RETHINKING CERTIFIED TRAINING EFFICIENCY: A BENCHMARKING FRAMEWORK AND EMPIRICAL INVESTIGATION FOR RANDOMIZED SMOOTHING

Master Thesis

by

Ali Salaheldin Ali Ahmed

1. Referee:
Dr. Kevin Baum

2. Referee:
Dr. Hanwei Zhang

July 3, 2026

Eidesstattliche Erklärung

Hiermit erkläre ich, dass ich die vorliegende Arbeit selbstständig und ohne die Beteiligung dritter Personen verfasst habe, und dass ich keine anderen als die angegebenen Quellen und Hilfsmittel benutzt habe. Alle Stellen der Arbeit, die wörtlich oder sinngemäß aus Veröffentlichungen oder aus anderweitigen fremden Äußerungen entnommen wurden, sind als solche kenntlich gemacht. Insbesondere bestätige ich hiermit, dass ich bei der Erstellung der nachfolgenden Arbeit mittels künstlicher Intelligenz betriebene Software (z. B. ChatGPT) ausschließlich für folgende zulässige Teilaufgaben: **Textumformulierung und Überarbeitung**, und nicht zur Bearbeitung der in der Arbeit aufgeworfenen Fragestellungen zu Hilfe genommen habe. Mir ist bewusst, dass der Verstoß gegen diese Versicherung zum Nichtbestehen der Prüfung bis hin zum Verlust des Prüfungsanspruchs führen kann.

Declaration of Original Authorship

I hereby declare that this thesis is my own original work and was completed independently, without unauthorized assistance or unacknowledged sources. Any content derived from publications or other external sources, whether quoted verbatim or paraphrased, has been duly acknowledged and clearly marked as such. I hereby confirm that, in preparing the following thesis, I have used artificial intelligence-based software (such as ChatGPT) solely for the following permitted tasks: **text rewriting and revision**, and not to address the core research questions presented in this thesis. I acknowledge that any breach of this declaration may result in failing the examination and, in severe cases, the right to be examined may be revoked.

Ort/Place, Datum/Date	Unterschrift/Signature

To my family, for always giving me a place to return to and the strength to keep moving forward.
And to the special someone who has walked this path beside me: thank you for your unwavering encouragement and for reminding me what matters most.

Abstract

Safety-critical deep learning requires formal reliability guarantees that extend beyond empirical robustness. Randomized Smoothing (RS) has emerged as the dominant certification framework for modern deep neural networks, yet the computational cost of certified training remains a major barrier to its practical adoption. This thesis develops an efficiency-first perspective on certified training by combining systematic benchmarking with hypothesis-driven investigation into the sources of computational overhead.

We first introduce BenchmarkCtrs, a modular benchmarking framework that jointly evaluates classification accuracy, certified robustness, and computational efficiency under a standardized experimental protocol. Our empirical evaluation of representative certified training methods reveals substantial efficiency–robustness trade-offs and identifies repeated noise sampling as the dominant computational bottleneck across the evaluated methods.

Building on these findings, we investigate two complementary hypotheses regarding the origins of certified training inefficiency: (i) we evaluate LossProj, a parameter-space gradient projection method that mitigates optimization conflicts between classification and robustness objectives. While LossProj improves clean accuracy in low-noise settings, these gains are offset by increased computational cost and do not generalize to stronger smoothing regimes, and (ii) we adapt Selective Backpropagation (SB) to robustness-aware sample selection, investigating whether certified training performs redundant computation. On ImageNet, SB reduces training time by 30% (approximately 60.4 GPU-hours) while incurring only minor reductions in certified robustness under the evaluated configuration.

These findings suggest that reducing redundant computation offers a more promising avenue for improving certified training efficiency than optimization-centric interventions. More broadly, this thesis establishes a systematic framework for evaluating certified training efficiency and demonstrates that computational cost should be treated as a first-class objective alongside certified robustness.

Contents

Chapter 1.

Introduction

The deployment of deep neural networks in safety-critical domains, from autonomous navigation to medical diagnostics, has expanded the objectives of machine learning from predictive accuracy alone to include verifiable reliability. While these models excel at pattern recognition, they remain fundamentally vulnerable to *adversarial perturbations*: small, intentional changes to input data that trigger catastrophic misclassifications. In regulated industries where failure carries significant human or economic costs, this vulnerability represents a primary barrier to the safe adoption of artificial intelligence. Addressing this risk through formal certification, however, introduces a severe conflict between the need for rigorous safety guarantees and the practical constraints of computational cost.

In this thesis, we argue that computational efficiency and robust certification are co-primary goals in scalable certified training. We develop a systematic approach to analyze the inherent trade-off between them and investigate how optimization-level and data-level strategies affect that trade-off. We further argue that the primary source of inefficiency in certified training arises not from optimization quality, but from the inherent data redundancy induced by a shared computational structure: multi-pass noise sampling.

1. Motivation

Practically addressing the vulnerability of neural networks requires a formal, certifiable, and performant approach to safety. While early research focused on empirical defenses, these methods frequently fail against newer adversaries and lack formal safety guarantees. In contrast, certified defenses provide a provable radius within which a classifier's prediction is guaranteed to remain constant. However, traditional certification methods, such as those based on interval bound propagation or semidefinite programming, suffer from limited scalability, often restricting application to small-scale models or simplified datasets like MNIST and CIFAR-10.

Randomized Smoothing (RS) [6] addresses this scalability wall. By converting a model's statistical consistency under random perturbations into a formal measure of decision-boundary stability, RS enables the calculation of a rigorous safety radius around individual inputs for large-scale architectures on high-dimensional datasets like ImageNet. However, the efficacy of the smoothed classifier depends heavily on specialized certified training methods, including data augmentation [6, 32], adversarial training [31, 16], and regularized risk minimization [48, 9, 15]. While both certification and certified training incur substantial computational cost, existing work has primarily focused on accelerating certification [5, 33, 43, 35, 45]. In contrast, this thesis is concerned exclusively with the efficiency of certified training.

Although the literature has produced increasingly complex certified training objectives, a critical research gap persists: *the computational cost of these methods is treated as a secondary engineering concern rather than a fundamental design pillar.* The multi-pass noise sampling required by these algorithms introduces a severe efficiency bottleneck, increasing training time by up to an order of magnitude. This inefficiency stems from a structural property of certified training: the repeated estimation of stochastic gradients under noise perturbations. As a result, it limits the applicability of certified training to modern workflows involving large-scale architectures or dynamic data regimes that require frequent retraining on continuously evolving datasets.

Because existing work primarily evaluates robustness while giving comparatively little attention to computational cost, there is no systematic understanding of the trade-offs between optimization objectives and resource consumption. We address this gap by placing computational efficiency at the forefront of certified training research, establishing systematic protocols for evaluating efficiency–robustness trade-offs, developing reproducible benchmarking methodologies, and investigating method-agnostic approaches that reveal the fundamental algorithmic and data-centric sources of computational inefficiency.

2. Problem Statement

This thesis addresses the computational efficiency challenges of certified training under randomized smoothing. Although certified training has achieved substantial progress in improving provable robustness, its computational requirements remain a major obstacle to practical deployment. Addressing this challenge requires moving beyond robustness alone and examining how certified training algorithms utilize computational resources throughout optimization.

Existing research has largely framed certified training as a robustness optimization problem, where success is measured primarily through certified accuracy and

certified radius. Under this perspective, computational cost is typically treated as a secondary concern: an unavoidable consequence of obtaining stronger robustness guarantees. This view, however, provides only a partial understanding of the practical scalability of certified training. In practice, robustness must be balanced against the time and computational resources required to obtain it. Models that require excessive training time or computational infrastructure are difficult to deploy, update, and maintain, particularly in safety-critical systems operating under strict hardware and latency constraints. Slow retraining also limits the ability to adapt models to newly available data or emerging threats, ultimately undermining the practical adoption of certified robustness.

This thesis argues that computational efficiency should be treated as a first-class objective alongside certified robustness and predictive performance. Rather than viewing the computational overhead of certified training as an unavoidable cost, we investigate whether it arises from identifiable and potentially addressable sources. In particular, we study two complementary hypotheses: (i) structural conflicts between the optimization objectives, and (ii) redundant computation across training samples.

Adopting this perspective exposes two broader gaps in the current research landscape.

(i) **Lack of systematic efficiency evaluation**. Existing work provides little support for comparing certified training methods from a computational perspective, making it difficult to distinguish algorithmic efficiency from implementation choices or hardware effects.

(ii) **Limited understanding of computational bottlenecks**. While the computational overhead of certified training is well recognized, comparatively little attention has been given to understanding where it originates or how it can be reduced without compromising certified robustness, particularly with respect to optimization conflicts and redundant computation.

3. Contributions

This thesis contributes a systematic perspective on the computational efficiency of certified training under randomized smoothing through standardized benchmarking and hypothesis-driven algorithmic investigation.

- **Unified Evaluation and Benchmark**. e develop BenchmarkCtrs, a modular benchmarking framework that jointly evaluates classification accuracy, certified robustness, and computational efficiency under a standardized protocol measuring GPU time, throughput, memory consumption, and convergence

behavior. Using this framework, we present a systematic comparison of five representative certified training methods: Gaussian Augmentation, SmoothAdv, MACER, ADRE, and Consistency. Our evaluation reveals previously unexplored efficiency–robustness trade-offs, showing that although Consistency achieves the strongest robustness, Gaussian Augmentation offers a more balanced trade-off on larger datasets due to the computational overhead of multi-pass noise sampling.

- **Empirical Investigation of Computational Bottlenecks**. Building on the benchmarking results, we investigate two complementary hypotheses regarding the sources of computational inefficiency in certified training.

  Structural Objective Conflicts (Negative Result). We evaluate parameter-space gradient projection (LOSSPROJ) to determine whether mitigating conflicts between classification and robustness objectives improves efficiency. Although gradient decoupling benefits optimization in specific settings, it does not consistently produce a favorable efficiency–robustness trade-off, suggesting that optimization conflicts are not the dominant source of inefficiency.

  Redundant Computation (Positive Result). We adapt Selective Backpropagation (SB) to robustness-aware sample selection and distributed training. Our results show that up to 74% of backward passes on ImageNet can be skipped while largely preserving certified robustness, identifying data redundancy as a more promising direction for improving certified training efficiency.

4. Thesis Organization

The remainder of this thesis is structured as follows:

- **Chapter 2** provides the theoretical foundation for adversarial robustness, randomized smoothing, certified training, and the computational challenges inherent in deep learning training.
- **Chapter 3** introduces the BENCHMARKCTRS framework and presents a controlled comparative analysis of existing certified training baselines.
- **Chapter 4** evaluates Loss Projection and Selective Backpropagation as experimental approaches for investigating the efficiency–robustness trade-off.
- **Chapter 5** summarizes our findings and discusses potential avenues for future research in scalable certified robustness.

Chapter 2.

Background and Related Work

This chapter introduces the background needed to understand the challenges of training models with certified robustness guarantees, with a particular focus on efficiency and scalability. The chapter follows the perspective of this thesis rather than attempting a full survey: although certified robustness methods have advanced substantially, the computational cost of training models with strong certified guarantees remains a major obstacle to practical adoption.

The discussion begins by introducing the foundations of adversarial robustness, together with the methods used to evaluate robustness and the training approaches developed to achieve it (Section 1). It then reviews robustness evaluation methods, culminating in randomized smoothing as the statistical certification framework that underpins the certified training methods studied throughout this thesis (Sections 2 and 3). Finally, the chapter introduces concepts of computational and data efficiency in deep learning (Section 5), establishing the broader perspective used to analyze the efficiency–robustness trade-off in certified training.

1. Adversarial Robustness

Deep neural networks have achieved strong predictive performance across many tasks. In many applications, however, predictive accuracy alone is insufficient. Models deployed in safety-critical or security-sensitive settings must also behave reliably when their inputs are subject to small variations, whether arising from natural noise, measurement errors, or deliberate manipulation. This requirement motivates the study of adversarial robustness, which characterizes the stability of model predictions under bounded input perturbations.

