Journal of Machine Learning Research 24 (2023) 1-42 Submitted 5/23; Revised 10/23; Published 12/23

RVCL: Evaluating the Robustness of Contrastive Learning via Verification

Zekai Wang WZEKAI99@GMAIL.COM
School of Computer Science
Wuhan University
Wuhan, Hubei, China

Weiwei Liu* LIUWEIWEI863@GMAIL.COM
School of Computer Science
Wuhan University
Wuhan, Hubei, China

Editor: Aapo Hyvarinen

Abstract

Contrastive adversarial training has successfully improved the robustness of contrastive learning (CL). However, the robustness metric in these methods depends on attack algorithms, image labels, and downstream tasks, introducing reliability concerns. To address these issues, this paper proposes a novel Robustness Verification framework for Contrastive Learning (RVCL). Specifically, we define the verification problem of CL from deterministic and probabilistic perspectives, then provide several effective metrics to evaluate the robustness of CL encoder. Furthermore, we use extreme value theory to reveal the relationship between the robust radius of the CL encoder and that of the supervised downstream task. Extensive experiments on various benchmark models and datasets validate theoretical findings, and further demonstrate RVCL's capability to evaluate the robustness of both CL encoders and images.

Keywords: Robustness Verification, Contrastive Learning, Adversarial Training

1 Introduction

While neural networks (NNs) have exhibited impressive performance across various applications, numerous studies (Goodfellow et al., 2015; Madry et al., 2018) have underscored the vulnerability of NNs to imperceptibly perturbed images. Consequently, a rapidly growing body of work aims to enhance the robustness of NNs.

One notably effective approach in this field is adversarial training (AT) (Moosavi-Dezfooli et al., 2016; Carlini and Wagner, 2017), which improves the robustness of NN by augmenting the training set with adversarial samples (Szegedy et al., 2014). Schmidt et al. (2018) highlight that AT requires a large amount of data, while labels of data are expensive to obtain. Consequently, efforts have been made to enhance the robustness of adversarially trained models by incorporating additional unlabeled data (Alayrac et al., 2019; Carmon et al., 2019; Zhai et al., 2019).

*. Corresponding Author.

Recently, attempts have been made to combine AT with contrastive learning (CL) (Kim et al., 2020; Fan et al., 2021). CL (Chen et al., 2020; He et al., 2020) has showcased exceptional capabilities in unsupervised learning. On downstream image classification tasks, CL can surpass the standard accuracy of supervised learning through the utilization of large-scale unlabeled dataset. It is therefore of primary interest to study the robust performance achieved by contrastive AT (Gowal et al., 2021).

However, existing contrastive AT methods employ the *empirical robustness metric*, specifically robust accuracy, to assess encoder robustness. These methods entail training a linear classifier on the labeled training dataset, utilizing features extracted by the CL encoder. Subsequently, they employ adversarial test images to assess encoder robustness. Robust accuracy is contingent upon attack algorithms, image labels, and downstream tasks, giving rise to three inherent issues:

1. Attack algorithm: Robust accuracy is given by specific attack algorithm, such as PGD attack (Madry et al., 2018). Consequently, the obtained results may not align consistently with potent adversaries like AutoAttack (Croce and Hein, 2020b)

2. Image label: The practice of training with unlabeled images while evaluating with labeled images introduces a substantial disconnect in the evaluation process.

3. Downstream task: Determining the factors influencing robust accuracy poses a challenge, as it could stem from either the training process of downstream task or the encoder's robustness.

In fact, the first issue has been explored in supervised learning. Prior research has revealed that the majority of defense heuristics prove ineffective against sufficiently potent attack algorithms (Carlini and Wagner, 2017; Uesato et al., 2018). Furthermore, even if the model successfully defends against the attack algorithm used in the evaluation, there is no assurance that it ill retain such robustness when confronted with novel and unforeseen attackers. This has encouraged researchers to develop *robustness verification* (Katz et al., 2017; Wong et al., 2018): the classifier is verified to remain consistent predictions within the neighborhood of a given point x, regardless of what attack algorithm is applied. Robustness verification finds the largest radius of this neighborhood, referred to as the *robust radius*, which serves as an important metric for assessing neural network robustness.

Regrettably, prior studies of robustness verification have exclusively relied on supervised learning, mandating true data labels for both the supervised classifier and verifier. A simplistic solution is to apply the existing verification framework to supervised downstream tasks. However, this approach would inherit the challenges outlined in Issues 2 & 3, which prompts us to pose the following questions:

- *Can we formulate a robustness verification framework for contrastive learning, eliminating the need for class labels and downstream tasks?*

- *Is there any correlation between the robust radius of the CL encoder and that of the downstream task?*

Our research endeavors to conduct a rigorous and comprehensive investigation to address these questions. Our principal contributions are outlined as follows:

1. We propose RVCL, a novel Robustness Verification framework for Contrastive Learning. We define the robust radius for CL, where points falling within this radius are identified as negative samples with zero probability (deterministic verification) or small probability (probabilistic verification). This novel metric assesses the robustness of encoders, obviating the need for attack algorithms, image labels, or downstream tasks.
2. We employ extreme value theory to establish that, in both binary and multi-class scenarios, the robust radius of the CL encoder serves as an upper bound for the downstream task's robust radius. This finding aligns with our experimental results.
3. To validate the efficacy of RVCL, we conduct extensive experiments on benchmark datasets (Mnist, CIFAR-10, CIFAR-100) and diverse models (CNN, ResNet). Our experimental results demonstrate that RVCL provides a suitable robustness metric for different models without requiring labels, thus corroborating our theoretical analysis. Furthermore, RVCL effectively evaluates the anti-disturbance ability of various images.

2 Related work

2.1 Self-supervised Contrastive Learning

Self-supervised learning (Jing and Tian, 2021), which involves training models using unlabeled data and various pretext tasks, has become popular as a means of extracting feature representation for deep NNs. Early advances have been used to solve image jigsaw puzzles (Noroozi and Favaro, 2016), predict rotation angles (Gidaris et al., 2018), fill image patches (Doersch et al., 2015), etc. Recently, contrastive learning (CL) (Chen et al., 2020; He et al., 2020; Tian et al., 2020; Wu et al., 2018) has been proposed by maximizing the agreement between positive samples (e.g., data augmentation) while contrasting with negative samples, and has further been shown to work well in learning effective representations. Some theoretical works have also been proposed; for example, Saunshi et al. (2019) provide the first generalization bound for CL, while Nozawa et al. (2020) extend it by means of a PAC-Bayesian approach.

2.2 Contrastive Adversarial Training

Due to the brittleness of NNs when faced with tiny input perturbations, AT (Madry et al., 2018) is one of the most powerful robust training methods used to enhance model robustness. Several recent works (Kim et al., 2020; Ho and Vasconcelos, 2020; Jiang et al., 2020; Fan et al., 2021) have explored how to improve robustness using contrastive AT. To obtain more robust data representations, AT is used on contrastive pretraining tasks following the "contrastive adversarial pretraining + supervised finetuning" paradigm. However, existing methods use an empirical robustness metric; the systematic study of the verified robustness of CL have been less explored.

2.3 Robustness Verification

In this paper, we consider two branches of verification approaches: *deterministic verification* and *probabilistic verification*, following the taxonomy of Li et al. (2020). However, previous works focus on supervised settings; verification with CL settings remain unknown.

Deterministic verification When the given input is non-robust against the attack, deterministic verification is guaranteed to identify this nonrobustness and outputs "not verified". The literature for this setting can be broadly divided into several categories: complete verifiers using satisfiability modulo theory (SMT) (Katz et al., 2017; Ehlers, 2017), mixed integer programming (MIP) (Tjeng et al., 2019; Anderson et al., 2020) and branch and bound (BaB) (Bunel et al., 2018; Wang et al., 2021); incomplete verifiers using bound propagation (Zhang et al., 2018; Xu et al., 2020), and convex relaxation by linear programming (Wong et al., 2018; Wong and Kolter, 2018).

Probabilistic verification Probabilistic verification is a promising method that can provide verified robustness for large NNs. Comparing with deterministic verification, probabilistic verification is guaranteed to output "not verified" with a high probability when the given input can be attacked successfully.

Probabilistic verification is based on randomized smoothing (RS), while RS originated from differential privacy (Lécuyer et al., 2019). Cohen et al. (2019) propose to add Gaussian noise to smooth the models, and thus derive the verified robustness for these smoothed classifiers. At present, RS is considered the state-of-the-art approach to offering a provable guarantee of robustness against ℓ_2-perturbations (Li et al., 2020), being the only type that can provide verification on large-scalse models and datasets. In light of this, many existing works focus on improving the robustness guarantee given by RS. such as by using different smoothing measures (Lee et al., 2019; Yang et al., 2020), different divergences (Dvijotham et al., 2020), etc.

2.4 Extreme Value Theory

Extreme value theory (EVT) has been recognized as a powerful tool, since it enables the limit distribution of properly normalized maxima to be effectively modeled (Scheirer et al., 2011). This success has produced strong empirical results for describable visual attributes (Scheirer et al., 2012), visual inspection tasks (Gibert et al., 2015) and open set recognition problems (Rudd et al., 2018), etc. Recently, CLEVER (Weng et al., 2018b) estimates the robust radius of supervised verification using EVT. The difference is discussed in more detail in Appendix D. In this paper, we creatively use EVT to theoretically analyze the relationship between the robust radius of the CL encoder and that of the downstream task.

3 Preliminaries

We first present notations and describe the frameworks for contrastive learning and supervised verification problem that will be essential for our analysis.

Notions: Let $k \quad k_p$ denote ℓ_p norm (p 2 $\mathbb{N}_+$ [$f+1$ g). E.g., $k \quad k_2$ and $k \quad k_\infty$ denote the Euclidean norm ℓ_2 and infinity norm ℓ_∞, respectively. $B_p(x_0, \epsilon) :=$ fx j $kx \quad x_0k_p \quad \epsilon$g denotes that the input x is constrained into the ℓ_p ball. $\mathbb{1}_{[Boolean\ expression]}$ is the indicator function (equal to 1 if the expression is True and 0 otherwise). We let sign$(x) = 1$ for $x \quad 0$ and sign$(x) = \quad 1$ for $x < 0$. If S is a set, jSj denotes its cardinality. The transpose of the vector/matrix is represented by the superscript $>$. u and v are column vectors with the same dimension. The ℓ_2 normalization is defined as $\widetilde{\rho}(u) = u/kuk_2$. The instance similarity

is defined as $\rho(u, v) = u^\top v / \|u\|_2 \|v\|_2$, which is the dot product between the ℓ_2 normalized u and v (i.e. cosine similarity). We use $[n]$ to represent the set $\{1, 2, \ldots, n\}$.

3.1 Contrastive Learning

Let $\mathcal{X}$ denote the set of all possible data points. Let $\mathcal{Y}$ denote the label set of the downstream task, which is a discrete and finite set. $f : \mathcal{X} \to \mathbb{R}^d$ denotes the *feature encoder*. To highlight the key ideas, we present the CL framework proposed by Saunshi et al. (2019) in a simplified binary scenario, i.e., $\mathcal{Y} = \{-1, 1\}$. The multi-class setting ($|\mathcal{Y}| > 2$) is discussed in Appendix B.1. CL assumes that we obtain the similar data in the form of pairs (x, x^+) and K independent and identically distributed (i.i.d.) negative samples $x_1^-, x_2^-, \ldots, x_K^-$. Given an unlabeled dataset $U = \{z_i\}_{i=1}^m$, where $z_i = (x_i, x_i^+, x_{i1}^-, \ldots, x_{iK}^-)$, we aim to learn an encoder f that makes $f(x)$ similar to $f(x^+)$, while keeping away from $f(x_1^-), \ldots, f(x_K^-)$ at the same time.

Linear evaluation One standard method for evaluating the performance of the CL model is *linear evaluation* (Chen et al., 2020; Kim et al., 2020), which learns a downstream linear layer after the base encoder, then uses a modified model for class-level classification. The test accuracy on the downstream task is used as a proxy for representation quality. The model with downstream layer is fine-tuned from a labeled dataset $S = \{(x_i, y_i)\}_{i=1}^n$. Both U and S are assumed to be i.i.d. collections.

Data distributions Let $\mathcal{C}$ denote the set of *latent classes* (Saunshi et al., 2019) that are all possible classes for points in $\mathcal{X}$. For each class $c \in \mathcal{C}$, there is a probability $\mathcal{D}_c$ over $\mathcal{X}$ that captures the probability that a point belongs to class c. The distribution on $\mathcal{C}$ is denoted by ϱ. Let c^+, c^- denote the positive and negative latent class drawn from ϱ; thus, $\mathcal{D}_{c^+}$ and $\mathcal{D}_{c^-}$ are the distributions to sample positive and negative samples, respectively. The process for generating an unlabeled sample $z = (x, x^+, \{x_i^-\}_{i=1}^K) \in U$ as follows: 1. Draw two latent classes $(c^+, c^-) \sim \varrho^2$; 2. Draw two positive samples $(x, x^+) \sim \mathcal{D}_{c^+}^2$ and K negative samples $\{x_i^-\}_{i=1}^K \sim \mathcal{D}_{c^-}$ i.i.d.

To set up the labeled dataset S for binary scenario, we build the binomial distribution ϱ_{sup} by fixing two classes c^+, c^-: $\varrho_{sup}(c^+) = \frac{\varrho(c^+)}{\varrho(c^-)+\varrho(c^+)}$, $\varrho_{sup}(c^-) = \frac{\varrho(c^-)}{\varrho(c^-)+\varrho(c^+)}$. We fix $y_{c^+} = +1$ and $y_{c^-} = -1$, then generate a labeled sample $(x, y) \in S$ as follows: 1. Draw a class $c \sim \varrho_{sup}$ and set the label $y = y_c$; 2. Draw a sample $x \sim \mathcal{D}_c$.

Loss function The learning process is divided into two steps: minimizing the contrastive loss on the encoder and fine-tuning on the downstream layer using supervised loss. We focus on *logistic loss*: $\ell(v) = \log_2(1 + \sum_j \exp(-v_j))$ for $v \in \mathbb{R}^K$. Thus, the *contrastive loss* (Chen et al., 2020; He et al., 2020) associated with the encoder f in this framework is defined as follows:

$$\mathcal{L}_{un}(f) = \mathop{\mathbb{E}}_{\substack{c^+, c \\ \sim \varrho^2}} \mathop{\mathbb{E}}_{\substack{x, x^+ \sim \mathcal{D}_{c^+}^2 \\ x_i \sim \mathcal{D}_c}} \ell\left(\left\{f(x)^T \left(f(x^+) - f(x_j^-)\right)\right\}\right). \tag{1}$$

For linear evaluation, the supervised learning algorithm is given the *mapped* dataset $\widehat{S} := \{(f(x_i), y_i)\}_{i=1}^n$ and returns a predictor $g : \mathbb{R}^d \to \mathbb{R}$. The label of $\widehat{x} \in \widehat{S}$ is obtained from $\widehat{y} = \text{sign}(g(\widehat{x}))$, $\widehat{y} \in \{-1, 1\}$. The logistic loss in (2) is $\ell(v) = \log_2(1 + \exp(-v))$ for $v \in \mathbb{R}$.

We aim to minimize the supervised loss as follows:

$$L_{sup}(g \quad f) = \mathop{\mathsf{E}}_{c\sim\varrho_{sup}} \mathop{\mathsf{E}}_{x\sim\mathcal{D}_c} \ell\left(y_c \quad g(f(x))\right). \tag{2}$$

3.2 Supervised Verification

In this section, we set up the verification problem for supervised learning. Existing studies of robustness verification (Wong et al., 2018; Cohen et al., 2019) mainly focus on defending *white-box* attacks, which is the strongest adversaries who have full knowledge of the victim model, including structures and parameters.

The supervised algorithm is given the labeled dataset $S = \mathsf{f}(x_i, y_i)\mathsf{g}_{i=1}^n, (x_i, y_i)$ 2 $X \quad Y$. We consider the classification task with $\mathsf{j}Y\mathsf{j}$ classes. Let $h : X \;!\; Y$ be the classifier, which is modeled by $h(x) \coloneqq \arg\max_{i\in\mathcal{Y}} H^i(x)$ with a differentiable mapping $H : X \;!\; \Delta^{|\mathcal{Y}|-1}$, where $\Delta^{|\mathcal{Y}|-1}$ denotes the probability simplex in $\mathsf{R}^{|\mathcal{Y}|}$. In this paper, H is an NN followed by a softmax layer. The final output after softmax function $H^i(x)$ measures how likely input x belongs to i-th class.

Local adversarial robustness We refer to $x' = x + \delta$ as an *adversarial sample* of x for classifier h if h correctly classifies x but assigns a different label to x'. In the context of *local adversarial robustness* for NNs, we require h not only to correctly classify (x, y), but also to be locally constant around x; i.e., h is certified not to contain any adversarial samples in the ℓ_p ball centered at x. Formally, we define this property as l_p^ϵ-*verified*.

Definition 1 (l_p^ϵ-*verified*)**.** *For the given input (x, y), the model h is l_p^ϵ-verified at (x, y) if it correctly classifies both x and x' as y for any x' 2 $B_p(x, \epsilon)$, i.e., $h(x') = y$. It means there are no adversarial samples around x.*

The l_p^ϵ-*verified* of h at (x, y) depends on the radius of the largest ℓ_p ball centered at x in which h does not change its prediction. This radius is called the *robust radius*, which is formally defined as follows:

$$\mathrm{R}(h; x, y) \coloneqq \inf_{h(x^0)\neq y} \left\|x' \quad x\right\|_p. \tag{3}$$

If $h(x)$ 6= y, then $\mathrm{R}(h; x, y) \coloneqq 0$. It is natural to regard the robust radius as a robustness metric. Unfortunately, computing the *robust radius* (3) is proven to be an NP-complete problem due to the need for complete verification (Katz et al., 2017; Sinha et al., 2018). In cases when h is too complex to control its predictions in practice (e.g. if h is a large NN on high-dimensional data), solving (3) directly will be time-consuming. We can thus derive a tight lower bound $\underline{\mathrm{R}}$ of R given by incomplete verifiers, referred to as the *certified radius*, which satisfies 0 $\underline{\mathrm{R}}(h; x, y)$ $\mathrm{R}(h; x, y)$.

