Learning Better Certified Models from Empirically-Robust Teachers

Alessandro De Palma*
Department of Statistics
London School of Economics and Political Science
London, United Kingdom
a.de-palma@lse.ac.uk

Abstract

Adversarial training attains strong empirical robustness to specific adversarial attacks by training on concrete adversarial perturbations, but it produces neural networks that are not amenable to strong robustness certificates through neural network verification. On the other hand, earlier certified training schemes directly train on bounds from network relaxations to obtain models that are certifiably robust, but display sub-par standard performance. Recent work has shown that state-of-the-art trade-offs between certified robustness and standard performance can be obtained through a family of losses combining adversarial outputs and neural network bounds. Nevertheless, differently from empirical robustness, verifiability still comes at a significant cost in standard performance. In this work, we propose to leverage empirically-robust teachers to improve the performance of certifiably-robust models through knowledge distillation. Using versatile feature-space distillation objectives, we show that distillation from adversarially-trained teachers can significantly improve the performance of state-of-the-art certified training algorithms for ReLU networks on robust computer vision benchmarks.

1 Introduction

Adversarial examples [7, 29, 62], imperceptible input perturbations that induce unintended behaviors such as misclassifications, hinder the applicability of deep learning systems to safety-critical applications. Formal robustness certificates can be obtained through neural network verification algorithms [23, 46, 63]. However, these techniques have a worst-case runtime that is exponential in network size even on piecewise-linear models [39]. While *empirical* robustness to specific adversarial attacks can be attained by training against adversarial inputs [47], a technique known as adversarial training, neural network verifiers fail to provide robustness certificates on the resulting networks in a reasonable time. This is in spite of sustained progress in verification algorithms, which now couple specialized network convex relaxations [20, 66, 69, 74] with efficient divide and conquer strategies [10, 32] within hardware-accelerated branch-and-bound frameworks [18, 25, 65].

Networks amenable to robustness certificates through neural network verification can be obtained using so-called certified training schemes. Earlier approaches [30, 53, 66, 75] proposed to train networks by computing the loss on neural network bounds obtained from convex relaxations, effectively deploying a building block of network verifiers within the training loop. Counter-intuitively, and owing to their relative smoothness and continuity, the loosest relaxations were found to outperform the others in this context [38, 43]. While the resulting networks enjoy strong verifiability using the same relaxations employed at training time, this is achieved at a significant cost in standard performance. Relying on branch-and-bound frameworks to perform post-training verification, a more recent line of

*Work partly carried out at Inria, École Normale Supérieure, PSL University, CNRS.

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

works has shown that better trade-offs between certified robustness and standard performance can be obtained by combining methods based on convex relaxations with adversarial training [6, 19, 48, 55]. This was then formalized into the notion of *expressivity* [21], which entails the ability of a certified training loss to span a continuous range of trade-offs between pure adversarial training and the earlier losses based on convex relaxations, and can be easily implemented through convex combinations.

While, as attested by a recent study [51], expressive losses produce state-of-the-art certifiably-robust networks for the ℓ_∞ threat model, their standard performance is still far from ideal. Noting that adversarially-trained models display significantly better standard performance and empirical robustness than certifiably-robust models [16], we believe they could be directly employed to improve the certified training process. Specifically, we aim to produce better certifiably-robust models by performing knowledge distillation [35, 58] from an empirically-robust teacher to the target model, leading to the following contributions:

- Inspired by a worst-case loss common in the adversarial robustness literature, we introduce a novel and versatile feature-space distillation framework for certified training (§3.1).
- Focusing on ReLU networks and on the ℓ_∞ threat model, we present distillation losses tailored to a series of popular certified training algorithms (§3.2, §3.3), demonstrating that distillation from adversarially-trained teachers can significantly improve the robustness-accuracy trade-offs of state-of-the-art certified training algorithms on robust computer vision benchmarks (§5.1).
- We show that students successfully learn from an adversarially-trained teacher while at the same time significantly surpassing it in terms of certified robustness (§5.2). In other words, they partly inherit the benefits of the representations from empirically-robust models while preserving verifiability through certified training losses.

We believe our work to be a further step towards bridging the ever-present gap between certified and empirical adversarial robustness. Code and sample trained models are available at `https://github.com/alessandrodepalma/feature-distillation-for-certified-training`.

2 Background

Let lowercase letters denote scalars (e.g., $a \in \mathbb{R}$) boldface lowercase letters denote vectors (e.g., $\mathbf{a} \in \mathbb{R}^n$), uppercase letters denote matrices (e.g., $A \in \mathbb{R}^{n \times m}$), calligraphic letters denote sets (e.g. $\mathcal{A} \subset \mathbb{R}^n$) and brackets denote intervals (e.g., $[\mathbf{a}, \mathbf{b}]$). Furthermore, let lowercase single-letter subscripts denote vector indices (for instance, $\mathbf{a}_i$, or $f(\mathbf{a})_i$ for a vector-valued function f), and let the vector of all entries of $\mathbf{a}$ except its i-th entry be denoted by $\mathbf{a}_{\bar{i}}$. We will write $\mathbf{1}^n \in \mathbb{R}^n$ for the unit vector, and $I^n \in \mathbb{R}^{n \times n}$ for the identity matrix. Abusing notation, given a vector-valued objective function $a(\mathbf{x})$ we will denote by $\min_{\mathbf{x} \in \mathcal{A}} a(\mathbf{x})$ the vector storing the minimum of $\min_{\mathbf{x} \in \mathcal{A}} a(\mathbf{x})_i$ in its i-th entry. Finally, we use the following shorthands: $[A]_+ := \max\{0, A\}$, $[A]_- := \min\{0, A\}$, $[\![a, b]\!] := \{a, a+1, \dots, b\}$. Appendix I provides further details and clarifications on the employed notation.

Let $(\mathbf{x}, y) \sim \mathcal{D}$ be a k-way classification dataset with points $\mathbf{x} \in \mathbb{R}^d$ and labels $y \in [\![1, k]\!]$. We aim to train a feed-forward ReLU neural network $f_{\boldsymbol{\theta}} : \mathbb{R}^d \to \mathbb{R}^k$ with parameters $\boldsymbol{\theta} \in \mathbb{R}^p$ such that each point $\mathbf{x}$ from $\mathcal{D}$ and the allowed adversarial perturbations $\mathcal{C}_\epsilon(\mathbf{x}) := \{\mathbf{x}' : \|\mathbf{x}' - \mathbf{x}\|_\infty \leq \epsilon\}$ around it are correctly classified:

$$\mathbf{x}' \in \mathcal{C}_\epsilon(\mathbf{x}) \implies \operatorname*{argmax}_i f_{\boldsymbol{\theta}}(\mathbf{x})_i = y. \tag{1}$$

For ease of notation, we will henceforth omit network parameters $\boldsymbol{\theta}$.

2.1 Neural network verification

Neural network verification can be used to check whether equation (1) holds on a given network f, providing a deterministic robustness certificate. Let $\mathbf{z}_f(\mathbf{x}, y)$ denote the differences between the ground truth logits, i.e., unnormalized network outputs corresponding to the correct class, and the other logits: $\mathbf{z}_f(\mathbf{x}, y) := \big(f(\mathbf{x})_y \mathbf{1}^k - f(\mathbf{x})\big)$. Verifiers solve the following optimization problem:

$$\mathbf{z}^{\mathcal{C}_\epsilon}(\mathbf{x}, y) := \min_{\mathbf{x}' \in \mathcal{C}_\epsilon(\mathbf{x})} \mathbf{z}_f(\mathbf{x}', y). \tag{2}$$

If all entries of $\mathbf{z}^{\mathcal{C}_\epsilon}(\mathbf{x}, y)_{\bar{y}}$ are positive, then equation (1) holds, implying that the network is guaranteed to be robust to any given attack in $\mathcal{C}_\epsilon(\mathbf{x})$. An algorithm computing $\mathbf{z}^{\mathcal{C}_\epsilon}(\mathbf{x}, y)$ exactly is called

a complete verifier: this is typically done through branch-and-bound [8, 20, 25, 32, 65]. As this was shown to be NP-complete [39], a series of algorithms propose to compute less expensive lower bounds $\underline{\mathbf{z}}^{\mathcal{C}_\epsilon}(\mathbf{x}, y) \leq \mathbf{z}^{\mathcal{C}_\epsilon}(\mathbf{x}, y)$ [22, 60, 66, 74] by operating on network relaxations (incomplete verifiers), with $\underline{\mathbf{z}}^{\mathcal{C}_\epsilon}(\mathbf{x}, y)_{\bar{y}} > 0$ successfully providing a robustness certificate. The least expensive incomplete verifier is called Interval Bound Propagation (IBP) [30, 53], which is obtained by applying interval arithmetics [34, 61] to the network operators. We refer the reader to appendix C for further details.

Beyond neural network verifiers, the robustness of a network can be certified using Lipschitz bounds [24], and through randomized smoothing [13], the latter providing a probabilistic certificate. Both techniques are at their most effective on the ℓ_2 threat model, which is beyond the scope of this work.

2.2 Certified training

In principle, a network can be trained for verified robustness (certified training) by replacing the employed classification loss $\mathcal{L} : \mathbb{R}^k \times [\![1, k]\!] \to \mathbb{R}$ (e.g., cross-entropy) with its worst-case across the adversarial perturbations [47]:

$$\mathcal{L}^{\mathcal{C}_\epsilon}(\mathbf{x}, y) := \max_{\mathbf{x}' \in \mathcal{C}_\epsilon(\mathbf{x})} \mathcal{L}(f(\mathbf{x}'), y). \tag{3}$$

Bounding the worst-case loss However, computing $\mathcal{L}^{\mathcal{C}_\epsilon}(\mathbf{x}, y)$ exactly is as hard as solving problem (2), and is hence typically replaced by an approximation. Let $\mathbf{x}_{\text{adv}}$ denote the output of an adversarial attack, for instance PGD [47], on the network f. Lower bounds to $\mathcal{L}^{\mathcal{C}_\epsilon}(\mathbf{x}, y)$ can be obtained by evaluating the loss at $\mathbf{x}_{\text{adv}}$:

$$\underline{\mathcal{L}}^{\mathcal{C}_\epsilon}(\mathbf{x}, y) := \mathcal{L}(f(\mathbf{x}_{\text{adv}}), y) \leq \mathcal{L}^{\mathcal{C}_\epsilon}(\mathbf{x}, y). \tag{4}$$

Networks trained using $\mathcal{L}(f(\mathbf{x}_{\text{adv}}), y)$ (adversarial training) typically enjoy strong empirical robustness to the same types of attacks employed during training, but are not amenable to formal guarantees, which are the focus of this work. Assume the loss is monotonically increasing with respect to the non-ground-truth network logits and that it is translation-invariant: $\mathcal{L}(-\mathbf{z}_f(\mathbf{x}, y), y) = \mathcal{L}(f(\mathbf{x}), y)$ [66]. This holds for common losses such as cross-entropy. Given bounds $\underline{\mathbf{z}}^{\mathcal{C}_\epsilon}(\mathbf{x}, y)$ from an incomplete verifier, let us set $\underline{\mathbf{z}}^{\mathcal{C}_\epsilon}(\mathbf{x}, y)_y = 0$ noting that, by definition, $\mathbf{z}_f(\mathbf{x}, y)_y = 0$. An upper bound to $\mathcal{L}^{\mathcal{C}_\epsilon}(\mathbf{x}, y)$ can then be obtained as:

$$\bar{\mathcal{L}}^{\mathcal{C}_\epsilon}(\mathbf{x}, y) := \mathcal{L}(-\underline{\mathbf{z}}^{\mathcal{C}_\epsilon}(\mathbf{x}, y), y) \geq \mathcal{L}^{\mathcal{C}_\epsilon}(\mathbf{x}, y). \tag{5}$$

While networks trained via $\mathcal{L}(-\underline{\mathbf{z}}^{\mathcal{C}_\epsilon}(\mathbf{x}, y), y)$ display strong verifiability through the same incomplete verifiers used to compute the $\underline{\mathbf{z}}^{\mathcal{C}_\epsilon}(\mathbf{x}, y)$ bounds, the most successful option being IBP [38], they display sub-par standard performance.

Expressive losses More recent and successful methods rely on losses featuring a combination between adversarial attacks and bounds from incomplete verifiers [6, 19, 48, 55], pairing better standard performance with stronger verifiability through branch-and-bound. In particular, the ability of a loss function to span a continuous range of trade-offs between lower and upper bounds to $\mathcal{L}^{\mathcal{C}_\epsilon}(\mathbf{x}, y)$, termed *expressivity*, was found to be crucial to maximize performance [21]. A parametrized loss function $\mathcal{L}^{\mathcal{C}_\epsilon}(\alpha; \mathbf{x}, y)$ is termed [21] expressive if: (i) $\mathcal{L}^{\mathcal{C}_\epsilon}(0; \mathbf{x}, y) = \underline{\mathcal{L}}^{\mathcal{C}_\epsilon}(\mathbf{x}, y)$ and $\mathcal{L}^{\mathcal{C}_\epsilon}(1; \mathbf{x}, y) = \bar{\mathcal{L}}^{\mathcal{C}_\epsilon}(\mathbf{x}, y)$; (ii) $\underline{\mathcal{L}}^{\mathcal{C}_\epsilon}(\mathbf{x}, y) \leq \mathcal{L}^{\mathcal{C}_\epsilon}(\alpha; \mathbf{x}, y) \leq \bar{\mathcal{L}}^{\mathcal{C}_\epsilon}(\mathbf{x}, y) \; \forall \; \alpha \in [0, 1]$; (iii) $\mathcal{L}^{\mathcal{C}_\epsilon}(\alpha; \mathbf{x}, y)$ is continuous and monotonically increasing for $\alpha \in [0, 1]$.

Nevertheless, the best-performing certifiably-robust models still display worse robustness-accuracy trade-offs than empirically-robust models [16, 21].

While we focus on certified training via convex relaxations in this work, we refer the reader to §4 for an overview of complementary techniques.

2.3 Knowledge distillation

Knowledge distillation [35] was introduced as a way to transfer the predictive ability of a large teacher model $t : \mathbb{R}^d \to \mathbb{R}^k$ onto the target model f, termed the student. Let $\text{KL}_T : \mathbb{R}^k \times \mathbb{R}^k \to \mathbb{R}$ denote a KL divergence incorporating a softmax operation with temperature T. In its standard form,

knowledge distillation pairs the employed classification loss with a KL divergence term where the teacher logits, termed soft labels, act as target distribution:

$$\mathcal{L}^{\text{KD}}(\lambda; \mathbf{x}, y) := \mathcal{L}(f(\mathbf{x}), y) + T^2 \lambda \, \text{KL}_T(f(\mathbf{x}), t(\mathbf{x})). \tag{6}$$

In order to provide more granular teacher information to the student, a series of works perform knowledge distillation on the intermediate activations [33, 58, 70]: this is referred to as feature-space distillation. Let us write both the teacher and the student as the composition of a classification head with a feature map: $t := g^t \circ h^t$ and $f := g \circ h$, respectively. In its simplest form, when the feature spaces of the teacher and student share the same dimensionality w, feature-space distillation encourages similarity between teacher and student features through a squared ℓ_2 term:

$$\mathcal{L}^{\text{F-KD}}(\lambda; \mathbf{x}, y) := \mathcal{L}(f(\mathbf{x}), y) + \lambda \left\| h(\mathbf{x}) - h^t(\mathbf{x}) \right\|_2^2. \tag{7}$$

Both $\mathcal{L}^{\text{KD}}(\lambda; \mathbf{x}, y)$ and $\mathcal{L}^{\text{F-KD}}(\lambda; \mathbf{x}, y)$ were originally designed to transfer standard network performance from teacher to student, and do not take robustness into account. Recent works have therefore focused on designing specialized distillation schemes that transfer either empirical adversarial robustness [28, 54, 78, 79] or probabilistic certified robustness [64] from teacher to student. Aiming to improve state-of-the-art trade-offs between deterministic certified robustness and standard performance, we here propose to transfer knowledge from an empirically-robust teacher through feature-space distillation.

3 Distillation for certified robustness

Motivated by their superior standard performance and empirical robustness, we aim to leverage adversarially-trained networks to train better certifiably-robust models. We will first present our distillation framework in §3.1, then show how they can be used in conjunction with existing certified training algorithms (§3.2, §3.3). Pseudo-code and proofs of technical results are respectively provided in appendices E and D.

3.1 Distillation from empirically-robust teachers

As outlined in §2.2, adversarial and certified training employ different approximations of the ideal robust loss in equation (3). While certified training schemes attain strong verifiability, they do so at the expense of both empirical robustness and standard performance compared to adversarial training, which arguably learns a more empirically-effective representation of data. We propose to employ feature-space distillation to transfer some of the benefits of this representation, while adopting state-of-the-art certified training losses to enforce verifiability.

Worst-case distillation loss Certified training is concerned with worst-case network behavior. Mirroring the robust loss in §2.2, a worst-case feature-space distillation loss over the perturbations would take the following form:

$$\mathcal{R}^{\mathcal{C}_\epsilon}(\mathbf{x}, y) := \max_{\mathbf{x}' \in \mathcal{C}_\epsilon(\mathbf{x})} \left\| h(\mathbf{x}') - h^t(\mathbf{x}) \right\|_2^2. \tag{8}$$

However, as for $\mathcal{L}^{\mathcal{C}_\epsilon}(\mathbf{x}, y)$, computing $\mathcal{R}^{\mathcal{C}_\epsilon}(\mathbf{x}, y)$ exactly amounts to solving a non-convex optimization problem over the features. Similarly to what done for the robust loss, we propose to use approximations of equation (8). These will be tailored to the employed certified training loss.