This section introduces the concept of adversarial robustness and the foundations used to evaluate it. These concepts form the basis of the certification methods and training procedures discussed throughout the remainder of this thesis.

Adversarial robustness describes the stability of a model's predictions under bounded perturbations of its input. While standard learning objectives emphasize average performance over a data distribution, robustness focuses on the local behavior of a model around individual inputs. A robust classifier should maintain its prediction even when an adversary is allowed to introduce small but worst-case perturbations.

Formally, a classifier $f : \mathcal{X} \to \mathcal{Y}$ is robust at an input $\mathbf{x} \in \mathcal{X}$ with respect to a perturbation budget $\epsilon > 0$ if its prediction remains invariant for any input $\mathbf{x}'$ within an allowable perturbation region $\mathcal{S}(\mathbf{x}, \epsilon) \subseteq \mathcal{X}$:

$$f(\mathbf{x}) = f(\mathbf{x}'), \quad \forall\, \mathbf{x}' \in \mathcal{S}(\mathbf{x}, \epsilon). \tag{2.1}$$

The typical choice for the geometry of the allowable perturbation region is based on the ℓ_p–norm such that,

$$\mathcal{B}_p(\mathbf{x}, \epsilon) = \{\mathbf{x}' \mid \|\mathbf{x}' - \mathbf{x}\|_p \leq \epsilon\}. \tag{2.2}$$

The ℓ_∞–norm (limiting maximum pixel-wise change) and the ℓ_0–norm (restricting the number of modified pixels) have been studied in the broader robustness literature. In our analysis of smoothing-based defenses however, we focus on the standard choice of the ℓ_2 norm, $\|\boldsymbol{\delta}\|_2 = \sqrt{\sum_i \boldsymbol{\delta}_i^2}$. We acknowledge that ℓ_p–norms are incomplete proxies for perceptual similarity: a small ℓ_p perturbation in a specific direction can be highly visible to humans, while larger perturbations following semantic shifts often go undetected. Nonetheless, the ℓ_p–norm remains the dominant paradigm for defining the bounds of the "adversarial region."

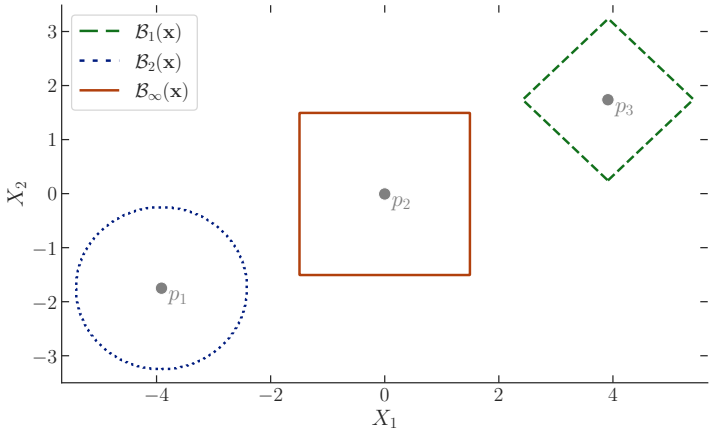


Figure 2.1.: The shape of the allowable perturbation region (the "adversarial ball") under select ℓ_p threat models.

1.1. Adversarial Attacks: Empirical Measures of Robustness

Adversarial attacks provide the most common approach for evaluating robustness in practice. Given an input, an attack attempts to identify a perturbation within the allowed threat model that causes the classifier to change its prediction. If such a perturbation exists, the resulting input is referred to as an *adversarial example*. Modern attacks typically formulate this search as an optimization problem, seeking perturbations that remain visually similar to the original input while inducing incorrect predictions.

Attack objectives can vary depending on the adversary's goal. In an *untargeted attack*, any incorrect prediction constitutes a successful attack. In a *targeted attack*, the adversary seeks to force the classifier to predict a specific target label chosen in advance.

Although adversarial attacks are valuable diagnostic tools, they provide only a partial view of robustness. A successful attack shows that a model is not robust by producing a concrete counterexample to the robustness property. The converse, however, does not hold: failure to find an adversarial example only indicates that a particular attack was unsuccessful. It does not establish that no such perturbation exists.

As a result, empirical robustness estimates are inherently tied to the strength of the attacks used for evaluation. This limitation motivates the development of certification methods, which seek to provide formal robustness guarantees rather than evidence obtained through adversarial search alone.

1.2. Empirical Defenses and Their Limitations

The discovery of adversarial vulnerabilities has motivated the development of various defense strategies aimed at improving the stability of model predictions under perturbations. Early approaches attempted to prevent attackers from exploiting model gradients rather than fundamentally improving robustness. For example, Defensive Distillation [28] was designed to reduce the sensitivity of model outputs by training on softened predictions from a teacher network. However, Athalye, Carlini, and Wagner [2] showed that many such approaches relied on *obfuscated gradients*, where the apparent robustness resulted from the failure of specific attack algorithms rather than from a genuinely more robust decision boundary.

Later approaches improved robustness by incorporating adversarial examples directly into the training process. *Adversarial Training* formulates robustness improvement as a min-max optimization problem:

$$\arg\min_{\boldsymbol{\theta}} \mathop{\mathbb{E}}_{(\mathbf{x},y)\sim\mathcal{D}} \left[\max_{\|\delta\|_p \leq \epsilon} \mathcal{L}\left(f_{\boldsymbol{\theta}}\left(\mathbf{x}+\boldsymbol{\delta}\right), y\right) \right], \tag{2.3}$$

where the inner maximization searches for a challenging perturbation using an adversarial attack algorithm, such as Projected Gradient Descent (PGD) [24]. By explicitly optimizing against adversarial examples during training, adversarial training can substantially improve robustness against the attacks considered during evaluation.

However, these empirical defenses remain fundamentally dependent on the attack procedures used to assess them. Although strong attacks provide evidence of robustness, they cannot guarantee that no stronger or previously unknown attack exists within the same perturbation budget. This limitation is particularly important for safety-critical applications, where the absence of observed failures is insufficient evidence of reliable behavior. Consequently, the field has moved toward certified robustness techniques, which aim to provide formal guarantees on model behavior under bounded perturbations.

2. Robustness Verification: Balancing Guarantees and Scalability

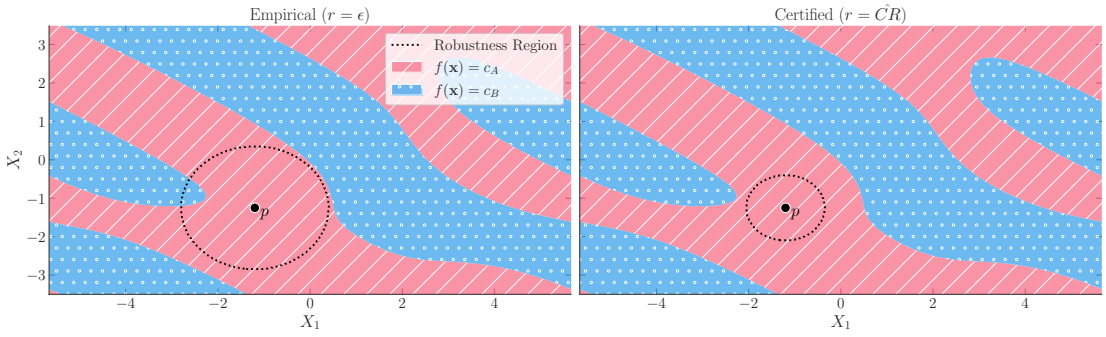


Figure 2.2.: **The distinction between empirical and certified robustness**. The safe region suggested by empirical defenses (left) depends on the attacks used during evaluation and is not guaranteed to exclude all adversarial examples. In contrast, certified defenses (right) compute a guaranteed safe region of radius $\hat{CR}$ that contains no adversarial examples. This guarantee is conservative, so the certified region is often smaller than the model's actual region of robustness.

While empirical defenses can reveal individual failures of a model, they cannot establish that no violating perturbation exists. *Certified robustness* addresses this limitation by providing a mathematical guarantee that a classifier's prediction remains unchanged for all perturbations within a specified budget. The conceptual distinction between empirical and certified robustness is illustrated in Figure 2.2: empirical methods provide only heuristic evidence of safety, whereas certified

methods guarantee that no adversarial example exists within the certified region. However, obtaining such guarantees requires reasoning about the behavior of the model over an entire region of possible inputs, making computational scalability a central challenge. Following the categorization of Huang et al. [13], verification approaches can be broadly distinguished by the strength of their guarantees, ranging from complete methods that provide exact guarantees to relaxed and randomized approaches that trade precision for improved scalability.

2.1. Complete Verification Methods

Complete verifiers provide the strongest form of guarantee by solving the robustness problem exactly. If the property holds, these methods produce a certificate proving that no valid perturbation can change the prediction. Otherwise, they return a concrete counterexample. Approaches based on Satisfiability Modulo Theories (SMT), such as Reluplex [19] and Marabou [20], as well as Mixed-Integer Linear Programming (MILP) formulations [41], achieve this by explicitly modeling the network's behavior.

The precision of complete verification comes at a significant computational cost. Since neural network verification is NP-hard in the general case, exact methods may require exploring an exponentially growing number of possible activation states as model complexity increases [19]. Consequently, complete verification remains limited to relatively small models and datasets [23], preventing its direct application to modern large-scale architectures.

2.2. Incomplete Verification Methods

Incomplete verification methods improve scalability by relaxing the exact verification problem. Rather than computing the precise set of possible model outputs, they derive a sound over-approximation of this set. If the entire approximation satisfies the desired robustness property, the model is certified. However, the relaxation may be too conservative, causing genuinely robust inputs to remain uncertified.

The primary motivation behind these approaches is reducing the computational burden of certification. *Interval Bound Propagation* (IBP) [11] provides an efficient approximation by propagating interval bounds through the network. More precise approaches, including linear relaxation methods such as CROWN [50] and abstract interpretation frameworks such as AI2 [10], achieve tighter bounds by representing dependencies between neurons through richer geometric domains. Convex relaxation methods further integrate these approximations into the training process, enabling optimization against worst-case losses [46].

Despite these improvements, the trade-off between certification tightness and computational efficiency remains unresolved. Hybrid approaches such as CROWN-

IBP [51] combine efficient and precise bounds to improve scalability, but deterministic verification methods continue to face challenges when applied to large and complex models. These limitations motivate alternative certification frameworks based on statistical guarantees, leading to the randomized smoothing approach discussed next.

2.3. Statistical Verification Methods

The scalability limitations of deterministic verification approaches have motivated alternative methods that provide robustness guarantees through statistical reasoning. Instead of exhaustively analyzing the complete behavior of a classifier, statistical verification methods establish guarantees based on probabilistic bounds derived from randomized model evaluations. These approaches trade the exact guarantees of complete verification for improved scalability, enabling certification of larger and more complex models.

A prominent example of this paradigm is *randomized smoothing* [6], which certifies robustness by transforming a classifier into a smoothed version whose predictions can be analyzed statistically. Due to its scalability and applicability to modern deep learning architectures, randomized smoothing has become one of the most widely studied approaches for statistical robustness certification. The next section introduces this framework in detail, as it forms the foundation of the certified training methods investigated in this thesis.

3. Randomized Smoothing Certification

Introduced by Cohen, Rosenfeld, and Kolter [6], randomized smoothing has emerged as the most scalable framework among statistical verification methods for certifying modern deep neural networks. Its ability to scale to large architectures has made it the foundation of many certified training approaches. This section introduces the randomized smoothing framework, the certification procedures used to obtain robustness guarantees, and the concepts required to understand the training methods discussed in the following section.