Formally, we define robustness verification (Li et al., 2020).

Definition 2 (Robustness verification)**.** A *is an algorithm of robustness verification. For any (x, y), as long as there exists x' 2 $B_p(x, \epsilon)$ making $h(x')$ 6= y (adversarial sample), $A(h, x, y, \epsilon) =$ `False` (deterministic verification) or $\mathsf{P}(A(h, x, y, \epsilon) =$ `False`$)$ 1 α (probabilistic verification), α is a small threshold. If $A(h, x, y, \epsilon) =$ `True`, h is l_p^ϵ-verified at (x, y) providing by A.*

Deterministic verification We provide notions for ReLU NN. Consider an input vector $x \in \mathsf{R}^{d_0}$ for a neural network with L layers. Let the number of neurons in the k-th layer be d_k, while $\mathbf{W}_k \in \mathsf{R}^{d_k \times d_{k-1}}$ and $b_k \in \mathsf{R}^{d_k}$ $(k \in [L])$ represent the weights and biases of NN. Let $\phi_k : \mathsf{R}^{d_0} \to \mathsf{R}^{d_k}$ be the operator mapping the input layer to layer k. $\sigma(\ )$ is the activation function, while the ReLU activation is $\sigma(\ ) = \max(\ , 0)$. $\sigma_{\mathrm{SM}}(\ )$ denotes the softmax function. For each $k \in [L]$, $\phi_k(x) = \mathbf{W}_k \widehat{\phi}_{k-1}(x) + b_k$, $\widehat{\phi}_k(x) = \sigma(\phi_k(x))$, $\widehat{\phi}_0(x) = x$. The output of the neural network is $\phi_L(x) \in \mathsf{R}^{d_L}$. d_L further denotes the number of input classes $|\mathcal{Y}|$.

Following the common multi-class setting of deterministic verification problem (Zhang et al., 2018; Xu et al., 2020), we consider feed-forward ReLU NN and ℓ_∞-bounded attack. We can view deterministic verification from optimization perspective.

Definition 3 (Multi-class deterministic verification problem)**.** *Given a point $x \in \mathsf{R}^{d_0}$, the true label y, attack target label y', class number $d_L = |\mathcal{Y}|$. We denote the NN output $H : \mathsf{R}^{d_0} \to \mathsf{R}^{d_L}$ with $H \coloneqq \sigma_{\mathrm{SM}}(\phi_L(x))$, where d_0 and d_L are the dimensions of the input and output, respectively. Let $h(x) \coloneqq \arg\max_{i \in \mathcal{Y}} H^i(x)$ be the final predictor. For any fixed ϵ, multi-class deterministic robustness verification problem is defined as follows:*

$$
\begin{aligned}
& \widetilde{h}(x, y, y', \epsilon) \coloneqq \min_{x^0} \ H^y(x') - H^{y^0}(x') \\
s.t. \ & \phi_k(x') = \mathbf{W}_k \widehat{\phi}_{k-1}(x') + b_k, k \in [L], \\
& \widehat{\phi}_k(x') = \sigma(\phi_k(x')), k \in [L-1], \\
& H(x') = \sigma_{\mathrm{SM}}(\phi_L(x')), \\
& x' \in \mathcal{B}_\infty(x, \epsilon).
\end{aligned}
\tag{4}
$$

If $\widetilde{h} \geq 0, \exists x' \in \mathcal{B}_\infty(x, \epsilon)$ fools the model into producing an incorrect label. h is l_∞^ϵ-*verified* if $\widetilde{h}(x, y, \epsilon) > 0$. The complete verifier aims to solve (4) and calculates $\widetilde{h}$ exactly. Recall that complete verification is proven to be an NP-complete problem (Katz et al., 2017). Therefore, many previous works of deterministic verification (Wong and Kolter, 2018; Zhang et al., 2018; Xu et al., 2020) propose incomplete verifiers that relax the non-convexity part of NN to derive a lower bound $\widetilde{h} \geq \underline{h}$. If $\underline{h}(x, y, y', \epsilon) > 0$ is given by the incomplete verifier, model h is also l_∞^ϵ-*verified* at (x, y).

Note that Theorem 3 is defined in terms of **target attack**. This enables us to provide the definition of robust radius under target attack, which means that H cannot be attack to the target label y' within this radius.

$$
\begin{aligned}
\mathrm{R}^{y^0}(h; x, y, y') &\coloneqq \inf_{h(x^0) = y^0} \left\| x' - x \right\|_\infty \\
&= \sup_\epsilon \ \epsilon \ \text{ s.t. } \widetilde{h}(x, y, y', \epsilon) > 0.
\end{aligned}
\tag{5}
$$

The certified radius $\underline{\mathrm{R}}^{y^0}$ is provided by incomplete verifier, which is the lower bound of the robust radius R given by $\underline{h}$.

$$
\underline{\mathrm{R}}^{y^0}(h; x, y, y') \coloneqq \sup_\epsilon \ \epsilon \ \text{ s.t. } \underline{h}(x, y, y', \epsilon) > 0. \tag{6}
$$

For the multi-class scenario, the l_∞^ϵ-*verified* of h at (x, y) depends on the robust radius in which h does not change its predicted label under **untarget attack**. Thus, the robust

radius for multi-class verification depends on the smallest robust radius R^{y^0}. The certified radius $\underline{\mathrm{R}}$ for untarget attack is defined similarly.

$$\mathrm{R}(h;x,y) \coloneqq \min_{y^0 \neq y} \mathrm{R}^{y^0},\ \underline{\mathrm{R}}(h;x,y) \coloneqq \min_{y^0 \neq y} \underline{\mathrm{R}}^{y^0}. \tag{7}$$

Probabilistic verification Besides deterministic verification, one recently emerging branch of studies proposes probabilistic verification based on *Randomized smoothing* (RS) (Cohen et al., 2019). Currently, only these approaches are scalable enough to verify large-scale NNs (e.g., ResNet) and datasets. RS constructs a new classifier $\widehat{h}$ from h that can more easily obtain robustness by "smoothly" transforming the base classifier h with the Gaussian distributions $\mathsf{N}(0, \sigma^2 I)$:

$$\widehat{h}(x) \coloneqq \arg\max_{c \in \mathcal{Y}} \mathsf{P}_{\eta \sim \mathcal{N}(0,\sigma^2 I)} \left(h(x+\eta) = c \right), \tag{8}$$

where σ^2 is a hyperparameter that controls the level of smoothing. For a given (x, y), (Cohen et al., 2019) show that robust radius $\mathrm{R}(\widehat{h}; x, y)$ can be lower-bounded by certified radius $\underline{\mathrm{R}}(\widehat{h}; x, y)$ derived from the *confidence* of $\widehat{h}$ at x, which we denote by $p_h(x)$, as follows:

$$\begin{aligned} &\underline{\mathrm{R}}(\widehat{h}; x, y) \coloneqq \sigma \quad \Phi^{-1}(p_h(x)) \quad \mathrm{R}(\widehat{h}; x, y), \\ &\text{where } p_h(x) \coloneqq \mathsf{P}_{\eta \sim \mathcal{N}(0,\sigma^2 I)}(h(x+\eta) = y), \end{aligned} \tag{9}$$

provided that $\widehat{h}(x) = y$; otherwise, $\mathrm{R}(\widehat{h}; x, y) \coloneqq 0$. Φ^{-1} denotes the inverse cumulative distribution function of the standard Gaussian distribution. This lower bound is known to be tight for the ℓ_2-minimum distance. The bound $\underline{\mathrm{R}}$ is optimal for linear classifiers (Cohen et al., 2019).

4 RVCL: Robustness Verification Framework for Contrastive Learning

In this section, we first introduce the deterministic verification problem on supervised downstream tasks by simply modifying supervised verification (Theorem 3), and further present several weaknesses of adopting (4) in CL. It motivates us to propose a novel RVCL framework to solve these issues.

4.1 Verification Problem for Linear Evaluation

By defining the supervised deterministic verification problem on linear evaluation, we can regard the robust radius R as a proxy robustness metric for CL.

We denote the encoder $f : \mathsf{R}^{d_0} \ ! \ \mathsf{R}^{d_L}$ with $f \coloneqq \phi_L(x)$, where d_0 and d_L are the dimensions of the encoder's input and output. g denotes the downstream linear predictor, which is modeled by $g(x) \coloneqq \arg\max_{i \in \mathcal{Y}} G^i(x)$, $G(x)$ is the downstream output after softmax function. $\mathbf{W}_{\mathrm{LE}}\ 2\ \mathsf{R}^{1 \times d_L}$ and $\mathbf{b}_{\mathrm{LE}}\ 2\ \mathsf{R}$ are the weight and bias of $g(x)$, respectively. The optimization problem for linear evaluation is defined by simply modifying the constraints of (4), as follows:

$$
\begin{aligned}
&\widetilde{g}(x, y, y', \epsilon) \coloneqq \min_{x^0} \; G^y(x') \quad G^{y^0}(x') \\
\text{s.t. } &\phi_k(x') = \mathbf{W}_k \widehat{\phi}_{k-1}(x') + \mathbf{b}_k, k \in [L], \\
&\widehat{\phi}_k(x') = \sigma(\phi_k(x')), k \in [L \quad 1], \\
&G(x') = \sigma_{\mathrm{SM}}\big(\mathbf{W}_{\mathrm{LE}} \phi_L(x') + \mathbf{b}_{\mathrm{LE}}\big), \\
&x' \in \mathbf{B}_\infty(x, \epsilon).
\end{aligned} \tag{10}
$$

The difference between (4) and (10) is that there is no active function $\sigma(\ )$ between the encoder and downstream layer. There is no barrier to applying incomplete verifiers on (10). Thus, the definition of robust radius $\mathrm{R}_{\mathrm{LE}}(g; x, y)$ and certified radius $\underline{\mathrm{R}}_{\mathrm{LE}}(g; x, y)$ on the downstream task are similar to (7).

We can therefore regard $\underline{\mathrm{R}}_{\mathrm{LE}}$ as a proxy robust radius at data point x. However, this approach has serious problems:

1. $\underline{\mathrm{R}}_{\mathrm{LE}}$ cannot be computed directly without a label.

2. Even if we have the label to compute $\underline{\mathrm{R}}_{\mathrm{LE}}$, and use $\underline{\mathrm{R}}_{\mathrm{LE}}$ to evaluate the model robustness, we do not know whether the robustness benefits from the encoder or downstream layer.

Probabilistic verification for linear evaluation also suffers from these problems. It motivates us to propose RVCL, a novel framework for verifying the robustness of encoders without the need for labels and downstream tasks.

4.2 RVCL Framework

Many existing works have studied the supervised verification problem stated in Section 3.2. However, the performing of robustness verification for CL has received less research attention. In this section, we present the formal definition of the robustness verification problem from deterministic and probabilistic perspectives on the encoder f, after which we provide two robustness metrics to study the performance of the CL encoder and incomplete verifier.

Deterministic verification for CL Following Section 3.2, we consider ℓ_∞ norm in this subsection. The core intuition of supervised verification is that the points in the small $\mathbf{B}_\infty(x, \epsilon)$ ball should have the same label as x. Inspired by this idea, we define *the conditions under which the disturbance successfully attacks the encoder*.

Given a positive sample x^+, let the negative sample x^- be the attack target of x^+. We hope that the points $x' \in \mathbf{B}_\infty(x^+, \epsilon)$ will be more similar to x^+ than any other negative samples x^-, while the instance-wise attack algorithm generates an adversarial sample $x' \in \mathbf{B}_\infty(x^+, \epsilon)$ with the attack strength ϵ, in order to fool the model by judging x' as similar to x^-.

If the instance similarity $\rho(f(x^+), f(x')) > \rho(f(x^-), f(x'))$, then x' is similar to x^+ (i.e. $\theta_1 < \theta_2$ in Figure 1), which means that the encoder f is not successfully attacked by x'. We say that the encoder f is l_∞^ϵ-*verified* at (x^+, x^-) if x' is more similar to x^+ than to x^- for

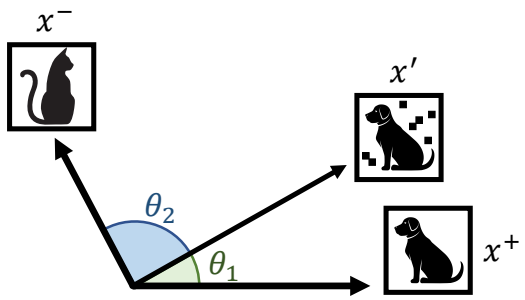


Figure 1: θ is the angle between features given by f. If $\theta_1 < \theta_2$, f is not attacked successfully by the adversarial sample x'.

any $x' \in B_\infty(x^+, \epsilon)$, i.e., there are no adversarial samples similar to x^- around x^+. Note that the comparison of instance similarity has an equivalent form:

$$\rho(f(x^+), f(x')) > \rho(f(x^-), f(x')) \quad (\ ) \tag{11}$$

$$\begin{pmatrix} \widetilde{\rho}(f(x^+)) & \widetilde{\rho}(f(x^-)) \end{pmatrix}^\top f(x') > 0 \tag{12}$$

Judging whether or not (12) is True can be regarded as a part of forward propagation; thus, we can define the optimization problem for CL by adding a linear layer after f with weight $\mathbf{W}_{\mathrm{CL}} = \begin{pmatrix} \widetilde{\rho}(f(x^+)) & \widetilde{\rho}(f(x^-)) \end{pmatrix}^\top$.

Definition 4 (Deterministic verification for CL)**.** *Given two positive and negative samples $x^+, x^- \in \mathbb{R}^{d_0}$, the feature encoder $f : \mathbb{R}^{d_0} \to \mathbb{R}^{d_L}$ with $f := \phi_L(x)$, ℓ_2 normalization $\widetilde{\rho}(u) = u/\|u\|_2$, for any fixed ϵ, the robustness verification problem for CL is defined as follows:*

$$\begin{aligned} & \widetilde{f}(x^+, x^-, \epsilon) := \min_{x^0} \ \mathbf{W}_{\mathrm{CL}} f(x') \\ s.t. \ & \phi_k(x') = \mathbf{W}_k \widehat{\phi}_{k-1}(x') + b_k, k \in [L], \\ & \widehat{\phi}_k(x') = \sigma(\phi_k(x')), k \in [L-1], \\ & \mathbf{W}_{\mathrm{CL}} = \begin{pmatrix} \widetilde{\rho}(f(x^+)) & \widetilde{\rho}(f(x^-)) \end{pmatrix}^\top \in \mathbb{R}^{1 \times d_L}, \\ & f(x') = \phi_L(x'), \ x' \in B_\infty(x^+, \epsilon). \end{aligned} \tag{13}$$

Moreover, the *robust radius* R_{CL} and *certified radius* $\underline{\mathrm{R}}_{\mathrm{CL}}$ for CL are defined as follows:

$$\begin{aligned} \mathrm{R}_{\mathrm{CL}}(f; x^+, x^-) &:= \inf_{\substack{\rho(f(x^0), f(x^+)) \\ < \rho(f(x^0), f(x^-))}} \|x' - x^+\|_\infty \\ &= \sup_\epsilon \ \epsilon \text{ s.t. } \widetilde{f}(x^+, x^-, \epsilon) > 0, \\ \underline{\mathrm{R}}_{\mathrm{CL}}(f; x^+, x^-) &:= \sup_\epsilon \ \epsilon \text{ s.t. } \underline{f}(x^+, x^-, \epsilon) > 0, \end{aligned} \tag{14}$$

where $\underline{f}$ is the verified lower bound of $\widetilde{f}$ given by the verifier; thus, $0 \le \underline{\mathrm{R}}_{\mathrm{CL}}(f; x^+, x^-) \le \mathrm{R}_{\mathrm{CL}}(f; x^+, x^-)$.

Algorithm 1 Deterministic verification for CL

 # judge f is l_∞^ϵ-verified at (x^+, x^-) or not
Input: encoder f, positive sample x^+, negative sample x^-, perturbation bound ϵ
Output: `True`: f is l_∞^ϵ-*verified* at (x^+, x^-); `False`: f is not l_∞^ϵ-*verified* at (x^+, x^-)
Function PREDICT(f, x^+, x^-, ϵ)
  $\underline{f} = \text{INCOMPLETEVERIFIER}(f, x^+, x^-, \epsilon)$
  if $\underline{f} > 0$ **then return** `True`
  else return `False`

 # compute the certified radius of (x^+, x^-) on encoder f
Input: encoder f, positive sample x^+, negative sample x^-, tolerance t, lower bound R_l, upper bound R_u
Output: certified radius $\underline{\mathrm{R}}_{\mathrm{CL}}$
Initialization: $t = 10^{-6}$, $\mathrm{R}_l = 0$, $\mathrm{R}_u = 1$
Function CERTIFY(f, x^+, x^-, t, R_l, R_u)
  while $j\mathrm{R}_u \quad \mathrm{R}_l j > t$ **do**
    $\epsilon = {}^{(\mathrm{R}_l + \mathrm{R}_u)}/_2$
    # f is l_∞^ϵ-verified at (x^+, x^-), the answer should be larger than current ϵ
    if PREDICT(f, x^+, x^-, ϵ) **then** $\mathrm{R}_l = \epsilon$
    else $\mathrm{R}_u = \epsilon$
  end while
  return ϵ

For verified prediction, $\underline{f}(x^+, x^-, \epsilon) > 0$ for a given strength ϵ implies that f is l_∞^ϵ-*verified* at (x^+, x^-). The procedure of verified prediction is presented in pseudocode as PREDICT. We utilize PREDICT to obtain the certified instance accuracy ($\underline{A}_{\mathrm{CL}}^\epsilon$ in Theorem 11) for the test dataset $U_{\mathtt{test}}$, since $\underline{A}_{\mathrm{CL}}^\epsilon$ is the fraction of the test dataset for which f is l_∞^ϵ-*verified* at (x^+, x^-), i.e., $\text{PREDICT}(f, x^+, x^-, \epsilon) = \mathtt{True}$.