Bounding the distillation loss As for equation (4), a lower bound $\underline{\mathcal{R}}^{\mathcal{C}_\epsilon}(\mathbf{x}, y)$ can be obtained by evaluating the left-hand side of equation (8) at the adversarial input $\mathbf{x}_{\text{adv}}$ computed on the student model:

$$\underline{\mathcal{R}}^{\mathcal{C}_\epsilon}(\mathbf{x}, y) := \left\| h(\mathbf{x}_{\text{adv}}) - h^t(\mathbf{x}) \right\|_2^2 \leq \mathcal{R}^{\mathcal{C}_\epsilon}(\mathbf{x}, y).$$

Similarly to $\bar{\mathcal{L}}^{\mathcal{C}_\epsilon}(\mathbf{x}, y)$, an upper bound $\bar{\mathcal{R}}^{\mathcal{C}_\epsilon}(\mathbf{x}, y)$ can be instead computed resorting to IBP.

Proposition 3.1. *Let $\underline{h}^{\mathcal{C}_\epsilon}(\mathbf{x})$ and $\bar{h}^{\mathcal{C}_\epsilon}(\mathbf{x})$ respectively denote IBP lower and upper bounds to the student features:*

$$\underline{h}^{\mathcal{C}_\epsilon}(\mathbf{x}) \leq h(\mathbf{x}') \leq \bar{h}^{\mathcal{C}_\epsilon}(\mathbf{x}), \quad \forall \mathbf{x}' \in \mathcal{C}_\epsilon(\mathbf{x}).$$

The loss function

$$\bar{\mathcal{R}}^{\mathcal{C}_\epsilon}(\mathbf{x}, y) := \sum_i \max\left\{ \left(\underline{h}^{\mathcal{C}_\epsilon}(\mathbf{x})_i - h^t(\mathbf{x})_i\right)^2, \left(\bar{h}^{\mathcal{C}_\epsilon}(\mathbf{x})_i - h^t(\mathbf{x})_i\right)^2 \right\}$$

is an upper bound to the worst-case distillation loss from equation (8)*:* $\bar{\mathcal{R}}^{\mathcal{C}_\epsilon}(\mathbf{x}, y) \geq \mathcal{R}^{\mathcal{C}_\epsilon}(\mathbf{x}, y)$.

Inspired by the parallels between the worst-case robust loss and distillation loss, and their approximations, we aim to use $\underline{\mathcal{R}}^{\mathcal{C}_\epsilon}(\mathbf{x}, y)$ and $\bar{\mathcal{R}}^{\mathcal{C}_\epsilon}(\mathbf{x}, y)$ as building blocks for algorithm-specific distillation losses introduced in the next subsections.

Teacher models We propose to use teacher models trained via pure adversarial training (see §2.2) on the target task, employing the same architecture as the student. Appendix G.8 empirically demonstrates that the empirical robustness of the teacher is essential for the success of the distillation process. Owing to its empirical efficacy (see appendix G.7) we define the features as the activations before the last affine network layer.

3.2 CC-Dist

Aiming to showcase the efficacy of our distillation framework, we primarily focus on distilling onto CC-IBP, a simple state-of-the-art expressive loss. The CC-IBP loss is obtained by evaluating $\mathcal{L}$ on convex combinations $\mathbf{z}_{\text{cc}}^{\mathcal{C}_\epsilon}(\alpha; \mathbf{x}, y) := (1-\alpha)\, \mathbf{z}_f(\mathbf{x}_{\text{adv}}, y) + \alpha\, \underline{\mathbf{z}}^{\mathcal{C}_\epsilon}(\mathbf{x}, y)$ between adversarial logit differences $\mathbf{z}_f(\mathbf{x}_{\text{adv}}, y)$ and their lower bounds $\underline{\mathbf{z}}^{\mathcal{C}_\epsilon}(\mathbf{x}, y)$ computed via IBP:

$$\mathcal{L}_{\text{cc}}^{\mathcal{C}_\epsilon}(\alpha; \mathbf{x}, y) := \mathcal{L}\left(-\mathbf{z}_{\text{cc}}^{\mathcal{C}_\epsilon}(\alpha; \mathbf{x}, y), y\right).$$

Mirroring the properties of the CC-IBP loss, we aim to design a parametrized feature-space distillation loss $\mathcal{R}_{\text{cc}}^{\mathcal{C}_\epsilon}(\alpha; \mathbf{x}, y)$ that can span a continuous range of trade-offs between $\underline{\mathcal{R}}^{\mathcal{C}_\epsilon}(\mathbf{x}, y)$ and $\bar{\mathcal{R}}^{\mathcal{C}_\epsilon}(\mathbf{x}, y)$. As we next show, $\mathcal{R}_{\text{cc}}^{\mathcal{C}_\epsilon}(\alpha; \mathbf{x}, y)$ can be realized by distilling onto the following convex combinations of the adversarial student features $h(\mathbf{x}_{\text{adv}})$ with $\bar{h}^{\mathcal{C}_\epsilon}(\mathbf{x})$ and $\underline{h}^{\mathcal{C}_\epsilon}(\mathbf{x})$:

$$\begin{aligned} \bar{h}_{\text{cc}}^{\mathcal{C}_\epsilon}(\alpha; \mathbf{x}) &:= (1-\alpha)\, h(\mathbf{x}_{\text{adv}}) + \alpha\, \bar{h}^{\mathcal{C}_\epsilon}(\mathbf{x}), \\ \underline{h}_{\text{cc}}^{\mathcal{C}_\epsilon}(\alpha; \mathbf{x}) &:= (1-\alpha)\, h(\mathbf{x}_{\text{adv}}) + \alpha\, \underline{h}^{\mathcal{C}_\epsilon}(\mathbf{x}). \end{aligned} \tag{9}$$

Proposition 3.2. *The loss function*

$$\mathcal{R}_{cc}^{\mathcal{C}_\epsilon}(\alpha; \mathbf{x}, y) := \sum_i \max\left\{ \left(\underline{h}_{cc}^{\mathcal{C}_\epsilon}(\alpha; \mathbf{x})_i - h^t(\mathbf{x})_i\right)^2, \left(\bar{h}_{cc}^{\mathcal{C}_\epsilon}(\alpha; \mathbf{x})_i - h^t(\mathbf{x})_i\right)^2 \right)$$

enjoys the following properties: (i) $\mathcal{R}_{cc}^{\mathcal{C}_\epsilon}(0; \mathbf{x}, y) = \underline{\mathcal{R}}^{\mathcal{C}_\epsilon}(\mathbf{x}, y)$ *and* $\mathcal{R}_{cc}^{\mathcal{C}_\epsilon}(1; \mathbf{x}, y) = \bar{\mathcal{R}}^{\mathcal{C}_\epsilon}(\mathbf{x}, y)$*; (ii)* $\underline{\mathcal{R}}^{\mathcal{C}_\epsilon}(\mathbf{x}, y) \leq \mathcal{R}_{cc}^{\mathcal{C}_\epsilon}(\alpha; \mathbf{x}, y) \leq \bar{\mathcal{R}}^{\mathcal{C}_\epsilon}(\mathbf{x}, y)\ \forall\ \alpha \in [0, 1]$*; (iii)* $\mathcal{R}_{cc}^{\mathcal{C}_\epsilon}(\alpha; \mathbf{x}, y)$ *is continuous and monotonically increasing for* $\alpha \in [0, 1]$.

While we expect special cases $\mathcal{R}_{\text{cc}}^{\mathcal{C}_\epsilon}(0; \mathbf{x}, y)$ and $\mathcal{R}_{\text{cc}}^{\mathcal{C}_\epsilon}(1; \mathbf{x}, y)$ to work well on settings where CC-IBP is employed with small or large α values, respectively, we aim to leverage the full flexibility from proposition 3.2 to obtain consistent performance across setups. Intuitively, $\mathcal{R}_{\text{cc}}^{\mathcal{C}_\epsilon}(\alpha; \mathbf{x}, y)$ will chiefly operate on the adversarial student features when the robust loss is close to $\mathcal{L}(f(\mathbf{x}_{\text{adv}}), y)$ (low α), and mainly focus on the student IBP bounds when robust loss is close to $\mathcal{L}(-\underline{\mathbf{z}}^{\mathcal{C}_\epsilon}(\mathbf{x}, y), y)$ (large α).

As we next show, if the student uses an affine classification head, the CC-IBP convex combinations $\mathbf{z}_{\text{cc}}^{\mathcal{C}_\epsilon}(\alpha; \mathbf{x}, y)$ correspond to a simple affine function of $\underline{h}_{\text{cc}}^{\mathcal{C}_\epsilon}(\alpha; \mathbf{x})$ and $\bar{h}_{\text{cc}}^{\mathcal{C}_\epsilon}(\alpha; \mathbf{x})$, the feature-space convex combinations onto which distillation is performed. As stated in §3.1, this assumption holds throughout our experiments, and is found to be empirically more effective through an ablation (appendix G.7).

Lemma 3.3. *Let the student classification head* $g(\mathbf{a})$ *be affine, and let* $\tilde{g}^y(\mathbf{a}) := \tilde{W}^n \mathbf{a} + \tilde{\mathbf{b}}^n$ *be its composition with the operator performing the difference between logits. We can write the CC-IBP convex combinations as:*

$$\mathbf{z}_{cc}^{\mathcal{C}_\epsilon}(\alpha; \mathbf{x}, y) = [\tilde{W}^n]_+ \underline{h}_{cc}^{\mathcal{C}_\epsilon}(\alpha; \mathbf{x}) + [\tilde{W}^n]_- \bar{h}_{cc}^{\mathcal{C}_\epsilon}(\alpha; \mathbf{x}) + \tilde{\mathbf{b}}^n.$$

Lemma 3.3 provides a strong coupling between $\mathcal{R}_{\text{cc}}^{\mathcal{C}_\epsilon}(\alpha; \mathbf{x}, y)$ and the CC-IBP loss when their convex combinations use the same α parameter. We hence incorporate this design choice into CC-Dist (short for CC-IBP Distillation). Omitting any regularization and introducing the distillation coefficient β, the CC-Dist training loss takes the following form:

$$\mathcal{L}^{\text{CC-Dist}}(\alpha, \beta; \mathbf{x}, y) := \mathcal{L}_{\text{cc}}^{\mathcal{C}_\epsilon}(\alpha; \mathbf{x}, y) + \beta\, \mathcal{R}_{\text{cc}}^{\mathcal{C}_\epsilon}(\alpha; \mathbf{x}, y). \tag{10}$$

In practice, denoting by w the dimensionality of the feature space, unless stated otherwise, we employ $\beta = 5/w$ in all reported experiments owing to its consistent performance.

3.3 Distilling onto other certified training schemes

Our distillation framework can be, in principle, applied on top of any existing certified training scheme for ReLU networks. We here propose distillation losses tailored to SABR [55], MTL-IBP [21], and IBP-R [19].

SABR-Dist SABR [55] proposes to compute IBP bounds over a parametrised subset $\mathcal{C}_\epsilon^\alpha$ of $\mathcal{C}_\epsilon$ that includes an adversarial attack $\mathbf{x}_{\text{adv}}$, termed a "small box". We denote the resulting bounds by $\underline{\mathbf{z}}^{\mathcal{C}_\epsilon^\alpha}(\mathbf{x}, y)$. The SABR loss $\mathcal{L}\left(-\underline{\mathbf{z}}^{\mathcal{C}_\epsilon^\alpha}(\mathbf{x}, y), y\right)$ is expressive owing to the monotonic growth of $-\underline{\mathbf{z}}^{\mathcal{C}_\epsilon^\alpha}(\mathbf{x}, y)$ with $\alpha \in [0, 1]$ from the adversarial logit differences to $-\underline{\mathbf{z}}^{\mathcal{C}_\epsilon}(\mathbf{x}, y)$ [21, proposition B.2]. Hence, in order to mirror the versatility of $\mathcal{R}_{\text{cc}}^{\mathcal{C}_\epsilon}(\alpha; \mathbf{x}, y)$, we propose to employ a distillation term that computes $\bar{\mathcal{R}}^{\mathcal{C}_\epsilon}(\mathbf{x}, y)$ on $\mathcal{C}_\epsilon^\alpha$, calling the resulting algorithm SABR-Dist:

$$\mathcal{L}^{\text{SABR-Dist}}(\alpha, \beta; \mathbf{x}, y) := \mathcal{L}\left(-\underline{\mathbf{z}}^{\mathcal{C}_\epsilon^\alpha}(\mathbf{x}, y), y\right) + \beta\, \bar{\mathcal{R}}^{\mathcal{C}_\epsilon^\alpha}(\mathbf{x}, y).$$

Similarly to $\mathcal{R}_{\text{cc}}^{\mathcal{C}_\epsilon}(\alpha; \mathbf{x}, y)$, $\bar{\mathcal{R}}^{\mathcal{C}_\epsilon^\alpha}(\mathbf{x}, y)$ will mainly operate on the student adversarial features when $\mathcal{C}_\epsilon^\alpha$ is a small neighborhood of the attack (low α), while focusing on the student IBP bounds for larger α. At the same time, differently from $\mathcal{R}_{\text{cc}}^{\mathcal{C}_\epsilon}(\alpha; \mathbf{x}, y)$, $\bar{\mathcal{R}}^{\mathcal{C}_\epsilon^\alpha}(\mathbf{x}, y)$ can be computed re-using the same IBP bounds employed for the SABR loss.

MTL-Dist MTL-IBP [21] is an expressive loss obtained by taking convex combinations between $\underline{\mathcal{L}}^{\mathcal{C}_\epsilon}(\mathbf{x}, y)$ and $\bar{\mathcal{L}}^{\mathcal{C}_\epsilon}(\mathbf{x}, y)$: $(1-\alpha)\underline{\mathcal{L}}^{\mathcal{C}_\epsilon}(\mathbf{x}, y) + \alpha\bar{\mathcal{L}}^{\mathcal{C}_\epsilon}(\mathbf{x}, y)$. Owing to its similarity with CC-IBP, whose loss it upper bounds for convex losses and $\alpha \in (0, 1)$ [21, proposition 4.2], we advocate for the use of $\mathcal{R}_{\text{cc}}^{\mathcal{C}_\epsilon}(\alpha; \mathbf{x}, y)$ as for CC-Dist, calling the resulting algorithm MTL-Dist:

$$\mathcal{L}^{\text{MTL-Dist}}(\alpha, \beta; \mathbf{x}, y) := (1-\alpha)\underline{\mathcal{L}}^{\mathcal{C}_\epsilon}(\mathbf{x}, y) + \alpha\bar{\mathcal{L}}^{\mathcal{C}_\epsilon}(\mathbf{x}, y) + \beta\mathcal{R}_{\text{cc}}^{\mathcal{C}_\epsilon}(\alpha; \mathbf{x}, y).$$

IBP-R-Dist IBP-R proposes to add verifiability on top of adversarial training through a regularizer $\mathcal{A}^{\mathcal{C}_\epsilon}(\mathbf{x}, y)$ on the area of network relaxations [19] (see appendix C for a definition). Owing to its strong reliance on the adversarial loss, we propose to perform distillation directly on the adversarial features. Specifically, we use the adversarial lower bound to the worst-case distillation loss, referring to the resulting algorithm as IBP-R-Dist:

$$\mathcal{L}^{\text{IBP-R-Dist}}(\lambda, \beta; \mathbf{x}, y) := \underline{\mathcal{L}}^{\mathcal{C}_\epsilon}(\mathbf{x}, y) + \lambda\mathcal{A}^{\mathcal{C}_\epsilon}(\mathbf{x}, y) + \beta\underline{\mathcal{R}}^{\mathcal{C}_\epsilon}(\mathbf{x}, y).$$

Differently from the other methods that we propose, IBP-R-Dist does not operate on an expressive loss, showcasing the versatility of the proposed distillation framework.

4 Related work

Owing to its favorable optimization properties [38], the relatively loose IBP (§2.1) is the method of choice [49] when computing lower bounds to equation (2) to be employed for certified training. Tighter techniques are however preferred within branch-and-bound-based complete verifiers. Branch-and-bound [8] couples a bounding algorithm with a strategy to refine the network relaxations by iteratively splitting problem (2) into subproblems (branching), which typically operates by splitting piecewise linearities into their linear components [10, 18, 25, 32]. Earlier bounding algorithms relied on linear network relaxations [2, 8, 23], with more effective techniques operating in the dual space [9, 20, 22]. State-of-the-art branch-and-bound frameworks have since converged to using fast bounds based on propagating couples of linear bounds through the network [60, 69, 74] with Lagrangian relaxations employed to capture additional constraints [25, 65, 76, 77].