3.1. The Smooth Classifier

Randomized smoothing constructs a *smoothed classifier* (g) from an arbitrary base classifier $f : \mathcal{X} \to \mathcal{Y}$. Rather than predicting the class assigned to a single input, the smoothed classifier predicts the class that is most likely to be produced by the base classifier under random perturbations of the input. Using isotropic Gaussian

noise, the smoothed classifier is defined as

$$g(\mathbf{x}) = \arg\max_{c \in \mathcal{Y}} \mathbb{P}_{\boldsymbol{\delta} \sim \mathcal{N}(\mathbf{0}, \sigma^2 \boldsymbol{I})}[f(\mathbf{x} + \boldsymbol{\delta}) = c]. \tag{2.4}$$

The noise level σ is a key hyperparameter that determines the smoothing distribution and controls the trade-off between clean accuracy and certified robustness. Larger values generally increase robustness at the cost of predictive accuracy on unperturbed inputs.

The central idea behind randomized smoothing is that averaging predictions over many noisy perturbations produces a classifier whose behavior is substantially more stable than that of the underlying base classifier. This stability enables robustness guarantees to be derived directly from the class probabilities induced by the noise distribution, without requiring explicit reasoning over the network's decision boundary.

Cohen, Rosenfeld, and Kolter [6] derive a provable robustness bound for g based on the Neyman-Pearson Lemma. They show that if the most probable class $c_A = g(\mathbf{x})$ occurs with probability p_A, and the second most probable class $c_B = \arg\max_{c \in \mathcal{Y} \setminus \{c_A\}} \mathbb{P}_{\boldsymbol{\delta} \sim \mathcal{N}(\mathbf{0}, \sigma^2 \boldsymbol{I})}[f(\mathbf{x} + \boldsymbol{\delta}) = c]$ occurs with probability p_B, then it follows that:

$$g(\mathbf{x}') = c_A, \qquad \forall\, \mathbf{x}' \in \mathcal{B}_2(\mathbf{x}, CR),$$

where CR is the certified radius defined by:

$$CR = \frac{\sigma}{2}\left(\Phi^{-1}(p_A) - \Phi^{-1}(p_B)\right), \tag{2.5}$$

and Φ^{-1} denotes the inverse cumulative distribution function of the standard normal distribution. In the standard binary case, or when utilizing an upper bound $\overline{p_B} \leq 1 - p_A$, this simplifies to $CR = \sigma\Phi^{-1}(p_A)$.

The certified radius therefore quantifies the largest ℓ_2 perturbation for which the prediction of the smoothed classifier is provably invariant. Larger separations between the probabilities of the most likely and competing classes yield stronger robustness guarantees, while increasing the smoothing parameter σ enlarges the potential certification radius at the cost of reduced clean accuracy. In practice, however, the class probabilities p_A and p_B cannot be computed exactly for modern neural networks and must instead be estimated statistically. The procedures used to obtain these estimates, and the resulting certified guarantees, are discussed next.

3.2. Certification Algorithms

The certification theorem presented in the previous subsection assumes access to the class probabilities of the smoothed classifier. Since these probabilities cannot

be computed exactly for modern neural networks, randomized smoothing instead relies on statistical estimation through Monte Carlo sampling. Every certification algorithm repeatedly evaluates the base classifier under independent Gaussian perturbations to estimate the probability of the predicted class. Consequently, the computational cost of certification is largely determined by the number of noisy forward passes required. Existing certification algorithms therefore differ primarily in how they balance computational efficiency against the tightness of the resulting probability, and therefore certified radius, estimates.

Fixed-Sample Certification: The Standard Protocol
The standard certification procedure proposed by Cohen, Rosenfeld, and Kolter [6], referred to as CERTIFY, employs a fixed-sample Monte Carlo protocol. In the first stage, the algorithm draws n_0 noise samples to identify the most probable class $\hat{c}_A$ through a majority vote. In the second stage, a larger set of n samples is used to count the occurrences of $\hat{c}_A$, denoted as $k = \sum_{i=1}^{n} \mathbb{1}_{f(\mathbf{x}+\boldsymbol{\delta}_i)=\hat{c}_A}$. To ensure that the resulting guarantee holds with at least $1-\alpha$ probability, the algorithm computes a lower confidence bound $\underline{p_A}$ for the probability $p_A = \mathbb{P}(f(\mathbf{x}+\boldsymbol{\delta}) = \hat{c}_A)$ using the Clopper-Pearson exact method:

$$\underline{p_A} = \mathrm{B}(\alpha; k, n-k+1), \tag{2.6}$$

where B is the inverse of the Beta distribution. The certified radius is then calculated as $CR = \sigma\Phi^{-1}(\underline{p_A})$ if $\underline{p_A} > 0.5$; otherwise, the method abstains and the sample is considered uncertified. Appendix A lists the CERTIFY algorithm.

Adaptive Certification
Many inputs can be certified long before the full sampling budget is exhausted, motivating adaptive certification algorithms that terminate sampling once sufficient statistical evidence has been collected. Voráček [45] proposed a certification procedure based on confidence sequences and betting strategies. Unlike fixed-sample confidence intervals, confidence sequences remain valid under arbitrary stopping times, allowing the algorithm to continuously monitor the probability estimate throughout sampling.

The algorithm maintains a betting process whose accumulated *capital* reflects the statistical evidence supporting the predicted class. Sampling terminates once enough evidence has been gathered to certify the prediction, once certification becomes impossible, or once a predefined maximum sampling budget is reached. Compared with the fixed-sample protocol, this adaptive strategy substantially reduces the average number of noisy forward passes while preserving valid statistical guarantees.

The original method was designed as a decision procedure for determining whether a specified certification radius is achievable. In this thesis, we employ

the implementation listed in Appendix A, which extends the original procedure to efficiently estimate certified radii by incorporating an additional early stopping criterion based on the stability of the probability estimate. When the lower confidence bound $\underline{p_A}$ changes by less than a threshold δ_p for a specified `patience` period, sampling terminates early.

Although certification is performed only during model evaluation, the repeated Monte Carlo sampling required by these algorithms alludes to the primary computational bottleneck associated with randomized smoothing. Similar sampling procedures are also required during certified training, where they become one of the dominant contributors to overall training cost.

4. Certified Training under Randomized Smoothing

The robustness guarantees of a smoothed classifier ultimately depend on the robustness of its underlying base classifier. Since standard empirical risk minimization does not explicitly encourage prediction stability under Gaussian perturbations, randomized smoothing requires specialized training objectives.

Existing methods can be broadly categorized according to how they promote this robustness. The simplest approaches align the training distribution with the smoothing distribution through *data augmentation*. More advanced methods explicitly optimize robustness, either by *incorporating adversarial examples* or by introducing *robustness-oriented regularization* terms into the training objective. More recent work has shifted attention toward improving the computational efficiency of certified training through *transfer learning and adaptive optimization* strategies.

This section briefly introduces each of these approaches and highlights the different optimization strategies they use to improve the certified robustness of randomized smoothing models.

4.1. Data Augmentation

The simplest approach to certified training is to expose the model to the same Gaussian perturbations used during randomized smoothing certification. By aligning the training distribution with the smoothing distribution, the base classifier is encouraged to produce consistent predictions under noisy inputs without modifying the standard supervised learning objective. This strategy was introduced by Cohen, Rosenfeld, and Kolter [6] as the original training procedure for randomized smoothing.

The resulting objective minimizes the expected cross-entropy loss over Gaussian perturbations,

$$\mathcal{L} = \mathop{\mathbb{E}}_{(\mathbf{x},y)\sim\mathcal{D}} \left[\mathop{\mathbb{E}}_{\boldsymbol{\delta}\sim\mathcal{N}(\mathbf{0},\sigma^2\boldsymbol{I})} \mathcal{L}_{\mathrm{CE}}(f_{\boldsymbol{\theta}}(\mathbf{x}+\boldsymbol{\delta}), y) \right]. \tag{2.7}$$

where the inner expectation encourages the classifier to remain accurate over the local neighborhood induced by the smoothing distribution.

Although this objective improves robustness through noise exposure, it does not explicitly optimize the quantities that determine the certified radius. Consequently, Gaussian augmentation serves as an efficient baseline but generally achieves weaker certified robustness than methods that directly optimize robustness-oriented objectives.

4.2. Adversarial Training

While Gaussian augmentation encourages robustness through random perturbations, adversarial training explicitly optimizes the model against worst-case perturbations. Salman et al. [31] extended the adversarial training paradigm to randomized smoothing through **SmoothAdv**, which seeks perturbations that maximize the loss of the *soft* smoothed classifier,

$$G(\mathbf{x}) = \mathop{\mathbb{E}}_{\boldsymbol{\delta}\sim\mathcal{N}(\mathbf{0},\sigma^2\boldsymbol{I})} \left[F(\mathbf{x}+\boldsymbol{\delta}) \right],$$

where $F(\mathbf{x})$ denotes the softmax output of the base classifier.

The resulting objective minimizes the worst-case loss with a fixed perturbation budget,

$$\mathcal{L}_{\mathrm{SmoothAdv}} = \mathop{\mathbb{E}}_{(\mathbf{x},y)\sim\mathcal{D}} \left[\max_{\|\boldsymbol{\eta}\|_2\leq\epsilon} \mathcal{L}_{\mathrm{CE}}(G_{\boldsymbol{\theta}}(\mathbf{x}+\boldsymbol{\eta}), y) \right], \tag{2.8}$$

where the inner maximization identifies the perturbation that most degrades the prediction of the smoothed classifier.

In practice, this optimization is approximated using iterative adversarial attacks, typically *Projected Gradient Descent* (PGD). Since each PGD step requires Monte Carlo estimation of the smoothed classifier and its gradients, adversarial training incurs substantially higher computational cost than Gaussian augmentation. This additional optimization generally improves certified robustness, but at the expense of significantly increased training time.

4.3. Robustness-Regularized Training

Rather than generating adversarial examples during optimization, robustness-regularized methods encourage certified robustness by augmenting the training

objective with a differentiable robustness penalty. The resulting objective takes the form

$$\mathcal{L} = \mathcal{L}_{\text{CE}} + \lambda \mathcal{L}_{\text{R}},$$

where $\mathcal{L}_{\text{CE}}$ is the standard classification loss and $\mathcal{L}_{\text{R}}$ promotes robustness using statistics estimated from multiple Gaussian perturbations. Unlike adversarial training, these methods avoid an explicit inner optimization problem while retaining a common training pipeline. They differ primarily in the formulation of the robustness regularizer. In this thesis, we consider three representative methods: MACER [48], ADRE [9], and Consistency [15].

(i) **MACER** [48] directly maximizes the certified radius by introducing a hinge loss on a differentiable approximation of the certification bound,

$$\mathcal{L}_{\text{MACER}} = \max(0, \gamma - \hat{CR}(\mathbf{x}, \sigma)), \tag{2.9}$$

where $\hat{CR}$ is a differentiable surrogate of the certified radius defined in Eq. (2.5).

(ii) **ADRE** [9] encourages robustness by increasing the margin between the true class and its strongest competitor in the smoothed prediction,

$$\mathcal{L}_{\text{ADRE}} = -\mathcal{L}_{\text{CE}}(G^{(y)}(\mathbf{x}), \arg\max_{c \in \mathcal{Y} \setminus \{y\}} G^{(c)}(\mathbf{x})), \tag{2.10}$$

where $G(\mathbf{x})$ denotes the smoothed prediction obtained by averaging over Gaussian perturbations.

(iii) **Consistency** [15] is based on the observation that locally consistent predictions under Gaussian perturbations imply improved robustness. It minimizes the divergence between individual noisy predictions and their smoothed average,

$$\mathcal{L}_{\text{Consistency}} = \lambda \mathop{\mathbb{E}}_{\boldsymbol{\delta} \sim \mathcal{N}(\mathbf{0}, \sigma^2 \boldsymbol{I})} [\text{KL}(G(\mathbf{x}) \parallel F(\mathbf{x} + \boldsymbol{\delta}))] + \eta \, \text{H}\left(G(\mathbf{x})\right), \tag{2.11}$$

where $\text{H}(\cdot)$ denotes the entropy of the smoothed prediction.