Besides prediction, we are also interested in the certified radius $\underline{\mathrm{R}}_{\mathrm{CL}}$ for a given (x^+, x^-). Apparently, $\underline{f}(x^+, x^-, \epsilon)$ is non-increasing with ϵ because of the `inf` operator. Thus, we can apply *binary search* to obtain $\underline{\mathrm{R}}_{\mathrm{CL}}$. The procedure is presented as CERTIFY. More precisely, we determine whether f is l_∞^ϵ-*verified* at (x^+, x^-) with current ϵ. If yes, it means that (x^+, x^-) is l_∞^ϵ-*verified* with an ϵ larger than the current one, then we increase ϵ; otherwise, we decrease ϵ. The final solution of ϵ is the certified radius $\underline{\mathrm{R}}_{\mathrm{CL}}$.

Note that (13) and (14) are both defined directly on the CL encoder without reference to any labels or downstream tasks, which resolves the issue articulated in Section 4.1. The pseudocode is presented in Algorithm 1.

Probabilistic verification for CL For probabilistic verification, motivated by (9), we define the probability of given $x' = x^+ + \delta$ being the positive sample of x^+. Since judging $x^+ + \delta$ is positive or negtive sample is a binary classification problem, we utilize logistic function $\ell(v) = (1 + \exp(\quad v))^{-1}$ (Section 3.1), which is widely used in binary logistic regression. Following common RS setting (Cohen et al., 2019), we discuss ℓ_2 norm in this subsection.

Algorithm 2 Probabilistic verification for CL

```
# judge f is l_2^ε-verified at (x^+, x^-) or not
Input: encoder f, positive sample x^+, negative sample x^-, perturbation bound ε, noise level σ, number of Gaussian samples N_G
Output: True: f is l_2^ε-verified at (x^+, x^-); False: f is not l_2^ε-verified at (x^+, x^-)
Function PREDICT(f, x^+, x^-, ε, σ, N_G)
    R_CL = CERTIFY(f, x^+, x^-, σ, N_G)
    if R_CL > ε then return True
    else return False

# compute the certified radius of (x^+, x^-) on encoder f
Input: encoder f, positive sample x^+, negative sample x^-, noise level σ, number of Gaussian samples N_G
Output: certified radius R_CL
Function CERTIFY(f, x^+, x^-, σ, N_G)
    Sample N_G i.i.d. Gaussian samples x_1 , x_k  N(x^+, σ^2 I)
    Compute empirical expectation of (15): p̂_f^+(x; x^+, x^-) = Σ_{i=1}^k p_f^+(x_i; x^+, x^-)/N_G
    return R_CL = σ  Φ^{-1}(p̂_f^+(x; x^+, x^-))
```

Definition 5 (Positive sample probability). *Given two positive and negative samples $x^+, x^- \in \mathcal{X}$, the feature encoder $f : \mathcal{X} \to \mathbb{R}^d$, the instance similarity $\rho(u, v) = u^\top v/\|u\|_2\|v\|_2$, temperature τ, the probability of the given input x' being the positive sample of x^+ is defined as follows:*

$$\begin{aligned} p_f^+(x'; x^+, x^-) :=& \frac{\exp(\rho(f(x^+), f(x'))/\tau)}{\exp(\rho(f(x^+), f(x'))/\tau) + \exp(\rho(f(x^-), f(x'))/\tau)} \\ =& \frac{1}{1 + \exp((\rho(f(x^-), f(x')) - \rho(f(x^+), f(x')))/\tau)}. \end{aligned} \tag{15}$$

Remark 6. *We use the logistic function to translate the numerical relationship of instance similarity into the probability of being a positive sample. It is motivated by the softmax function of binary classification, which converts NN output into the probability simplex. We follow (Chen et al., 2020; Zhai et al., 2020) to define (15) with temperature τ; τ is used to control the softness of the probability distribution. When τ tends to 0, $p_f^+(x'; x^+, x^-)$ goes to $\mathbb{1}_{[\rho(f(x^0), f(x^+)) - \rho(f(x^0), f(x^-)) > 0]}$.*

Recall that when $\rho(f(x^+), f(x')) > \rho(f(x^-), f(x'))$ for any $x' \in \mathcal{B}_p(x^+, \epsilon)$, the encoder f is l_p^ϵ-*verified* at (x^+, x^-) determinately. However, verifying every point in $\mathcal{B}_p(x^+, \epsilon)$ is difficult and time-consuming for large and complicated NN. For probability verification, instead of directly verifying f, we turn to verify amounts of points in $\mathcal{B}_p(x^+, \epsilon)$. If the majority of points are the positive sample of x^+, we say that f is l_p^ϵ-*verified* with high probability. We define this intuition formally as follows.

Theorem 7 (Probabilistic verification for CL). *Given two positive and negative samples $x^+, x^- \in \mathcal{X}$, the feature encoder $f : \mathcal{X} \to \mathbb{R}^d$, let $\eta \sim \mathcal{N}(0, \sigma^2 I)$, if*

$$\mathbb{E}_\eta(p_f^+(x^+ + \eta; x^+, x^-)) > 0.5, \tag{16}$$

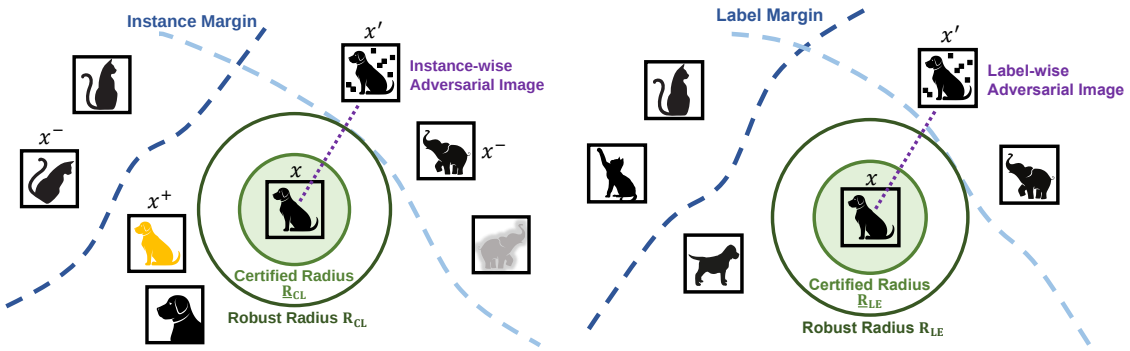
then f *is* $l_2^{\underline{\mathrm{R}}_{\mathrm{CL}}}$*-verified at* (x^+, x^-)*. The certified radius* $\underline{\mathrm{R}}_{\mathrm{CL}}$ *is given by*

$$\underline{\mathrm{R}}_{\mathrm{CL}}(f; x^+, x^-) = \sigma \quad \Phi^{-1}(\mathsf{E}_\eta(p_f^+(x^+ + \eta; x^+, x^-))), \tag{17}$$

where Φ^{-1} *is the inverse cumulative distribution function of standard Gaussian distribution.*

See proof in Appendix A.1. In Theorem 7, instead of directly verifying f, we define a "smoothed" f whose positive sample probability (15) is the majority vote of f applied to x^+ convolved with some noise distribution. The Gaussian distribution is applied here to provide ℓ_2 verification. We are able to guarantee that the encoder output $f(x^+ + \eta)$ will not be recognized as the negative sample within a certain radius $\underline{\mathrm{R}}_{\mathrm{CL}}$, where the magnitude of this radius is a function of the probability by which the majority vote wins: the more points vote for l_p^ϵ*-verified*, the larger the certified radius.

The expectation in (17) cannot be exactly solved. Thus, Monte Carlo sampling (Metropolis and Ulam, 1949) is used to estimate the expectation. As a result, the verification is probabilistic rather than deterministic. The pseudocodes Predict and Certify are presented in Algorithm 2. Note that $\underline{\mathrm{R}}_{\mathrm{CL}}$ 0 implies $\widehat{p}_f^+$ 0.5, which means Certify fails to verify f at (x^+, x^-).



(a) RVCL (b) Multi-class supervised verification

Figure 2: Illustration for RVCL and multi-class supervised verification. $\|x' - x\|_p$ must be larger than the robust radius if x' is a adversarial sample. It is verified that no adversarial sample exists in $B_p(x, \underline{\mathrm{R}})$, $\underline{\mathrm{R}}$ is provided by the incomplete verifier, which is the lower bound of the robust radius. (a) The latent class of instance-wise adversarial sample x' is "dog", while the feature of x' is similar to that of x^-, which is an "elephant". (b) The true label of label-wise adversarial sample x' is "dog", while the prediction of x' is attacked to "elephant".

Remark 8. *We illustrate RVCL in Figure 2(a). Note that the differences between Figure 2(a) (RVCL) and Figure 2(b) (supervised verification) are the margin and the type of data points. The CL framework (we adopt SimCLR (Chen et al., 2020) in our paper) aims to maximize the agreement between different augmentations of the same image (a.k.a. positive samples), while minimizing the agreement between negative samples. The data points in the instance margin are augmentations (rotation, crop, resize, etc.) of the specific image Figure 2(a), while the data points within the label margin are the images with the same label (Figure 2(b)).*

Robustness metrics for CL This subsection provides two robustness metrics used to study the robustness of the CL encoder and the performance of incomplete verifiers.

Average certified radius (ACR): For supervised verification, ACR (Zhai et al., 2020) is an important metric used in evaluating robustness. Specifically, it is the average of the certified radius on the test dataset. We can define it directly on the supervised downstream task: $\mathrm{ACR}_{\mathrm{LE}} \coloneqq \frac{1}{|S_{\mathtt{test}}|} \sum_{(x,y)\in S_{\mathtt{test}}} \underline{\mathrm{R}}_{\mathrm{LE}}(g; x, y)$, where $S_{\mathtt{test}}$ is a labeled test dataset satisfying $g(x) = y$ for all (x, y) 2 $S_{\mathtt{test}}$. However, $\mathrm{ACR}_{\mathrm{LE}}$ still suffers from the problems discussed in Section 4.1. We therefore define ACR for CL based on $\underline{\mathrm{R}}_{\mathrm{CL}}$, which directly reflects the robustness of CL encoder without the label:

Definition 9 (Average certified radius for CL)**.** *Given an unlabeled test dataset* $U_{\mathtt{test}}$ *generated following Section 3.1,* $z = (x^+, \mathsf{f}x_i^-\mathsf{g}_{i=1}^K)$ 2 $U_{\mathtt{test}}$*, K is the number of negative samples,* $\underline{\mathrm{R}}_{\mathrm{CL}}$ *is defined in (14). The average certified radius for CL is defined as follows:*

$$\mathrm{ACR}_{\mathrm{CL}} \coloneqq \frac{1}{K\mathsf{j}U_{\mathtt{test}}\mathsf{j}} \sum_{z\in U_{\mathtt{test}}} \sum_{i=1}^{K} \underline{\mathrm{R}}_{\mathrm{CL}}(f; x^+, x_i^-). \tag{18}$$

Certified instance accuracy: For supervised deterministic verification, certified accuracy is a metric used to evaluate the performance of incomplete verifiers. Wang et al. (2021) state that the verifier will be stronger if the certified accuracy is the tighter lower bound of supervised robust accuracy. Since there is no definition of "robust accuracy" provided for the CL encoder, we propose a novel robust accuracy without label and downstream task for CL — called *robust instance accuracy* — based on (11):

Definition 10 (Robust instance accuracy)**.** *Given an unlabeled test dataset* $U_{\mathtt{test}}$*,* $z = (x^+, x^-)$ 2 $U_{\mathtt{test}}$*, we use* instance-wise PGD attack *(Kim et al., 2020) to generate the adversarial point* x' 2 $\mathsf{B}(x^+, \epsilon)$ *by maximizing the contrastive loss (1). The robust instance accuracy with strength* ϵ *is defined as follows:*

$$\mathsf{A}_{\mathrm{CL}}^{\epsilon} = \frac{1}{\mathsf{j}U_{\mathtt{test}}\mathsf{j}} \sum_{z\in U_{\mathtt{test}}} \mathbb{1}_{[\rho(f(x^0),f(x^+))-\rho(f(x^0),f(x\ \ ))>0]}. \tag{19}$$

We then define the certified instance accuracy with strength ϵ, which is the fraction of the test dataset for which f is l_p^{ϵ}*-verified* at (x^+, x^-), i.e., $\underline{f} > 0$.

Definition 11 (Certified instance accuracy)**.** *Given an unlabeled test dataset* $U_{\mathtt{test}}$*,* $z = (x^+, x^-)$ 2 $U_{\mathtt{test}}$*. The certified instance accuracy with strength* ϵ *is defined as*

$$\underline{\mathsf{A}}_{\mathrm{CL}}^{\epsilon} = \frac{1}{\mathsf{j}U_{\mathtt{test}}\mathsf{j}} \sum_{z\in U_{\mathtt{test}}} \mathbb{1}_{[\underline{f}(x^+,x\ \ ,\epsilon)>0]}. \tag{20}$$

f being l_p^{ϵ}*-verified* at (x^+, x^-) is the sufficient but not necessary condition of correctly classifying x' generated by a specific attack algorithm with attack strength ϵ. This means that the hold of the judgement condition in (20) implies the hold of that in (19), but not vice versa. Thus, (20) is the lower bound of (19).

Remark 12. *The certified instance accuracy* $\underline{A}^{\epsilon}_{\mathrm{CL}}$ *is used to compare the deterministic verifiers on the CL encoder without labels. The smaller gap between robust instance accuracy* $A^{\epsilon}_{\mathrm{CL}}$ *and* $\underline{A}^{\epsilon}_{\mathrm{CL}}$ *implies a stronger incomplete verifier (see experiments in Section 6.3). However, as* $\underline{A}^{\epsilon}_{\mathrm{CL}}$ *is a function of the fixed attack strength* ϵ*, it is difficult to compare the robustness of two models unless one is uniformly better than the other for all strength* ϵ*. What's more,* $\mathrm{ACR}_{\mathrm{CL}}$ *can be applied to both probabilistic and deterministic verification. Thus,* $\mathrm{ACR}_{\mathrm{CL}}$ *is a more suitable choice than* $\underline{A}^{\epsilon}_{\mathrm{CL}}$ *for evaluating the robustness of CL encoders.*

5 Analysis of Robust Radius

What is the relationship between the robust radius R_{CL} and R_{LE}? This section demonstrates that R_{CL} is the upper bound of R_{LE}, which is further verified by the experimental results in Section 6.1.

To provide the main insights, we first consider the situation in which only one positive sample is used. Formally, given an unlabeled sample $z = (x^+, \mathsf{f}x_i^-\mathsf{g}_{i=1}^K)$, we introduce the *margin distance* of x^+ as half of the minimum distance between $f(x^+)$ and $f(x_i^-)$, defined as $M \coloneqq \min_{i\in[K]} D_i$, where $D_i \coloneqq (1 \quad \rho(f(x^+), f(x_i^-))/2$.

The idea is to estimate the lower tail of the distribution of M by fitting the λ smallest D_i of the negative samples x_i^-. We can then use this estimated distribution to produce the probability of a new point x falling into the margin of x^+, which can be interpreted as the probability of x being a positive sample of x^+. x is classified as a positive sample of x^+ if it is inside the margin of x^+ with high probability.

To estimate the distribution of the margin distance, we turn to the Fisher-Tippett-Gnedenko Theorem in extreme value theory (see the complete statement in Theorem 21).

Lemma 13 (Fisher-Tippett-Gnedenko theorem (Coles et al., 2001))**.** *Let* $X_1, X_2, \ldots$ *be a sequence of independent random variables with common distribution function* F*. Let* $M_n = \sup(X_1, \ldots, X_n)$*. If there exists a sequence* $a_n > 0, b_n$ *2* R *such that*

$$\lim_{n\to\infty} \mathsf{P}(\frac{M_n \quad b_n}{a_n} \quad z) = G(z),$$

where G *is a non-degenerate distribution function, then* G *belongs to either the Gumbel family, the Fréchet family or the Reverse Weibull family.*

The theorem states that the maximum of a sequence of i.i.d. random variables after proper normalization can only converge to one of three possible distributions.

Theorem 14 (Margin distribution)**.** *Assume a continuous non-degenerate margin distribution exists. The distribution for margin distance* M *is then given by the Reverse Weibull distribution. The probability of* x *being a positive sample of* x^+ *is given by the following:*

$$\Psi(x; x^+, \alpha, \sigma) = \exp\left\{ \quad \left(\frac{1 \quad \rho(f(x), f(x^+))}{\sigma}\right)^{\alpha}\right\},$$

where $\rho(f(x), f(x^+))$ *is the instance similarity between* x *and* x^+*.* $\alpha, \sigma > 0$ *are Weibull shape and scale parameters, obtained from fitting to the* λ *smallest margin distances* D_i*.*

See proof in Appendix A.3. Theorem 14 demonstrates that the probability of x being a positive sample can be given by the Reverse Weibull distribution fitting on finite samples. This enables us to compare the robust radius of the encoder and that of the downstream task. Intuitively, if the classifier can predict correctly with higher confidence, this implies that the classifier provides better verified robustness.