Earlier certified training works add geometric regularizers to the standard adversarial training loss [15, 45, 67]. Recent certified training schemes mix adversarial training with bounds from network relaxations to obtain strong post-training verifiability using branch-and-bound (see §2.2). Balunovic and Vechev [6] perform adversarial attacks over latent-space over-approximations. The good performance of the latter at lower perturbation radii was matched by IBP-R, and later refined by connecting the gradients of the over-approximation with those of the attack (TAPS) [48]. The later SABR (see §3.3) obtained significant performance improvements, then generalized into the notion of expressive losses [21] (see §2.2), which includes losses based on convex combinations forming the basis of this work. All the above methods train standard feedforward ReLU networks using convex relaxations, with IBP bounds being the most prevalent among them. Recent work has highlighted the superior representational power of networks trained via tighter relaxations [4, 50], and sought to overcome the associated optimization challenges [5].

Certifiably-robust networks can also be obtained by training 1-Lipschitz models using alternative architectures [71, 72], the best-performing being SortNet [73]. These were also shown to benefit from additional generated data [1], whose use is beyond the focus of our work. While we here concentrate on ℓ_∞ perturbations and deterministic certificates, 1-Lipschitz architectures [52, 57] and Lipschitz regularization [36, 44] are very effective when $\mathcal{C}_\epsilon$ is defined using the ℓ_2 norm, setting under which probabilistic robustness certificates can be effectively obtained using randomised smoothing [13].

Related work on adversarial distillation is provided in appendix B.

5 Experimental evaluation

This section presents an experimental evaluation of the proposed distillation framework on image classification datasets from the certified training literature. We first present results on the effectiveness of distillation from adversarially-trained teachers for various certified training algorithms (§5.1), followed by a comparison of CC-Dist models with their teachers (§5.2), and by an analysis of the effect of the distillation coefficient on CC-Dist (§5.3).

In line with previous work [21, 48, 55, 59], all the models in the main evaluation are based on a 7-layer convolutional architecture named `CNN-7`, regularized using an ℓ_1 term. The teacher models are trained via adversarial training based on a 10-step PGD adversary [47] and ℓ_1 regularization. As this was found to be beneficial in practice, we employ the `CNN-7` architecture for the teacher model too. Similarly to the expressive losses codebase [21], our implementation is in PyTorch [56] and relies on `auto_LiRPA` [68] to compute IBP bounds. As common in the wider certified training literature [48, 55], we employ the custom regularization and initialization introduced by Shi et al. [59]. Post-training verification is performed using the OVAL branch-and-bound framework [8, 9, 18] similarly to De Palma et al. [19, 21], using a configuration based on α-β-CROWN network bounds [65, 69] and employing at most 600 seconds per image. Additional details and supplementary experiments can be found in appendices F and G, respectively.

Table 1: Evaluation of the effect of CC-Dist compared to pure CC-IBP [21] when training for certified robustness against ℓ_∞ norm perturbations. Means over 6 repetitions and their 95% Confidence Intervals (CIs) are reported: improvements with non-overlapping CIs are highlighted in bold.

Dataset	ϵ	Method	Standard acc. [%]	AA acc. [%]	Certified acc. [%]
CIFAR-10	$\frac{2}{255}$	CC-DIST	**81.59 ± 0.15**	**70.62 ± 0.17**	**64.43 ± 0.24**
		CC-IBP	79.83 ± 0.29	69.03 ± 0.31	63.51 ± 0.25
	$\frac{8}{255}$	CC-DIST	**55.18 ± 0.25**	36.40 ± 0.35	35.05 ± 0.50
		CC-IBP	54.13 ± 0.32	36.06 ± 0.42	34.89 ± 0.39
TinyImageNet	$\frac{1}{255}$	CC-DIST	**43.86 ± 0.21**	**32.30 ± 0.07**	**27.61 ± 0.21**
		CC-IBP	41.31 ± 0.40	30.00 ± 0.25	26.45 ± 0.24
ImageNet64	$\frac{1}{255}$	CC-DIST	**28.62 ± 0.94**	**18.35 ± 0.69**	**14.12 ± 0.19**
		CC-IBP	26.41 ± 0.95	16.75 ± 0.52	13.56 ± 0.15

5.1 Distilling onto certified training

We here evaluate our distillation framework by assessing its effect on top of state-of-the-art certified training schemes, demonstrating that distillation from empirically-robust teachers can significantly enhance their trade-offs between certified robustness and standard performance. We first focus on CC-Dist, then demonstrate the more general efficacy of distillation through a set of SABR-Dist, MTL-Dist and IBP-R-Dist experiments. We rely on the popular AutoAttack [14] to measure empirical robustness in this context.

CC-Dist Table 1 shows that the distillation loss from §3.2 results in improved average robustness-accuracy trade-offs compared to CC-IBP, a state-of-the-art certified training algorithm, on all the considered CIFAR-10 [41], TinyImageNet [42], and downscaled ImageNet (64×64) [12] benchmarks. On CIFAR-10 with $\epsilon = 2/255$, and on TinyImageNet and downscaled ImageNet, CC-Dist yields relatively large and statistically-significant improvements on all metrics (see appendix H for a comparison with previously-attained improvements on these benchmarks). In particular, CC-Dist improves standard accuracy by more than 2% on both TinyImageNet and downscaled ImageNet while at the same time providing non-negligible average certified accuracy improvements. For the larger $\epsilon = 8/255$ on CIFAR-10, distillation results in a statistically-significant improvement on standard accuracy while leaving robustness metrics roughly unvaried, demonstrating a strictly-better performance trade-off. Appendix G.1 compares CC-Dist with literature results in the area, demonstrating that the distillation improvements lead to a new state-of-the-art for certified training on ReLU networks.

SABR-Dist, MTL-Dist and IBP-R-Dist Recalling that (§3.3) knowledge distillation from empirically-robust teachers can be applied to a wide range of certified training algorithms, we aim to show its efficacy beyond CC-Dist. In particular, we here show results pertaining to the distillation losses tailored to SABR, MTL-IBP and IBP-R (see §3.3). We forego downscaled ImageNet experiments owing to their large computational cost, and IBP-R distillation on CIFAR-10 with $\epsilon = 8/255$ because of its significantly worse performance trade-offs on this benchmark (see table 5). Table 2 shows that the distillation process significantly improves the robustness-accuracy trade-offs of all the three considered methods on CIFAR-10 with $\epsilon = 2/255$ and TinyImageNet, with statistically-significant improvements on standard accuracy and empirical robustness on all three methods. In both settings, MTL-Dist also displays statistically-significant certified accuracy improvements, owing to smaller experimental variability. Differently from CC-Dist, distillation does

Table 2: Effect of distillation from empirically-robust teachers onto SABR [55], MTL-IBP [21], and IBP-R [19]. We report means over 4 repetitions and their 95% CIs, with improvements having non-overlapping CIs highlighted in bold.

Dataset	ϵ	Method	Standard acc. [%]	AA acc. [%]	Certified acc. [%]
CIFAR-10	$\frac{2}{255}$	SABR-DIST	**81.22 ± 0.19**	**70.09 ± 0.39**	64.33 ± 0.61
		SABR	79.38 ± 0.28	68.25 ± 0.38	63.54 ± 0.46
		MTL-DIST	**81.69 ± 0.29**	**70.51 ± 0.50**	**64.09 ± 0.28**
		MTL-IBP	79.87 ± 0.31	68.71 ± 0.33	63.21 ± 0.13
		IBP-R-DIST	**80.96 ± 0.19**	**68.98 ± 0.49**	61.74 ± 0.29
		IBP-R	79.55 ± 0.20	67.53 ± 0.62	60.84 ± 0.70
	$\frac{8}{255}$	SABR-DIST	53.14 ± 0.69	36.16 ± 0.55	35.11 ± 0.54
		SABR	52.34 ± 0.18	36.03 ± 0.30	35.16 ± 0.29
		MTL-DIST	54.86 ± 0.48	36.16 ± 0.86	34.95 ± 0.70
		MTL-IBP	54.22 ± 0.55	36.09 ± 0.32	35.06 ± 0.47
TinyImageNet	$\frac{1}{255}$	SABR-DIST	**43.92 ± 0.25**	**32.32 ± 0.40**	27.31 ± 0.36
		SABR	41.13 ± 0.37	29.77 ± 0.85	26.46 ± 0.85
		MTL-DIST	**43.85 ± 0.38**	**32.23 ± 0.28**	**27.26 ± 0.20**
		MTL-IBP	41.20 ± 0.39	29.86 ± 0.39	26.37 ± 0.18
		IBP-R-DIST	**44.19 ± 0.18**	**32.15 ± 0.38**	27.02 ± 0.59
		IBP-R	41.10 ± 0.76	29.59 ± 0.98	25.95 ± 0.66

Table 3: Teacher-student comparison for CC-Dist models (first repetition from table 1).

Dataset	ϵ	Model	Standard acc. [%]	AA acc. [%]	Certified acc.† [%]
CIFAR-10	$\frac{2}{255}$	STUDENT	81.55	70.71	64.4
		TEACHER	88.23	73.26	1.2
	$\frac{8}{255}$	STUDENT	55.13	36.80	35.8
		TEACHER	78.16	41.62	0.0
TinyImageNet	$\frac{1}{255}$	STUDENT	43.78	32.30	32.6
		TEACHER	47.18	34.17	17.4
ImageNet64	$\frac{1}{255}$	STUDENT	28.17	17.87	5.6
		TEACHER	40.30	25.34	0.0

† Certified accuracy reported over the first 500 test images.

not appear to provide significant improvements for neither MTL-IBP nor SABR on CIFAR-10 with $\epsilon = 8/255$. Given the well-know superiority of Lipschitz-based techniques on this benchmark (see table 5), we believe that ReLU networks may have fundamental limitations in this regime. Exploring distillation on Lipschitz networks is an exciting direction for future work. Overall, these results demonstrate the wider potential of our framework, and confirm that distillation can improve on the state-of-the-art for certified training through convex relaxations.

Supplementary results Appendix G.2 investigates the effectiveness of special cases of the $\mathcal{R}_{\text{cc}}^{\mathcal{C}_\epsilon}(\alpha; \mathbf{x}, y)$ distillation loss on CIFAR-10, showing that $\mathcal{R}_{\text{cc}}^{\mathcal{C}_\epsilon}(0; \mathbf{x}, y)$ and standard clean distillation from equation (7), which requires an extra clean pass, perform competitively with $\mathcal{R}_{\text{cc}}^{\mathcal{C}_\epsilon}(\alpha; \mathbf{x}, y)$ for $\epsilon = 2/255$. Appendix G.6 shows that CC-Dist is effective on a different model architecture and activation function. These further point to the benefits of knowledge distillation from adversarially-robust teachers for certified training. Appendices G.3, G.7, and G.9 respectively investigate the performance of logit-based distillation, of using earlier layer features for $\mathcal{R}_{\text{cc}}^{\mathcal{C}_\epsilon}(\alpha; \mathbf{x}, y)$, and of distilling from adversarial teacher features.

5.2 Understanding distillation: teacher-student comparison

In order to provide insights on the distillation process, we now compare CC-Dist models from table 1 (their first repetition) with their respective adversarially-trained teacher models. Owing to the large computational cost associated to running branch-and-bound on networks trained via pure adversarial training (the teachers), we restrict the certified robustness comparison on the first 500 images of the test set of each benchmark. Table 3 shows a clear difference in terms of standard accuracy and empirical robustness to AutoAttack [14] between teacher and student, the latter displaying worse performance as expected. On the other hand, the teachers fail to attain good certified robustness across any of the considered benchmarks, highlighting that the students significantly surpass the respective teachers in terms of verifiability.

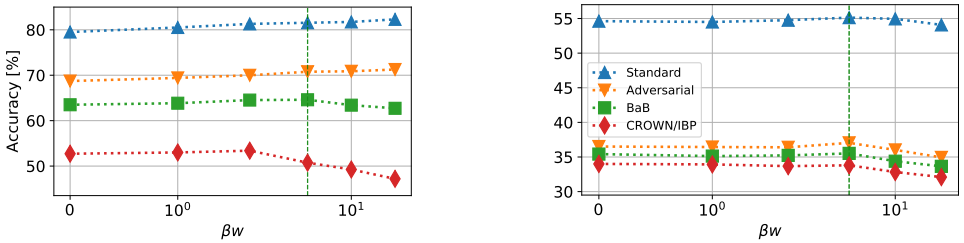
In other words, the distillation process successfully exploits empirically-robust teachers to improve the performance of certifiably-robust models by (i) transferring some of their superior natural accuracy and empirical robustness, and (ii) preserving a large degree of verifiability via the employed certified training loss.

Appendix G.8 shows that employing clean teachers results in worse performance profiles, highlighting the importance of robust representations.

5.3 Sensitivity to distillation coefficient

Owing to its strong performance across the considered settings and to avoid the introduction of an additional hyper-parameter, $\beta = 5/w$ throughout all experiments. We now present an empirical study of its effect on CC-Dist performance.

Figure 1 compares the behavior at $\beta = 5/w$ (the first repetition from table 1), where both certified robustness via branch-and-bound and standard performance are enhanced, with results for different



(a) $\epsilon = 2/255$. (b) $\epsilon = 8/255$.

Figure 1: Standard, empirical adversarial and certified accuracies (BaB and CROWN/IBP) under ℓ_∞ perturbations of networks trained using CC-Dist under varying distillation coefficient β. The legend is reported once for all subfigures in plot 1(b). Metrics are reported on the CIFAR-10 test set. The β value employed throughout the paper (see §3.2) is marked by a dashed vertical line.

distillation coefficients. It reports standard performance, empirical adversarial robustness (measured through the attacks within the employed branch-and-bound framework), and certified robustness (both using branch-and-bound and the best results between incomplete verifiers CROWN [74] and IBP [30]) across β values on the CIFAR-10 test set. For $\epsilon = 2/255$, increasing β leads to a steady increase in both standard accuracy and empirical robustness. At the same time, verifiability increases until an intermediate β value, then steadily decreases, with the certified accuracy via incomplete verifiers peaking at a lower value than when using branch-and-bound. This is similar to the behavior of the α parameter in expressive losses such as CC-IBP [21]. As also visible in table 1, distillation appears to be less helpful for $\epsilon = 8/255$, where standard performance is enhanced only for intermediate β values. We explain this difference through the larger degree of similarity between teacher and student on $\epsilon = 2/255$: see table 3.

6 Conclusions

We presented a novel distillation framework that successfully leverages the superior standard accuracy and empirical robustness of adversarially-trained models to train better certifiably-robust neural networks. Mirroring approximations to a worst-case loss popular in the adversarial machine learning literature, we present a series of novel and versatile feature-space distillation losses tailored to popular certified training algorithms. We show that these can significantly improve the robustness-accuracy trade-offs of state-of-the-art certified training schemes on benchmarks from the robust vision literature, in spite of the relatively low certified robustness of the employed teacher models. This points to the large potential of knowledge distillation for certified robustness. Indeed, while our work focuses on teachers that employ the same architecture as the student, we are hopeful that further progress can be made by appropriately training teachers with larger effective capacity.

Acknowledgments and Disclosure of Funding

The work was supported by the “SAIF" project, funded by the “France 2030” government investment plan managed by the French National Research Agency, under the reference ANR-23-PEIA-0006. The authors have no competing interests to declare. We thank the anonymous reviewers for providing valuable and constructive feedback.

References

[1] T. Altstidl, D. Dobre, A. Kosmala, B. Eskofier, G. Gidel, and L. Schwinn. On the scalability of certified adversarial robustness with generated data. In *Advances in Neural Information Processing Systems*, 2024.

[2] R. Anderson, J. Huchette, W. Ma, C. Tjandraatmadja, and J. P. Vielma. Strong mixed-integer programming formulations for trained neural networks. *Mathematical Programming*, 2020.

[3] M. Andriushchenko and N. Flammarion. Understanding and improving fast adversarial training. In *Advances in Neural Information Processing Systems*, 2020.

[4] M. Baader, M. N. Mueller, Y. Mao, and M. Vechev. Expressivity of relu-networks under convex relaxations. In *International Conference on Learning Representations*, 2024.

[5] S. Balauca, M. N. Müller, Y. Mao, M. Baader, M. Fischer, and M. Vechev. Gaussian loss smoothing enables certified training with tight convex relaxations. *Transactions on Machine Learning Research*, 2025.

[6] M. Balunovic and M. Vechev. Adversarial training and provable defenses: Bridging the gap. In *International Conference on Learning Representations*, 2020.

[7] B. Biggio, I. Corona, D. Maiorca, B. Nelson, N. Šrndić, P. Laskov, G. Giacinto, and F. Roli. Evasion attacks against machine learning at test time. In *European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases (ECML PKDD)*, 2013.

[8] R. Bunel, I. Turkaslan, P. H. Torr, P. Kohli, and M. P. Kumar. A unified view of piecewise linear neural network verification. In *Neural Information Processing Systems*, 2018.

[9] R. Bunel, A. De Palma, A. Desmaison, K. Dvijotham, P. Kohli, P. H. Torr, and M. P. Kumar. Lagrangian decomposition for neural network verification. In *Conference on Uncertainty in Artificial Intelligence*, 2020.

[10] R. Bunel, J. Lu, I. Turkaslan, P. Kohli, P. Torr, and M. P. Kumar. Branch and bound for piecewise linear neural network verification. *Journal of Machine Learning Research*, 21(2020), 2020.

[11] T. Chen, Z. Zhang, S. Liu, S. Chang, and Z. Wang. Robust overfitting may be mitigated by properly learned smoothening. In *International Conference on Learning Representations*, 2020.