Although these methods optimize different robustness objectives, they share the same computational structure. Each requires estimating smoothed statistics such as $G(\mathbf{x})$ or $\hat{CR}$ from multiple Gaussian perturbations of every training sample. Consequently, the number of noise samples m directly controls the trade-off between the quality of the robustness objective and the computational cost of training, making repeated noise sampling one of the principal bottlenecks in certified training. In the existing literature, however, this trade-off is typically treated as a fixed design choice or tuned primarily for robustness, with comparatively little attention given to its computational implications or to systematically improving the efficiency of the training process.

4.4. Adaptive Training Methods

More recent work has explored improving certified training by adapting the optimization process itself rather than introducing new robustness objectives. These methods modify how training samples are selected or how supervision is provided, with the goal of improving robustness, reducing training cost, or achieving a better balance between the two.

One direction is based on *knowledge transfer*. **Certified Robustness Transfer** (CRT) [44] trains a student model by aligning its predictions with those of a pretrained robust teacher, reporting training speedups of up to 8× compared to direct certified training.

A second direction employs *adaptive optimization*. Methods such as **Cat-RS** [14] and **TRC** [22] dynamically adjust the training process according to sample-specific properties such as confidence or difficulty. By allocating computational effort adaptively, these methods seek to improve the effectiveness of certified training while avoiding regressions due to less informative samples.

4.5. Research Gap: The Missing Efficiency Perspective

The certified training methods presented in this section differ substantially in their optimization strategies, ranging from Gaussian data augmentation and adversarial optimization to robustness regularization and adaptive training. Despite these differences, they share a common characteristic: all require repeated evaluations of the base classifier under Gaussian perturbations to estimate smoothed predictions or robustness-related quantities during optimization. Consequently, the computational overhead of certified training is not a consequence of any individual training objective, but rather a structural property of the randomized smoothing framework itself.

Although this shared computational structure is prevalent, the question of its efficiency remains comparatively underexplored. In particular, three challenges motivate the work presented in this thesis:

(i) *Limited Efficiency Evaluation.* Existing work provides limited support for systematically comparing the computational efficiency of certified training methods.

(ii) *Incomplete Trade-off Characterization.* The relationship between computational cost and certified robustness remains insufficiently characterized.

(iii) *Efficiency Is Rarely a Primary Objective.* Most certified training methods primarily optimize for robustness, with comparatively less emphasis placed on computational efficiency as a design objective.

The discussion throughout this section suggests that understanding certified training efficiency requires more than the slight modification of individual training algorithms. Since the computational overhead arises from characteristics shared across randomized smoothing methods, a principled understanding of computational efficiency is needed before these methods can be systematically evaluated or modified. The following section therefore reviews computational and data efficiency in deep learning, establishing the concepts and perspective used throughout the remainder of this thesis.

5. Computational and Data Efficiency

While efficiency has been widely studied in general deep learning, its role in certified training remains underexplored. The unique structure of randomized smoothing introduces distinct efficiency challenges that are not addressed by standard optimization techniques. By revisiting advances in efficient deep learning through the lens of randomized smoothing, we can develop a more principled understanding of these challenges and identify factors that warrant direct empirical investigation.

The advancement of machine learning has been inextricably linked to an exponential escalation in training compute and data volume. Sevilla et al. [36] characterize this progression through distinct eras, noting that doubling times for training compute have shortened significantly from 21 months in the pre-deep learning era to as little as 6 months in the modern regime. As the environmental and economic costs of training state-of-the-art models become increasingly prohibitive [34], algorithmic efficiency has emerged from a secondary implementation concern to a primary research frontier.

We categorize algorithmic speedup techniques into two parallel research objectives: improving *time efficiency* by reducing the computational cost per iteration, and enhancing *data efficiency* by minimizing the total number of samples required to reach a target performance level. The latter is particularly relevant given the limitations of standard neural scaling laws. While scaling models and datasets typically yields performance gains, standard power-law scaling suggests diminishing returns where incremental accuracy improvements require exponential increases in resources. Recent work proposes this barrier can be bypassed through targeted data-centric strategies. Sorscher et al. [38] show that data pruning, the removal of redundant or uninformative samples, can lead to exponential gains in data efficiency. Central to this approach is the identification of sample importance heuristics, such as the loss magnitude and the loss gradient norm [29], which identify the most informative samples in the training process.

Efficiency Metrics and Taxonomy

Evaluating the efficiency of training algorithms requires a multidimensional approach, as no single metric captures the entirety of the resource-performance trade-off. Bartoldson, Kailkhura, and Blalock [3] distinguish between direct performance indicators, such as wall-clock training time, and hardware-agnostic proxies like Floating Point Operations (FLOPs). While wall-clock time is the most intuitive measure for practical deployment, it is highly sensitive to hardware configurations and software optimizations. Conversely, while FLOPs provide a theoretical measure of algorithmic complexity, they often fail to account for memory bandwidth bottlenecks, communication overhead in distributed settings, and varying hardware utilization rates.

Algorithmic strategies for improving training efficiency follow several paths, including advanced optimization schedules, architectural innovations like depthwise separable convolutions [25], and numerical optimizations such as mixed-precision training [26]. In the following, we focus on data-centric acceleration techniques that directly inform the approaches evaluated later in this thesis.

Curriculum Learning

Inspired by the hierarchical nature of human education, Bengio et al. [4] formalized **curriculum learning**, a strategy that orders training examples by difficulty. The model is initially exposed to "easier" samples, with the complexity of the training set gradually increasing according to a predefined *pacing function*. From a theoretical perspective, curriculum learning can be viewed as a continuation method that smooths the optimization landscape early in training, steering the model toward better local minima and accelerating convergence under appropriate conditions. While its benefits in standard, well-behaved datasets are sometimes marginal [47], curriculum learning is effective in scenarios involving limited computational budgets or noisy data distributions, which are conditions inherent to robust training.

Selective Backpropagation

Selective backpropagation addresses efficiency by reducing the number of samples processed during the computationally expensive backward pass. Jiang et al. [17] proposed prioritizing samples based on their individual contribution to learning, typically measured by the loss value obtained during the forward pass. Under this paradigm, samples with low loss ("easy" samples) are discarded or processed with lower probability, while resources are concentrated on high-loss examples that drive significant weight updates. Further refinements, such as RHO–LOSS [27], prioritize training points that are expected to most significantly reduce the model's generalization error. By dynamically filtering the training stream, these methods achieve significant wall-clock speedups while maintaining or even improving the final model accuracy. While selective backpropagation has been shown to be effective

in standard training, its application to certified robustness remains non-trivial due to the stochastic nature of noise sampling. This raises the question, whether robustness-aware selection metrics can provide more stable signals than loss-based criteria, which is one of the questions we evaluate in this thesis.

Integrating these general speedup techniques into the randomized smoothing pipeline constitutes a direct probe of the intrinsic inefficiency of estimating robust gradients. While curriculum learning and selective backpropagation traditionally target standard optimization, their potential application to certified training changes the m-fold scaling of computational overhead by reducing how often samples enter the backward pass. By leveraging robustness-aware importance sampling, the expensive multi-pass noise sampling process, where m noisy samples $\boldsymbol{\delta} \sim \mathcal{N}(\mathbf{0}, \sigma^2 \boldsymbol{I})$ are processed per input, can be concentrated on examples with stronger robustness impact. This motivates a data-level analysis that measures how selective computation changes training cost, clean accuracy, and certified robustness. Consequently, the interaction between general algorithmic acceleration and domain-specific robustness requirements forms the foundation for the selective-computation analysis in this work.

Chapter 3.

Benchmarking Certified Training Efficiency

Randomized smoothing has scaled to large datasets and diverse model architectures, enabling rapid advances in certified training. Despite large increases in training duration and memory requirements, the computational overhead of these methods remains largely unexplored, leaving comparative efficiency analysis fragmented. In this chapter, we address this gap by developing a systematic, unified framework for evaluating the efficiency–robustness trade-off in certifiably robust training. By jointly assessing model accuracy, certified robustness, and multi-faceted computational resources, this framework addresses the limitations of existing evaluations. We also define efficiency metrics that quantify training dynamics rather than relying solely on final performance.

The remainder of this chapter is structured as follows: Section 1 reviews the efficiency benchmarking gap in the certified training literature, outlines insights from general deep learning training efficiency literature, and categorizes the key infrastructural and domain-specific challenges. Section 2 describes our evaluation methodology, its objectives and core characteristics, the performance indicators measured, and the baseline methods selected for evaluation. Section 3 details the hardware and software specifications, the datasets considered, and the hyperparameter configurations for each baseline. Section 4 presents the benchmark results and analyzes the quality–efficiency trade-off. Section 5 states the scope and interpretation limits of the benchmark. Finally, Section 6 summarizes the benchmark findings and motivates the follow-up analysis.

1. Limitations of Existing Efficiency Evaluations

Empirical Gap: Inconsistent Efficiency Reporting

Computational overhead remains largely unaddressed in the certified training literature. When researchers report training efficiency, they often present brief evaluations that prohibit direct comparison. These comparisons typically fail due to under-characterized hardware and software environments, insufficient metrics, or reproducibility limitations. Inconsistent implementations fail to isolate relevant components of the training pipeline, preventing correct attribution of performance gains.

Prior studies sparsely evaluate computational efficiency. Zhai et al. [48] report training times on CIFAR-10 and ImageNet to compare their regularization-based method, MACER, with SmoothAdv and Gaussian augmentation. The lack of a standardized framework to isolate individual training loops limits the reliability of their findings. While targeting training acceleration directly, Vaishnavi, Eykholt, and Rahmati [44] evaluate SmoothAdv, MACER, and SmoothMix by reporting a "slowdown" factor relative to standard training. They report training times for their transfer learning method, Certified Robustness Transfer (CRT), across various architectures on CIFAR-10. This analysis remains limited by the use of non-standard architectures, a focus on smaller-scale datasets, and the lack of implementation isolation. Most recently, Li, Fang, and Yang [22] report training durations for Teacher Reweighted Consistency (TRC) against Consistency and CRT, yet they do not discuss their underlying benchmarking methodology.

Another issue concerns implementation fragmentation, where authors often bundle the core training logic with data-loading pipelines, logging utilities, and certification scripts, hindering isolation of the training algorithm's computational cost.

Taken together, existing evaluations lack standardization, reproducibility, and isolation of algorithmic effects, making cross-method comparisons unreliable.

Methodological Challenges in Measuring Training Efficiency

Measuring training efficiency is not merely a matter of poor experimental practice; it is a difficult scientific problem. This difficulty arises from two key sources: infrastructure variability and hyperparameter sensitivity.

Infrastructure variability. Quantifying training efficiency requires overcoming general infrastructural hurdles. Deep learning training is fundamentally stochastic, with performance varying across random initializations and data shuffling orders. Measurements of computational overhead depend directly on the underlying hardware and software stack. Variations in GPU architecture, memory bandwidth, and CUDA kernel optimizations produce inconsistent execution times if the environment

is not strictly controlled. Hence, insufficient environmental specification prevents direct replication of reported throughput metrics or slowdown factors.

Hyperparameter sensitivity. Hyperparameter sensitivity exacerbates these issues. Both the result quality and computational overhead of certified training methods depend on the training noise level σ, number of noise samples taken per input sample, method-specific regularization coefficients (such as λ in MACER or ADRE), and standard optimization parameters like batch size and learning rate schedules. Certified training typically requires tuning for each method to obtain competitive robustness. A benchmark that fixes these parameters across all methods unfairly disadvantages certain approaches, whereas exhaustively tuning each method measures the quality of the tuning process rather than the algorithm itself.