Theorem 15 (Robust radius bound). *Given an encoder $f : \mathcal{X} \to \mathbb{R}^d$ and an unlabeled sample $z = (x^+, \{x_i^-\}_{i=1}^K)$, the downstream predictor $g : \mathbb{R}^d \to \mathbb{R}$ is trained on $\widehat{S} = \{(f(x^+), y_{c^+}), (f(x_i^-), y_{c^-})\}_{i=1}^K\}$. Then, for different negative samples x_i^-, we have*

$$\mathrm{R}_{\mathrm{CL}}(f; x^+, x_i^-) \quad \mathrm{R}_{\mathrm{LE}}(g; x^+, y_{c^+}).$$

Proof Sketch The probability of downstream layer predicting x as positive can be given by Ψ_{LE}, which is fitted from margin distances $\{D_i\}_{i=1}^K$ computed by $\widehat{S}$. Ψ_{CL} is fitted from specific D_i, since $\mathrm{R}_{\mathrm{CL}}(f; x^+, x_i^-)$ is the robust radius between specific pair of positive and negative sample.

Figure 3 plots the cumulative distribution function (CDF) of Ψ_{CL} and Ψ_{LE}. $\Psi \to 1$ when $x \to x^+$, which means x is very likely to be the positive sample of x^+. If there exists negative samples between x_i^- and x^+ (" " in Figure 3), then Ψ_{LE} will fit to these negative samples and make CDF grows slower than Ψ_{CL}, i.e., $\Psi_{\mathrm{CL}}(x) \quad \Psi_{\mathrm{LE}}(x)$. Theorem 24 offers the correspondence that a higher probability to be a positive sample implies a larger robust radius, which recovers the theorem statement. See complete proof in Appendix A.4. ■

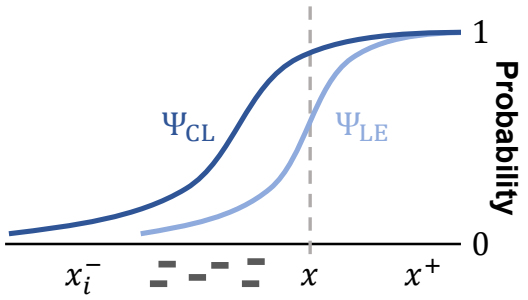


Figure 3: Probability of x as a positive sample. " ": the negative sample other than x_i^-.

Remark 16. *Theorem 15 implies that improving the robustness of the CL encoder enlarges the robust radius R_{CL}, which can benefit the robustness of the downstream layer by providing a large upper bound of R_{LE}. A larger R_{CL} implies that the model will achieve a higher robust performance on the downstream task; thus, it is reasonable to regard R_{CL} as a robustness metric.*

Intuitively, an incorrect prediction will be more easily produced when there are many classes to attack, which implies a lower robust radius than that of binary classification. Theorem 17 demonstrates that the multi-class classifier is more fragile than a binary classifier under perturbation. We present the multi-class experimental results in Section 6.1.

Theorem 17 (Multi-class case)**.** *g_M is trained on the labeled dataset with $p+1$ classes, g_j is the binary classifier of the specific latent class pair $(c^+, c_j^-), j \in [p]$. Then, for different negative samples x_{ij}^- and latent class pairs (c^+, c_j^-), we have*

$$\mathrm{R_{CL}}(f; x^+, x_{ij}^-) \quad \mathrm{R_{LE}}(g_j; x^+, y_{c^+}) \quad \mathrm{R_{LE}}(g_M; x^+, y_{c^+}).$$

We prove the theorem by constructing multiple margin distributions Ψ^j for every class. See proof in Appendix B.2.

In the above, we discuss the decision margin by analyzing the case with single x^+ and multiple x^-. Here, we discuss the robust radius of different x^+ by making the following theroem:

Theorem 18. *Given an encoder $f : \mathcal{X} \to \mathbb{R}^d$, two positive samples x_1^+, x_2^+ and one negative sample x^-, if $\rho(f(x_1^+), f(x^-)) \geq \rho(f(x_2^+), f(x^-))$, then*

$$\mathrm{R_{CL}}(f; x_1^+, x^-) \geq \mathrm{R_{CL}}(f; x_2^+, x^-).$$

See proof in Appendix A.5. Theorem 18 proves that if $f(x^+)$ is similar to $f(x^-)$, it is easy to recognize the adversarial sample x' of x^+ as a negative sample. In Section 6.2, we empirically show that a vague image for which it is difficult to identify the class characteristic is easy to attack.

6 Experiments

In this section, we verify the effectiveness of our proposed RVCL by means of numerical experiments. More specifically, Section 6.1 demonstrates the effectiveness of the average certified radius for CL. Section 6.2 shows that $\mathrm{ACR_{CL}}$ can evaluate the anti-disturbance ability of individual images. Section 6.3 compares the strength of deterministic verifiers to illustrate the effectiveness of RVCL. Section 6.4 provides the efficiency analysis for RVCL. Section 6.5 provides the sensitivity analysis for parameters in RVCL.

Set-up. SimCLR (Chen et al., 2020) and RoCL (Kim et al., 2020) are used in this paper for CL training and contrastive AT, respectively. Contrastive AT is trained with instance-wise adversarial samples with different attack strengths ϵ_{train}; $\epsilon_{train} = 0$ indicates that the encoder is trained with benign images.

For **deterministic verification**, the experiments utilize four architectures: **Base**, **Deep** from Wang et al. (2021), **CNN-A**, **CNN-B** from Dathathri et al. (2020), from which the last layer is removed to form the CL encoders. Using the same dataset as in previous deterministic verification works, all CL encoders are trained on Mnist (LeCun and Cortes, 2010) and CIFAR-10 (Krizhevsky and Hinton, 2009). Further details of the models and experimental settings are presented in Appendix E.

We utilize two incomplete verifiers with different verified tightness in the RVCL framework: CROWN (Zhang et al., 2018) and CBC (β-CROWN (Wang et al., 2021) for CL). A more detailed introduction to incomplete verifiers is presented in Appendix C.

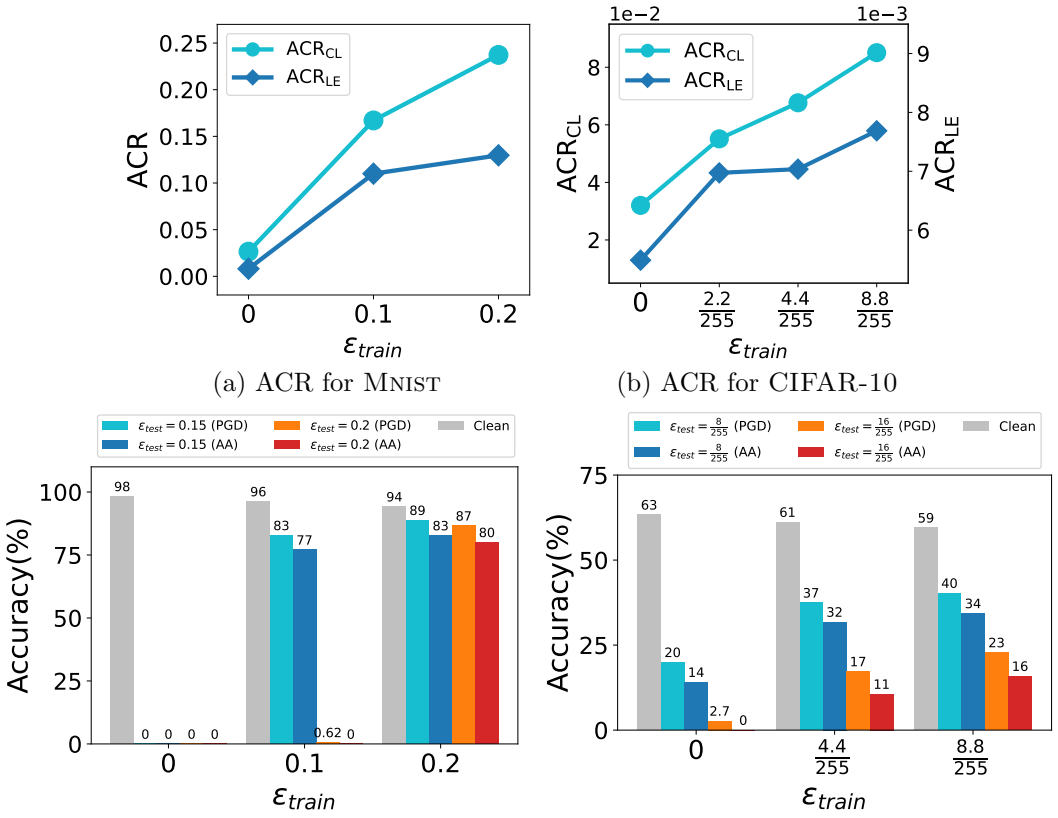
For **probabilistic verification**, we use ResNet-18 and ResNet-50 (He et al., 2016) as the base encoder NN, which are trained on CIFAR-10 and CIFAR-100 (Krizhevsky and Hinton, 2009). The training and evaluation details of contrastive adversarial training are consistent with Kim et al. (2020).

6.1 Average certified radius

In this section, we study the robustness of CL encoders with different structures via average certified radius (ACR in Section 4.2). The experimental results show that ACR is an effective robustness metric for deterministic and probabilistic verification, and reveals the robustness properties of both two verifications.

For deterministic verification (Figure 4), we compute ACR over 100 test samples (i.e., $jS_{\texttt{test}}j = jU_{\texttt{test}}j = 100$) following previous verification works (Zhang et al., 2018; Weng et al., 2018a; Wang et al., 2021; Shi et al., 2022). For ACR_{CL}, we set the number of negative samples $K = 10$. In the interests of efficiency, we use CROWN to compute the ACR_{CL}, which is discussed further in Section 6.4.

For probabilistic verification (Table 1), we use 100 and 500 test samples for CIFAR-10 and CIFAR-100, respectively. We set $K = 10$, noise level $\sigma = 0.1$, number of Gaussian samples $N_G = 256$, and temperature parameter $\tau = 0.1$.



(a) ACR for Mnist (b) ACR for CIFAR-10

(c) Mnist Robust Test (d) CIFAR-10 Robust Test

Figure 4: Results for deterministic verification. (a,b) ACR_{CL} and ACR_{LE} on Mnist and CIFAR-10 with different values of ϵ_{train}. (c,d) Supervised robust accuracy on the downstream classifier under PGD attack and AutoAttack(AA). On Mnist, attack strengths ϵ_{test} are **0.15**, **0.2**; on CIFAR-10, attack strengths ϵ_{test} are $\mathbf{8/255}$, $\mathbf{16/255}$. **Clean** represents testing with benign images. (a,c) run on CNN-A, (b,d) run on CNN-B.

Table 1: ACR and Supervised robust test (under PGD attack and AutoAttack(AA)) with different values of ϵ_{train} for probabilistic verification. The attack strength ϵ_{test} of robust accuracy is $^{16}/_{255}$.

Dataset	Model	*train*	$\mathbf{ACR_{CL}}$ (10^{-2})	$\mathbf{ACR_{LE}}$ (10^{-3})	Clean Acc (%)	PGD Acc (%)	AA Acc (%)
CIFAR-10	ResNet-18	0	0.832	1.24	84.8	0.15	0
		$^{2.2}/_{255}$	2.57	4.87	80.4	11.2	7.37
		$^{4.4}/_{255}$	4.67	6.54	77.3	18.3	15.0
		$^{8.8}/_{255}$	5.71	9.39	75.5	22.8	19.7
	ResNet-50	0	1.58	2.28	88.2	0.25	0
		$^{2.2}/_{255}$	2.43	6.94	83.5	15.9	10.7
		$^{4.4}/_{255}$	3.30	8.19	81.1	21.6	16.2
		$^{8.8}/_{255}$	4.06	9.81	79.2	24.5	19.3
CIFAR-100	ResNet-18	0	1.60	0.085	56.4	0	0
		$^{2.2}/_{255}$	3.56	0.102	53.2	4.58	2.81
		$^{4.4}/_{255}$	4.69	0.249	50.8	8.96	6.32
		$^{8.8}/_{255}$	5.86	0.426	47.7	12.6	9.72

To determine whether the model robustness benefits from the encoder or linear classifier, we fix the encoder and fine-tune the downstream layer with benign images without perturbations. For the empirical robust test, we utilize supervised robust accuracy on the whole test dataset; this is the downstream accuracy over adversarial samples via label-wise PGD attack (Madry et al., 2018) and AutoAttack (Croce and Hein, 2020b) with attack strength ϵ_{test}. Autoattack is an ensemble of four strong diverse attacks: APGD-CE, APGD-DLR, FAB (Croce and Hein, 2020a), and Square Attack (Andriushchenko et al., 2020), which has been proven to be reliable in evaluating the robustness (Croce et al., 2021).

Note that a larger value of ϵ_{train} implies that a more robust encoder is obtained by contrastive AT. From the experimental results of deterministic (Figure 4) and probabilistic verification (Table 1), we make following observations:

1. The results in Figure 4(a,b) show that $\mathrm{ACR_{CL}}$ and $\mathrm{ACR_{LE}}$ increase with increasing ϵ_{train} on the two datasets.

2. For deterministic verification (Figure 4), AutoAttack brings a 5-7% drop in robust accuracy compared to PGD attack, while probabilistic verification experiences a 2-5% drop (Table 1). This drop is due to AutoAttack being a stronger attacker than PGD.

3. Figure 4(c,d) show that with different values of ϵ_{test}, supervised robust accuracies under PGD attack and AutoAttack increase as ϵ_{train} increases. Table 1 shows the same tendency.

From these observations, we can conclude that:

Table 2: Sample Pearson correlation coefficient (PCC) between ACR and supervised robust accuracy (under PGD attack and AutoAttack(AA)). The attack strength ϵ_{test} of robust accuracy is **0.15** for Mnist, and **16/255** for CIFAR-10 and CIFAR-100.

Verification	Dataset	Model	PGD		AA	
			r_{CL}	r_{LE}	r_{CL}	r_{LE}
Deterministic	Mnist	CNN-A	0.963	**0.996**	0.961	**0.994**
	CIFAR-10	CNN-B	**0.998**	0.966	**0.996**	0.939
Probabilistic	CIFAR-10	ResNet-18	**0.990**	0.987	**0.997**	0.986
		ResNet-50	0.943	**0.995**	0.970	**0.993**
	CIFAR-100	ResNet-18	**0.994**	0.938	**0.982**	0.966

1. It is effective to measure the robustness using $\mathrm{ACR}_{\mathrm{CL}}$ without labels and downstream tasks, because both the $\mathrm{ACR}_{\mathrm{CL}}$ and supervised metric ($\mathrm{ACR}_{\mathrm{LE}}$ and robust accuracy) grow consistently with increasing ϵ_{train}.

2. Figure 4(a,b), "$\mathrm{ACR}_{\mathrm{CL}}$", "$\mathrm{ACR}_{\mathrm{LE}}$" columns in Table 1 show that $\mathrm{ACR}_{\mathrm{CL}}$ is larger than $\mathrm{ACR}_{\mathrm{LE}}$ with the same ϵ_{train}, which supports our theory: Theorem 17 demonstrates that R_{CL} is the upper bound of R_{LE}, while $\mathrm{ACR}_{\mathrm{CL}}$ and $\mathrm{ACR}_{\mathrm{LE}}$ are related to R_{CL} and R_{LE} respectively.

3. A robust encoder can significantly improve the model's robust performance on downstream tasks, since $\mathrm{ACR}_{\mathrm{LE}}$ grows with increasing ϵ_{train} even though the downstream layer is learned on benign images.

Pearson correlation coefficient To further assess the efficacy of ACR as a robustness metric, we computed the sample Pearson correlation coefficient (PCC) between ACR and supervised robust accuracy in Table 2. PCC measures the linear correlation between two sets of data. r_{CL} and r_{LE} denote the sample PCC between $\mathrm{ACR}_{\mathrm{CL}}$/$\mathrm{ACR}_{\mathrm{LE}}$ and supervised robust accuracy, respectively. The results in Table 2 are calculated using the data in Figure 4 and Table 1.

As shown in Table 2, all r_{CL} and r_{LE} values exceed 0.9, indicating a strong positive relationship between ACR and supervised robust accuracy. In Line 2, 3, 5 of Table 2, the unsupervised metric $\mathrm{ACR}_{\mathrm{CL}}$ exhibits an even stronger positive relationship than the supervised metric $\mathrm{ACR}_{\mathrm{LE}}$. These findings confirm that $\mathrm{ACR}_{\mathrm{CL}}$ effectively evaluates the robustness of the CL encoder.

Adversarially trained downstream layer $\epsilon_{train_{LE}}$ denotes adversarial training strength for the downstream layer. In prior work (Kim et al., 2020; Ho and Vasconcelos, 2020; Jiang et al., 2020; Fan et al., 2021), the downstream layer was trained on clean samples ($\epsilon_{train_{LE}} = 0$) for linear evaluation. We further conduct experiments with adversarially trained downstream layers on CL encoders, comparing their properties (ACR, clean and robust accuracy).

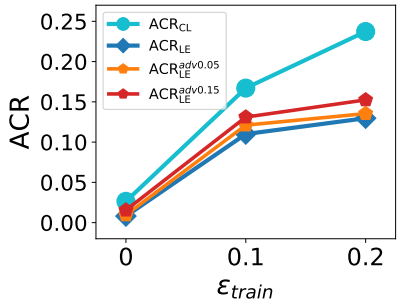


Figure 5: ACR_{CL} and ACR_{LE} with respect to various strengths $\epsilon_{train_{LE}}$ for downstream layer.

Table 3: Robust accuracy on adversarially trained downstream layer with strength $\epsilon_{train_{LE}}$. The encoder is trained with clean samples ($\epsilon_{train} = 0$) or adversarial samples with attack strength ϵ_{train}.