[12] P. Chrabaszcz, I. Loshchilov, and F. Hutter. A downsampled variant of ImageNet as an alternative to the CIFAR datasets. *arXiv:1707.08819*, 2017.

[13] J. Cohen, E. Rosenfeld, and Z. Kolter. Certified adversarial robustness via randomized smoothing. In *International Conference on Machine Learning*, 2019.

[14] F. Croce and M. Hein. Reliable evaluation of adversarial robustness with an ensemble of diverse parameter-free attacks. In *International Conference on Machine Learning*, 2020.

[15] F. Croce, M. Andriushchenko, and M. Hein. Provable robustness of relu networks via maximization of linear regions. In *Conference on Artificial Intelligence and Statistics*. PMLR, 2019.

[16] F. Croce, M. Andriushchenko, V. Sehwag, E. Debenedetti, N. Flammarion, M. Chiang, P. Mittal, and M. Hein. Robustbench: a standardized adversarial robustness benchmark. In *Neural Information Processing Systems, Datasets and Benchmarks Track*, 2021.

[17] P. de Jorge, A. Bibi, R. Volpi, A. Sanyal, P. Torr, G. Rogez, and P. K. Dokania. Make some noise: Reliable and efficient single-step adversarial training. In *Advances in Neural Information Processing Systems*, 2022.

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

[19] A. De Palma, R. Bunel, K. Dvijotham, M. P. Kumar, and R. Stanforth. IBP regularization for verified adversarial robustness via branch-and-bound. In *ICML 2022 Workshop on Formal Verification of Machine Learning*, 2022.

[20] A. De Palma, H. S. Behl, R. Bunel, P. H. S. Torr, and M. P. Kumar. Scaling the convex barrier with sparse dual algorithms. *Journal of Machine Learning Research*, 2024.

[21] A. De Palma, R. Bunel, K. Dvijotham, M. P. Kumar, R. Stanforth, and A. Lomuscio. Expressive losses for verified robustness via convex combinations. In *International Conference on Learning Representations*, 2024.

[22] K. Dvijotham, R. Stanforth, S. Gowal, T. Mann, and P. Kohli. A dual approach to scalable verification of deep networks. In *Conference on Uncertainty in Artificial Intelligence*, 2018.

[23] R. Ehlers. Formal verification of piece-wise linear feed-forward neural networks. *Automated Technology for Verification and Analysis*, 2017.

[24] M. Fazlyab, A. Robey, H. Hassani, M. Morari, and G. Pappas. Efficient and accurate estimation of lipschitz constants for deep neural networks. In *Advances in Neural Information Processing Systems*, 2019.

[25] C. Ferrari, M. N. Mueller, N. Jovanović, and M. Vechev. Complete verification via multi-neuron relaxation guided branch-and-bound. In *International Conference on Learning Representations*, 2022.

[26] T. Furlanello, Z. Lipton, M. Tschannen, L. Itti, and A. Anandkumar. Born again neural networks. In *International Conference on Machine Learning*, 2018.

[27] S. Ghazanfari, A. Araujo, P. Krishnamurthy, F. Khorrami, and S. Garg. Lipsim: A provably robust perceptual similarity metric. In *International Conference on Learning Representations*, 2024.

[28] M. Goldblum, L. Fowl, S. Feizi, and T. Goldstein. Adversarially robust distillation. In *AAAI conference on artificial intelligence*, 2020.

[29] I. J. Goodfellow, J. Shlens, and C. Szegedy. Explaining and harnessing adversarial examples. In *International Conference on Learning Representations*, 2015.

[30] S. Gowal, K. D. Dvijotham, R. Stanforth, R. Bunel, C. Qin, J. Uesato, R. Arandjelovic, T. Mann, and P. Kohli. Scalable verified training for provably robust image classification. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019.

[31] K. He, X. Zhang, S. Ren, and J. Sun. Identity mappings in deep residual networks. In *European Conference on Computer Vision (ECCV)*, 2016.

[32] P. Henriksen and A. Lomuscio. Deepsplit: An efficient splitting method for neural network verification via indirect effect analysis. In *Proceedings of the 30th International Joint Conference on Artificial Intelligence (IJCAI21)*, 2021.

[33] B. Heo, J. Kim, S. Yun, H. Park, N. Kwak, and J. Y. Choi. A comprehensive overhaul of feature distillation. In *International Conference on Computer Vision*, 2019.

[34] T. Hickey, Q. Ju, and M. H. Van Emden. Interval arithmetic: From principles to implementation. *Journal of the ACM (JACM)*, 2001.

[35] G. Hinton, O. Vinyals, and J. Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015.

[36] K. Hu, A. Zou, Z. Wang, K. Leino, and M. Fredrikson. Unlocking deterministic robustness certification on imagenet. In *Neural Information Processing Systems*, 2023.

[37] P. Indri, P. Blohm, A. Athavale, E. Bartocci, G. Weissenbacher, M. Maffei, D. Nickovic, T. Gärtner, and S. Malhotra. Distillation based robustness verification with pac guarantees. In *ICML 2024 Next Generation of AI Safety Workshop*, 2024.

[38] N. Jovanović, M. Balunović, M. Baader, and M. T. Vechev. On the paradox of certified training. *Transactions on Machine Learning Research*, 2022.

[39] G. Katz, C. Barrett, D. Dill, K. Julian, and M. Kochenderfer. Reluplex: An efficient SMT solver for verifying deep neural networks. In *Computer Aided Verification*, 2017.

[40] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations*, 2015.

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

[42] Y. Le and X. S. Yang. Tiny imagenet visual recognition challenge. 2015.

[43] S. Lee, W. Lee, J. Park, and J. Lee. Towards better understanding of training certifiably robust models against adversarial examples. In *Neural Information Processing Systems*, 2021.

[44] K. Leino, Z. Wang, and M. Fredrikson. Globally-robust neural networks. In *International Conference on Machine Learning*. PMLR.

[45] C. Liu, M. Salzmann, and S. Süsstrunk. Training provably robust models by polyhedral envelope regularization. *IEEE Transactions on Neural Networks and Learning Systems*, 34(6), 2021.

[46] A. Lomuscio and L. Maganti. An approach to reachability analysis for feed-forward ReLU neural networks. *arXiv:1706.07351*, 2017.

[47] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu. Towards deep learning models resistant to adversarial attacks. In *International Conference on Learning Representations*, 2018.

[48] Y. Mao, M. N. Müller, M. Fischer, and M. Vechev. TAPS: Connecting certified and adversarial training. In *Neural Information Processing Systems*, 2023.

[49] Y. Mao, M. N. Müller, M. Fischer, and M. Vechev. Understanding certified training with interval bound propagation. In *International Conference on Learning Representations*, 2024.

[50] Y. Mao, Y. Zhang, and M. Vechev. On the expressiveness of multi-neuron convex relaxations. *arXiv preprint arXiv:2410.06816*, 2024.

[51] Y. Mao, S. Balauca, and M. Vechev. CTBench: A library and benchmark for certified training. In *International Conference on Machine Learning*, 2025.

[52] L. Meunier, B. J. Delattre, A. Araujo, and A. Allauzen. A dynamical system perspective for Lipschitz neural networks. In *International Conference on Machine Learning*, 2022.

[53] M. Mirman, T. Gehr, and M. Vechev. Differentiable abstract interpretation for provably robust neural networks. In *International Conference on Machine Learning*, 2018.

[54] A. Muhammad, F. Zhou, C. Xie, J. Li, S.-H. Bae, and Z. Li. Mixacm: Mixup-based robustness transfer via distillation of activated channel maps. 2021.

[55] M. N. Müller, F. Eckert, M. Fischer, and M. Vechev. Certified training: Small boxes are all you need. In *International Conference on Learning Representations*, 2023.

[56] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala. Pytorch: An imperative style, high-performance deep learning library. In *Neural Information Processing Systems*, 2019.

[57] B. Prach, F. Brau, G. Buttazzo, and C. H. Lampert. 1-lipschitz layers compared: Memory speed and certifiable robustness. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024.

[58] A. Romero, N. Ballas, S. E. Kahou, A. Chassang, C. Gatta, and Y. Bengio. Fitnets: Hints for thin deep nets. In *International Conference on Learning Representations*, 2015.

[59] Z. Shi, Y. Wang, H. Zhang, J. Yi, and C.-J. Hsieh. Fast certified robust training with short warmup. In *Neural Information Processing Systems*, 2021.

[60] G. Singh, T. Gehr, M. Püschel, and M. Vechev. An abstract domain for certifying neural networks. In *Proceedings of the ACM on Programming Languages*, 2019.

[61] T. Sunaga. Theory of an interval algebra and its application to numerical analysis. *RAAG Memoirs*, 1958.

[62] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. J. Goodfellow, and R. Fergus. Intriguing properties of neural networks. In *International Conference on Learning Representations*, 2014.

[63] V. Tjeng, K. Xiao, and R. Tedrake. Evaluating robustness of neural networks with mixed integer programming. *International Conference on Learning Representations*, 2019.

[64] P. Vaishnavi, V. Krish, F. Ahmed, K. Eykholt, and A. Rahmati. On the feasibility of compressing certifiably robust neural networks. In *Workshop on Trustworthy and Socially Responsible Machine Learning, NeurIPS 2022*, 2022.

[65] S. Wang, H. Zhang, K. Xu, X. Lin, S. Jana, C.-J. Hsieh, and J. Z. Kolter. Beta-CROWN: Efficient bound propagation with per-neuron split constraints for complete and incomplete neural network verification. In *Neural Information Processing Systems*, 2021.

[66] E. Wong and Z. Kolter. Provable defenses against adversarial examples via the convex outer adversarial polytope. In *International Conference on Machine Learning*, 2018.

[67] K. Xiao, V. Tjeng, N. M. Shafiullah, and A. Madry. Training for faster adversarial robustness verification via inducing relu stability. *International Conference on Learning Representations*, 2019.

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

[69] K. Xu, H. Zhang, S. Wang, Y. Wang, S. Jana, X. Lin, and C.-J. Hsieh. Fast and complete: Enabling complete neural network verification with rapid and massively parallel incomplete verifiers. In *International Conference on Learning Representations*, 2021.

[70] S. Zagoruyko and N. Komodakis. Paying more attention to attention: Improving the performance of convolutional neural networks via attention transfer. In *International Conference on Learning Representations*, 2017.

[71] B. Zhang, T. Cai, Z. Lu, D. He, and L. Wang. Towards certifying l-infinity robustness using neural networks with l-inf-dist neurons. In *International Conference on Machine Learning*, 2021.

[72] B. Zhang, D. Jiang, D. He, and L. Wang. Boosting the certified robustness of l-infinity distance nets. In *International Conference on Learning Representations*, 2022.

[73] B. Zhang, D. Jiang, D. He, and L. Wang. Rethinking lipschitz neural networks and certified robustness: A boolean function perspective. In *Neural Information Processing Systems*, 2022.

[74] H. Zhang, T.-W. Weng, P.-Y. Chen, C.-J. Hsieh, and L. Daniel. Efficient neural network robustness certification with general activation functions. In *Neural Information Processing Systems*, 2018.

[75] H. Zhang, H. Chen, C. Xiao, S. Gowal, R. Stanforth, B. Li, D. Boning, and C.-J. Hsieh. Towards stable and efficient training of verifiably robust neural networks. In *International Conference on Learning Representations*, 2020.

[76] H. Zhang, S. Wang, K. Xu, L. Li, B. Li, S. Jana, C.-J. Hsieh, and J. Z. Kolter. General cutting planes for bound-propagation-based neural network verification. In *Neural Information Processing Systems*, 2022.

[77] D. Zhou, C. Brix, G. A. Hanasusanto, and H. Zhang. Scalable neural network verification with branch-and-bound inferred cutting planes. In *Neural Information Processing Systems*, 2024.

[78] J. Zhu, J. Yao, B. Han, J. Zhang, T. Liu, G. Niu, J. Zhou, J. Xu, and H. Yang. Reliable adversarial distillation with unreliable teachers. In *International Conference on Learning Representations*, 2022.

[79] B. Zi, S. Zhao, X. Ma, and Y.-G. Jiang. Revisiting adversarial robustness distillation: Robust soft labels make student better. In *International Conference on Computer Vision*, 2021.

A Limitations and broader impact

While our work demonstrates that knowledge distillation from empirically-robust teachers can be employed to improve the state-of-the-art in certified training, it still shares many of the limitations of work in the area. We here focus on higher-level arguments, pointing the reader to appendix F.2.4 for lower-level experimental details. Our work focuses on ℓ_∞ perturbations, the context in which certified training via convex relaxations on `CNN-7` is the state-of-the-art deterministic certified defense (except on larger perturbations, were Lipschitz-based methods display better performance: see appendix G.1). A large certified accuracy to local ℓ_∞ perturbations of a fixed radius does not generally guarantee neither robustness to other types of perturbations, nor to the same perturbations when applied to a point subject to a strong distribution shift. Devising effective distillation techniques for other perturbation types, network architectures, and training methodologies are exciting avenues for future work. Furthermore, while our work targets verifiability via branch-and-bound, we acknowledge the large associated runtime cost, which prevents such verifiers to be deployed on a per-point basis in a real-time environment. Finally, the considered benchmarks all belong to the vision domain: effectively deploying certified training schemes to other data domains, for instance natural language processing, will involve overcoming additional domain-specific difficulties.

We do not anticipate any short-term negative impact of certifiably-robust networks. Indeed, we believe that efficient certified training techniques should be primarily leveraged towards providing guarantees to deep learning systems operating in safety-critical contexts. Nevertheless, we acknowledge that adversarial attacks may have social utility when deployed against unethical systems, which constitute unintended use of machine learning technologies, pointing to a potential shortcoming of provable robustness. Effective mitigation strategies may include targeted regulations.

B Related work on knowledge distillation

Many specialized knowledge distillation schemes designed to transfer empirical robustness focus on modifying the logit-space distillation loss from equation (6). Assuming an adversarially-trained teacher, ARD [28] proposes to modify the KL term from equation (6) to evaluate the student on an adversarially-perturbed input. IAD [78], instead, replaces the cross-entropy loss with a KL term between the student's perturbed and unperturbed outputs, weighted according to the teacher's inability to correctly classify the adversarial input. RSLAD [79] removes the dynamic weighting and trains with two KL terms between teacher and student: one using clean inputs, the other where the student is evaluated on a perturbation seeking to maximize the distance from the clean teacher output. In this context, less attention has been devoted to feature-space distillation methods [54]. CRD [64] presented a logit-space distillation loss designed to transfer probabilistic certified robustness, relying on a sum of the student cross-entropy loss with an ℓ_2 term on the models softmax outputs, all evaluated on random Gaussian perturbations. The focus of all the above works is to transfer some of the performance of larger and more capable models onto smaller and less expensive architectures. However, it was shown that training a network for a given task, and then re-using it to distill knowledge onto a second model with the same architecture and training goal can improve performance through regularization. This was observed both in the context of standard training [26], and of pure adversarial training for empirical robustness [11]. Finally, closely-related works at the intersection between knowledge distillation and deterministic certified robustness include: (i) investigating how robustness guarantees obtained on a student can be transferred from a distilled student to its teacher [37], and (ii) distilling a non-robust perceptual similarity metric onto a 1-Lipschitz architecture to obtain a certifiably-robust perceptual similarity metric [27]. In this work, targeting supervised classification tasks from the robust vision literature and using techniques based on convex relaxations, we aim to leverage an empirically-robust yet hard-to-verify teacher to improve the deterministic certified robustness of a student with the same architecture.

C Interval Bound Propagation

We here provide details concerning the computations of IBP bounds omitted from §2.1, and define the IBP-R regularizer. Assuming a feed-forward network structure, let W^j and b^j respectively be the weight and bias of the j-th layer, and let σ denote a monotonic element-wise activation function, which, as stated in §2, we assume to be the ReLU activation in this work. Let us denote

by ${}_jI^n \in \mathbb{R}^{n\times n}$ a matrix whose j-th column is the unit vector and filled with zeros otherwise. Furthermore, let $\tilde{W}^n := \left({}_yI^k - I^k\right) W^n$ and $\tilde{\mathbf{b}}^n := \left({}_yI^k - I^k\right) \mathbf{b}^n$ denote the weight and bias corresponding to the composition of the last network layer and of the difference between logits.