These challenges indicate that efficiency benchmarking cannot be treated as a purely engineering concern, but instead requires a principled experimental framework. Bartoldson, Kailkhura, and Blalock [3] formalize comparative training speedup and establish best practices for profiling training efficiency. The ALGOPERF framework [7, 18], designed for benchmarking general-purpose training algorithms, highlights several vital concepts that inspire our methodology: namely, a strict separation of concerns when measuring training performance and the explicit treatment of hyperparameters. Building on these principles, we draw inspiration from the broader deep learning literature while tailoring the methodology to the certified training setting. While our scope is more focused, where we consider a fixed workload and a narrower definition of the training algorithm that excludes the optimizer, we adjust their proposed "time-to-result" approach. This adjustment is necessary because, as we will explore next, estimating certifiable robustness is a computationally expensive operation, thus limiting its direct viability in evaluating training efficiency.

Domain-Specific Bottlenecks in Certified Training

Certified training also poses domain-specific difficulties that complicate holistic performance benchmarking. We attribute these difficulties to two sources: evaluation bottlenecks and training bottlenecks.

Evaluation bottlenecks. Robustness certification constitutes a major evaluation bottleneck in certified training due to its high computational cost. Estimating certified radii typically requires 10^4–10^5 Monte Carlo samples per test image to produce statistically sound lower bounds [6]. Certifying the full CIFAR-10 test set takes hours, whereas ImageNet certification can take days on a single GPU. This expense makes frequent robustness evaluation impractical, limiting visibility into training dynamics and the progression of robustness throughout training.

Training bottlenecks. Certified training introduces computational costs that are absent from standard empirical risk minimization. At the core of most state-of-the-

art methods is the estimation of gradients of a smoothed loss,

$$\nabla_{\boldsymbol{\theta}} \mathop{\mathbb{E}}_{\boldsymbol{\delta} \sim \mathcal{N}(\mathbf{0}, \sigma^2 \boldsymbol{I})} \left[\mathcal{L}(f_{\boldsymbol{\theta}}(\mathbf{x} + \boldsymbol{\delta}), y) \right],$$

which typically requires m independent noise samples per input. As a result, each forward and backward computation is scaled by a factor of m, whose range in the literature is $[1, 16]$. This requirement underlies methods such as MACER [48] and Consistency [15], leading to substantial reductions in throughput. The additional memory consumption often forces smaller batch sizes, which in turn affects both optimization dynamics and convergence behavior. Training costs increase further in methods such as SMOOTHADV [31], which combine smoothing with adversarial inner-loop optimization. These interacting sources of overhead make it difficult to identify and attribute the dominant efficiency bottlenecks in certified training methods.

In summary, certified training introduces domain-specific constraints that complicate both the measurement and interpretation of computational efficiency. In combination with the general limitations of current efficiency evaluation practices, these factors indicate that existing reported figures and benchmarks capture only an incomplete perspective on certified training performance. Characterizing the interaction between training-time complexity, evaluation cost, and implementation practices therefore motivates a systematic, multi-faceted benchmarking approach.

Key Insights

A certified training benchmark must isolate training logic from infrastructure, control for hardware and software environments, manage hyperparameter sensitivity, and address the extreme computational cost of certification to enable result reproduction and fair efficiency comparisons.

2. The BENCHMARKCTRS Framework and Evaluation Protocol

We introduce the BENCHMARKCTRS framework as a systematic approach to addressing the limitations outlined in the previous section. First, we define the design objectives that guide the construction of a benchmark for certified training efficiency. Second, we describe the implementation principles, which enable isolating algorithmic efficiency and reducing confounding environmental factors. Finally, we present the evaluation protocol used to interpret results, contextualize performance differences, and enable consistent comparison across methods.

2.1. Design Objectives

The validity of a benchmarking depends not only on the metrics it reports, but also on the structure through which experiments are conducted. To ensure trustworthy results, we define three design objectives that we consider necessary for a successful and reliable benchmarking framework for certified training: isolation of training logic, multifaceted efficiency characterization, and reproducibility and extensibility.

Isolation of Training Logic. To address the prevalent entanglement of algorithmic and infrastructural costs, it is necessary to decouple the training algorithm from the experimental pipeline. By enforcing a shared data-loading and evaluation infrastructure across all methods, the benchmarking framework ensures that observed differences in runtime and memory consumption are attributable to the training algorithm itself rather than implementation or system-level effects.

Multi-Faceted Efficiency Characterization. To overcome the limitations of evaluation practices that focus primarily on final performance metrics, we argue that a broader notion of efficiency is necessary for any meaningful benchmarking of certified training. As discussed in the previous section, such practices obscure both the computational cost of training and the evolution of robustness over time. A benchmarking framework must account for wall-clock training time, throughput, and memory consumption alongside the accuracy and certified-radius metrics commonly used in the certified training literature. These measurements capture efficiency dimensions that remain hidden under standard evaluation protocols.

We further consider it essential that a benchmarking framework provide observability beyond final outcomes by continuously tracking accuracy and robustness throughout training. This is necessary to address the lack of visibility into convergence behavior and intermediate training dynamics, which results from infrequent and cost-prohibitive evaluation in existing approaches.

Reproducibility and Extensibility. To mitigate inconsistencies arising from heterogeneous implementations and experimental setups, we consider it important that a benchmarking framework support reproducibility and extensibility through a modular design. In particular, it is necessary to ensure that new training methods can be integrated consistently under a shared experimental environment, so that comparisons are not confounded by implementation-specific or system-level variations. This is essential to establish a reliable baseline for comparative evaluation across methods.

These three objectives jointly establish the core requirements for a reliable benchmarking framework for certified training. In the following section, we describe how BenchmarkCtrs is structured to satisfy these requirements in a unified framework architecture.

2.2. Framework Architecture

The architecture of BenchmarkCtrs is designed to produce valid, trustworthy, and interpretable benchmarks by realizing the design objectives above. The framework is built around three core principles: strict separation of concerns, extensibility through a plugin-based architecture, and detailed instrumentation of the training process. Together, these components provide a controlled experimental environment that isolates algorithmic behavior, simplifies the integration of new methods, and enables detailed characterization of training efficiency and robustness dynamics.

Modular Separation of Components. To satisfy the objective of isolating training logic from experimental infrastructure, BenchmarkCtrs adopts a modular architecture that enforces a strict separation between the training algorithm, the data pipeline, and the evaluation engine. This design directly addresses the benchmarking challenges discussed in the previous section, where implementation-specific and infrastructural costs can obscure the true computational characteristics of a training method.

Built upon the PyTorch Lightning [1] framework, BenchmarkCtrs leverages a standardized training lifecycle that abstracts common training operations from algorithm-specific logic. This separation is realized through a common base module that defines the core responsibilities of a certified training method, including the specification of the optimization objective and stochastic perturbation mechanisms. By enforcing a uniform interface across methods, the framework ensures that differences in runtime and memory consumption can be attributed to the training algorithm itself rather than variations in the surrounding infrastructure.

Extensibility via Plugin Architecture. To satisfy the objective of reproducibility and extensibility, BenchmarkCtrs employs a hook-based plugin architecture that supports the integration of third-party training methods, datasets, and certification algorithms. Beyond its core functionality and baseline implementations, the framework allows new components to be introduced as self-contained plugin modules without requiring modifications to the underlying infrastructure.

This design lowers the barrier to benchmarking new methods while preserving a consistent experimental environment. Environmental controls and system-level concerns, such as thread management and CUDA synchronization, remain centralized within the shared infrastructure. As a result, newly integrated methods can be evaluated under the same conditions as existing ones, ensuring that comparisons remain reproducible and attributable to algorithmic differences rather than implementation artifacts.

Instrumentation and Monitoring. A central architectural component of BenchmarkCtrs addresses the objective of multi-faceted efficiency characterization through integrated instrumentation and training-time robustness observability. As discussed in the previous section, existing evaluation practices provide limited

visibility into convergence behavior because robustness metrics such as ACR are typically evaluated only after training. To overcome this limitation, the framework incorporates a dedicated certification layer that performs efficient robustness estimation throughout the training process.

To address the domain-specific challenge posed by the cost of certification, BenchmarkCtrs introduces an intermediate validation stage at the end of each epoch. Rather than performing the standard, computationally expensive certification, the framework employs the adaptive certification procedure described in Section 3.2 of Chapter 2 to obtain conservative estimates of the Average Certified Radius (ACR) on a hold-out validation set. This enables continuous tracking of robustness throughout training at a fraction of the cost, exposing convergence dynamics and allowing the identification of peak robustness under different training budgets.

The final evaluation stage transitions back to fixed-sample certification on the test set, ensuring that reported quality metrics remain directly comparable with the existing literature. This two-tiered evaluation strategy, efficient adaptive certification for training-time monitoring and rigorous fixed-sample certification for final assessment, provides both robustness observability and reliable end-of-training evaluation, forming a key component of the framework's ability to characterize quality–efficiency trade-offs.

Collectively, this architectural design directly implements the stated design objectives, enabling isolated, reproducible, and fully observable benchmarking of certified training methods. By combining modular separation, extensibility, and training-time instrumentation, the BenchmarkCtrs framework provides a controlled environment for consistent efficiency evaluation. Next, we focus on the evaluation protocol used to interpret framework results and extract comparative insights across methods.

2.3. Evaluation Protocol

In this section, we describe the evaluation protocol used to interpret and analyze results produced by BenchmarkCtrs. The protocol is organized into two components. First, we define the set of metrics used to characterize performance, spanning computational overhead, classification and robustness quality, and optimization dynamics. Second, we define the tools used to interpret these results, including Pareto-based trade-off visualizations and a supplementary, composite certified utility index.

The Evaluation Metrics

The selected metrics characterize both efficiency and quality in certified training. Computational overhead metrics quantify the resources required during training,

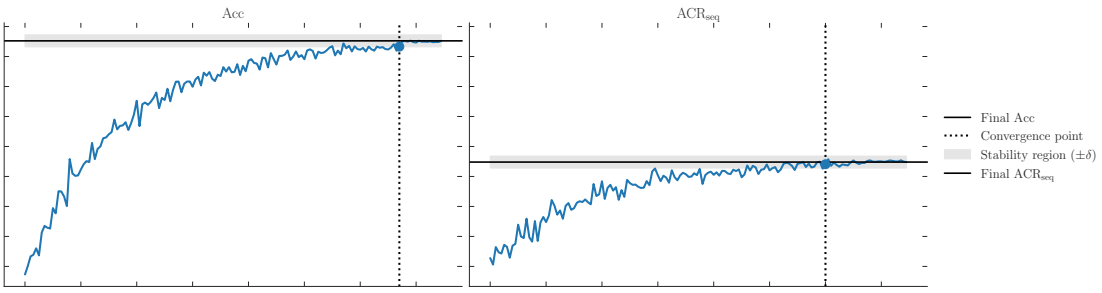


Figure 3.1.: A visualization of the convergence metrics Conv_{Acc} (left) and Conv_{ACR} (right) measured on a sample training process using a fixed threshold $\delta_{\text{conv}} = 0.02$. The x-axes correspond to the training epoch, the dotted line is the convergence epoch, and the shaded stability region is centered around each final metric value (the solid lines).

while classification and robustness metrics assess the quality of the resulting model. Linking computational cost and model quality, optimization dynamics metrics complement these perspectives by measuring how quickly a method reaches a stable level of performance.

1. **Computational Overhead Metrics.** To characterize the resource requirements of certified training methods, we define a set of metrics that quantify training cost along complementary dimensions:

(i) *Total training cost* (Time). We measure the total computational budget required for training in terms of GPU-hours. This is computed as the product of wall-clock training time and the number of GPUs used, i.e., GPU-h $= (T \times N_{\text{GPUs}})/3600$. This metric captures the overall resource consumption of a method in a way that is more suitable for comparison across different configurations with similar hardware than the standalone wall-clock time.

(ii) *Training throughput* (TP). We report throughput as the number of training samples processed per second (samples/s). This provides a hardware-agnostic measure of computational efficiency, reflecting the effective speed of the training procedure notwithstanding resource allocation.