	$train_{LE} = 0.05$			$train_{LE} = 0.15$		
test	Clean	0.15	0.2	Clean	0.15	0.2
$\epsilon_{train} = 0$	96.9%	0%	0%	94.7%	87.2%	1.95%
$\epsilon_{train} = 0.1$	95.5%	86.2%	1.65%	94.3%	90.8%	81.9%
$\epsilon_{train} = 0.2$	94.1%	90.0%	88.3%	93.6%	91.3%	89.6%

Figure 5 displays ACR_{CL} and ACR_{LE} for various $\epsilon_{train_{LE}}$ values. Table 3 presents supervised clean and robust accuracy on downstream layers. The attack strength ϵ_{test} of robust accuracy is 0.15 or 0.2. $\epsilon_{train_{LE}}$ is set to 0.05 or 0.15. Experiments run on Mnist. From experimental results, we conclude that:

1. Adversarial training for the downstream layer enhances model robustness. Figure 5 and Table 3 illustrate that as ϵ_{train} for the encoder and $\epsilon_{train_{LE}}$ for the downstream task increase, both ACR and supervised robust accuracy improve.

2. Figure 5 demonstrates that ACR_{LE} consistently grows with increasing $\epsilon_{train_{LE}}$, but a gap remains with ACR_{CL}. This affirms that R_{CL} R_{LE} (Theorem 15) still holds when the downstream task is adversarially trained.

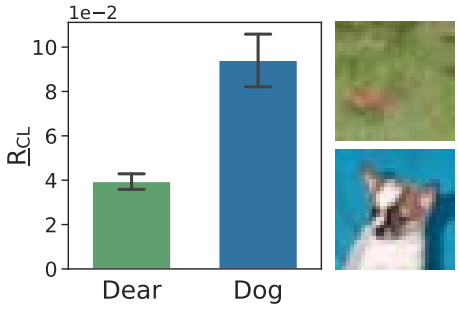
6.2 Anti-disturbance ability of images

To study the robustness property of each image, in this subsection, we compute ACR_{CL} for a single test sample; i.e., $jU_{\texttt{test}}j = 1$, then $ACR_{CL} := \frac{1}{K}\sum_{i=1}^{K} \underline{R}_{CL}(f; x^+, x_i^-)$.

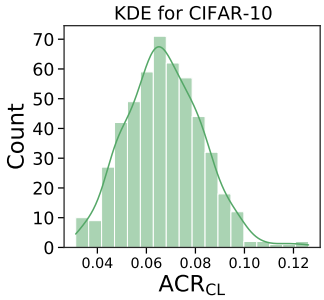
We sample two images from CIFAR-10, as shown in Figure 6(b). The above image is labeled as *deer*, which is vague and makes it difficult to identify the latent class. The below image is labeled as *dog*, which is much clearer than *deer*. We calculate $\underline{R}_{CL}$ with fifty negative samples ($K = 50$). Our findings suggest that the $\underline{R}_{CL}$ of *deer* is significantly smaller than that of *dog*; this means that the distance between the feature of *deer* and its negative samples is smaller than that between the feature of *dog* and its negative samples, which supports our Theorem 18. We can therefore conclude that the anti-disturbance ability of *dog* is stronger than that of *deer*, which means that *dog* is l_∞^ϵ-*verified* with a larger ϵ than *deer*. We sample 10 more images from CIFAR-10, and plot the images and their ACR_{CL} in Figure 6(a). It comes to the same conclusion: the vague image which is difficult to identify the latent class has a low ACR_{CL}. These results verify that ACR_{CL} is able to quantify the anti-disturbance ability of images.

We further visualize the distribution of ACR_{CL} for 500 images from CIFAR-10 and CIFAR-100 by calculating ACR_{CL} with $K = 20$, and additionally provide the kernel density plot to show the distribution (see Figure 6(c,d)). Figure 6(c) is given by deterministic

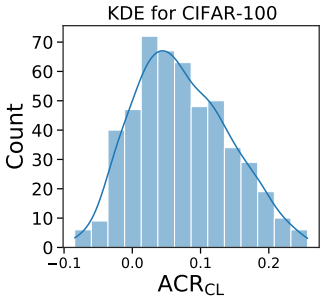
(a) ACR_{CL} for 10 images chosen from CIFAR-10



(b) $\underline{\text{R}}_{\text{CL}}$ for *deer* and *dog*



(c) KDE of ACR_{CL}



(d) KDE of ACR_{CL}

Figure 6: The models are trained with $\epsilon_{train} = 4.4/255$. (a,b,c) run on CNN-B, (d) run on ResNet-18. (a) ACR_{CL} for images, which are randomly chosen from CIFAR-10; $K = 50$. (b) $\underline{\text{R}}_{\text{CL}}$ for two images from CIFAR-10; $K = 50$. (c,d) Frequency distribution histogram and kernel density estimation (KDE) of ACR_{CL} for 500 images; (c) KDE for CIFAR-10 given by deterministic verification; (d) KDE for CIFAR-100 given by probabilistic verification.

verification, about 90% of images' ACR_{CL} are distributed within the range $[0.044, 0.093]$, concentrated around 0.07. We use probabilistic verification with $\sigma = 0.1, N_G = 256, \tau = 0.1$ to plot Figure 6(d). Recall that $\underline{\text{R}}_{\text{CL}} < 0$ means the verifier fails to verify f at (x^+, x^-). About 90% ACR_{CL} are distributed within $[\ 0.028, 0.19]$, concentrated around 0.065; 84.6% images are verified successfully.

6.3 Tightness of deterministic verification

In this subsection, we illustrate that RVCL is an effective verification framework by comparing the strength of different deterministic verifiers.

Certified instance accuracy Refer to Theorem 12, we use the certified instance accuracy ($\underline{A}^{\epsilon}_{\text{CL}}$ in Theorem 11) to compare the strength of deterministic verifiers on the CL encoder without labels. We study the performance of incomplete verifiers via $\underline{A}^{\epsilon}_{\text{CL}}$. Table 4 summarizes the results of *certified instance accuracy* provided by our proposed RVCL on

Table 4: Complete comparison of certified instance accuracy across various networks and attack strength ϵ_{test}. The number of test samples $|U_{\texttt{test}}| = 100$. CBC uses 3 minutes for each sample.

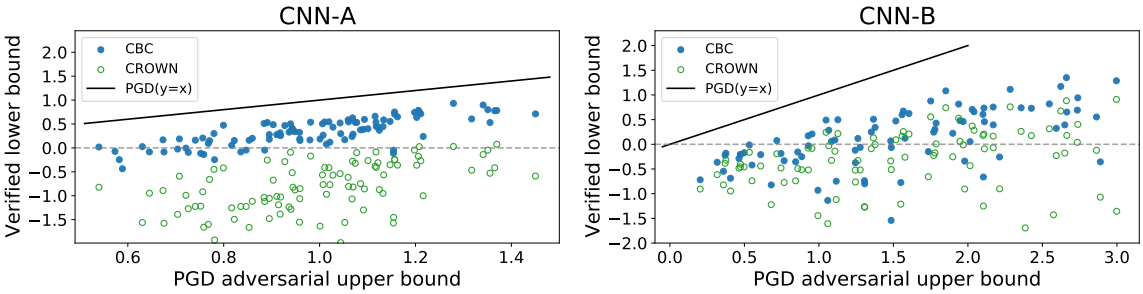
Dataset	ϵ_{neg}	ϵ_{test}	Model	ϵ_{train}	Instance Accuracy	Certified Instance Accuracy	
					PGD	**CBC**	**CROWN**
Mnist	0.3	0.1	Based	0	20%	2%	0%
			CNN-A		6%	0%	0%
			Based	0.1	100%	100%	100%
			CNN-A		100%	100%	99%
	0.5	0.3	Based	0.3	100%	98%	42%
			CNN-A		100%	85%	3%
CIFAR-10	$\frac{16}{255}$	$\frac{2}{255}$	CNN-B	0	100%	97%	96%
				$\frac{2.2}{255}$	100%	100%	100%
				$\frac{4.4}{255}$	91%	26%	11%
				$\frac{8.8}{255}$	100%	55%	34%
		$\frac{4}{255}$			100%	68%	52%
			Based	$\frac{4.4}{255}$	100%	99%	95%
			Deep		100%	96%	84%
			CNN-A		99%	91%	81%
		$\frac{8}{255}$	CNN-B	$\frac{8.8}{255}$	1%	0%	0%
	$\frac{24}{255}$	$\frac{6}{255}$		$\frac{4.4}{255}$	100%	8%	2%
				$\frac{8.8}{255}$	100%	24%	11%

Mnist and CIFAR-10. In order to control the similarity $\rho(f(x^-), f(x'))$ between x' and x^-, we generate the negative sample x^- via instance-wise PGD attack (Kim et al., 2020) with strength ϵ_{neg} which is much larger than ϵ_{test}. The two incomplete verifiers we utilize herein are CROWN and CBC; CBC is the state-of-the-art verifier, which is more powerful than CROWN for supervised verification (Wang et al., 2021).

$\underline{A}^{\epsilon}_{\text{CL}}$ is the lower bound of the robust instance accuracy A^{ϵ}_{CL} (Theorem 10, obtained by instance-wise PGD attack). The tighter lower bound given by $\underline{A}^{\epsilon}_{\text{CL}}$ indicates a stronger incomplete verifier. Table 4 shows that the gap between $\underline{A}^{\epsilon}_{\text{CL}}$ given by CBC and A^{ϵ}_{CL} given by PGD is smaller than that between CROWN and PGD on both Mnist and CIFAR-10 datasets, which shows that CBC is a stronger verifier than CROWN. The results demonstrate the efficacy of our proposed RVCL: a stronger supervised verifier can still achieve a tighter certified radius in the RVCL framework.

From Table 4, we can also make the following observations and remarks:

Influence of ϵ_{test}: If ϵ_{test} is small, the certified instance accuracy $\underline{A}^{\epsilon}_{\text{CL}}$ of both CROWN and CBC approach the robust instance accuracy A^{ϵ}_{CL} given by instance-wise PGD. However, the gap between CROWN and CBC becomes large as ϵ_{test} increases. The results show that CBC is a more powerful verifer than CROWN. The reason for this is that CBC



(a) Mnist, $\epsilon_{train} = 0.3$, $\epsilon_{test} = 0.3$ (b) CIFAR-10, $\epsilon_{train} = \frac{4.4}{255}$, $\epsilon_{test} = \frac{4}{255}$

Figure 7: **Comparing the tightness of verifiers.** For 100 test samples ($jU_{\texttt{test}}j = 100$) on Mnist and CIFAR-10. We plot the verified lower bound $\underline{f}(x^+, x^-, \epsilon)$ against PGD upper bound $\overline{f}$. Some points exceed the plotted axes limits.

optimizes the intermediate layer bounds and then iteratively tightens the lower bound. We can further observe that instance-wise PGD successfully attacks the model on all images of CIFAR-10 under $\epsilon_{test} = 8/255$.

Remark 19. *The results in Table 4 often achieve a high robust instance accuracy* A^{ϵ}_{CL}. *The direct reason is that the instance-wise attack is more difficult than the label-wise attack. Theoretically, Theorem 15 shows that the robust radius* R_{CL} R_{LE}. *This implies that one may label-wise attack an image successfully with a small* ϵ_{test}, *but that it is nearly impossible to successfully instance-wise attack with the same small* ϵ_{test}, *which results in a high robust instance accuracy. Figure 4(b) experimentally certifies this conclusion,* ACR_{CL} *is an order of magnitude larger than* ACR_{LE}. *However, without loss of effectiveness, we can still evaluate the tightness of verifiers by comparing the gap between* A^{ϵ}_{CL} *and* $\underline{A}^{\epsilon}_{\text{CL}}$.

Influence of ϵ_{train}: The certified instance accuracy $\underline{A}^{\epsilon}_{\text{CL}}$ increases with increasing ϵ_{train} (consistent with Figure 4(c,d)), which demonstrates that $\underline{A}^{\epsilon}_{\text{CL}}$ can also evaluate the model robustness. Howerver, as we discuss in Theorem 12, $\underline{A}^{\epsilon}_{\text{CL}}$ is a function of specific attack strength ϵ_{test}, it's hard to compare the robustness of two models by comparing $\underline{A}^{\epsilon}_{\text{CL}}$ of various values of ϵ_{test}. Thus ACR_{CL} is a more suitable choice to campare models with different robust performance.

Illustration of verified lower bound $\underline{f}$ We can also evaluate the strength of deterministic verifiers by the verified lower bound $\underline{f}$. Instance-wise PGD attack (Kim et al., 2020) provides the upper bound of minimum distortion, $\overline{f}$ $\widetilde{f}$, while deterministic verifier provides the lower bound, $\widetilde{f}$ $\underline{f}(x^+, x^-, \epsilon)$. Distinguish from supervised verification (Dathathri et al., 2020; Wang et al., 2021), all distortion here is instance-wise. The closer the verified lower bound $\underline{f}(x^+, x^-, \epsilon)$ is to the PGD upper bound $\overline{f}$ ($y = x$ in Figure 7), the stronger the verifier would be.

As Figure 7 shows, the points above the dotted line are successfully verified. CBC achieves tight verification across all samples, and furthermore consistently outperforms CROWN on Mnist and CIFAR-10. This conclusion is consistant with Table 4 and supervised verification results (Wang et al., 2021), which further demonstrates the effectiveness of RVCL.

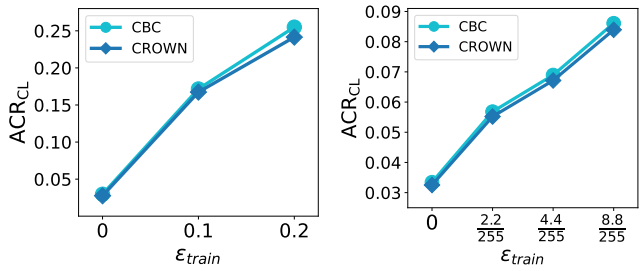


Figure 8: ACR_{CL} given by CBC and CROWN on Mnist (**Left**) and CIFAR-10 (**Right**). $|U_{\texttt{test}}| = 100$, $K = 10$.

Time(s)	CROWN	CBC
Mnist	1.16	464.52
CIFAR-10	3.66	921.78

Table 5: Average verification time per image, the number of negative samples $K = 5$, and timeout is set to 0.3.

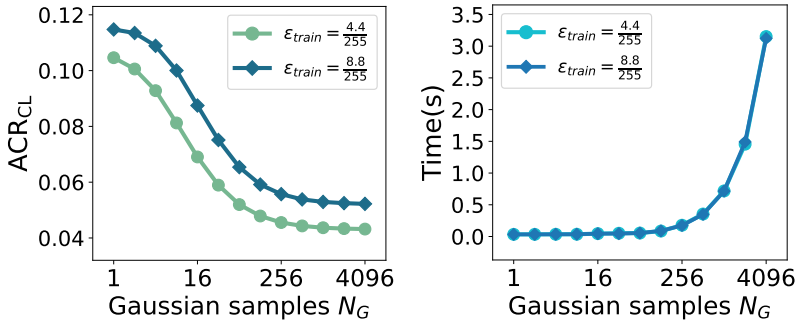


Figure 9: Sensitivity analysis of the number of Gaussian samples N_G for probabilistic verification. **Left:** N_G influencing ACR_{CL}. **Right:** N_G influencing average verification time per image. Experiments run on CIFAR-10 and ResNet-18.

6.4 Efficiency of verification

In this subsection, we explore key factors within RVCL that can affect verification efficiency.

Efficiency of deterministic verifiers Section 6.1 and Section 6.2 use CROWN to compute the certified radius ACR_{CL} in the interests of efficiency. This subsection shows that CBC achieves similar ACR_{CL} with CROWN, but takes more time than CROWN.

Timeout is set to $0.3s$ for each step of CBC binary search. CNN-A is run on Mnist and CNN-B is run on CIFAR-10. The experimental setting of Figure 8 is the same with that in Section 6.1. Table 5 provides the average time to caculate ACR_{CL} over 20 images.

The results in Figure 8 show that ACR_{CL} over $U_{\texttt{test}}$ provided by CBC and CROWN is nearly the same, and both of them show the tendency of robust performance. This is because the time for each step of binary search is too short for CBC to tighten the lower bound. However, Table 5 shows that the time cost of CBC is about **300 times** slower than CROWN, even using a small value of timeout for CBC. Therefore, we conclude that CBC is able to achieve a tight bound, but CBC is time-consuming in computing $\underline{\text{R}}_{\text{CL}}$, and it is reasonable to use CROWN to compute ACR_{CL} of models and images.

The number of Gaussian samples N_G We conducted experiments (Figure 9) to examine how the number of Gaussian samples N_G affects both ACR_{CL} and probabilistic verification time per image. Our investigation spanned a wide range of N_G, from 2^0 to $2^{12}(4096)$. Larger N_G provides a more accurate estimation of the empirical expectation of p_f^+ in (15). As

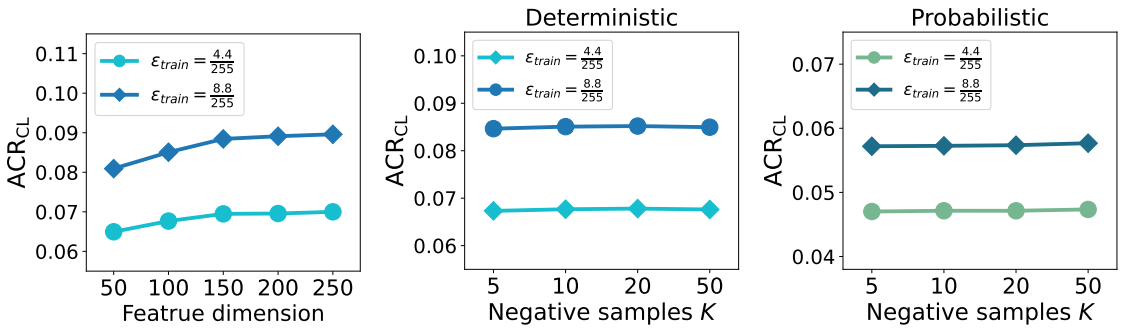


Figure 10: Sensitivity analysis of RVCL. **Left:** feature dimension influencing ACR_{CL} of deterministic verification. **Middle and Right:** the number of negative samples K influencing ACR_{CL} of deterministic and probabilistic verification, respectively.

shown in Figure 9(Left), ACR_{CL} exhibits efficacy regardless of the chosen value for N_G, as ACR_{CL} for $\epsilon_{train} = 4.4/255$ consistently remains lower than that for $\epsilon_{train} = 8.8/255$ across various N_G. Figure 9(Left) also indicates that ACR_{CL} stabilizes when N_G exceeds 256, while Figure 9(Right) reveals exponential verification time growth when $N_G > 128$. To ensure both stable ACR and efficient computation, we set $N_G = 256$ for probabilistic verification.