IBP pass IBP proceeds by first computing $\hat{\mathbf{l}}^1 = W^1\mathbf{x} - \epsilon \left|W^1\right| \mathbf{1} + \mathbf{b}^1$ and $\hat{\mathbf{u}}^1 = W^1\mathbf{x} + \epsilon \left|W^1\right| \mathbf{1} + \mathbf{b}^1$, and then derives $\underline{\mathbf{z}}^{\mathcal{C}_\epsilon}(\mathbf{x}, y)$ by iteratively computing lower and upper bounds ($\hat{\mathbf{l}}^j$, and $\hat{\mathbf{u}}^j$, respectively) to the network pre-activations at layer j:

$$\begin{gathered}\underline{\mathbf{z}}^{\mathcal{C}_\epsilon}(\mathbf{x}, y) = [\tilde{W}^n]_+ \sigma\left(\hat{\mathbf{l}}^{n-1}\right) + [\tilde{W}^n]_- \sigma\left(\hat{\mathbf{u}}^{n-1}\right) + \tilde{\mathbf{b}}^n, \quad \text{where:} \\ \left[\begin{array}{l} \hat{\mathbf{l}}^j = [W^j]_+ \sigma\left(\hat{\mathbf{l}}^{j-1}\right) + \frac{1}{2}[W^j]_- \sigma\left(\hat{\mathbf{u}}^{j-1}\right) + \mathbf{b}_j \\ \hat{\mathbf{u}}^j = [W^j]_+ \sigma\left(\hat{\mathbf{u}}^{j-1}\right) + \frac{1}{2}[W^j]_- \sigma\left(\hat{\mathbf{l}}^{j-1}\right) + \mathbf{b}_j \end{array}\right] \forall\, j \in [\![2, n-1]\!]\,.\end{gathered} \tag{11}$$

IBP-R regularizer Given pre-activation bounds from equation (11), and denoting the batch size as B, we compute the IBP-R regularizer $\mathcal{A}^{\mathcal{C}_\epsilon}(\mathbf{x}, y)$ [19] as:

$$\mathcal{A}^{\mathcal{C}_\epsilon}(\mathbf{x}, y) := B \sum_{j=1}^{n-1} \left[\hat{\mathbf{u}}^j\right]_+^T \left[\hat{\mathbf{l}}^j\right]_-\,.$$

D Proofs of technical results

We here provide the proofs omitted from §3.

Remark D.1. Let $f : \mathbb{R}^n \to \mathbb{R}$. If $\mathcal{A} \neq \emptyset$ and $\mathcal{A} \subseteq \mathcal{B}$, then $\max_{\mathbf{x}\in\mathcal{A}} f(\mathbf{x}) \leq \max_{\mathbf{x}\in\mathcal{B}} f(\mathbf{x})$.

Proposition 3.1. *Let $\underline{h}^{\mathcal{C}_\epsilon}(\mathbf{x})$ and $\bar{h}^{\mathcal{C}_\epsilon}(\mathbf{x})$ respectively denote IBP lower and upper bounds to the student features:*

$$\underline{h}^{\mathcal{C}_\epsilon}(\mathbf{x}) \leq h(\mathbf{x}') \leq \bar{h}^{\mathcal{C}_\epsilon}(\mathbf{x}), \quad \forall \mathbf{x}' \in \mathcal{C}_\epsilon(\mathbf{x}).$$

The loss function

$$\bar{\mathcal{R}}^{\mathcal{C}_\epsilon}(\mathbf{x}, y) := \sum_i \max\left\{\left(\underline{h}^{\mathcal{C}_\epsilon}(\mathbf{x})_i - h^t(\mathbf{x})_i\right)^2, \left(\bar{h}^{\mathcal{C}_\epsilon}(\mathbf{x})_i - h^t(\mathbf{x})_i\right)^2\right\}$$

is an upper bound to the worst-case distillation loss from equation (8)*: $\bar{\mathcal{R}}^{\mathcal{C}_\epsilon}(\mathbf{x}, y) \geq \mathcal{R}^{\mathcal{C}_\epsilon}(\mathbf{x}, y)$.*

Proof. Let us start by upper bounding the worst-case distillation loss $\mathcal{R}^{\mathcal{C}_\epsilon}(\mathbf{x}, y)$:

$$\begin{aligned}\mathcal{R}^{\mathcal{C}_\epsilon}(\mathbf{x}, y) &= \max_{\mathbf{x}'\in\mathcal{C}_\epsilon(\mathbf{x})} \left\|h(\mathbf{x}') - h^t(\mathbf{x})\right\|_2^2 = \max_{\mathbf{x}'\in\mathcal{C}_\epsilon(\mathbf{x})} \left[\sum_i \left(h(\mathbf{x}')_i - h^t(\mathbf{x})_i\right)^2\right] \\ &\leq \sum_i \max_{\mathbf{x}'\in\mathcal{C}_\epsilon(\mathbf{x})} \left(h(\mathbf{x}')_i - h^t(\mathbf{x})_i\right)^2.\end{aligned}$$

Let $\mathcal{M}_i = \left\{h(\mathbf{x}')_i \;\middle|\; \mathbf{x}' \in \operatorname{argmax}_{\mathbf{x}'\in\mathcal{C}_\epsilon(\mathbf{x})} \left(h(\mathbf{x}')_i - h^t(\mathbf{x})_i\right)^2\right\}$. By the definition of the IBP feature bounds $\underline{h}^{\mathcal{C}_\epsilon}(\mathbf{x})$ and $\bar{h}^{\mathcal{C}_\epsilon}(\mathbf{x})$:

$$\mathcal{M}_i \subseteq \left[\underline{h}^{\mathcal{C}_\epsilon}(\mathbf{x})_i, \bar{h}^{\mathcal{C}_\epsilon}(\mathbf{x})_i\right].$$

Recalling remark D.1, we can hence further bound $\mathcal{R}^{\mathcal{C}_\epsilon}(\mathbf{x}, y)$ as follows:

$$\begin{aligned}\mathcal{R}^{\mathcal{C}_\epsilon}(\mathbf{x}, y) \leq \sum_i \max_{\mathbf{x}'\in\mathcal{C}_\epsilon(\mathbf{x})} \left(h(\mathbf{x}')_i - h^t(\mathbf{x})_i\right)^2 &= \sum_i \max_{h\in\mathcal{M}_i} \left(h - h^t(\mathbf{x})_i\right)^2 \\ &\leq \sum_i \max_{h\in\left[\underline{h}^{\mathcal{C}_\epsilon}(\mathbf{x})_i, \bar{h}^{\mathcal{C}_\epsilon}(\mathbf{x})_i\right]} \left(h - h^t(\mathbf{x})_i\right)^2.\end{aligned}$$

Let us note that (for any $b, l, u \in \mathbb{R}$, $l \leq u$):

$$\max_{x\in[l,u]} (x-b)^2 = \max\left\{(l-b)^2, (u-b)^2\right\}. \tag{12}$$

We can hence write $\mathcal{R}^{\mathcal{C}_\epsilon}(\mathbf{x}, y) \leq \bar{\mathcal{R}}^{\mathcal{C}_\epsilon}(\mathbf{x}, y)$. □

Proposition 3.2. *The loss function*

$$\mathcal{R}_{cc}^{\mathcal{C}_\epsilon}(\alpha;\mathbf{x},y) := \sum_i \max\left\{\left(\underline{h}_{cc}^{\mathcal{C}_\epsilon}(\alpha;\mathbf{x})_i - h^t(\mathbf{x})_i\right)^2, \left(\bar{h}_{cc}^{\mathcal{C}_\epsilon}(\alpha;\mathbf{x})_i - h^t(\mathbf{x})_i\right)^2\right)$$

enjoys the following properties: (i) $\mathcal{R}_{cc}^{\mathcal{C}_\epsilon}(0;\mathbf{x},y) = \underline{\mathcal{R}}^{\mathcal{C}_\epsilon}(\mathbf{x},y)$ *and* $\mathcal{R}_{cc}^{\mathcal{C}_\epsilon}(1;\mathbf{x},y) = \bar{\mathcal{R}}^{\mathcal{C}_\epsilon}(\mathbf{x},y)$*; (ii)* $\underline{\mathcal{R}}^{\mathcal{C}_\epsilon}(\mathbf{x},y) \leq \mathcal{R}_{cc}^{\mathcal{C}_\epsilon}(\alpha;\mathbf{x},y) \leq \bar{\mathcal{R}}^{\mathcal{C}_\epsilon}(\mathbf{x},y)\ \forall\ \alpha \in [0,1]$*; (iii)* $\mathcal{R}_{cc}^{\mathcal{C}_\epsilon}(\alpha;\mathbf{x},y)$ *is continuous and monotonically increasing for* $\alpha \in [0,1]$.

Proof. Let us define $\mathcal{I}_i^\alpha := \left[\underline{h}_{\text{cc}}^{\mathcal{C}_\epsilon}(\alpha;\mathbf{x})_i, \bar{h}_{\text{cc}}^{\mathcal{C}_\epsilon}(\alpha;\mathbf{x})_i\right]$. Owing to equation (12), we can write the following alternative definition of $\mathcal{R}_{\text{cc}}^{\mathcal{C}_\epsilon}(\alpha;\mathbf{x},y)$:

$$\mathcal{R}_{\text{cc}}^{\mathcal{C}_\epsilon}(\alpha;\mathbf{x},y) = \sum_i \max_{h\in\mathcal{I}_i^\alpha}\left(h - h^t(\mathbf{x})_i\right)^2. \tag{13}$$

Let us recall that $\bar{h}_{\text{cc}}^{\mathcal{C}_\epsilon}(\alpha;\mathbf{x}) := (1-\alpha)\ h(\mathbf{x}_{\text{adv}}) + \alpha\ \bar{h}^{\mathcal{C}_\epsilon}(\mathbf{x})$, and $\underline{h}_{\text{cc}}^{\mathcal{C}_\epsilon}(\alpha;\mathbf{x}) := (1-\alpha)\ h(\mathbf{x}_{\text{adv}}) + \alpha\ \underline{h}^{\mathcal{C}_\epsilon}(\mathbf{x})$. By pointing out that $h(\mathbf{x}_{\text{adv}}) \in \left[\underline{h}^{\mathcal{C}_\epsilon}(\mathbf{x}), \bar{h}^{\mathcal{C}_\epsilon}(\mathbf{x})\right]$, we can see that, $\forall\alpha \in [0,1]$:

$$\underline{h}^{\mathcal{C}_\epsilon}(\mathbf{x}) \leq \underline{h}_{\text{cc}}^{\mathcal{C}_\epsilon}(\alpha;\mathbf{x}) \leq h(\mathbf{x}_{\text{adv}}), \quad h(\mathbf{x}_{\text{adv}}) \leq \bar{h}_{\text{cc}}^{\mathcal{C}_\epsilon}(\alpha;\mathbf{x}) \leq \bar{h}^{\mathcal{C}_\epsilon}(\mathbf{x}).$$

Consequently, $\forall\alpha \in [0,1]$ and $\forall i \in [\![1,m]\!]$ (m being the dimensionality of the feature space):

$$\mathcal{I}_i^0 = \{h(\mathbf{x}_{\text{adv}})_i\} \subseteq \mathcal{I}_i^\alpha \subseteq \left[\underline{h}^{\mathcal{C}_\epsilon}(\mathbf{x})_i, \bar{h}^{\mathcal{C}_\epsilon}(\mathbf{x})_i\right] = \mathcal{I}_i^1.$$

And hence, as per remark D.1:

$$\sum_i \max_{h\in\mathcal{I}_i^0}\left(h - h^t(\mathbf{x})_i\right)^2 \leq \sum_i \max_{h\in\mathcal{I}_i^\alpha}\left(h - h^t(\mathbf{x})_i\right)^2 \leq \sum_i \max_{h\in\mathcal{I}_i^1}\left(h - h^t(\mathbf{x})_i\right)^2. \tag{14}$$

Using equation (13) with $\alpha = 0$ we can see that:

$$\begin{aligned}\mathcal{R}_{\text{cc}}^{\mathcal{C}_\epsilon}(0;\mathbf{x},y) &= \sum_i \max_{h\in\mathcal{I}_i^0}\left(h - h^t(\mathbf{x})_i\right)^2 = \sum_i \left(h(\mathbf{x}_{\text{adv}})_i - h^t(\mathbf{x})_i\right)^2 \\ &= \left\|h(\mathbf{x}_{\text{adv}}) - h^t(\mathbf{x})\right\|_2^2 = \underline{\mathcal{R}}^{\mathcal{C}_\epsilon}(\mathbf{x},y).\end{aligned}$$

Using it again with $\alpha = 1$ and recalling equation (12), we obtain:

$$\begin{aligned}\mathcal{R}_{\text{cc}}^{\mathcal{C}_\epsilon}(1;\mathbf{x},y) &= \sum_i \max_{h\in\mathcal{I}_i^1}\left(h - h^t(\mathbf{x})_i\right)^2 = \\ &= \sum_i \max\left\{\left(\underline{h}^{\mathcal{C}_\epsilon}(\mathbf{x})_i - h^t(\mathbf{x})_i\right)^2, \left(\bar{h}^{\mathcal{C}_\epsilon}(\mathbf{x})_i - h^t(\mathbf{x})_i\right)^2\right\} = \bar{\mathcal{R}}^{\mathcal{C}_\epsilon}(\mathbf{x},y),\end{aligned}$$

Hence proving the first point of the proposition.

The second point can be now proved by again using equation (13) within equation (14):

$$\underline{\mathcal{R}}^{\mathcal{C}_\epsilon}(\mathbf{x},y) = \mathcal{R}_{\text{cc}}^{\mathcal{C}_\epsilon}(0;\mathbf{x},y) \leq \mathcal{R}_{\text{cc}}^{\mathcal{C}_\epsilon}(\alpha;\mathbf{x},y) \leq \mathcal{R}_{\text{cc}}^{\mathcal{C}_\epsilon}(1;\mathbf{x},y) = \bar{\mathcal{R}}^{\mathcal{C}_\epsilon}(\mathbf{x},y).$$

Concerning the third point, the continuity of $\mathcal{R}_{\text{cc}}^{\mathcal{C}_\epsilon}(\alpha;\mathbf{x},y)$ for $\alpha \in [0,1]$ can be proved by pointing out that $\mathcal{R}_{\text{cc}}^{\mathcal{C}_\epsilon}(\alpha;\mathbf{x},y)$ is composed of operators preserving continuity (pointwise $\max$), linear combinations and compositions of continuous functions of α. In order to prove monotonicity, note that, for any $\alpha, \alpha' \in [0,1]$ with $\alpha \leq \alpha'$, $\bar{h}_{\text{cc}}^{\mathcal{C}_\epsilon}(\alpha';\mathbf{x}) \geq \bar{h}_{\text{cc}}^{\mathcal{C}_\epsilon}(\alpha;\mathbf{x})$ and $\underline{h}_{\text{cc}}^{\mathcal{C}_\epsilon}(\alpha';\mathbf{x}) \leq \underline{h}_{\text{cc}}^{\mathcal{C}_\epsilon}(\alpha;\mathbf{x})$, implying $\mathcal{I}_i^\alpha \subseteq \mathcal{I}_i^{\alpha'}$. Hence, recalling remark D.1, for any $\alpha, \alpha' \in [0,1]$ with $\alpha \leq \alpha'$, we have:

$$\mathcal{R}_{\text{cc}}^{\mathcal{C}_\epsilon}(\alpha;\mathbf{x},y) = \sum_i \max_{h\in\mathcal{I}_i^\alpha}\left(h - h^t(\mathbf{x})_i\right)^2 \leq \sum_i \max_{h\in\mathcal{I}_i^{\alpha'}}\left(h - h^t(\mathbf{x})_i\right)^2 = \mathcal{R}_{\text{cc}}^{\mathcal{C}_\epsilon}(\alpha';\mathbf{x},y),$$

which proves that $\mathcal{R}_{\text{cc}}^{\mathcal{C}_\epsilon}(\alpha;\mathbf{x},y)$ is monotonically increasing for $\alpha \in [0,1]$. □

Lemma 3.3. *Let the student classification head* $g(\mathbf{a})$ *be affine, and let* $\tilde{g}^y(\mathbf{a}) := \tilde{W}^n\mathbf{a} + \tilde{\mathbf{b}}^n$ *be its composition with the operator performing the difference between logits. We can write the CC-IBP convex combinations as:*

$$\mathbf{z}_{cc}^{\mathcal{C}_\epsilon}(\alpha;\mathbf{x},y) = [\tilde{W}^n]_+\underline{h}_{cc}^{\mathcal{C}_\epsilon}(\alpha;\mathbf{x}) + [\tilde{W}^n]_-\bar{h}_{cc}^{\mathcal{C}_\epsilon}(\alpha;\mathbf{x}) + \tilde{\mathbf{b}}^n.$$

Proof. We can write the student logit differences as: $\mathbf{z}_f(\mathbf{x}, y) = \tilde{W}^n h(\mathbf{x}) + \tilde{\mathbf{b}}^n$. Hence, given IBP bounds to the student features $\underline{h}^{\mathcal{C}_\epsilon}(\mathbf{x})$ and $\bar{h}^{\mathcal{C}_\epsilon}(\mathbf{x})$ (corresponding to $\sigma\left(\hat{\mathbf{l}}^{n-1}\right)$ and $\sigma\left(\hat{\mathbf{u}}^{n-1}\right)$ in equation (11), respectively), we can compute the logit differences lower bounds $\underline{\mathbf{z}}^{\mathcal{C}_\epsilon}(\mathbf{x}, y)$ as per equation (11):

$$\underline{\mathbf{z}}^{\mathcal{C}_\epsilon}(\mathbf{x}, y) = [\tilde{W}^n]_+ \underline{h}^{\mathcal{C}_\epsilon}(\mathbf{x}) + [\tilde{W}^n]_- \bar{h}^{\mathcal{C}_\epsilon}(\mathbf{x}) + \tilde{\mathbf{b}}^n.$$

Let us write the adversarial logit differences $\mathbf{z}_f(\mathbf{x}_{\text{adv}}, y)$ as a function of the adversarial student features:

$$\mathbf{z}_f(\mathbf{x}_{\text{adv}}, y) = \tilde{W}^n h(\mathbf{x}_{\text{adv}}) + \tilde{\mathbf{b}}^n = \left([\tilde{W}^n]_+ + [\tilde{W}^n]_-\right) h(\mathbf{x}_{\text{adv}}) + \tilde{\mathbf{b}}^n.$$