(iii) *Peak GPU memory footprint* (Mem). We measure the maximum GPU memory allocated during training. While temporal cost captures total computation, memory usage determines scalability with respect to model size, batch size, and hardware constraints, and is therefore essential for assessing practical deployability.

Fair comparison of training time also requires accounting for memory consumption. Since training throughput can often be improved by allocating additional memory resources, for example through larger batch sizes, runtime measurements alone may provide a misleading picture of computational

efficiency. Memory usage therefore serves as an important complementary indicator when assessing the practical cost of a training method.

Together, these metrics provide a broad view of computational budget and address the limitations of prior evaluations that rely on isolated or partial cost indicators.

2. **Classification and Robustness Metrics.** To quantify the final model quality, we employ standard metrics from the randomized smoothing literature. These metrics capture both threshold-based performance under a fixed robustness requirement and aggregate robustness across the test distribution.

(i) *Approximate Certified Accuracy* (Acc(ϵ)). It measures the proportion of correctly classified test set samples whose certified radius meets a specified lower-bound ϵ with confidence level $1 - \alpha$. This metric reflects the practical utility of a model under a fixed adversarial budget.

(ii) *Average Certified Radius* (ACR). Introduced by Zhai et al. [48], it aggregates certified radii over the test set $\mathcal{D}_{\text{test}}$, assigning zero radius to misclassified samples. This offers a threshold-independent assessment of model robustness, which is a more interpretable view compared to the certified accuracy.

To ensure consistency with prior work while accounting for computational constraints, we distinguish between fixed-sample certification and the adaptive certification procedures. We follow the standard practice of using fixed-sample certification for final evaluation on the test set, whereas we utilize adaptive certification whenever assessing performance mid-training on the validation set is needed. Irrespective of the certification procedure used to obtain the certified radius $\hat{CR}(g, \sigma, x)$, both Acc and ACR follow the same definitions as follows:

$$\text{Acc}(\epsilon) = \frac{1}{|\mathcal{D}_{\text{test}}|} \sum_{(\mathbf{x},y)\in\mathcal{D}_{\text{test}}} \mathbb{1}_{\hat{CR}(g,\sigma,\mathbf{x})\geq\epsilon \,\wedge\, g(\mathbf{x})=y},$$

$$\text{ACR} = \frac{1}{|\mathcal{D}_{\text{test}}|} \sum_{(\mathbf{x},y)\in\mathcal{D}_{\text{test}}} \hat{CR}(g, \sigma, \mathbf{x}) \mathbb{1}_{g(\mathbf{x})=y},$$

where the same definitions apply under both certification regimes.

From a benchmarking perspective, reporting the ACR alone can obscure important aspects of model behavior. Sun et al. [39] show that optimizing for the ACR can incentivize methods to further increase the certified radii of already robust samples, while providing comparatively little benefit to more challenging inputs. As a result, methods that achieve higher average certified radii may nevertheless certify fewer difficult samples at practically relevant threat levels. Certified accuracy complements this view by measuring the proportion of inputs certified at a fixed adversarial budget, providing a more application-oriented assessment of robustness.

Together, ACR and certified accuracy provide a more complete characterization of certified robustness than either metric in isolation.

3. **Training Dynamics Metrics.** To characterize the efficiency of the optimization process, we measure *the convergence speed of both the clean accuracy and robustness*. These metrics capture how quickly a model reaches a stable performance regime, complementing the final quality measures defined in the previous section.

We define convergence based on the epoch at which a metric stabilizes within a tolerance δ_{conv} of its final value over the training horizon. Let $Acc^{(t)}$ and $\text{ACR}_{\text{seq}}{}^{(t)}$ denote the validation accuracy and the adaptively estimated validation average certified radius at epoch t respectively, and T the total number of training epochs.

We then define:

$$\text{Conv}_{Acc} := \min \left\{ t \in [1, T] \;\middle|\; \delta_{\text{conv}} \geq |Acc^{(u)} - Acc^{(T)}|, \forall\, u \geq t \right\}$$
$$\text{Conv}_{ACR} := \min \left\{ t \in [1, T] \;\middle|\; \delta_{\text{conv}} \geq |\text{ACR}_{\text{seq}}{}^{(u)} - \text{ACR}_{\text{seq}}{}^{(T)}|, \forall\, u \geq t \right\}$$

While Conv_{Acc} reflects standard convergence behavior in deep learning, Conv_{ACR} provides a robustness-specific notion of convergence, capturing how quickly certified performance stabilizes during training. As highlighted in the discussion of evaluation limitations, this distinction is particularly relevant in certified training settings where robustness and accuracy may evolve at different rates. Figure 3.1 provides a visual illustration of the two convergence measures.

Collectively, the proposed metrics define a structured, multidimensional space for evaluating certified training methods along efficiency, quality, and optimization behavior. While they enable consistent quantitative comparison, they do not prescribe how trade-offs between these dimensions should be interpreted. The next subsection addresses this gap by introducing the analysis tools used to contextualize and compare results across methods.

Result Interpretation

We now turn to the interpretation of the benchmark results. This section introduces two complementary tools for analyzing and comparing certified training methods: Pareto-front visualizations for examining trade-offs across objectives, and a composite certified utility index for aggregating multidimensional performance into a single comparative score.

Pareto-front Plots. To complement scalar metric comparisons, we use Pareto-based visualizations to explicitly represent trade-offs between the competing objectives. Each method is mapped into a multidimensional space defined by computational efficiency and model quality, where Pareto fronts highlight non-dominated solutions. We plot the results at the end of every training epoch, allowing computational investment to be directly related to the corresponding performance outcome

over time. In this way, the resulting trajectories capture how efficiency and quality evolve jointly throughout training, rather than only at the end of the training process.

We consider three primary views of this space: (i) ACR versus Time, (ii) validation accuracy versus Time, and (iii) ACR versus validation accuracy. This set of projections allows us to disentangle trade-offs between robustness, accuracy, and computational cost from complementary perspectives. Together, these visualizations enable more intuitive identification of methods that achieve favorable balances between cost and performance compared to raw result tables.

Certified Utility Index. To complement Pareto-based analysis, we introduce the Certified Utility Index (CUI), a scalar measure designed to facilitate direct comparison of trade-offs between methods within the multidimensional evaluation space. While Pareto fronts identify which methods are non-dominated, they do not quantify whether differences in individual dimensions justify the observed trade-offs. The CUI addresses this by providing a compact summary that allows controlled comparisons across methods with different efficiency–quality profiles.

The CUI is defined as a harmonic mean of normalized indicators for robustness and computational efficiency in terms of time to convergence and memory consumption. Concretely, it combines four *normalized* indicators: average certified radius ($\mathrm{ACR}_{\mathrm{fixed}}$), certified accuracy ($\mathrm{Acc}_{\mathrm{fixed}}(0.0)$), time-to-convergence efficiency (inverse of Time in GPU-h until Convergence), and memory efficiency (inverse of Mem):

$$CUI = \frac{4}{\frac{\mathrm{ACR}_{\mathrm{fixed}}{}^{*}}{\mathrm{ACR}_{\mathrm{fixed}}} + \frac{100}{\mathrm{Acc}_{\mathrm{fixed}}} + \frac{T_{Acc}^{\mathrm{Conv}}}{T_{Acc}^{\mathrm{Conv}*}} + \frac{\mathrm{Mem}}{\mathrm{Mem}^{*}}} \tag{3.1}$$

where $(\cdot)^{*}$ denotes the best observed value for the corresponding quantity within the benchmark suite.

By construction, the harmonic mean penalizes imbalanced performance across dimensions. This property is particularly useful for evaluating trade-offs: a method that reduces computational cost at the expense of robustness will only achieve a higher CUI if the gain compensates for the corresponding degradation in quality. Conversely, disproportionate improvements in a single dimension are discouraged if they come at significant cost elsewhere. Unlike Pareto-based analysis, which characterizes the structure of the trade-off landscape, the CUI enables direct quantitative comparison between methods with different trade-off profiles, making it possible to assess whether observed efficiency gains justify reductions in robustness or accuracy.

Together, the evaluation metrics and interpretation tools form a unified framework for analyzing certified training methods along efficiency, quality, and convergence dimensions. While the metrics provide the basis for consistent measurement, the Pareto-based analysis and CUI enable complementary perspectives on trade-offs and

comparative performance. We apply this protocol consistently across all experiments in this work and suggest it as a general evaluation approach for comparing existing methods and future experiments. The following section describes the experimental setup under which this protocol is applied throughout the remainder of the thesis

Key Insights

BenchmarkCtrs systematically isolates training logic and collects quality, efficiency, and convergence metrics. Alongside result tables, we use Pareto-front plots and the Certified Utility Index (CUI) to interpret the inherent multidimensional trade-off.

3. Experimental Setup

This section details the experimental setup used to instantiate the evaluation framework described above, including datasets, hardware configuration, baseline methods, and hyperparameters. These components combined define a controlled environment for reproducible and comparable evaluation of all methods.

3.1. Datasets

We evaluate all methods on the CIFAR-10 and ImageNet benchmarks, two widely used datasets in the certified robustness literature that provide complementary small-scale and large-scale evaluation settings.

CIFAR-10 [21]. 60 000 RGB images of size 32 × 32 pixels, 50 000 for training which we further split into 49 000 images for training and 1000 images for validation, and 10 000 for testing. Images are assigned one of 10 classes, each with 5000 training samples and 1000 test samples. We use the same data augmentation technique standard in the certified training literature: random horizontal flipping and random translation up to 4 pixels. The images are also pixel-wise normalized by the mean and standard deviation calculated on the training set. The full dataset is publicly available for download[1].

ImageNet [30]. Published as 1 281 167 training images, and 50 000 images for validation. Images are assigned one of 1000 classes. We split the original training set into 1 280 167 images for training and 1000 images for validation, and use the original validation set for testing. We apply the standard augmentation scheme from the certified training literature: 224 × 224 random cropping and random horizontal flipping for the training images, and resizing into 256 × 256 resolution

[1] `https://www.cs.toronto.edu/~kriz/cifar.html`

followed by 224 × 224 center cropping for validation and test images. The full dataset is publicly available for download[2].

3.2. Controlled Training Setup

To ensure fair and reproducible comparisons, all methods are evaluated under a shared experimental environment. Consistent with the design objectives outlined in Section 2.1, we distinguish between the common training setup described in this section and the method-specific hyperparameters presented in the following subsection, which are selected according to the original implementations.

For CIFAR-10, all methods are trained using a 110-layer residual network [12] for 150 epochs. For ImageNet, all methods are trained using a ResNet-50 architecture [12] for 90 epochs. Unless otherwise specified, optimization is performed using stochastic gradient descent (SGD) with momentum 0.9 and weight decay 10^{-4}. Method-specific choices, including learning rates, scheduling policies, and batch sizes, are reported separately in Section 3.3.

To eliminate system-level sources of variability, all experiments are executed on a fixed hardware configuration. CIFAR-10 experiments are conducted on a single NVIDIA A100 40 GiB PCIe GPU, with 4 AMD EPYC 7662 CPU cores and 16 GiB of process memory. ImageNet experiments use synchronized SGD across 4 NVIDIA A100 40 GiB PCIe GPUs, with 16 AMD EPYC 7662 CPU cores and 80 GiB of process memory.

All shared training and evaluation procedures are implemented within BenchmarkCtrs, which decouples data loading, model construction, validation, and certification from the training algorithm itself. The framework[3] is built using PyTorch [1] and PyTorch Lightning [8].

3.3. Benchmark Configuration

Our experiments include a representative set of methods spanning the principal training paradigms in the randomized smoothing literature. The selected baselines are intended to capture the dominant methodological directions in the randomized smoothing literature, while spanning a variety of computational and robustness characteristics. This group of methods enable a meaningful study of the trade-offs between efficiency and certified performance.