6.5 Sensitivity analysis

Deterministic verification We conduct a sensitivity analysis for deterministic verification to examine the effect of hyperparameters. Experiments run on CNN-B with different ϵ_{train} on CIFAR-10.

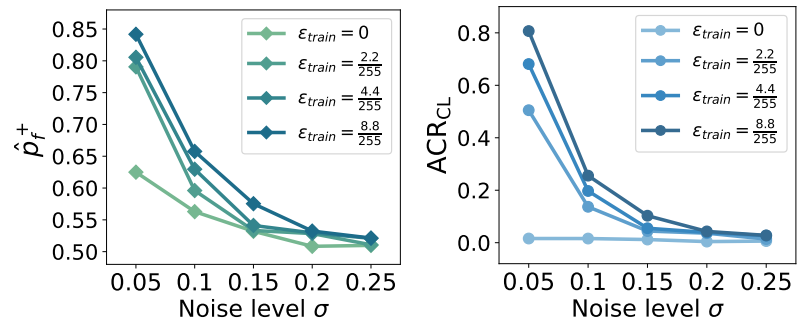
Feature dimension: We investigate the influence of the feature dimension on ACR_{CL} (see **Left** of Figure 10). From the result, we can determine that ACR_{CL} increases slightly with the growing feature dimension, then remains stable on dimensions larger than 150. The results illustrate that ACR_{CL} is not sensitive to feature dimension.

Number of negative samples: We validate the influence of the number of negative samples K on ACR_{CL} (see **Middle** of Figure 10). The result shows that ACR_{CL} is not sensitive to K with different ϵ_{train}, which means that we can use small values of K to efficiently evaluate the model robustness or the anti-disturbance ability of an image. We set $K = 10$ throughout the experiments.

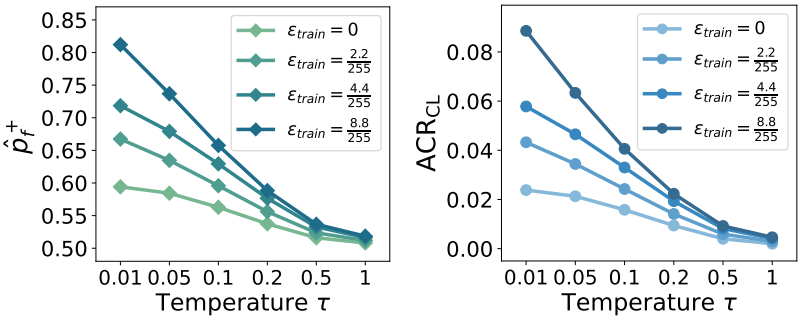
Probabilistic verification In this subsection, experiments run on CIFAR-10 with different ϵ_{train}, $N_G = 256$.

Number of negative samples: Similarly to the deterministic verification, we evaluate the effect of different numbers of negative samples K on ACR_{CL} (see **Right** of Figure 10). Experiments run on ResNet-18 with $\epsilon_{train} = 4.4/255, 8.8/255$. The results show that ACR_{CL} is also not sensitive to K with different values of ϵ_{train}. We set $K = 10$ for consistency with deterministic verification.

Noise level σ: Experiments for Figure 11 run on ResNet-50 with $\epsilon_{train} = 4.4/255$. Figure 11(a,b) plots the average of positive sample probability $\widehat{p}_f^+$ (Theorem 5) and ACR_{CL} with respect to the noise level σ. Recall that ACR_{CL} is a function of $\widehat{p}_f^+$ (Theorem 7). From Figure 11(a), we observe that $\widehat{p}_f^+$ decreases as σ increases. $\widehat{p}_f^+$ decreases to 0.5 when σ is



(a) σ influencing p_f^+ (b) σ influencing ACR$_{\text{CL}}$



(c) τ influencing p_f^+ (d) τ influencing ACR$_{\text{CL}}$

Figure 11: Sensitivity analysis for probabilistic verification. (a,b) noise level σ influencing average probability p_f^+ and ACR$_{\text{CL}}$. (c,d) temperature τ influencing average probability p_f^+ and ACR$_{\text{CL}}$.

large ($\sigma = 0.25$), which means the verifier degenerates to the random guess and fails to verify. This is because $\sigma = 0.25$ is a large noise level and models perform poorly to recognize the positive sample. $\sigma = 0.1$ is the best to distinguish the encoders with different robustness, thus we choose $\sigma = 0.1$ in Section 6.1 to evaluate the CL encoder by ACR$_{\text{CL}}$.

Temperature τ: In Figure 11(c,d), we illustrate $\widehat{p}_f^+$ and ACR$_{\text{CL}}$ with respect to the temperature τ in (15). When τ ! 1, $\widehat{p}_f^+$! 0.5 and ACR$_{\text{CL}}$! 0. This is because the margin between $\rho(f(x'), f(x^+))$ and $\rho(f(x'), f(x^-))$ is small when τ ! 1. We choose $\tau = 0.1$ in Section 6.1 consistent with (Chen et al., 2020).

7 Conclusion

In this paper, we tackle both deterministic and probabilistic robustness verification problems for CL without labels, and accordingly propose a novel RVCL framework that does not depend on any class labels, downstream tasks or specific attack algorithms. We then use extreme value theory to reveal the quantitative relationship between the robust radius of the CL encoder and that of the downstream task. All our experiments show that RVCL is an efficient robustness framework for CL encoders, and can also be used to evaluate the anti-disturbance ability of images. Moreover, our experimental results verify our theoretical analysis. We believe that RVCL is a novel perspective from which to understand robustness on contrastive learning.

Appendix A. Proofs

A.1 Proof of Theorem 7

Theorem 7 (Probabilistic verification for CL). *Given two positive and negative samples $x^+, x^- \in \mathcal{X}$, the feature encoder $f : \mathcal{X} \to \mathbb{R}^d$, let $\eta \sim \mathcal{N}(0, \sigma^2 I)$, if*

$$\mathbb{E}_\eta(p_f^+(x^+ + \eta; x^+, x^-)) > 0.5, \tag{21}$$

then f is $l_2^{\mathrm{R}_{\mathrm{CL}}}$-verified at (x^+, x^-). The certified radius $\underline{\mathrm{R}}_{\mathrm{CL}}$ is given by

$$\underline{\mathrm{R}}_{\mathrm{CL}}(f; x^+, x^-) = \sigma \Phi^{-1}(\mathbb{E}_\eta(p_f^+(x^+ + \eta; x^+, x^-))), \tag{22}$$

where Φ^{-1} is the inverse cumulative distribution function of the standard Gaussian distribution.

The proof is based on the following lemma:

Lemma 20. *For any measurable function $f : \mathcal{X} \to [0, 1]$, define $\widehat{f}(x) = \mathbb{E}_{\eta \sim \mathcal{N}(0,\sigma^2 I)} f(x + \eta)$, then $x \mapsto \Phi^{-1}(\widehat{f}(x))$ is $1/\sigma$-Lipschitz.*

This lemma is the generalized version of Lemma 2 in Salman et al. (2019). Then we formally prove Theorem 7.

Proof $p_f^+(x; x^+, x^-)$ is the probability of x being a positive sample of x^+ (Theorem 5). We define $\widehat{p}_f^+$ as:

$$\widehat{p}_f^+(x; x^+, x^-) = \mathbb{E}_{\eta \sim \mathcal{N}(0,\sigma^2 I)}(p_f^+(x + \eta; x^+, x^-)). \tag{23}$$

We simply use $\widehat{p}_f^+(x)$ to refer to $\widehat{p}_f^+(x; x^+, x^-)$ when the context is clear. By Theorem 20, we know that under any perturbation δ of x^+,

$$\Phi^{-1}(\widehat{p}_f^+(x^+)) \quad \Phi^{-1}(\widehat{p}_f^+(x^+ + \delta)) \quad \frac{k\delta k_2}{\sigma}. \tag{24}$$

For an adversarial perturbation δ, $\Phi^{-1}(\widehat{p}_f^+(x^+ + \delta)) \geq 0.5$, leading to $\Phi^{-1}(\widehat{p}_f^+(x^+)) \geq \frac{\|\delta\|_2}{\sigma}$, i.e.,

$$\underline{\mathrm{R}}_{\mathrm{CL}}(f; x^+, x^-) = \sigma \Phi^{-1}(\widehat{p}_f^+(x^+)). \tag{25}$$

■

A.2 Extreme Value Theory

Before provding the proofs of main results, we first provide two important lemmas in extreme value theory (EVT).

Lemma 21 (Fisher-Tippett-Gnedenko theorem (Coles et al., 2001))**.** *Let $X_1, X_2, \ldots$ be a sequence of independent random variables with common distribution function F. Let $M_n = \max(X_1, \ldots, X_n)$. If there exists a sequence $a_n > 0, b_n \in \mathbb{R}$ such that*

$$\lim_{n\to\infty} \mathbb{P}(\frac{M_n - b_n}{a_n} \leq z) = G(z), \tag{26}$$

where G is a non-degenerate distribution function, then G belongs to either the Gumbel family (Type I), the Fréchet family (Type II) or the Reverse Weibull family (Type III) with their CDFs as follows:

$$\text{Gumbel family (Type I):} \quad G(z) = \exp\{ \quad \exp[\quad (\frac{z \quad b}{a})]\}, \quad z \; 2 \; \mathsf{R},$$

$$\text{Fréchet family (Type II):} \quad G(z) = \begin{cases} 0, & \text{if } z < b, \\ \exp\{ \quad (\frac{z-b}{a})^{-\alpha}\}, & \text{if } z \quad b, \end{cases}$$

$$\text{Reverse Weibull family (Type III):} \quad G(z) = \begin{cases} \exp\{ \quad (\frac{b-z}{a})^{\alpha}\}, & \text{if } z < b, \\ 1, & \text{if } z \quad b, \end{cases}$$

where $a > 0$, $b \; 2 \; \mathsf{R}$ and $\alpha > 0$ are the scale, location and shape parameters, respectively.

Theorem 21 states that the rescaled sample maxima $(M_n \quad b_n)/a_n$ converge in distribution to a variable that has a distribution within one of three families. Furthermore, these three families can be combined into a single family called generalized extreme value (GEV) distribution, which is a family of continuous probability distributions developed within extreme value theory. The Gumbel, Fréchet and Reverse Weibull families are special cases of GEV distribution.

Lemma 22 (Generalized Extreme Value (GEV) distribution (Coles et al., 2001))**.** *Let $X_1, X_2, \ldots$ be a sequence of i.i.d. samples from the distribution function F. Let $M_n = \max(X_1, \ldots, X_n)$. If there exists a sequence $a_n > 0$, $b_n \; 2 \; \mathsf{R}$ such that $\lim_{n\to\infty} \mathsf{P}(\frac{M_n - b_n}{a_n} z) = G(z)$, then if G is a non-degenerate distribution function, it belongs to the class of generalized extreme value (GEV) distributions with*

$$G(z) = \exp\left[\quad \left\{1 + \xi\left(\frac{z \quad \mu}{\sigma}\right)\right\}^{-1/\xi}\right], \tag{27}$$

where $z \; 2 \; \mathsf{R} : 1 + \xi\left(\frac{z-\mu}{\sigma}\right) > 0$, $\xi \; 2 \; \mathsf{R}$, $\mu \; 2 \; \mathsf{R}$ and $\sigma > 0$ are the shape, location and scale parameters, respectively.

As the special case, the Reverse Weibull family in Theorem 21 can be derived by Theorem 22, which let $\xi < 0$ and the upper endpoint of F be denoted by b, then $\alpha = \quad 1/\xi > 0$.

A.3 Proof of Theorem 14

Theorem 14 (Margin distribution). *Assume a continuous non-degenerate margin distribution exists. The distribution for margin distance M is then given by the Reverse Weibull distribution. The probability of x being a positive sample of x^+ is given by the following:*

$$\Psi(x; x^+, \alpha, \sigma) = \exp\left\{ \quad \left(\frac{1 \quad \rho(f(x), f(x^+))}{\sigma}\right)^{\alpha}\right\}, \tag{28}$$

where $\rho(f(x), f(x^+))$ is the instance similarity between x and x^+. $\alpha, \sigma > 0$ are Weibull shape and scale parameters, obtained from fitting to the λ smallest margin distances D_i.

Proof From the assume we know that $G(z)$ in Theorem 21 exists. Since Theorem 21 applies to maxima, we transform the variables via $\overline{M} = \max_{i\in[K]} \quad D_i$, $D_i \coloneqq (1 \quad \rho(f(x^+), f(x_i^-))/2$. Because $\quad D_i$ is bounded ($\quad D_i < 0$), so the asymptotic distribution of $\overline{M}$ converges to the Reverse Weibull distribution:

$$W(z) = \begin{cases} \exp\left\{ \quad \left(\quad \frac{z}{\sigma} \right)^{\alpha} \right\}, & \text{if } z < 0, \\ 1, & \text{if } z \quad 0, \end{cases} \tag{29}$$

where $\alpha > 0$, b is the upper endpoint of F, σ is the scale parameters. $b = 0$ in (29) since $\overline{M}$ is bounded above by 0 as a negative distance. We use margin distances D_i of the λ closest samples with x^+ to estimate the parameters α and σ, which means to estimate $\widehat{W}$ of the distribution funstion W.

The margin distance between x and x^+ defined as $1 \quad \rho(f(x), f(x^+))$. Then we focus on the probability of x included in the margin of x^+, which can be written as:

$$\begin{aligned} &\mathsf{P}(1 \quad \rho(f(x), f(x^+)) < \min(D_1, \ldots, D_n)) \\ =&\mathsf{P}(\quad \min(D_1, \ldots, D_n) < \rho(f(x), f(x^+)) \quad 1) \\ =&\mathsf{P}(\max(\quad D_1, \ldots, \quad D_n) < \rho(f(x), f(x^+)) \quad 1) \\ =&\mathsf{P}(\overline{M} < \rho(f(x), f(x^+)) \quad 1) \\ =&\widehat{W}(\rho(f(x), f(x^+)) \quad 1). \end{aligned} \tag{30}$$

Since $\rho(f(x), f(x^+)) \quad 1 < 0$, we can rewrite (30) as (28). We conclude our proof. ■

A.4 Proof of Theorem 15

Theorem 15 (Robust radius bound). *Given an encoder $f : \mathcal{X} \; ! \; \mathsf{R}^d$ and an unlabeled sample $z = (x^+, \mathsf{f}x_i^- \mathsf{g}_{i=1}^K)$, the downstream predictor $g : \mathsf{R}^d \; ! \; \mathsf{R}$ is trained on $\widehat{S} = \mathsf{f}(f(x^+), y_{c^*}), (f(x_i^-), y_c \;)_{i=1}^K \mathsf{g}$. Then, for different negative samples x_i^-, we have*

$$\mathrm{R}_{\mathrm{CL}}(f; x^+, x_i^-) \quad \mathrm{R}_{\mathrm{LE}}(g; x^+, y_{c^*}). \tag{31}$$

Before we formally prove Theorem 15, we first provide the following two lemmas.

Theorem 23 focuses on a model for the k largest order statistics. It extends the result in Theorem 21 to extreme order statistics, by defining $M_n^{(k)} = k$-th largest of $\mathsf{f}X_1, \ldots, X_n\mathsf{g}$ and further identifying the limiting behavior of this variable, for fixed k, as $n \; ! \; \mathcal{1}$. Theorem 23 implies that, if the k-th largest order statistic is normalized in exactly the same way as the maximum, then its limiting distribution is of the form given by (33).

Lemma 23 (k-th largest order statistic (Coles et al., 2001))**.** *Let $X_1, X_2, \ldots$ be a sequence of i.i.d. samples from the distribution function F. Let $M_n = \max(X_1, \ldots, X_n)$. If there exists a sequence $a_n > 0$, $b_n \; \mathcal{2} \; \mathsf{R}$ such that $\lim_{n\to\infty} \mathsf{P}(\frac{M_n - b_n}{a_n} \quad z) = G(z)$ for a non-degenerate distribution function G, so that G is the GEV distribution function given by (27), then, for fixed k,*

$$\lim_{n\to\infty} \mathsf{P}((M_n^{(k)} \quad b_n)/a_n) = G_k(z) \tag{32}$$

on $\mathsf{f}z : 1 + \frac{\xi(z-\mu)}{\sigma} > 0\mathsf{g}$*, where*

$$G_k(z) = \exp\mathsf{f} \quad \tau(z)\mathsf{g}\sum_{s=0}^{k-1}\frac{\tau(z)^s}{s!} \tag{33}$$

with $\tau(z) = \left\{1 + \xi\left(\frac{z-\mu}{\sigma}\right)\right\}^{-1/\xi}$.

Intuitively, the classifier predicting correctly with higher confidence implies that the classifier can provide better certified robustness. Theorem 24 provides this relationship directly.