Replacing the above two equations in the definition of the CC-IBP convex combinations $\mathbf{z}_{\text{cc}}^{\mathcal{C}_\epsilon}(\alpha; \mathbf{x}, y)$ we get:

$$\begin{aligned}
\mathbf{z}_{\text{cc}}^{\mathcal{C}_\epsilon}(\alpha; \mathbf{x}, y) &= (1-\alpha)\, \mathbf{z}_f(\mathbf{x}_{\text{adv}}, y) + \alpha\, \underline{\mathbf{z}}^{\mathcal{C}_\epsilon}(\mathbf{x}, y) \\
&= (1-\alpha) \left[\left([\tilde{W}^n]_+ + [\tilde{W}^n]_-\right) h(\mathbf{x}_{\text{adv}})\right] + \alpha \left([\tilde{W}^n]_+ \underline{h}^{\mathcal{C}_\epsilon}(\mathbf{x}) + [\tilde{W}^n]_- \bar{h}^{\mathcal{C}_\epsilon}(\mathbf{x})\right) + \tilde{\mathbf{b}}^n \\
&= [\tilde{W}^n]_+ \left[(1-\alpha) h(\mathbf{x}_{\text{adv}}) + \alpha \underline{h}^{\mathcal{C}_\epsilon}(\mathbf{x})\right] + [\tilde{W}^n]_- \left[(1-\alpha) h(\mathbf{x}_{\text{adv}}) + \alpha \bar{h}^{\mathcal{C}_\epsilon}(\mathbf{x})\right] + \tilde{\mathbf{b}}^n.
\end{aligned}$$

Recalling that $\bar{h}_{\text{cc}}^{\mathcal{C}_\epsilon}(\alpha; \mathbf{x}) := (1-\alpha)\, h(\mathbf{x}_{\text{adv}}) + \alpha\, \bar{h}^{\mathcal{C}_\epsilon}(\mathbf{x})$, and $\underline{h}_{\text{cc}}^{\mathcal{C}_\epsilon}(\alpha; \mathbf{x}) := (1-\alpha)\, h(\mathbf{x}_{\text{adv}}) + \alpha\, \underline{h}^{\mathcal{C}_\epsilon}(\mathbf{x})$, we get:

$$\mathbf{z}_{\text{cc}}^{\mathcal{C}_\epsilon}(\alpha; \mathbf{x}, y) = [\tilde{W}^n]_+ \underline{h}_{\text{cc}}^{\mathcal{C}_\epsilon}(\alpha; \mathbf{x}) + [\tilde{W}^n]_- \bar{h}_{\text{cc}}^{\mathcal{C}_\epsilon}(\alpha; \mathbf{x}) + \tilde{\mathbf{b}}^n,$$

which concludes the proof. □

E Pseudo-code

Let us denote the Euclidean projection of $\mathbf{a}$ on set $\mathcal{A}$ by $\text{Proj}(\mathbf{a}, \mathcal{A})$. We here provide pseudo-code for CC-Dist and MTL-Dist (algorithm 1), SABR-Dist (algorithm 2), and IBP-R-Dist (algorithm 3).

Algorithm 1 CC-Dist/MTL-Dist training losses $\mathcal{L}^{\text{CC-Dist/MTL-Dist}}(\alpha, {}^5\!/_w; \mathbf{x}, y)$

Input: Student $f = g \circ h$, teacher feature map h^t, loss function $\mathcal{L}$, data point $(\mathbf{x}, y)$, perturbation set $\mathcal{C}_\epsilon(\mathbf{x})$, hyper-parameter α
Compute $\mathbf{x}_{\text{adv}}$ via an adversarial attack on $\mathcal{C}_\epsilon(\mathbf{x})$
Compute $\underline{h}^{\mathcal{C}_\epsilon}(\mathbf{x})$, $\bar{h}^{\mathcal{C}_\epsilon}(\mathbf{x})$ and $\underline{\mathbf{z}}^{\mathcal{C}_\epsilon}(\mathbf{x}, y)$ via IBP
$\bar{h}_{\text{cc}}^{\mathcal{C}_\epsilon}(\alpha; \mathbf{x}) \leftarrow (1-\alpha)\, h(\mathbf{x}_{\text{adv}}) + \alpha\, \bar{h}^{\mathcal{C}_\epsilon}(\mathbf{x})$
$\underline{h}_{\text{cc}}^{\mathcal{C}_\epsilon}(\alpha; \mathbf{x}) \leftarrow (1-\alpha)\, h(\mathbf{x}_{\text{adv}}) + \alpha\, \underline{h}^{\mathcal{C}_\epsilon}(\mathbf{x})$
$\mathcal{R}_{\text{cc}}^{\mathcal{C}_\epsilon}(\alpha; \mathbf{x}, y) \leftarrow \sum_i \max\left\{\left(\underline{h}_{\text{cc}}^{\mathcal{C}_\epsilon}(\alpha; \mathbf{x})_i - h^t(\mathbf{x})_i\right)^2, \left(\bar{h}_{\text{cc}}^{\mathcal{C}_\epsilon}(\alpha; \mathbf{x})_i - h^t(\mathbf{x})_i\right)^2\right)$
if CC-Dist **then**
    $\mathbf{z}_f(\mathbf{x}_{\text{adv}}, y) \leftarrow \left(g\left(h(\mathbf{x}_{\text{adv}})\right)_y \mathbf{1}^k - g\left(h(\mathbf{x}_{\text{adv}})\right)\right)$
    $L \leftarrow \mathcal{L}\left(-\left[(1-\alpha)\, \mathbf{z}_f(\mathbf{x}_{\text{adv}}, y) + \alpha\, \underline{\mathbf{z}}^{\mathcal{C}_\epsilon}(\mathbf{x}, y)\right], y\right)$
else if MTL-Dist **then**
    $L \leftarrow (1-\alpha)\mathcal{L}(f(\mathbf{x}_{\text{adv}}), y) + \alpha\mathcal{L}(-\underline{\mathbf{z}}^{\mathcal{C}_\epsilon}(\mathbf{x}, y), y)$
end if
Return: $L + {}^5\!/_w\, \mathcal{R}_{\text{cc}}^{\mathcal{C}_\epsilon}(\alpha; \mathbf{x}, y)$

Algorithm 2 SABR-Dist training loss $\mathcal{L}^{\text{SABR-Dist}}(\alpha, {}^{5}\!/_{w}; \mathbf{x}, y)$

Input: Student $f = g \circ h$, teacher feature map h^t, loss function $\mathcal{L}$, data point $(\mathbf{x}, y)$, perturbation set $\mathcal{C}_\epsilon(\mathbf{x})$, hyper-parameter α
Compute $\mathbf{x}_{\text{adv}}$ via an adversarial attack on $\mathcal{C}_\epsilon(\mathbf{x})$
$\mathbf{x}_\alpha \leftarrow \text{Proj}(\mathbf{x}_{\text{adv}}, \mathcal{C}_{\epsilon - \lambda\epsilon}(\mathbf{x}))$
$\mathcal{C}_\epsilon^\alpha(\mathbf{x}) \leftarrow \mathcal{C}_{\alpha\epsilon}(\mathbf{x}_\alpha)$
Compute $\underline{h}^{\mathcal{C}_\epsilon^\alpha}(\mathbf{x})$, $\bar{h}^{\mathcal{C}_\epsilon^\alpha}(\mathbf{x})$ and $\underline{\mathbf{z}}^{\mathcal{C}_\epsilon^\alpha}(\mathbf{x}, y)$ via IBP
$\bar{\mathcal{R}}^{\mathcal{C}_\epsilon^\alpha}(\mathbf{x}, y) \leftarrow \sum_i \max\left\{\left(\underline{h}^{\mathcal{C}_\epsilon^\alpha}(\mathbf{x})_i - h^t(\mathbf{x})_i\right)^2, \left(\bar{h}^{\mathcal{C}_\epsilon^\alpha}(\mathbf{x})_i - h^t(\mathbf{x})_i\right)^2\right\}$
Return: $\mathcal{L}\left(-\underline{\mathbf{z}}^{\mathcal{C}_\epsilon^\alpha}(\mathbf{x}, y), y\right) + {}^{5}\!/_{w}\ \bar{\mathcal{R}}^{\mathcal{C}_\epsilon^\alpha}(\mathbf{x}, y)$

Algorithm 3 IBP-R-Dist training loss $\mathcal{L}^{\text{IBP-R-Dist}}(\lambda, {}^{5}\!/_{w}; \mathbf{x}, y)$

Input: Student $f = g \circ h$, teacher feature map h^t, loss function $\mathcal{L}$, data point $(\mathbf{x}, y)$, perturbation set $\mathcal{C}_\epsilon(\mathbf{x})$, hyper-parameter α, batch size B
Compute $\mathbf{x}_{\text{adv}}$ via an adversarial attack on $\mathcal{C}_\epsilon(\mathbf{x})$
Compute pre-activation bounds $\hat{\mathbf{l}}^j$, $\hat{\mathbf{u}}^j$ for $j \in [\![1, n-1]\!]$ via IBP
$\mathcal{A}^{\mathcal{C}_\epsilon}(\mathbf{x}, y) \leftarrow B \sum_{j=1}^{n-1} \left[\hat{\mathbf{u}}^j\right]_+^T \left[\hat{\mathbf{l}}^j\right]_-$
$\underline{\mathcal{R}}^{\mathcal{C}_\epsilon}(\mathbf{x}, y) \leftarrow \|h(\mathbf{x}_{\text{adv}}) - h^t(\mathbf{x})\|_2^2$
Return: $\mathcal{L}(f(\mathbf{x}_{\text{adv}}), y) + \lambda \mathcal{A}^{\mathcal{C}_\epsilon}(\mathbf{x}, y) + {}^{5}\!/_{w}\ \underline{\mathcal{R}}^{\mathcal{C}_\epsilon}(\mathbf{x}, y)$

F Experimental details

This appendix provides experimental details omitted from §5.

F.1 Experimental setup

Experiments were each carried out on an internal cluster using a single GPU and employing from 6 to 12 CPU cores. GPUs belonging to the following models were employed: Nvidia GTX 1080 Ti, Nvidia RTX 2080 Ti, Nvidia RTX 6000, Nvidia RTX 8000, Nvidia V100, Nvidia A100. CPU memory was capped at 10 GB for the training experiments, and at 100 GB for the verification experiments. The runtime of training runs ranges from roughly 5 hours for CIFAR-10, to roughly 4 days for downscaled ImageNet. For verification experiments, runtime ranges from below 8 hours for CIFAR-10 with $\epsilon = {}^{8}\!/_{255}$ to roughly 12 days for downscaled ImageNet.

F.2 Training setup and hyper-parameters

We now report the details of the employed training setup and hyper-parameters.

F.2.1 Datasets

As stated in §5.1, the experiments were carried out on the following datasets: CIFAR-10 [41], TinyImageNet [42], and downscaled ImageNet (64×64) [12]. For all three, we train using random horizontal flips, random crops, and normalization as done in [21]. The employed dataset splits comply with previous work in the area [21, 48, 55]. Specifically, for CIFAR-10, which consists of 32×32 RGB images divided into 10 classes, we trained on the original $50,000$ training points, and evaluated on the original $10,000$ test points. For TinyImageNet, consisting of 64×64 RGB images and 200 classes, we trained on the original $100,000$ training points, and evaluated on the original $10,000$ validation images. Finally, for downscaled ImageNet, which consists of 64×64 RGB images and 1000 classes, we employed the original training set of $1,281,167$ images, and evaluated on the original $50,000$ validation images.

F.2.2 Training schedules and implementation details

For distilled models, their teachers and their baselines, trained networks are initialized using the specialized scheme by Shi et al. [59], which reduces the magnitude of the IBP bounds. Teachers are trained using pure adversarial training, relying on a 10-step PGD attack [47] with step size $\eta = 0.25\epsilon$. Complying with De Palma et al. [21], $\mathbf{x}_{\text{adv}}$ is computed using a single-step PGD attack with uniform random initialization and step size $\eta = 10.0\epsilon$ on all setups except for CIFAR-10 with $\epsilon = 2/255$, which instead relies on a 8-step PGD attack with step size $\eta = 0.25\epsilon'$ run on a larger effective input perturbation ($\epsilon' = 2.1\epsilon$). All methods use a batch size of 128 throughout the experiments, and gradient clipping with norm equal to 10. We do not perform any early stopping.

As relatively common in the adversarial training literature [3, 17], teachers are trained using the SGD optimizer (with momentum set to 0.9), shorter schedules and a cyclic learning rate, which linearly increases the learning rate from 0 to 0.2 during the first half of the training, and then decreases it back to 0. As a longer number of teacher training epochs was found to be beneficial on TinyImageNet, we report it among the hyper-parameters in table 4. As instead common in the certified training literature, student models and their baselines are trained using the Adam optimizer [40] with longer training schedules and a learning rate of 5×10^{-4}, which is decayed twice by a factor 0.2. Their training starts with one epoch of "warm-up", where the certified training loss is replaced by the standard cross entropy on the clean inputs, and proceeds with a "ramp-up" phase, during which the training perturbation radius ϵ is gradually increased from 0 to its target value, and the IBP regularization from Shi et al. [59] is added to the training loss (with coefficient $\lambda = 0.5$ on CIFAR-10, and $\lambda = 0.2$ for TinyImageNet and downscaled ImageNet). Benchmark-specific details, which are in accordance with De Palma et al. [21], follow:

- CIFAR-10 with $\epsilon = 2/255$: the training schedule for students and baselines is 160-epochs long, with 80 epochs of ramp-up, and with the learning rate decayed at epochs 120 and 140.
- CIFAR-10 with $\epsilon = 8/255$: 260 epochs, with 80 epochs of ramp-up, and with the learning rate decayed at epochs 180 and 220.
- TinyImageNet: 160 epochs, with 80 ramp-up epochs, and decaying the learning rate at epochs 120 and 140.
- Downscaled ImageNet: 80 epochs, with 20 epochs of ramp-up, and decaying the learning rate at epochs 60 and 70.

In order to separate the effect of distillation from any other potential implementation detail, the baseline evaluations (CC-IBP, MTL-IBP, SABR, and IBP-R) from tables 1 and 2 were obtained by employing the student implementation with $\beta = 0$.

F.2.3 Network architecture

The employed `CNN-7` architecture is left unvaried with respect to previous work [21, 48, 51, 55] except on downscaled ImageNet, where we applied a small modification to the linear layers. Specifically, in order to make sure that the feature space employed for distillation is larger than the network output space, we set the output size of the penultimate layer (and the input size of the last) equal to 1024, instead of 512 as for the other datasets and in De Palma et al. [21]. As originally suggested by Shi et al. [59] to improve performance, and complying with previous work [21, 48, 51, 55], we employ batch normalization (BatchNorm) after every network layer except the last. In our implementation, adversarial attacks are carried out with the network in evaluation mode, hence using the current BatchNorm running statistics. Except during the warm-up phase, where we also perform an evaluation on the unperturbed data points, the network is exclusively evaluated on the computed adversarial inputs $\mathbf{x}_{\text{adv}}$, which hence dominate the computed running statistics for most of the training and for the final network. Finally, except for the clean loss during warm-up, which is computed using the unperturbed current batch statistics, all the training loss computations (including those requiring IBP bounds) are carried out using the batch statistics from the computed adversarial inputs (hence in training mode).

F.2.4 Hyper-parameters

As done in previous work [21, 30, 51, 55, 59, 75], and hence ensuring a fair comparison in table 5 within appendix G.1, tuning was carried out directly on the evaluation sets. Table 4 reports a list of

Table 4: Hyper-parameter settings for the models from tables 1 and 2, and their respective teachers.

Dataset	ϵ	Method	α or λ	ℓ_1	Teacher n. epochs	Teacher ℓ_1
CIFAR-10	$\frac{2}{255}$	CC-DIST	10^{-2}	3×10^{-6}	30	10^{-5}
		CC-IBP	10^{-2}	3×10^{-6}	/	/
		SABR-DIST	10^{-2}	3×10^{-6}	30	10^{-5}
		SABR	10^{-2}	3×10^{-6}	/	/
		MTL-DIST	4×10^{-3}	3×10^{-6}	30	10^{-5}
		MTL-IBP	4×10^{-3}	3×10^{-6}	/	/
		IBP-R-DIST	3×10^{-4}	3×10^{-6}	30	2×10^{-5}
		IBP-R	2×10^{-4}	3×10^{-6}	/	/
	$\frac{8}{255}$	CC-DIST	0.5	0	30	5×10^{-6}
		CC-IBP	0.5	0	/	/
		SABR-DIST	0.7	0	30	1×10^{-6}
		SABR	0.7	0	/	/
		MTL-DIST	0.5	1×10^{-7}	30	5×10^{-6}
		MTL-IBP	0.5	1×10^{-7}	/	/
TinyImageNet	$\frac{1}{255}$	CC-DIST	5×10^{-3}	5×10^{-5}	100	5×10^{-5}
		CC-IBP	3×10^{-3}	5×10^{-5}	/	/
		SABR-DIST	5×10^{-3}	5×10^{-5}	100	5×10^{-5}
		SABR	4×10^{-3}	5×10^{-5}	/	/
		MTL-DIST	3×10^{-3}	5×10^{-5}	100	5×10^{-5}
		MTL-IBP	2×10^{-3}	5×10^{-5}	/	/
		IBP-R-DIST	2×10^{-5}	5×10^{-5}	100	5×10^{-5}
		IBP-R	1×10^{-5}	5×10^{-5}	/	/
ImageNet64	$\frac{1}{255}$	CC-DIST	5×10^{-3}	10^{-5}	30	5×10^{-6}
		CC-IBP	5×10^{-3}	10^{-5}	/	/

the main method and regularization hyper-parameter values for our evaluations from tables 1 and 2. The details of the employed training schedules, taken from De Palma et al. [21], are instead reported in appendix F.2.2.

