $w \times h + s$" corresponds to a 2D convolutional layer with k filters of size $w \times h$ using a stride of s in both dimensions. "FC n" corresponds to a fully connected layer with n outputs.

Dataset preprocessing For MNIST, we use full 28×28 images without any augmentations and normalisation. For CIFAR-10, we use random horizontal flips and random crops as data augmentation, and normalise images according to per-channel statistics. For Tiny ImageNet, we use random crops of 56×56 and random flips during training. During testing, we use a central 56×56 crop. We also normalise images according to per-channel statistics.

Robust evaluation We consider ℓ_2-ball perturbations. We evaluate the robust error using the most expensive version of AutoAttack (AA+) [7]. Specifically, we include the following attacks: untargeted APGD-CE (5 restarts), untargeted APGD-DLR (5 restarts), untargeted APGD-DLR (5 restarts), Square Attack (5000 queries), targeted APGD-DLR (9 target classes) and targeted FAB (9 target classes).

AT training details For MNIST, we train 100 epochs using Adam optimiser [20] with a learning rate of 0.001, momentum of 0.9 and a batch size of 128; we reduce the learning rate by a factor 0.1 at epochs 40 and 80. For CIFAR-10 with ResNet, we train 150 epochs using SGD with a learning rate of 0.05 and a batch size of 128; we reduce the learning rate by a factor

0.1 at epochs 80 and 120. For Tiny Imagenet and CIFAR-10 with Wide-Resnet we train 200 epochs using SGD with a learning rate of 0.1 and a batch size of 512; we reduce the learning rate by a factor 0.1 at epochs 100 and 150. For the inner optimisation of all models and datasets, adversarial examples are generated with 10 iterations of Auto-PGD [7].

COAP training details We follow the settings proposed by the authors and report them here. For MNIST, we use the Adam optimiser [20] with a learning rate of 0.001 and a batch size of 50. We schedule ϵ starting from 0.01 to the desired value over the first 20 epochs, after which we decay the learning rate by a factor of 0.5 every 10 epochs for a total of 60 epochs. For CIFAR-10, we use the SGD optimiser with a learning rate of 0.05 and a batch size of 50. We schedule ϵ starting from 0.001 to the desired value over the first 20 epochs, after which we decay the learning rate by a factor of 0.5 every 10 epochs for a total of 60 epochs. For all datasets and models, we use random projection of 50 dimensions. For all experiments, we use the implementation provided in Wong et al. [40].

CROWN-IBP training details We follow the settings proposed by the authors and report them here. For MNIST, we train 200 epochs with a batch size of 256. We use Adam optimiser [20] and set the learning rate to 5×10^{-4}. We warm up with 10 epochs of regular training, and gradually ramp up ϵ_{train} from 0 to ϵ in 50 epochs. We reduce the learning rate by a factor 0.1 at epoch 130 and 190. For CIFAR-10, we train 2000 epochs with a batch size of 256, and a learning rate of 5×10^{-4}. We warm up for 100 epochs, and ramp-up ϵ for 800 epochs. Learning rate is reduced by a factor 0.1 at epoch 1400 and 1700. For Tiny ImageNet, we train 600 epochs with batch size 128. The first 100 epochs are clean training, then we gradually increase ϵ_{train} with a schedule length of 400. For all datasets, an hyper-parameter β to balance LiRPA bounds and IBP bounds for the output layer is gradually decreased from 1 to 0 (1 for only using LiRPA bounds and 0 for only using IBP bounds), with the same schedule of ϵ. For all experiments, we use the implementation provided in the auto LiRPA library [42].