The benchmark comprises: (i) *Gaussian augmentation*, serving as the foundational baseline for smoothed classifiers; (ii) *SmoothAdv*, representing adversarial

[2] `https://image-net.org/download`

[3] Publicly available on GitHub `https://github.com/ratedali/benchmark-ctrs`

training; and (iii) and the regularization-based methods MACER, ADRE, and *Consistency*, which constitute some of the most widely studied approaches for improving certified robustness with comparatively modest training overhead.

Together, these methods capture the dominant algorithmic directions explored in certified training and provide a diverse set of efficiency and robustness profiles for evaluation. Detailed descriptions and theoretical background for each method are provided in Section 4 of Chapter 2.

To preserve the validity of efficiency comparisons, we use the authors' original implementations whenever possible and integrate them into the common benchmarking environment through the framework's plugin architecture[4].

3.4. Hyperparameter Selection

Shared Hyperparameters

Unless otherwise specified, hyperparameters are selected to match the configurations reported in the original implementations. We apply the same Gaussian noise standard deviation during both training and evaluation, using $\sigma \in \{0.25, 0.5, 1.0\}$ for CIFAR-10 and $\sigma \in \{0.5, 1.0\}$ for ImageNet.

For robustness evaluation, we employ the two certification procedures as established in Section 2.3. Both fixed-sample and adaptive certification use the shared parameters $n = 100,000,\ n_0 = 100,$ and $\alpha = 0.001$. Fixed-sample certification is applied during final evaluation on the test set. On CIFAR-10, all test samples are certified, whereas on ImageNet, every 100^{th} sample is evaluated to remain consistent with common practice in the literature.

During training, robustness is monitored on the validation set using adaptive certification. In addition to the shared certification parameters, we set `patience` $= 1000$ and $\delta_{\text{seq}} = 10^{-4}$.

Method-Specific Hyperparameters

The remainder of this section details the method-specific hyperparameter configurations used for Gaussian augmentation, SMOOTHADV, MACER, ADRE, and Consistency. As a guiding principle, we adopt the best-performing configurations reported by the original authors in order to ensure faithful reproduction of each method's intended operating regime.

Gaussian Augmentation.[5] (Cohen, Rosenfeld, and Kolter [6]). on CIFAR-10, we use a batch size of 256 and initial learning rate is 0.1, and decays by a factor of 0.1 on a 50 epoch interval. On ImageNet, we use an effective batch size of 400 (or

[4]The plugin collection is publicly available on `https://github.com/ratedali/benchmark-ctrs-methods`

[5]We reuse training code publicly available on `https://github.com/locuslab/smoothing`

100 per GPU) and initial learning rate is 0.1, and decays by a factor of 0.1 on a 30 epoch interval.

SmoothAdv[6] (Salman et al. [31]). uses the same batch size and learning rate configuration as Gaussian augmentation. It has three hyperparameters when using the projected gradient descent (PGD) attack variant: (a) the ℓ_2–norm attack radius ϵ, (b) the number of PGD steps T, and (c) the number of noise samples m. We apply a warmup strategy for ϵ in which it starts at 0.0 and linearly increases to its intended value over 10 epochs. On CIFAR-10, we fix $T = 10$. When $\sigma = 0.25$, we set $\epsilon = 1.0$ and $m = 4$. For $\sigma = 0.5$, we set $\epsilon = 1.0$ and $m = 8$. With $\sigma = 1.0$ we set $\epsilon = 2.0$ and $m = 2$. On ImageNet, we fix $T = 1, m = 1$ and set $\epsilon = 1.0$ for $\sigma = 0.5$, and $\epsilon = 2.0$ for $\sigma = 1.0$.

MACER[7] (Zhai et al. [48]). on CIFAR-10, we use a batch size of 64 and initial learning rate is 0.01, and decays by a factor of 0.1 on a 50 epoch interval. On ImageNet, we use an effective batch size of 256 (64 per GPU) and initial learning rate is 0.1, and decays by a factor of 0.1 on a 30 epoch interval. It has four method-specific hyperparameters: (a) the number of noise samples k, (b) the regularization coefficient λ, (c) the certified radius clamping parameter γ, and (d) the softmax temperature for estimating the certified radius β.

We follow the configuration used by Zhai et al. [48]: we set $\gamma = 8.0, \beta = 16.0$. We also fix $k = 16$ on CIFAR-10 and set $\lambda = 16.0$ for $\sigma = 0.25$, and $\lambda = 4.0$ for $\sigma = 0.5$. For $\sigma = 1.0$, we initially set $\lambda = 0.0$ for 50 epochs and proceed with $\lambda = 12.0$. On ImageNet, we fix $k = 2$ and $\lambda = 3.0$.

ADRE[8] (Feng et al. [9]).on CIFAR-10, we use the same batch size and learning rate configuration as Gaussian augmentation. On ImageNet, we use a 256 effective batch size (64 per GPU) and linear schedule for the first 8 epochs, by setting the learning rate to $0.1 \cdot t/8$ at epoch $1 \leq t \leq 8$. Then, we use a cosine-annealing schedule, setting the learning rate to $0.05 \cdot (1 + \cos(\pi \cdot t/(90 - 8)))$ for epoch $8 \leq t \leq 90$. We use a momentum of 0.875 and weight decay of $1/32\,768$ for the synchronized SGD optimizer, and apply label smoothing [40] with smoothing factor $\alpha = 0.1$.

The method further introduces two further hyperparameters: (a) the number of noise samples m, and (b) the regularization coefficient λ. Following the values used by Feng et al. [9], we fix $m = 8$ on CIFAR-10, and use $\lambda = 0.1$ for $\sigma = 0.25$ and $\lambda = 0.3$ for $\sigma \in \{0.5, 1.0\}$. On ImageNet, we fix $m = 1$ and $\lambda = 0.1$.

[6] We reuse training code publicly available on `https://github.com/Hadisalman/smoothing-adversarial`

[7] Using training code publicly available on `https://github.com/RuntianZ/macer`

[8] Using our own implementation.

Consistency[9] (Jeong and Shin [15]). uses the same configuration as Gaussian augmentation. For training stability, we had to utilize gradient clipping which is not reported in the original work. We applied ℓ_2–norm clipping with max norm of 1.0 on CIFAR-10 and max norm 0.5 on ImageNet. Also on ImageNet, we use an effective batch size of 200 (50 per GPU). Otherwise, Jeong and Shin [15] introduce two penalty terms (a) λ for the KL divergence loss, and, and (b) η for the entropy loss. On CIFAR-10, we fix $\eta = 0.5$ and set $\lambda = 20$ for $\sigma = 0.25$, and $\lambda = 10$ for $\sigma \in \{0.5, 1.0\}$. On ImageNet, we fix $\lambda = 5.0$. We also set $\eta = 0.5$ for $\sigma = 0.5$, and $\eta = 0.1$ for $\sigma = 1.0$.

4. Empirical Results and Discussion

This section presents the empirical evaluation of the certified training methods listed in Section 3.3 under the experimental protocol established in Section 2.3. Our primary objective is to characterize and compare their computational efficiency, which constitutes the primary focus of this benchmark study.

While classification and robustness metrics are included where necessary to contextualize efficiency measurements and trade-offs, we do not focus on absolute quality comparisons. These results are not novel and are primarily included to establish performance baselines for the experiments conducted in Chapter 4.

We begin by discussing several considerations that are important for interpreting the reported results. We then present the benchmark results on CIFAR-10 and ImageNet, respectively, before concluding with a cross-method analysis of the principal trends and trade-offs observed throughout the evaluation.

Experimental Considerations

Before discussing the results, we first outline several experimental considerations that affect their interpretation and comparability.

Fixed-sample vs. Adaptive Certification. Unless otherwise noted, the final accuracy and robustness metrics reported in this chapter are taken directly from the original publications of the corresponding methods. The only exception is ADRE, for which we are not aware of any previously reported ACR values and therefore compute them ourselves. These results correspond to the standard fixed-sample certification procedure.

The training-time robustness measurements presented throughout this chapter are obtained using adaptive certification. As discussed in Section 1, the computational cost of fixed-sample certification makes continuous monitoring during training impractical. Adaptive certification alleviates this limitation by providing

[9] Using training code publicly available on `https://github.com/jh-jeong/smoothing-consistency`

substantially faster, albeit more conservative, estimates of robustness under the same certification parameters (n, n_0, α).

Consequently, the $\mathrm{ACR}_{\mathrm{seq}}$ values reported during training should not be interpreted as direct substitutes for their fixed-sample counterparts. Rather, they serve as efficient indicators of robustness progression and convergence dynamics. A detailed comparison between fixed-sample and adaptive certification is provided in Appendix B, where both procedures are evaluated on the test set under identical conditions.

Reproducibility. The interpretation of benchmark results must be viewed in light of the reproducibility challenges that remain prevalent in the randomized smoothing literature. Even when official implementations are available, reproducing published quality metrics can be difficult due to sensitivity to environmental factors, implementation details, and hyperparameter choices that are not always fully specified.

Our experience with MACER [48] illustrates this challenge. Despite following the original implementation and reported hyperparameters, we frequently observed quality metrics substantially below those reported by the authors. Similar difficulties have also been documented by Jeong, Kim, and Shin [14]. A detailed comparison between the originally reported MACER values and our reproduced results is provided in Appendix C.

To account for this discrepancy, we adopt a dual-reporting strategy throughout this work. When comparing established baseline methods, as we do in this chapter, we report the quality metrics published by the original authors whenever available. In contrast, the evaluations of our experiments in Chapter 4 are performed relative to our own reproduced baselines. This approach preserves comparability with the existing literature while maintaining consistency in the evaluation of our follow-up experiments. As will become apparent, our later experiments modify the corresponding baselines to isolate specific factors in the efficiency–robustness trade-off. Hence, any observed changes should be measured relative to the performance obtained from those baselines within the same experimental environment rather than against independently reported results.

Runtime Fairness. Training runtime is sensitive to several hyperparameter choices, in particular the batch size and the number of noise samples used during stochastic estimation. As noted in Section 3.4, we follow the best-performing configurations reported in the original works, prioritizing model quality rather than enforcing uniform computational settings across methods. An alternative strategy would have been to standardize batch size and noise sample counts across all methods. However, this would typically require compromising the resulting model quality.

To ensure a fair comparison of computational efficiency, we instead condition runtime comparisons on memory usage. Specifically, we compare the training time of configurations with comparable peak memory consumption. This provides a practical normalization, as increases in batch size or noise sample count that reduce runtime inevitably induce a corresponding increase in memory requirements, making memory a natural constraint for fair efficiency comparisons.

Feasibility of Training-Time Evaluation The continuous robustness monitoring described in Section 2.3 is applied to the CIFAR-10 experiments, where adaptive certification is used to track robustness throughout training. While this functionality can be readily extended to ImageNet given sufficient computational budget, it is not included in the present ImageNet evaluation. Consequently, convergence analyses and Pareto-based trade-off visualizations are reported only for CIFAR-10 (Section 4.1). For ImageNet (Section 4.2), evaluation is restricted to post-training fixed-sample certification, focusing on final robustness and efficiency metrics.

Taken together, the above considerations define the evaluation assumptions under which all subsequent results should be interpreted. We now present the empirical results, beginning with CIFAR-10.

4.1. CIFAR-10 Results

We first present the empirical results on CIFAR-10. This section summarizes the performance of all baseline methods under the evaluation protocol introduced in Section 2.3, focusing on computational efficiency, robustness, and their associated trade-offs. We begin with an overview of the aggregated results and convergence behavior, followed by a trade-off analysis using Pareto-based visualizations and CUI-based evaluation.

Overall Trends

We begin by examining the high-level efficiency and robustness patterns across training methods, as reported in Table 3.1.

Scaling Patterns. A clear trend emerges in the computational cost of certified training: the costs increase approximately linearly with the number of noise samples used during gradient estimation. This observation supports the discussion in Section 1, where multi-pass gradient estimation was identified as a primary source of training overhead.