Lemma 24 (Robust radius (Yang et al., 2020))**.** *The robust radius in any norm* $\mathsf{k} \quad \mathsf{k}$ *is at least*

$$\mathrm{R} \coloneqq \int_{1-\lambda}^{1/2}\frac{1}{\Phi(p)}\mathrm{d}p, \tag{34}$$

where $\Phi(p) \coloneqq \sup_{\|v\|=1}\sup_{U\subseteq\mathsf{R}^d:q(U)=p}\lim_{r\searrow 0}\frac{q(U-rv)-p}{r}$*,* λ *is the probability that the binary classifier predicts the right label under perturbation,* $q(U)$ *is the measure of* U *under* q*, i.e.* $q(U) = \Pr_{\delta\sim q}(\delta \; \mathsf{2} \; U)$*,* v *is the perturbation vector.*

Finally, we prove Theorem 15.

Proof Ψ_{LE} is obtained from fitting to margin distances $\mathsf{f}D_i\mathsf{g}_{i=1}^K, D_i \coloneqq (1 \quad \rho(f(x^+), f(x_i^-))/2$. From Theorem 21 we know that (28) is the Reverse Weibull distribution and can be written as the form of (27):

$$\Psi_{\mathrm{LE}}(x; x^+, \alpha, \sigma) = \exp\left[\quad\left\{1 + \xi\left(\frac{z \quad \mu}{\sigma}\right)\right\}^{-1/\xi}\right] \tag{35}$$

with $z = \quad D = \rho(f(x), f(x^+)) \quad 1$.

$\mathrm{R}_{\mathrm{CL}}(f; x^+, x_i^-)$ is defined on the positive and negative sample pair (x^+, x_i^-); it means that Ψ_{CL} is fitted from specific (x^+, x_i^-). We denote $\quad D^{(k)}$ for x_i^- is the k-th largest order statistic of $\mathsf{f} \quad D_1, \ldots, \quad D_K\mathsf{g}$. By Theorem 23, the distribution function Ψ_{CL} for $\quad D^{(k)}$ can be written as:

$$\Psi_{\mathrm{CL}}(x; x^+, \alpha, \sigma, k) = \exp\mathsf{f} \quad \tau(z)\mathsf{g}\sum_{s=0}^{k-1}\frac{\tau(z)^s}{s!} \tag{36}$$

with $\tau(z) = \left\{1 + \xi\left(\frac{z-\mu}{\sigma}\right)\right\}^{-1/\xi}, z = \rho(f(x), f(x^+)) \quad 1$. ξ, μ, σ are the same with (35).

Let $x \quad \mathsf{D}_{c^+}$ be the positive test sample of x^+. As we discuss in Section 3.1, the latent label for unlabeled positive sample can be given by y_{c^+}. In Section 4, we propose that judging positive sample correctly on CL encoder and downstream task can be transformed to judge whether or not $\mathbf{W}_{\mathrm{CL}}f(x) > 0$ and $y_{c^+} \quad g(x) > 0$ are True, respectively. Thus, for

every negative sample x_i^-, we have

$$\begin{aligned}
&\mathsf{P}(\mathbf{W}_{\mathrm{CL}} f(x) > 0 \,\mathsf{j}\, x) \quad \mathsf{P}(y_{c^+} \quad g(x) > 0 \,\mathsf{j}\, x) \\
=&\Psi_{\mathrm{CL}}(x; x^+, \alpha, \sigma, k) \quad \Psi_{\mathrm{LE}}(x; x^+, \alpha, \sigma) \\
=&\exp\mathsf{f} \quad \tau(z)\mathsf{g} \sum_{s=0}^{k-1} \frac{\tau(z)^s}{s!} \quad \exp\left[\quad \left\{1 + \xi\left(\frac{z \quad \mu}{\sigma}\right)\right\}^{-1/\xi}\right] \\
=&\exp\mathsf{f} \quad \tau(z)\mathsf{g} \sum_{s=1}^{k-1} \frac{\tau(z)^s}{s!} \\
&0
\end{aligned} \tag{37}$$

with $\tau(z) = \left\{1 + \xi\left(\frac{z-\mu}{\sigma}\right)\right\}^{-1/\xi}$, $z = \rho(f(x), f(x^+)) \quad 1$. Equality holds if and only if $k = 1$. Recall that $x = x^+ + \delta, \mathsf{k}\delta\mathsf{k}_\infty \quad \epsilon$, i.e. $x \; \mathsf{2} \; \mathsf{B}_\infty(x^+, \epsilon)$, which is in the constraints of (10) and (13). By Theorem 24, we have:

$$\begin{aligned}
&\mathrm{R}_{\mathrm{CL}}(f; x^+, x_i^-) \quad \mathrm{R}_{\mathrm{LE}}(g; x^+, y_{c^+}) \\
=&\int_{1-\Psi_{\mathsf{CL}}(x;x^+,\alpha,\sigma,k)}^{1/2} \frac{1}{\Phi(p)}\mathrm{d}p \quad \int_{1-\Psi_{\mathsf{LE}}(x;x^+,\alpha,\sigma)}^{1/2} \frac{1}{\Phi(p)}\mathrm{d}p \\
=&\int_{1-\Psi_{\mathsf{CL}}(x;x^+,\alpha,\sigma,k)}^{1-\Psi_{\mathsf{LE}}(x;x^+,\alpha,\sigma)} \frac{1}{\Phi(p)}\mathrm{d}p \\
&0,
\end{aligned} \tag{38}$$

which recovers the theorem statement. Equality holds if and only if $k = 1$. ■

A.5 Proof of Theorem 18

Theorem 18. *Given an encoder $f : \mathsf{X} \;\; ! \;\; \mathsf{R}^d$, two positive samples x_1^+, x_2^+ and one negative sample x^-, if $\rho(f(x_1^+), f(x^-)) \quad \rho(f(x_2^+), f(x^-))$, then*

$$\mathrm{R}_{\mathrm{CL}}(f; x_1^+, x^-) \quad \mathrm{R}_{\mathrm{CL}}(f; x_2^+, x^-). \tag{39}$$

Proof In this proposition, we consider the situation in which multiple positive samples x^+ and only one negative sample x^- are used. Formally, given an unlabeled sample $z = (\mathsf{f}x_i^+\mathsf{g}_{i=1}^K, x^-)$, we define the margin distance of x^- as $M^- \coloneqq \min_{i\in[K]} D_i^-$, where $D_i^- \coloneqq (1 \quad \rho(f(x_i^+), f(x^-))/2$.

We denote the lower tail of the distribution of M^- as Ψ_{neg}. Similar to the idea of Theorem 14, we use Ψ_{neg} to produce the probability of x falling into the margin of x^-, which can be interpreted as the probability of x being a negative sample. From Theorem 21, we know that Ψ_{neg} also converges to the Reverse Weibull distribution, and can be written to the form as following:

$$\Psi_{\mathrm{neg}}(x; x^-, \alpha, \sigma) = \exp\left\{ \quad \left(\frac{1 \quad \rho(f(x), f(x^-))}{\sigma}\right)^{\alpha}\right\}, \tag{40}$$

where $\rho(f(x), f(x^-))$ is the instance similarity between x and x^-. $\alpha, \sigma > 0$ are Weibull shape and scale parameters, obtained from fitting to the λ smallest margin distances D_i^-.

Cumulative distribution function Ψ_{neg} is monotonic increasing. Given two positive samples x_1^+, x_2^+, if $\rho(f(x_1^+), f(x^-))$ $\rho(f(x_2^+), f(x^-))$, we have:

$$\Psi_{\text{neg}}(x_1^+; x^-, \alpha, \sigma) \quad \Psi_{\text{neg}}(x_2^+; x^-, \alpha, \sigma) \tag{41}$$

x being the positive sample of x^+ means that x is more similar to x^+ than to x^-. From Theorem 24 we have:

$$\begin{aligned} & \mathrm{R}_{\mathrm{CL}}(f; x_1^+, x^-) \quad \mathrm{R}_{\mathrm{CL}}(f; x_2^+, x^-) \\ = & \int_{\Psi_{\text{neg}}(x_1^+; x \quad, \alpha, \sigma)}^{1/2} \frac{1}{\Phi(p)} \mathrm{d}p \quad \int_{\Psi_{\text{neg}}(x_2^+; x \quad, \alpha, \sigma)}^{1/2} \frac{1}{\Phi(p)} \mathrm{d}p \\ = & \int_{\Psi_{\text{neg}}(x_2^+; x \quad, \alpha, \sigma)}^{\Psi_{\text{neg}}(x_1^+; x \quad, \alpha, \sigma)} \frac{1}{\Phi(p)} \mathrm{d}p \\ & 0, \end{aligned} \tag{42}$$

which recovers the theorem statement. Equality holds if and only if $\rho(f(x_1^+), f(x^-)) = \rho(f(x_2^+), f(x^-))$. ■

Appendix B. Analysis for Multi-class Setting

B.1 Contrastive Learning

We provide the contrastive framework for multi-class setting by extending the binary framework (Section 3.1) in Saunshi et al. (2019). We consider the data generation process with p negetive latent classes and p K negative samples. The scheme for generating an unlabeled sample $z = (x, x^+, \{x_{i1}^-\}_{i=1}^K, \ldots, \{x_{ip}^-\}_{i=1}^K)$ U is that:

1. Draw $p+1$ latent classes $(c^+, \{c_j^-\}_{j=1}^p)$ η^{p+1};
2. Draw two similar samples (x, x^+) $\mathcal{D}_{c^+}^2$;
3. Draw K negative samples from $\mathcal{D}_{c_j}$, j $[p]$: $\{x_{ij}^-\}$ $\mathcal{D}_{c_j}$ j i $[K]$.

We focus on *logistic loss*: $\ell(v) = \log_2(1 + \sum_m \exp(\quad v_m))$ for v $\mathbb{R}^K$, then we define the unsupervised contrastive loss with p negative latent classes:

$$L_{un}(f) = \mathop{\mathbb{E}}_{\substack{c^+, \{c_j\} \\ \sim \eta^{p+1}}} \mathop{\mathbb{E}}_{\substack{x, x^+ \sim \mathcal{D}_{c^+}^2 \\ x_{ij} \sim \mathcal{D}_{c_j}}} \ell\left(\left\{f(x)^T \left(f(x^+) \quad f(x_{ij}^-)\right)\right\}\right). \tag{43}$$

B.2 Proof of Theorem 17

Theorem 17 (Multi-class case for Theorem 15). *g_M is trained on the labeled dataset with $p+1$ classes, g_j is the binary classifier of the specific latent class pair (c^+, c_j^-), j $[p]$. Then, for different negative samples x_{ij}^- and latent class pairs (c^+, c_j^-) generated following Appendix B.1, we have*

$$\mathrm{R}_{\mathrm{CL}}(f; x^+, x_{ij}^-) \overset{(i)}{} \mathrm{R}_{\mathrm{LE}}(g_j; x^+, y_{c^+}) \overset{(ii)}{} \mathrm{R}_{\mathrm{LE}}(g_M; x^+, y_{c^+}).$$

Proof As the definition in Appendix B.1, we sample an unlabeled sample $z = (x, x^+, \mathsf{f}x_{i1}^- \mathsf{g}_{i=1}^K, \ldots, \mathsf{f}x_{ip}^- \mathsf{g}_{i=1}^K)$. For every latent class pair (c^+, c_j^-), we define the margin distance as $M_j \coloneqq \min_{i\in[K]} D_{ij}$, where $D_{ij} \coloneqq (1 \quad \rho(f(x^+), f(x_{ij}^-))/2$. Following the proof of Theorem 15 in Appendix A.4, we can define Ψ_{LE}^j and Ψ_{CL}^j for (c^+, c_j^-) as follows:

$$\begin{aligned}
\Psi_{\mathrm{LE}}^j(x; x^+, \alpha, \sigma) &= \exp\left[\quad \left\{1 + \xi\left(\frac{z \quad \mu}{\sigma}\right)\right\}^{-1/\xi}\right], \\
\Psi_{\mathrm{CL}}^j(x; x^+, \alpha, \sigma, k) &= \exp\mathsf{f} \quad \tau(z)\mathsf{g}\sum_{s=0}^{k-1}\frac{\tau(z)^s}{s!},
\end{aligned} \tag{44}$$

with $\tau(z) = \left\{1 + \xi\left(\frac{z-\mu}{\sigma}\right)\right\}^{-1/\xi}, z = \rho(f(x), f(x^+)) \quad 1$. Ψ_{LE}^j and Ψ_{CL}^j are obtained from fitting to the λ smallest margin distances D_{ij}. By Theorem 15, we have:

$$\begin{aligned}
&\mathrm{R}_{\mathrm{CL}}(f; x^+, x_{ij}^-) \quad \mathrm{R}_{\mathrm{LE}}(g_j; x^+, y_{c^+}) \\
&= \int_{1-\Psi_{\mathsf{CL}}^j(x;x^+,\alpha,\sigma,k)}^{1-\Psi_{\mathsf{LE}}^j(x;x^+,\alpha,\sigma)} \frac{1}{\Phi(t)}\mathrm{d}t \\
&\quad 0,
\end{aligned} \tag{45}$$

where $\Phi(t)$ is defined in Theorem 24, j 2 $[p]$. So the inequality (i) is proved.

For the multi-class scenario, the robust radius depends on the most vulnerable class, i.e.

$$\mathsf{P}(y_{c^+} \quad g_M(x) > 0 \,\mathsf{j}\, x) = \min_{j\in[p]} \Psi_{\mathrm{LE}}^j(x; x^+, \alpha, \sigma). \tag{46}$$

By Theorem 24, we have:

$$\begin{aligned}
&\mathrm{R}_{\mathrm{LE}}(g_j; x^+, y_{c^+}) \quad \mathrm{R}_{\mathrm{LE}}(g_M; x^+, y_{c^+}) \\
&= \int_{1-\Psi_{\mathsf{LE}}^j(x;x^+,\alpha,\sigma)}^{1-\min_{j\,2\,[p]}\Psi_{\mathsf{LE}}^j(x;x^+,\alpha,\sigma)} \frac{1}{\Phi(t)}\mathrm{d}t \\
&\quad 0,
\end{aligned} \tag{47}$$

then inequality (ii) is proved. Overall, we conclude our proof. ■

Appendix C. Incomplete verifiers

We introduce two incomplete verifiers with different verified tightness used for RVCL. Due to the nonlinear activations $\sigma(\ )$, the feasible set of (10) and (13) is nonconvex. One intuitive idea is to perform the convex relaxation of the feasible set to build incomplete verifiers. This paper discusses ReLU networks with CROWN (Zhang et al., 2018), which is a method used to relax the nonconvex equality constraints $\widehat{\phi}_k(\ ) = \sigma(\phi_k(\ ))$ to convex inequality constraints.

Let $\mathsf{l}_k^{(j)}$ and $\mathsf{u}_k^{(j)}$ be the lower and upper bound of $\phi_k^{(j)}$, i.e. $\mathsf{l}_k^{(j)} < \phi_k^{(j)} < \mathsf{u}_k^{(j)}$, k 2 $[L]$. Given the ReLU activation function $\sigma(y) = \max(y, 0)$, CROWN uses linear constraints to relax ReLU: $\overset{(i)}{j}\phi_k^{(j)} \quad \widehat{\phi}_k^{(j)} \quad \frac{\mathsf{u}_k^{(j)}}{\mathsf{u}_k^{(j)} - \mathsf{l}_k^{(j)}}\left(\phi_k^{(j)} \quad \mathsf{l}_k^{(j)}\right)$, where $0 \quad \overset{(i)}{j} \quad 1$. After convex relaxation, (10) and (13) can be efficiently solved, as follows:

Lemma 25 (CROWN bound (Zhang et al., 2018)). *Given an L-layer NN* $\phi_L : \mathsf{R}^{d_0} \; ! \; \mathsf{R}$ *with weights* $\mathbf{W}_k$ *and bias* b_k*, and the pre-activation bound* $\mathsf{l}_k^{(j)} < \phi_k^{(j)} < \mathsf{u}_k^{(j)}$ *(*$k \; 2 \; [L], j \; 2 \; [d_k]$*),* $x' \; 2 \; \mathsf{B}(x, \epsilon)$*, we have:*

$$\min_{x^0} \mathbf{W}_L \widehat{\phi}_{L-1}(x') + \mathsf{b}_L \quad \min_{x^0} \mathsf{c}^\top x' + c_0, \tag{48}$$

where c *and* c_0 *can be computed by* $\mathbf{W}_k$*,* b_k*,* $\mathsf{l}_k^{(j)}$*,* $\mathsf{u}_k^{(j)}$*.*

Another incomplete verifier stronger than CROWN is β-CROWN (Wang et al., 2021) which is the state-of-the-art verification method. β-CROWN uses a few steps of gradient ascent to achieve bounds as tight as possible but suffer from high time cost.

Lemma 26 (β-CROWN bound (Wang et al., 2021)). *Given an L-layer NN* $\phi_L : \mathsf{R}^{d_0} \; ! \; \mathsf{R}$ *with weights* $\mathbf{W}_k$ *and bias* b_k*, the pre-activation bound* $\mathsf{l}_k^{(j)} < \phi_k^{(j)} < \mathsf{u}_k^{(j)}$*,* $x' \; 2 \; \mathsf{B}(x, \epsilon)$*, and split constraints* $z \; 2 \; \mathcal{Z}$*, we have:*

$$\min_{x^0, z \in \mathcal{Z}} \mathbf{W}_L \widehat{\phi}_{L-1}(x') + \mathsf{b}_L \quad \max_{\geq 0} \min_{x^0} (\mathsf{a} + \mathsf{P} \quad)^\top x' + \mathsf{q}^\top \quad + c_0, \tag{49}$$

where $\mathsf{a}, \mathsf{P}, \mathsf{q}$ *and* c_0 *can be computed by* $\mathbf{W}_k$*,* b_k*,* $\mathsf{l}_k^{(j)}$*,* $\mathsf{u}_k^{(j)}$*,* *is the multiplier of Lagrange function.*

We modify two supervised verifiers with different verification tightness in order to show that: a stronger verifier can still achieve a tighter certified radius $\underline{\mathrm{R}}_{\mathrm{CL}}$ in RVCL framework, which illustrates the efficacy of RVCL. The related experiment results are in Section 6.3.