CC-Dist As explained in §3.2, we advocate for a constant distillation coefficient for CC-Dist: we noticed that $\beta = 5/w$ yielded good performance across our earlier CIFAR-10 experiments, and then decided to keep it to the same value across all evaluations. On CIFAR-10, we did not tune the α coefficient for either method: it was set to be the CC-IBP α coefficient employed in De Palma et al. [21]. Complying with common practice in the certified training literature [21, 51, 55], ℓ_1 regularization is applied on top of the employed training losses: the corresponding values for CC-Dist and CC-IBP models were not tuned but taken from the CC-IBP values reported in the expressive losses work [21]. On TinyImageNet and ImageNet64, we found that both the standard performance and the certified accuracy of CC-IBP could be significantly improved compared to the original results from De Palma et al. [21], which respectively employ $\alpha = 10^{-2}$ and $\alpha = 5 \times 10^{-2}$ for CC-IBP on TinyImageNet and ImageNet64, by decreasing α. In particular, we selected the smallest α value resulting in strictly better robustness-accuracy trade-offs (i.e., before certified accuracy started decreasing with lower α values). We trained a series of potential teachers for CC-Dist with varying ℓ_1 coefficient and number of training epochs, and selected the teacher depending on the performance of the resulting CC-Dist model. On all datasets except TinyImageNet, where longer schedules were beneficial to CC-Dist performance, we found a teacher trained with the relatively short 30-epoch schedule to result in strong CC-Dist performance. Finally, generally speaking, we advocate for the re-use of the selected CC-IBP α coefficient for CC-Dist. On TinyImageNet, the only setting where the employed α values for the reported models differ across CC-Dist and CC-IBP, we found a larger CC-Dist α to result in simultaneously better standard performance and certified robustness compared to all results from the literature. Nevertheless, re-using $\alpha = 3 \times 10^{-3}$ for CC-Dist produced a model with 44.08% standard accuracy and 27.40% certified accuracy (compare with table 5), which, albeit not improving on both metrics at once compared to the MTL-IBP results from [51], significantly

improves upon the overall performance trade-offs seen in the literature. Unless otherwise stated, all hyper-parameters throughout the experimental evaluations comply with those reported in table 4.

SABR-Dist, MTL-Dist and IBP-R-Dist On CIFAR-10 with $\epsilon = 2/255$ various teachers, method-specific regularization (λ) and distillation coefficients were tested for IBP-R-Dist and IBP-R. On the same benchmark, we only varied β and kept the same teacher as table 3 for SABR-Dist and MTL-Dist, keeping the α coefficients to the values tuned in De Palma et al. [21] for both and for their respective baselines. $\beta = 5/w$ was chosen for all methods, suggesting that, as for CC-Dist, it should not be treated as a hyper-parameter: we hence kept it fixed for both $\epsilon = 8/255$ experiments and TinyImageNet. On CIFAR-10 with $\epsilon = 8/255$, we kept α values as in De Palma et al. [21], and tested various teachers. As this was beneficial for CC-Dist, on TinyImageNet we tuned α and λ values for both distillation-based methods and their baselines, keeping the teachers fixed to the one employed for CC-Dist to minimize computational overhead.

F.3 Verification setup

The employed verification setup is analogous to the one from De Palma et al. [21]. As stated in §5, we use the open source OVAL verification framework [8, 9, 18], which performs branch-and-bound, and a configuration based on alpha-beta-CROWN bounds [65]. Before running branch-and-bound, we try verifying the property via IBP and CROWN bounds from `auto_LiRPA` [68], or to falsify it through a PGD attack [47]. Differently from De Palma et al. [21], we use a timeout of 600 seconds (as opposed to 1800 seconds). We perform verification by running the framework to compute $\min_i \mathbf{z}^{\mathcal{C}_\epsilon}(\mathbf{x}, y)_i$, where the $\min$ operator is converted into an equivalent auxiliary ReLU network to append to $\mathbf{z}_f(\mathbf{x}, y)$ [10, 21]. For TinyImageNet and downscaled ImageNet, differently from De Palma et al. [21], in order to reduce the size of the auxiliary network, we exclude from the $\min$ operator all the logit differences that were already proved to be positive by the IBP or CROWN bounds computed before running branch-and-bound. Finally, the configuration employed to verify the TinyImageNet and downscaled ImageNet teacher models (results reported in table 3) computes looser pre-activation bounds (using CROWN [74] as opposed to 5 iterations of alpha-CROWN [69]) at the root of branch-and-bound: this was found to be a more effective verifier for networks trained via pure adversarial training.

F.4 Software acknowledgments and licenses

As described above, our code relies on the OVAL verification framework to verify the models, which was released under an MIT license. The training codebase is analogous to the one from the expressive losses work [21], also released under an MIT license: this was in turn based on the codebase from Shi et al. [59], released under a 3-Clause BSD license. Both above repositories, and hence our codebase, rely on the `auto_LiRPA` [68] framework for incomplete verification, which has a 3-Clause BSD license. Concerning datasets: downscaled ImageNet was obtained from the ImageNet website (`https://www.image-net.org/download.php`), and TinyImageNet from the website of the CS231n Stanford class (`http://cs231n.stanford.edu/TinyImageNet-200.zip`). MNIST and CIFAR-10 were instead downloaded using `torchvision.datasets` [56].

G Supplementary experiments

This appendix reports experimental results omitted from the main paper.

G.1 Comparison of CC-Dist with literature results

In line with common practice in the area [21, 48, 55], we here compare the first repetition of the CC-Dist experiments from §5.1 with results taken from the certified training literature. For each method from the literature, we report the best attained performance across previous works and network architectures. In order to isolate the effect of any modification within our codebase and experimental setup, we report both (first-repetition) CC-IBP results from our evaluations, and the literature results corresponding to the best-performing expressive loss (CC-IBP, MTL-IBP, or Exp-IBP) from the original work [21].

Table 5 shows that CC-Dist establishes a new state-of-the-art, improving on both standard and certified accuracy compared to previously-reported results, on all benchmarks except CIFAR-10 with $\epsilon = 8/255$.

Table 5: Comparison of CC-Dist with literature results when training for certified robustness against ℓ_∞ norm perturbations. We highlight in bold the entries corresponding to the largest standard or certified accuracy for each benchmark, and, when they do not coincide, underline the best accuracies for ReLU-based architectures.

Dataset	ϵ	Method	Source	Standard acc. [%]	Certified acc. [%]
CIFAR-10	$\frac{2}{255}$	CC-DIST	this work	**81.55**	**64.60**
		CC-IBP	this work	79.51	63.50
		CC-IBP	De Palma et al. [21]	80.09	63.78
		MTL-IBP†	Mao et al. [51]	78.82	64.41
		STAPS	Mao et al. [48]	79.76	62.98
		SABR†	Mao et al. [49]	79.89	63.28
		SORTNET	Zhang et al. [73]	67.72	56.94
		IBP-R†	Mao et al. [49]	80.46	62.03
		IBP†	Mao et al. [49]	68.06	56.18
		CROWN-IBP†	Mao et al. [51]	67.60	53.97
	$\frac{8}{255}$	CC-DIST	this work	**55.13**	<u>35.52</u>
		CC-IBP	this work	54.46	35.42
		CC-IBP	De Palma et al. [21]	53.71	35.27
		MTL-IBP†	Mao et al. [51]	54.28	35.41
		STAPS	Mao et al. [48]	52.82	34.65
		SABR†	Mao et al. [51]	52.71	35.34
		SORTNET	Zhang et al. [73]	54.84	**40.39**
		IBP-R	De Palma et al. [19]	52.74	27.55
		IBP	Shi et al. [59]	48.94	34.97
		CROWN-IBP†	Mao et al. [51]	48.25	32.59
TinyImageNet	$\frac{1}{255}$	CC-DIST	this work	**43.78**	**27.88**
		CC-IBP	this work	41.28	26.53
		EXP-IBP	De Palma et al. [21]	38.71	26.18
		MTL-IBP†	Mao et al. [51]	35.97	27.73
		STAPS†	Mao et al. [51]	30.63	22.31
		SABR†	De Palma et al. [21]	38.68	25.85
		SORTNET	Zhang et al. [73]	25.69	18.18
		IBP†	Mao et al. [51]	26.77	19.82
		CROWN-IBP†	Mao et al. [51]	28.44	22.14
ImageNet64	$\frac{1}{255}$	CC-DIST	this work	**28.17**	**13.96**
		CC-IBP	this work	25.94	13.69
		EXP-IBP	De Palma et al. [21]	22.73	13.30
		SABR†	De Palma et al. [21]	20.33	12.39
		SORTNET	Zhang et al. [73]	14.79	9.54
		CROWN-IBP	Xu et al. [68]	16.23	8.73
		IBP	Gowal et al. [30]	15.96	6.13

†Evaluation from later work attaining a larger standard or certified accuracy than reported in the original work.

Differently from the other settings, the best certified accuracy on this benchmark is attained by SortNet [73], which uses a specialized 1-Lipschitz network architecture (see §4). Its performance on this setup can be further improved through the use of additional generated data [1], which is outside the scope of our work. While the sub-par performance of ReLU networks on this benchmark is well known in the literature [21, 55], we are hopeful that distillation from empirically-robust teachers may be beneficial to 1-Lipschitz networks too, and defer the investigation to future work. On TinyImageNet and downscaled ImageNet we found that both the standard and certified accuracies of CC-IBP benefit from a smaller α coefficient than those employed in De Palma et al. [21], explaining the difference between our CC-IBP results and those reported in the original work. Owing to the combined effect of this and of the benefits of our novel distillation loss, the difference between CC-Dist and the results from the literature is particularly remarkable on these two larger-scale benchmarks.

G.2 Other special cases of the CC-Dist distillation loss

As seen in proposition 3.2, when choosing $\alpha \in [0, 1]$, $\mathcal{R}^{\mathcal{C}_\epsilon}_{\text{cc}}(\alpha; \mathbf{x}, y)$ can continuously interpolate between lower and upper bounds to the worst-case feature-space distillation loss in equation (8).

While §3.2 advocates for the use of the same α coefficient employed by CC-IBP in order to closely mirror its loss (see lemma 3.3), we expect other choices of the α coefficient to work well on selected benchmarks.

G.2.1 CC-Dist$_0$ and CC-Dist$_1$

We here evaluate the behavior of two training schemes that correspond to other special cases of the CC-Dist distillation loss. Specifically, we consider the use of $\mathcal{R}_{\text{cc}}^{\mathcal{C}_\epsilon}(0; \mathbf{x}, y) = \underline{\mathcal{R}}^{\mathcal{C}_\epsilon}(\mathbf{x}, y)$ and $\mathcal{R}_{\text{cc}}^{\mathcal{C}_\epsilon}(1; \mathbf{x}, y) = \bar{\mathcal{R}}^{\mathcal{C}_\epsilon}(\mathbf{x}, y)$ on top of CC-IBP, calling the resulting training schemes CC-Dist$_0$ and CC-Dist$_1$, respectively:

$$\mathcal{L}^{\text{CC-Dist}_0}(\alpha, \beta; \mathbf{x}, y) := \mathcal{L}_{\text{cc}}^{\mathcal{C}_\epsilon}(\alpha; \mathbf{x}, y) + \beta\, \mathcal{R}_{\text{cc}}^{\mathcal{C}_\epsilon}(0; \mathbf{x}, y),$$
$$\mathcal{L}^{\text{CC-Dist}_1}(\alpha, \beta; \mathbf{x}, y) := \mathcal{L}_{\text{cc}}^{\mathcal{C}_\epsilon}(\alpha; \mathbf{x}, y) + \beta\, \mathcal{R}_{\text{cc}}^{\mathcal{C}_\epsilon}(1; \mathbf{x}, y).$$

Keeping ℓ_1 regularization and the α coefficient of the CC-IBP loss fixed, we first tested different distillation coefficients β while using the teachers employed for the CC-Dist models (see tables 3 and 4), finding that $\beta = 5/w$ yields the largest natural accuracy improvement for both methods on $\epsilon = 8/255$, and a strong trade-off between certified robustness and standard performance for CC-Dist$_0$ on $\epsilon = 2/255$. We found $\beta = (5\times10^{-4})/w$ to work better for CC-Dist$_1$ on $\epsilon = 2/255$. Then, using the above distillation coefficients, and similarly to what done for CC-Dist (see appendix F.2.4), we tested the two methods using teachers with varying ℓ_1 regularization and trained for 30 epochs. Aiming to compare them with CC-Dist, Table 6 reports results for the best-performing CC-Dist$_0$ and CC-Dist$_1$ models from the above procedure that display similar performance profiles with CC-Dist.

Table 6 shows that both CC-Dist$_0$ and CC-Dist$_1$ work well across the considered CIFAR-10 settings. As we would expect considering the low employed α coefficient (see table 4), CC-Dist$_0$ strictly outperforms CC-Dist$_1$ on $\epsilon = 2/255$, where both methods nevertheless improve on both the standard performance and the certified accuracy of CC-IBP. While CC-Dist$_0$ displays performance comparable to CC-Dist in this setting, CC-Dist$_1$ performs markedly worse. On $\epsilon = 8/255$, where the employed α coefficient is instead relatively large (see table 4), CC-Dist$_1$ strictly outperforms CC-Dist$_0$ on both reported performance profiles, indicating the benefit of distilling onto feature IBP bounds in this context. Nevertheless, both methods can produce models improving on CC-IBP for both metrics on $\epsilon = 8/255$, with the corresponding CC-Dist$_0$ model strictly outperformed by CC-Dist.

We believe that the above results further confirm the effectiveness of using empirically-robust models to improve certified training via knowledge distillation. As the relative performance of CC-Dist$_0$ and CC-Dist$_1$ varies depending on the experimental setting, we advocate for the use of CC-Dist as described in §3.2 due to its adaptive nature. Nevertheless, given that most considered benchmarks require relatively low α coefficients (see table 4), we expect CC-Dist$_0$ to be a valid alternative to CC-Dist on TinyImageNet and downscaled ImageNet. We note, however, that re-running the above experiments with multiple seeds could lead to different conclusions: this does not affect the main contribution of this work, which is to show the benefits of distillation from adversarially-trained teachers.

Table 6: Comparison between the CC-Dist models from table 5 and other special cases of the CC-Dist parametrized distillation loss.

Dataset	ϵ	Method	β	Teacher ℓ_1	Standard acc. [%]	Certified acc. [%]
CIFAR-10	$\frac{2}{255}$	CC-DIST	$5/w$	10^{-5}	81.55	64.60
		CC-DIST$_0$	$5/w$	10^{-5}	81.74	64.22
		CC-DIST-Clean	$5/w$	5×10^{-5}	81.52	64.50
		CC-DIST$_1$	$(5\times10^{-4})/w$	5×10^{-5}	79.84	63.93
		CC-IBP	0	/	79.51	63.50
	$\frac{8}{255}$	CC-DIST	$5/w$	5×10^{-6}	55.13	35.52
		CC-DIST$_0$	$5/w$	2×10^{-5}	55.63	35.08
		CC-DIST-Clean	$2/w$	5×10^{-6}	55.47	35.06
		CC-DIST$_1$	$5/w$	2×10^{-6}	55.91	35.35
		CC-DIST$_0$	$2/w$	5×10^{-6}	54.65	35.46
		CC-DIST-Clean	$2/w$	5×10^{-6}	54.71	35.13
		CC-DIST$_1$	$2/w$	5×10^{-6}	54.87	35.55
		CC-IBP	0	/	54.46	35.42

G.2.2 CC-Dist-Clean

Recalling that $\mathcal{R}_{\text{cc}}^{\mathcal{C}_\epsilon}(0; \mathbf{x}, y) = \|h(\mathbf{x}_{\text{adv}}) - h^t(\mathbf{x})\|_2^2$, we point out that setting $\mathbf{x}_{\text{adv}} = \mathbf{x}$ would result in the standard feature-space distillation loss (see §2.3). We hence consider this as an additional special case of $\mathcal{R}_{\text{cc}}^{\mathcal{C}_\epsilon}(\alpha; \mathbf{x}, y)$, calling the resulting algorithm CC-Dist-Clean:

$$\mathcal{L}^{\text{CC-Dist-Clean}}(\alpha, \beta; \mathbf{x}, y) := \mathcal{L}_{\text{cc}}^{\mathcal{C}_\epsilon}(\alpha; \mathbf{x}, y) + \beta \left\|h(\mathbf{x}) - h^t(\mathbf{x})\right\|_2^2 .$$

Keeping the teachers fixed to those employed in table 1 for CC-Dist, we tested the performance of CC-Dist-Clean for varying β parameters on CIFAR-10, and report the best results in table 6. CC-Dist-Clean displays performance profiles similar to CC-Dist$_0$, suggesting that the use of adversarial features is not crucial when lower bounding the worst-case distillation loss via $\mathcal{R}^{\mathcal{C}_\epsilon}(\mathbf{x}, y)$. We advocate for CC-Dist$_0$ over CC-Dist-Clean as the latter requires an extra clean forward pass, whereas the adversarial inputs come at no additional cost to CC-IBP.