The procedure of incomplete verifier is denoted by IncompleteVerifier in Algorithm 1, which aims to give the verified lower bound $\underline{f}(x^+, x^-, \epsilon)$ of the function $\widetilde{f}$ in (13).

Appendix D. Difference with CLEVER

CLEVER (Cross-Lipschitz Extreme Value for nEtwork Robustness) (Weng et al., 2018b) estimates the robust radius R using extreme value theory (EVT). In this paper, we utilize EVT in a totally different way compared with CLEVER:

1. To produce the probability of x being a positive sample of x^+, we utilize EVT to estimate the lower tail of the **margin distance** (Theorem 14); thus, we can compare the probability given by the CL encoder and the downstream task. While CLEVER focuses on estimating R by proposing a sampling based approach with EVT to estimate the **local Lipschitz constant**; it is based on a theoretical analysis of formal robustness guarantee with Lipschitz continuity assumption.

2. We use EVT to reveal the quantitative relationship between the robust radius of the CL encoder and that of the downstream task. While CLEVER only focuses on the robust radius for supervised verification.

Appendix E. Experimental Settings

For probabilistic verification, we adopt the same setup as RoCL (Kim et al., 2020). In this section, we provide the setup for deterministic verification.

E.1 Datasets

For model training, we use Mnist (LeCun and Cortes, 2010) and CIFAR-10/CIFAR-100 (Krizhevsky and Hinton, 2009). Mnist is a dataset of 28 28 pixel grayscale images of handwritten single digits between 0 and 9, which contains 60,000 training images and 10,000 testing images with 10 classes. CIFAR-10 and CIFAR-100 contain 50,000 training and 10,000 testing images with 10 classes and 100 classes, respectively. Size for color image in CIFAR-10/CIFAR-100 is 32 32.

E.2 Model architectures

Table 6 summarizes the CNN encoder architectures used for deterministic verification. Each layer (except the last linear layer) is followed by ReLU activation function. **Based** and **Deep** are used in Wang et al. (2021), **CNN-A** and **CNN-B** are used in Dathathri et al. (2020). To study the sensitivity of featrue dimension, the ouput dimension of encoder can be changed from 50 to 250 on **CNN-B**.

Table 6: Model structures used in our experiments. For example, Conv(1, 8, 4) stands for a conventional layer with 1 input channel, 8 output channels and a 4 4 kernel. Linear(754, 100) stands for a fully connected layer with 754 input features and 100 output features.

Datasets	**Model name**	**Encoder structure**
Mnist	Base	Conv(1, 8, 4) - Conv(8, 16, 4) - Linear(784, 100)
	CNN-A	Conv(1, 16, 4) - Conv(16, 32, 4) - Linear(1568, 100)
CIFAR-10	Base	Conv(3, 8, 4) - Conv(8, 16, 4) - Linear(1024, 100)
	Deep	Conv(3, 8, 4) - Conv(8, 8, 3) - Conv(8, 8, 3) - Conv(8, 8, 4) - Linear(412, 100)
	CNN-A	Conv(3, 16, 4) - Conv(16, 32, 4) - Linear(2048, 100)
	CNN-B	Conv(3, 32, 5) - Conv(32, 128, 4) - Linear(8192, 100)

E.3 Training setup

All NNs are trained with verification-agnostic setting (Dathathri et al., 2020), which means without using any tricks to promote verifiability. For deterministic verification, we use the model mentioned in Appendix E.2 as the base encoder network and 2-layer multi-layer perceptron as the projection head (Chen et al., 2020). We set the step size of instance-wise attack $\alpha = 0.007$, the number of PGD maximize iteration $K = 10$. For the rest, we follow the similar setup of SimCLR (Chen et al., 2020) and RoCL (Kim et al., 2020).

For optimization, we train the encoder with 500 epochs under Adam (Kingma and Ba, 2015) optimizer with the learning rate of 0.001. For the learning rate scheduling, the learning rate is dropped by a factor of 10 for every 100 epochs. The batch size in training is 256.

E.4 Evaluation setup

Linear evaluation We train the downstream linear layer on the top of the frozen encoder, and the training images are clean. We train the linear layer for 100 epochs with the learning

rate of 0.001, and use the cross-entropy loss. The learning rate is dropped by a factor of 10 for every 50 epochs.

Robust test To evaluate the adversarial robustness, we use white-box project gradient descent (PGD) attack. We set ℓ_∞ attack with 20 iteration steps. We set $\epsilon_{test} = 0, 0.15, 0.2$ for Mnist, and $\epsilon_{test} = 0, {}^{8}/_{255}, {}^{16}/_{255}$ for CIFAR-10/CIFAR-100.

Incomplete verifiers If not special specified, CBC (β-CROWN (Wang et al., 2021) for CL) working as an incomplete verifier uses three minutes for each image.

E.5 Training efficiency

Our experiments are conducted on a Ubuntu 64-Bit Linux workstation, having 10-core Intel Xeon Silver CPU (2.20 GHz) and NVIDIA GeForce RTX 3090 GPU with 24GB graphics memory. For adversarial contrastive learning on base encoder, it takes about 10 hours to train 500 epochs on Mnist **CNN-A** and CIFAR-10 **CNN-B** with a single GPU. And it takes about 120 minutes to train 100 epochs on downstream linear layer.

References

Jean-Baptiste Alayrac, Jonathan Uesato, Po-Sen Huang, Alhussein Fawzi, Robert Stanforth, and Pushmeet Kohli. Are labels required for improving adversarial robustness? In *NeurIPS*, pages 12192–12202, 2019.

Ross Anderson, Joey Huchette, Will Ma, Christian Tjandraatmadja, and Juan Pablo Vielma. Strong mixed-integer programming formulations for trained neural networks. *Math. Program.*, 183(1):3–39, 2020.

Maksym Andriushchenko, Francesco Croce, Nicolas Flammarion, and Matthias Hein. Square attack: A query-efficient black-box adversarial attack via random search. In *ECCV*, volume 12368, pages 484–501, 2020.

Rudy Bunel, Ilker Turkaslan, Philip H. S. Torr, Pushmeet Kohli, and Pawan Kumar Mudigonda. A unified view of piecewise linear neural network verification. In *NeurIPS*, pages 4795–4804, 2018.

Nicholas Carlini and David A. Wagner. Towards evaluating the robustness of neural networks. In *SP*, pages 39–57, 2017.

Yair Carmon, Aditi Raghunathan, Ludwig Schmidt, John C. Duchi, and Percy Liang. Unlabeled data improves adversarial robustness. In *NeurIPS*, pages 11190–11201, 2019.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey E. Hinton. A simple framework for contrastive learning of visual representations. In *ICML*, volume 119, pages 1597–1607, 2020.

Jeremy M. Cohen, Elan Rosenfeld, and J. Zico Kolter. Certified adversarial robustness via randomized smoothing. In *ICML*, volume 97, pages 1310–1320, 2019.

Stuart Coles, Joanna Bawa, Lesley Trenner, and Pat Dorazio. *An introduction to statistical modeling of extreme values*, volume 208. Springer, 2001.

Francesco Croce and Matthias Hein. Minimally distorted adversarial examples with a fast adaptive boundary attack. In *ICML*, volume 119, pages 2196–2205, 2020a.

Francesco Croce and Matthias Hein. Reliable evaluation of adversarial robustness with an ensemble of diverse parameter-free attacks. In *ICML*, volume 119, pages 2206–2216, 2020b.

Francesco Croce, Maksym Andriushchenko, Vikash Sehwag, Edoardo Debenedetti, Nicolas Flammarion, Mung Chiang, Prateek Mittal, and Matthias Hein. Robustbench: a standardized adversarial robustness benchmark. In *NeurIPS*, 2021.

Sumanth Dathathri, Krishnamurthy Dvijotham, Alexey Kurakin, Aditi Raghunathan, Jonathan Uesato, Rudy Bunel, Shreya Shankar, Jacob Steinhardt, Ian J. Goodfellow, Percy Liang, and Pushmeet Kohli. Enabling certification of verification-agnostic networks via memory-efficient semidefinite programming. In *NeurIPS*, 2020.

Carl Doersch, Abhinav Gupta, and Alexei A. Efros. Unsupervised visual representation learning by context prediction. In *ICCV*, pages 1422–1430, 2015.

Krishnamurthy (Dj) Dvijotham, Jamie Hayes, Borja Balle, J. Zico Kolter, Chongli Qin, András György, Kai Xiao, Sven Gowal, and Pushmeet Kohli. A framework for robustness certification of smoothed classifiers using f-divergences. In *ICLR*, 2020.

Rüdiger Ehlers. Formal verification of piece-wise linear feed-forward neural networks. In *Automated Technology for Verification and Analysis*, volume 10482, pages 269–286, 2017.

Lijie Fan, Sijia Liu, Pin-Yu Chen, Gaoyuan Zhang, and Chuang Gan. When does contrastive learning preserve adversarial robustness from pretraining to finetuning? In *NeurIPS*, 2021.

Xavier Gibert, Vishal M. Patel, and Rama Chellappa. Sequential score adaptation with extreme value theory for robust railway track inspection. In *ICCV*, pages 131–138, 2015.

Spyros Gidaris, Praveer Singh, and Nikos Komodakis. Unsupervised representation learning by predicting image rotations. In *ICLR*, 2018.

Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. In *ICLR*, 2015.

Sven Gowal, Po-Sen Huang, Aäron van den Oord, Timothy Mann, and Pushmeet Kohli. Self-supervised adversarial robustness for the low-label, high-data regime. In *ICLR*, 2021.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, pages 770–778, 2016.

Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross B. Girshick. Momentum contrast for unsupervised visual representation learning. In *CVPR*, pages 9726–9735, 2020.

Chih-Hui Ho and Nuno Vasconcelos. Contrastive learning with adversarial examples. In *NeurIPS*, 2020.

Ziyu Jiang, Tianlong Chen, Ting Chen, and Zhangyang Wang. Robust pre-training by adversarial contrastive learning. In *NeurIPS*, 2020.

Longlong Jing and Yingli Tian. Self-supervised visual feature learning with deep neural networks: A survey. *IEEE Trans. Pattern Anal. Mach. Intell.*, 43(11):4037–4058, 2021.

Guy Katz, Clark W. Barrett, David L. Dill, Kyle Julian, and Mykel J. Kochenderfer. Reluplex: An efficient SMT solver for verifying deep neural networks. In *CAV*, volume 10426, pages 97–117, 2017.

Minseon Kim, Jihoon Tack, and Sung Ju Hwang. Adversarial self-supervised contrastive learning. In *NeurIPS*, 2020.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *ICLR*, 2015.

Alex Krizhevsky and Geoffrey Hinton. Learning multiple layers of features from tiny images. Technical report, Citeseer, 2009.

Yann LeCun and Corinna Cortes. Mnist handwritten digit database. *ATT Labs [Online]. Available: http://yann.lecun.com/exdb/mnist*, 2010.

Mathias Lécuyer, Vaggelis Atlidakis, Roxana Geambasu, Daniel Hsu, and Suman Jana. Certified robustness to adversarial examples with differential privacy. In *SP*, pages 656–672, 2019.

Guang-He Lee, Yang Yuan, Shiyu Chang, and Tommi S. Jaakkola. Tight certificates of adversarial robustness for randomly smoothed classifiers. In *NeurIPS*, pages 4911–4922, 2019.

Linyi Li, Xiangyu Qi, Tao Xie, and Bo Li. Sok: Certified robustness for deep neural networks. *CoRR*, abs/2009.04131, 2020.

Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. In *ICLR*, 2018.

Nicholas Metropolis and Stanislaw Ulam. The monte carlo method. *Journal of the American statistical association*, 44(247):335–341, 1949.

Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, and Pascal Frossard. Deepfool: A simple and accurate method to fool deep neural networks. In *CVPR*, pages 2574–2582, 2016.

Mehdi Noroozi and Paolo Favaro. Unsupervised learning of visual representations by solving jigsaw puzzles. In *ECCV*, volume 9910, pages 69–84, 2016.

Kento Nozawa, Pascal Germain, and Benjamin Guedj. Pac-bayesian contrastive unsupervised representation learning. In *UAI*, volume 124, pages 21–30, 2020.

Ethan M. Rudd, Lalit P. Jain, Walter J. Scheirer, and Terrance E. Boult. The extreme value machine. *IEEE Trans. Pattern Anal. Mach. Intell.*, 40(3):762–768, 2018.

Hadi Salman, Jerry Li, Ilya P. Razenshteyn, Pengchuan Zhang, Huan Zhang, Sébastien Bubeck, and Greg Yang. Provably robust deep learning via adversarially trained smoothed classifiers. In *NeurIPS*, pages 11289–11300, 2019.

Nikunj Saunshi, Orestis Plevrakis, Sanjeev Arora, Mikhail Khodak, and Hrishikesh Khandeparkar. A theoretical analysis of contrastive unsupervised representation learning. In *ICML*, volume 97, pages 5628–5637, 2019.

Walter J. Scheirer, Anderson Rocha, Ross J. Micheals, and Terrance E. Boult. Meta-recognition: The theory and practice of recognition score analysis. *IEEE Trans. Pattern Anal. Mach. Intell.*, 33(8):1689–1695, 2011.

Walter J. Scheirer, Neeraj Kumar, Peter N. Belhumeur, and Terrance E. Boult. Multi-attribute spaces: Calibration for attribute fusion and similarity search. In *CVPR*, pages 2933–2940, 2012.

Ludwig Schmidt, Shibani Santurkar, Dimitris Tsipras, Kunal Talwar, and Aleksander Madry. Adversarially robust generalization requires more data. In *NeurIPS*, pages 5019–5031, 2018.

Zhouxing Shi, Yihan Wang, Huan Zhang, J. Zico Kolter, and Cho-Jui Hsieh. Efficiently computing local lipschitz constants of neural networks via bound propagation. In *NeurIPS*, 2022.

Aman Sinha, Hongseok Namkoong, and John C. Duchi. Certifying some distributional robustness with principled adversarial training. In *ICLR*, 2018.

Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian J. Goodfellow, and Rob Fergus. Intriguing properties of neural networks. In *ICLR*, 2014.

Yonglong Tian, Dilip Krishnan, and Phillip Isola. Contrastive multiview coding. In *ECCV*, volume 12356, pages 776–794, 2020.

Vincent Tjeng, Kai Yuanqing Xiao, and Russ Tedrake. Evaluating robustness of neural networks with mixed integer programming. In *ICLR*, 2019.

Jonathan Uesato, Brendan O'Donoghue, Pushmeet Kohli, and Aäron van den Oord. Adversarial risk and the dangers of evaluating against weak attacks. In *ICML*, volume 80, pages 5032–5041, 2018.

Shiqi Wang, Huan Zhang, Kaidi Xu, Xue Lin, Suman Jana, Cho-Jui Hsieh, and J. Zico Kolter. Beta-crown: Efficient bound propagation with per-neuron split constraints for complete and incomplete neural network verification. In *NeurIPS*, 2021.

Tsui-Wei Weng, Huan Zhang, Hongge Chen, Zhao Song, Cho-Jui Hsieh, Luca Daniel, Duane S. Boning, and Inderjit S. Dhillon. Towards fast computation of certified robustness for relu networks. In *ICML*, volume 80, pages 5273–5282, 2018a.

Tsui-Wei Weng, Huan Zhang, Pin-Yu Chen, Jinfeng Yi, Dong Su, Yupeng Gao, Cho-Jui Hsieh, and Luca Daniel. Evaluating the robustness of neural networks: An extreme value theory approach. In *ICLR*, 2018b.

Eric Wong and J. Zico Kolter. Provable defenses against adversarial examples via the convex outer adversarial polytope. In *ICML*, volume 80, pages 5283–5292, 2018.

Eric Wong, Frank R. Schmidt, Jan Hendrik Metzen, and J. Zico Kolter. Scaling provable adversarial defenses. In *NeurIPS*, pages 8410–8419, 2018.

Zhirong Wu, Yuanjun Xiong, Stella X. Yu, and Dahua Lin. Unsupervised feature learning via non-parametric instance discrimination. In *CVPR*, pages 3733–3742, 2018.

Kaidi Xu, Zhouxing Shi, Huan Zhang, Yihan Wang, Kai-Wei Chang, Minlie Huang, Bhavya Kailkhura, Xue Lin, and Cho-Jui Hsieh. Automatic perturbation analysis for scalable certified robustness and beyond. In *NeurIPS*, 2020.

Greg Yang, Tony Duan, J. Edward Hu, Hadi Salman, Ilya P. Razenshteyn, and Jerry Li. Randomized smoothing of all shapes and sizes. In *ICML*, volume 119, pages 10693–10705, 2020.

Runtian Zhai, Tianle Cai, Di He, Chen Dan, Kun He, John E. Hopcroft, and Liwei Wang. Adversarially robust generalization just requires more unlabeled data. *CoRR*, abs/1906.00555, 2019.

Runtian Zhai, Chen Dan, Di He, Huan Zhang, Boqing Gong, Pradeep Ravikumar, Cho-Jui Hsieh, and Liwei Wang. MACER: attack-free and scalable robust training via maximizing certified radius. In *ICLR*, 2020.

Huan Zhang, Tsui-Wei Weng, Pin-Yu Chen, Cho-Jui Hsieh, and Luca Daniel. Efficient neural network robustness certification with general activation functions. In *NeurIPS*, pages 4944–4953, 2018.