G.3 Logit-based distillation

We now compare the results of $\mathcal{R}_{\text{cc}}^{\mathcal{C}_\epsilon}(\alpha; \mathbf{x}, y)$ with a loss that instead seeks to directly distill onto the CC-IBP convex combinations $\mathbf{z}_{\text{cc}}^{\mathcal{C}_\epsilon}(\alpha; \mathbf{x}, y)$. Specifically, it employs a KL term between $\mathbf{z}_{\text{cc}}^{\mathcal{C}_\epsilon}(\alpha; \mathbf{x}, y)$ and the teacher logit differences $\mathbf{z}_t(\mathbf{x}, y)$, resulting in the following training loss:

$$\mathcal{L}_{\text{cc}}^{\mathcal{C}_\epsilon}(\alpha; \mathbf{x}, y) + T^2 \beta \, \text{KL}_T \left(-\mathbf{z}_{\text{cc}}^{\mathcal{C}_\epsilon}(\alpha; \mathbf{x}, y), -\mathbf{z}_t(\mathbf{x}, y)\right) . \tag{15}$$

Keeping all other hyper-parameters fixed (including the teacher model) to the values reported in table 4, we tested different distillation coefficients β, using $T = 20$. We found that, compared to CC-IBP, the best performance profile using equation (15) were obtained using $\beta = {}^{(10^4)}/_{w}$ for both considered CIFAR-10 setups, whose results are reported in table 7. CC-Dist outperforms the logit-space distillation loss in both settings. We hypothesize this may be linked to the more informative content of the model features: allowing the student to learn a markedly different classification head from the teacher may be beneficial in the context of the employed loss functions.

G.4 PreActResNet18 teachers

We now present a preliminary evaluation of the effect of employing a PreActResNet18 (`PRN18`) architecture [31] as teacher for CC-Dist. In order to ensure that the feature space of the teacher is set to the output of a ReLU, in compliance with the student model, we place the PreActResNet18 average pooling layer before the last ReLU and batch normalization (the standard PreActResNet18 architecture places it instead before the last linear layer). Focusing on TinyImageNet, and keeping $\alpha = 3 \times 10^{-3}$ and the ℓ_1 coefficient to 5×10^{-5} as for the CC-IBP model on this setup (see table 4), we tested various `PRN18` teachers trained for 30 epochs with varying ℓ_1 regularization as CC-Dist teachers. Table 8 compares the performance of the ensuing CC-Dist model having the largest natural accuracy with the best model obtained through same-architecture teachers for these hyper-parameters, and the respective teachers. CC-Dist draws no benefit from the `PRN18` teacher (which was trained with ℓ_1 coefficient equal to 5×10^{-5}), in spite of the fact that it displays stronger performance than the `CNN-7` teacher. While we are hopeful that better CC-Dist results could be obtained by training `PRN18` teachers for longer (the considered teachers are trained for 30 epochs, as opposed to the 100 epochs employed for the `CNN-7` teacher), we leave this for future work.

G.5 Runtime measurements

In order to assess the training overhead associated with our distillation scheme, we here present runtime measurements and estimates for CC-Dist and CC-IBP. Specifically, we provide the training

Table 7: Comparison between the logit-space distillation from equation (15) and the CC-Dist models from table 5.

Dataset	ϵ	Method	Standard acc. [%]	Certified acc. [%]
CIFAR-10	$\frac{2}{255}$	CC-DIST	81.55	64.60
		EQ. (15)	80.61	63.65
	$\frac{8}{255}$	CC-DIST	55.13	35.52
		EQ. (15)	54.76	35.20

Table 8: Evaluation of the effect of PreActResNet18 (PRN18) teachers (tch.) on the TinyImageNet performance of CC-Dist, compared to teachers with the same architecture as the student.

Dataset	ϵ	Tch. arch.	Std. acc. [%]	Cert. acc. [%]	Tch. std. acc. [%]	Tch. PGD-40 acc. [%]
TinyImageNet	$\frac{1}{255}$	CNN-7	44.08	27.40	47.18	36.07
		PRN18	43.03	26.09	50.90	39.98

runtime of both methods, separately including also the teacher training runtime for CC-Dist, and estimates of the verification runtimes for the trained models from table 1. These experiments were carried out using an Nvidia RTX 8000 GPU, and 6 cores of an AMD EPIC 7302 CPU. Table 9 shows that, under the training schedules of appendix F.2.2, CC-Dist is associated with minimal training overhead on the considered CIFAR-10 settings. This overhead increases to respectively almost 70% and 40% for TinyImageNet and downscaled ImageNet, where stronger teachers (with relatively longer training schedules) are required in order to maximize performance. Finally, we note that the increased certified accuracy of the CC-Dist models (see table 1) comes with an increased verification runtime.

Table 9: Training runtime measurements for CC-Dist, their respective teachers, and CC-IBP under the training schedules of appendix F.2.2.

Dataset	ϵ	Method	Training runtime [s]	Teacher training runtime [s]	Estimated[†] verification runtime [s]
CIFAR-10	$\frac{2}{255}$	CC-Dist	1.605×10^4	2.225×10^3	1.097×10^5
		CC-IBP	1.568×10^4	/	7.089×10^4
	$\frac{8}{255}$	CC-Dist	1.653×10^4	2.223×10^3	1.090×10^4
		CC-IBP	1.600×10^4	/	6.238×10^3
TinyImageNet	$\frac{1}{255}$	CC-Dist	5.432×10^4	3.470×10^4	2.778×10^5
		CC-IBP	5.246×10^4	/	1.608×10^5
ImageNet64	$\frac{1}{255}$	CC-Dist	3.228×10^5	1.089×10^5	1.210×10^6
		CC-IBP	3.095×10^5	/	7.794×10^5

[†]Extrapolated from measurements over the first 500 test images.

G.6 Different architecture and activation function

The main results focus on the CNN-7 architecture owing to its state-of-the-art performance and prevalence in the relevant literature [21, 48, 55, 59]. We here investigate whether the distillation process is beneficial beyond this context by studying the relative performance of CC-Dist on a different activation function and on a different architecture for both the teacher and the model, focussing on CIFAR-10 with $\epsilon = 2/255$. For the activation experiment, we modify CNN-7 to employ the hyperbolic tangent (tanh), testing different teachers trained for 100 epochs (choosing a teacher ℓ_1 coefficient of 2×10^{-5}), and varying β values (settling for $\beta = 5/w$ as for the experiments in table 1). For the architecture experiment, we use PreActResNet18 (PRN18) for both the teacher and the student, testing different teachers trained for 30 epochs (settling on a teacher with ℓ_1 coefficient of 5×10^{-6}), and using $\beta = 5/w$. In both cases, owing to the lack of support for either model from the OVAL branch-and-bound framework [8, 9, 18] employed throughout the paper, we use CROWN [74] from auto_LiRPA [68] as post-training verification algorithm. Table 10 shows that CC-Dist successfully improves robustness-accuracy trade-offs for both the considered settings, demonstrating the wider applicability of the proposed distillation technique.

Table 10: Effect of distillation beyond CNN-7 students on CIFAR-10 with $\epsilon = 2/255$ for CC-Dist.

(a) Modified CNN-7 with tanh activation functions.

Method	Std. acc. [%]	CROWN acc. [%]
CC-Dist	72.38	49.42
CC-IBP	71.29	47.39

(b) PreActResNet18 architecture.

Method	Std. acc. [%]	CROWN acc. [%]
CC-Dist	74.01	53.63
CC-IBP	73.40	52.88

Table 11: Effect of distillation from the adversarial teacher features on CC-Dist.

Dataset	ϵ	Teacher features	Standard acc. [%]	AA acc. [%]	Certified acc.† [%]
CIFAR-10	$\frac{2}{255}$	CLEAN	81.55	70.71	64.60
		ADVERSARIAL	81.39	70.16	64.15
	$\frac{8}{255}$	CLEAN	55.13	36.80	35.52
		ADVERSARIAL	55.64	36.84	35.04

G.7 Distillation on earlier latents

Throughout the paper, we define the feature space as the activations before last affine network layer, in accordance with the conditions of lemma 3.2. We here investigate the effect of computing on the distillation loss $\mathcal{R}_{\text{cc}}^{\mathcal{C}_\epsilon}(\alpha; \mathbf{x}, y)$ on an earlier feature space, corresponding to the activations before the penultimate affine layer of `CNN-7`, focusing on CC-Dist. In particular, we focus on CIFAR-10 with $\epsilon = 2/255$, keeping the teacher model fixed to the one used for tables 1 and 3 and varying the β coefficient to account for the change in the feature space: we found $\beta = 2/w$ to yield the maximize performance in this context. Table 13 shows that distillation for the last affine layer yields strictly better robustness-accuracy trade-offs than those presented in table 5.1. As distilling on earlier layers implies that a larger portion of the network is exclusively trained using the CC-IBP loss, we ascribe this to insufficient teacher-student coupling.

G.8 Clean teachers

In spite of our focus on learning from adversarially-trained teachers, $\mathcal{R}_{\text{cc}}^{\mathcal{C}_\epsilon}(\alpha; \mathbf{x}, y)$ only makes use of the clean features of the teacher model. In order to investigate whether the empirical robustness of the teacher is indeed necessary for an effective distillation process, we here study the effect of employing standard-trained teachers for CC-Dist. In particular, focusing on CIFAR-10 with $\epsilon = 2/255$, we keep $\beta = 2/w$ and test the use of SGD-trained teachers trained with varying ℓ_1 regularization coefficients (5×10^{-6} being the chosen ℓ_1 coefficient). Table 14 shows that the use of standard teachers markedly worsens the certified accuracy of the student, which is now inferior to the one associated to CC-IBP (see table 5). These results demonstrate that a robust teacher representation is a key requirement for the success of the distillation process

G.9 Distilling from adversarial teacher features

As explained in the previous appendix, all the distillation losses from §3 perform distillation from the clean teacher features $h^t(\mathbf{x})$. Having shown that robustness of the teacher model is crucial to maximize performance, we investigate how the input point on which the teacher features are computed impacts the effectiveness of the distillation process. Specifically, focusing on CC-Dist and CIFAR-10, we consider the effect of using teacher features computed on $\mathbf{x}_{\text{adv}}$, the adversarial input computed on the student model, referring to $h^t(\mathbf{x}_{\text{adv}})$ as adversarial teacher features. Table 11 shows that clean teacher features yield strictly better performance for $\epsilon = 2/255$ and are hence to be preferred. We hypothesize that, in spite of the teacher's robustness, its clean features may provide a higher-quality representation to learn from.

H Significance of the empirical results

In order to put the improvements from our distillation framework into context, table 12 reports those associated with relevant recent work in ℓ_∞ certified training for ReLU networks. We exclude CIFAR-10 with $\epsilon = 8/255$, where Lipschitz-based methods have been the unquestioned state-of-the-art [73], and consider both improvements to previous literature results, and to re-tuned baselines within each work (for instance, results reported in De Palma et al. [21] are compared against their own SABR results [21, table 9]). While, as shown in table 9, distillation improvements are associated with a non-negligible additional computational overhead at training or verification time, they are in line with the best algorithmic improvements in the area, pointing to their clear significance within the context of certified training.

Table 12: Comparison of CC-Dist average improvements (from table 1) with those obtained by relevant previous work in the area on CIFAR-10 with $\epsilon = 2/255$, TinyImageNet and downscaled ImageNet. These are computed as the difference between the single-seed results from each of these works with the best-performing model from earlier methods. We consider both previously-reported literature results and re-tuned baselines within each of these works.

Dataset	ϵ	Method	Source	Standard acc. impr. [%]	Certified acc. impr. [%]
CIFAR-10	$\frac{2}{255}$	CC-DIST	this work	1.76	0.92
		CC-IBP	De Palma et al. [21]	0.20	0.50
		STAPS	Mao et al. [48]	0.52	0.14
		SABR	Müller et al. [55]	1.05	0.87
TinyImageNet	$\frac{1}{255}$	CC-DIST	this work	2.55	1.16
		EXP-IBP	De Palma et al. [21]	0.03	0.33
		STAPS	Mao et al. [48]	0.13	1.70
		SABR	Müller et al. [55]	2.93	2.59
ImageNet64	$\frac{1}{255}$	CC-DIST	this work	2.21	0.56
		EXP-IBP	De Palma et al. [21]	2.40	0.91

I Details on the employed notation

We here provide additional details on the employed notation.

Student and teacher models Input-to-logits maps are defined as the composition of a feature map, denoted using the letter h, and classification heads, denoted using the letter g. The (sub-)networks for the teacher model are denoted by the use of the t superscript (e.g, h^t).

Bounds to worst-case quantities The worst-case classification loss, equation (3), and the worst-case distillation loss, equation (8), are superscripted by the relative local perturbation set $\mathcal{C}_\epsilon$ around the input $\mathbf{x}$. Lower and upper bounds to both quantities are denoted through lower and upper bars, respectively (e.g., $\underline{\mathcal{L}}^{\mathcal{C}_\epsilon}(\mathbf{x}, y)$), with the superscript preserved to stress their local validity. A similar notation is employed for the local bounds to the worst-case logit differences (e.g, $\underline{\mathbf{z}}^{\mathcal{C}_\epsilon}(\mathbf{x}, y)$, which lower bounds equation 2. Convex combinations between lower and upper bounds, parametrized by α, are denoted by subscript "cc" standing for convex combination (e.g., $\mathcal{R}^{\mathcal{C}_\epsilon}_{\text{cc}}(\alpha; \mathbf{x}, y)$), without any bars. For all these worst-case quantities, lower bounds corresponding to evaluations on concrete inputs from adversarial attacks, and the upper bounds are obtained through network convex relaxations (specifically, IBP).

Bounds to the student features We again employ a similar notation when bounding the student features $h(\mathbf{x})$, defined in proposition 3.1. In this context, both the lower and the upper bounds are obtained through IBP. Convex combinations between the adversarial student latents $h(\mathbf{x}_{\text{adv}})$ and the IBP lower and upper bounds, as defined in equation 9, are denoted by subscript "cc" and lower and upper bars, respectively (e.g., $\bar{h}^{\mathcal{C}_\epsilon}_{\text{cc}}(\alpha; \mathbf{x})$).

Table 13: Effect of computing the distillation loss $\mathcal{R}^{\mathcal{C}_\epsilon}_{\text{cc}}(\alpha; \mathbf{x}, y)$ on an earlier feature space for CC-Dist.

Dataset	ϵ	Feature space	Std. acc. [%]	Cert. acc. [%]
CIFAR-10	$\frac{2}{255}$	`Last`	81.55	64.60
		`Penultimate`	80.84	63.11

Table 14: Effect of performing distillation from clean teachers for CC-Dist.

Dataset	ϵ	Teacher training	Std. acc. [%]	Cert. acc. [%]
CIFAR-10	$\frac{2}{255}$	`PGD-10`	81.55	64.60
		`Standard`	81.48	62.52

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: Our claims match the presented techniques and the provided experimental results, which hold on popular benchmarks from the certifiably robust vision literature, which are the target of our work.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: Limitations are discussed in appendix A.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [Yes]

Justification: Assumptions are stated in the result statements. Proofs of technical results are provided in appendix D.

Guidelines:

- The answer [N/A] means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: Pseudo-code is provided in appendix E. Information to reproduce the experiments can be found in appendix F.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification: Code for CC-Dist, SABR-Dist, MTL-Dist and IBP-R-Dist, along with sample CC-Dist student and teacher models, is provided at `https://github.com/alessandrodepalma/feature-distillation-for-certified-training`. The employed datasets are freely available online.

Guidelines:

- The answer [N/A] means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

Answer: [Yes]

Justification: Experimental settings are outlined in §5; details, including hyper-parameters, are provided in appendix F.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
- The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

Answer: [Yes]

Justification: Our main CC-Dist experiments (table 1) report means over six repetitions and their 95% Confidence Intervals (CIs). Our main SABR-Dist, MTL-Dist and IBP-R-Dist results (table 2) report means over four repetitions and 95% CIs. The other experiments are run using a single seed, owing to the large cost associated with verification (a timeout of 600 seconds per evaluation image is employed), and in line with related previous works [21, 48, 55].

Guidelines:

- The answer [N/A] means that the paper does not include experiments.

- The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: Information on the employed compute resources is provided in appendix F.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: The authors have read and adhered to NeurIPS Code of Ethics.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: See appendix A.

Guidelines:

- The answer [N/A] means that there is no societal impact of the work performed.
- If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: Owing to the relatively small scale of the models and to the foundational nature of the research, we believe that certifiably-robust models pose no such risks.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: the sources of data and models are cited throughout the paper. Software acknowledgments are provided in appendix F.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.

- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: a `README` file is provided in the code repository.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: the paper does not involve neither crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: the paper does not involve neither crowdsourcing nor research with human subjects.

    Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

    Answer: [N/A]

    Justification: No LLM impact on core methodology.

    Guidelines:

    - The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.