The effect is particularly visible when comparing the training time of regularization-based methods. At $\sigma = 0.25$, Gaussian augmentation $m = 1$ requires 0.59 GPU-h, while Consistency $m = 2$, ADRE $m = 8$, and MACER $m = 16$ incur progressively higher costs of 1.19, 4.35, and 9.12 respectively. SmoothAdv deviates from this pattern, exhibiting substantially higher overhead than would be expected from

σ	Method	Time ↓	TP ↑	Mem ↓	Conv_{Acc} ↓	Acc_{full} ↑	$\text{ACR}_{\text{fixed}}$ ↑	CUI ↑
0.25	Gaussian	**0.59**	**3,456**	87.3	139	76.6	0.424	0.694
	MACER	9.12	224	**85.0**	62	**79.5**	0.531	0.273
	ADRE	4.35	469	89.1	92	73.0	0.528	0.347
	Consistency	1.19	1,713	87.3	**51**	75.8	**0.552**	**0.809**
	SmoothAdv	14.95	137	90.3	136	73.4	0.544	0.094
0.50	Gaussian	**0.60**	**3,403**	87.3	88	65.7	0.525	0.765
	MACER	9.09	225	**85.0**	97	64.2	0.691	0.192
	ADRE	4.36	469	89.1	114	**67.0**	0.675	0.294
	Consistency	1.20	1,706	87.3	**50**	64.3	**0.720**	**0.819**
	SmoothAdv	31.29	65	88.8	148	65.3	0.684	0.044
1.00	Gaussian	**0.60**	**3,419**	87.3	132	**47.1**	0.511	0.567
	MACER	9.05	226	**85.0**	**109**	41.4	0.744	0.170
	ADRE	4.36	469	89.1	141	47.0	0.589	0.239
	Consistency	1.19	1,722	87.3	122	46.3	0.756	**0.569**
	SmoothAdv	7.66	267	94.3	149	43.7	**0.790**	0.152

Table 3.1.: **The key baseline metrics on CIFAR-10**. An upward pointing arrow ↑ in a column header indicates that higher values are better, and a downward pointing arrow ↓ indicates that lower values are better. Values highlighted in **bold** represent the best value for the metric in that section of the table.

noise sampling alone. This additional cost originates from the inner optimization loop inherent in adversarial training.

Robustness–Efficiency Trade-off. The relationship between computational cost and robustness is less straightforward. Increased training overhead does not translate directly into higher certified accuracy or ACR. Nevertheless, a broader trend is visible: methods with lower computational requirements, such as Gaussian augmentation and ADRE, generally achieve weaker robustness than the more computationally intensive approaches, namely MACER and SMOOTHADV.

Consistency provides a notable exception to the broader trend. Despite requiring only a modest increase in training cost over Gaussian augmentation, it achieves robustness levels comparable to substantially more expensive methods. This observation suggests that improvements in certified robustness need not arise solely from proportional increases in computational expenditure.

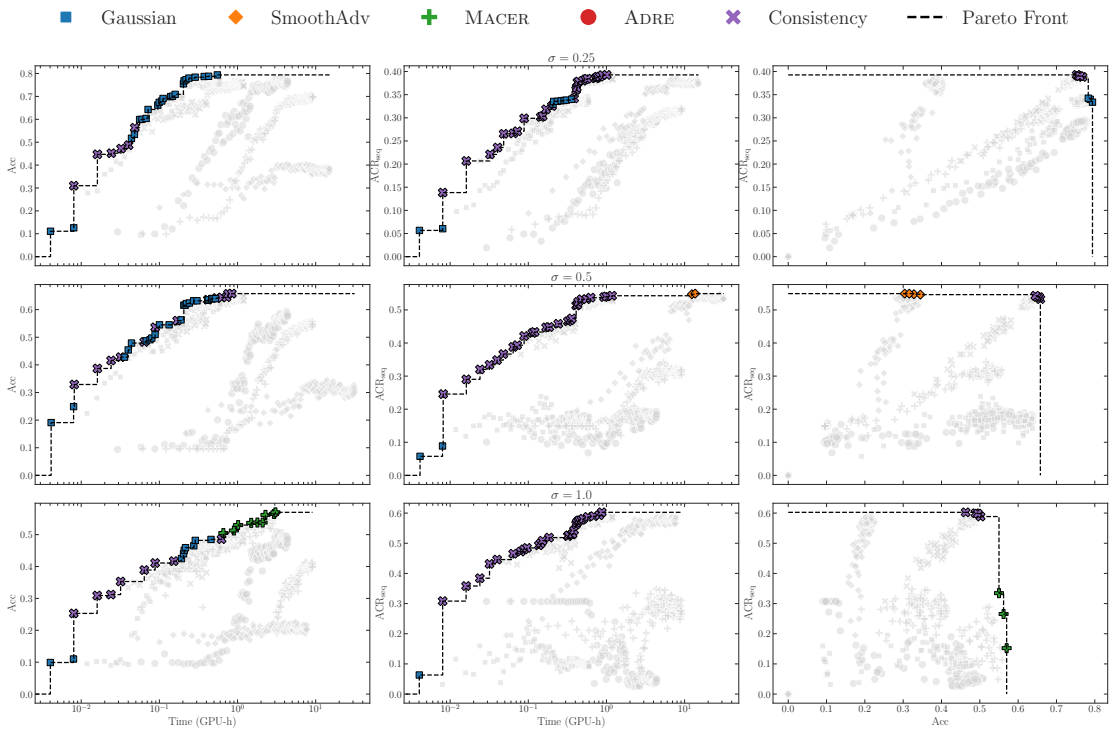


Figure 3.2.: The plots illustrate the inherent trade-offs during training, the metrics are evaluated on the CIFAR-10 validation set. Each method from Table 3.1 adds 150 points for its result at the end of each training epoch. The rows correspond to σ values: 0.25, 0.5 and 1.0 in order. The training time in the first two columns is represented on the x-axis in log-scale. The Pareto front line connects the methods that offer dominating performance on at least one metric.

However, this result should not be interpreted as evidence that the robustness–efficiency trade-off can be entirely avoided. Relative to standard training, Consistency still incurs approximately twice the training cost and exhibits a noticeable reduction in classification accuracy. As the ImageNet experiments show, its efficiency advantage becomes less pronounced at larger scales. Rather than eliminating the trade-off, Consistency appears to operate in a more favorable regime, making it a particularly interesting subject for the detailed analysis that follows.

Convergence Behaviour

The convergence analysis suggests that, for most methods, accuracy has not fully stabilized within the standard training budget of 150 epochs. This indicates that the observed performance may still be influenced by under-training and that several methods may benefit from extended optimization schedules.

An exception is observed for Consistency at lower noise levels ($\sigma \in \{0.25, 0.5\}$), where both accuracy and robustness metrics reach a stable regime earlier in

training. This suggests a more efficient optimization trajectory in these settings, where satisfactory robustness is achieved without requiring the full training budget.

However, convergence behavior should be interpreted in conjunction with final performance metrics, as early stabilization does not necessarily imply superior ultimate robustness. Instead, it reflects differences in optimization dynamics and the rate at which methods trade classification performance for certified robustness.

Trade-off Interpretation

We now examine the efficiency–robustness trade-offs induced by the evaluated training methods. We combine Pareto-based visualizations, which expose the structure of the multi-objective performance space, with the Certified Utility Index (CUI), which provides a scalar summary of how improvements in one dimension are balanced against degradations in others. This dual perspective enables both qualitative and quantitative assessment of the observed trade-offs.

Pareto-based Multi-Objective Analysis. To better understand the structure of these trade-offs, Figure 3.2 visualizes the relationship between efficiency, accuracy, and robustness across three complementary views: (i) ACR versus training time, (ii) accuracy versus training time, and (iii) ACR versus accuracy.

In the plots, each point corresponds to the performance of a training method on the CIFAR-10 validation set at a particular epoch. Methods that are not strictly dominated in the corresponding objective space form the Pareto frontier and are highlighted in the figure.

A clear pattern emerges across all noise levels: Consistency occupies a large portion of the ACR–efficiency Pareto frontier. In the ACR–time plots, the method consistently achieves the highest robustness for a broad range of training-time budgets. A similar trend is visible in the accuracy–robustness plots, where Consistency frequently dominates competing methods in both dimensions. These observations reinforce the conclusion that the method attains a particularly favorable robustness–efficiency balance on CIFAR-10.

Gaussian augmentation occupies the complementary end of the trade-off spectrum. While it generally provides the strongest accuracy–time characteristics, its robustness remains substantially lower than that of the competing certified training methods. Nevertheless, the ACR–time plots suggest that Gaussian augmentation can still offer a competitive compromise between robustness and computational cost, particularly at lower noise levels ($\sigma \in 0.25, 0.5$), where modest robustness gains may not justify the additional training overhead of more sophisticated methods.

MACER exhibits a more nuanced behavior. At $\sigma = 1.0$, it initially appears highly competitive in the accuracy–time space. However, this effect is largely attributable to its delayed robustness regularization schedule (see Section 3.4), which causes the early stages of training to resemble Gaussian augmentation with $m = 16$ noise

samples. As training progresses, the method gradually trades classification accuracy for robustness, explaining the lower final accuracy reported in Table 3.1.

Finally, while SmoothAdv achieves some of the strongest robustness levels in the benchmark, these gains are accompanied by substantially higher computational costs and a reduction in clean accuracy. The resulting trade-off is particularly evident at $\sigma = 0.5$, where improvements in robustness require a disproportionately larger training-time investment than competing methods.

Overall, the Pareto analysis supports the broader conclusion that robustness improvements are rarely obtained without sacrifice. The primary distinction between methods lies not in whether a trade-off exists, but in where they position themselves along the efficiency–accuracy–robustness spectrum.

CUI-based Improvement/Degradation Evaluation. Finally, as introduced in Section 2.3, the Certified Utility Index (CUI) provides a compact view of the efficiency–robustness trade-offs discussed above by aggregating the different performance dimensions into a single comparative measure. Rather than serving as a strict ranking, it is primarily intended to quantify whether gains in one dimension justify the corresponding losses in others.

Referring back to Table 3.1, Consistency attains the highest CUI across noise levels (0.809, 0.826, 0.569). This reflects its balanced profile, combining strong robustness and accuracy with relatively modest computational requirements.

Similarly, Gaussian augmentation obtains the second-best CUI results (0.694, 0.756, 0.567), primarily driven by its computational efficiency. While it does not match the robustness of methods such as Consistency, Macer, or SmoothAdv, the CUI indicates that the corresponding reductions in training time more than compensate for this loss under several operating regimes. This highlights the value of evaluating trade-offs directly rather than considering robustness in isolation.

In contrast, SmoothAdv achieves the lowest CUI values (0.094, 0.044, 0.152) despite its consistently strong robustness performance, as its substantial computational overhead dominates the aggregated trade-off. This indicates that the robustness gains come at a disproportionately high efficiency cost under the current benchmark setting.

Overall, the CIFAR-10 results indicate a consistent separation between methods along an efficiency–accuracy–robustness spectrum, with no single approach dominating across all dimensions. We next extend the evaluation to ImageNet to examine whether these trade-offs remain stable under a more computationally demanding setting.

Key Insights

- **Noise sampling is the primary bottleneck**: training throughput scales inversely with the number of samples, with corresponding increases in memory pressure.
- **Efficiency opportunities with regularization**: these methods offer more flexible efficiency–robustness profiles, though they still incur substantial overhead relative to Gaussian augmentation.
- **Robustness is not proportional to cost**: higher computational overhead does not reliably translate into higher certified robustness.

4.2. ImageNet Results

σ	Method	Time ↓	TP ↑	Mem ↓	Conv_{Acc} ↓	Acc_{full} ↑	$\text{ACR}_{\text{fixed}}$ ↑	CUI ↑
0.50	Gaussian	**78.18**	**1,637**	519.0	**60**	57.2	0.733	**0.537**
	MACER	132.09	969	500.1	89	**64.0**	**0.831**	0.344
	ADRE	142.39	899	504.6	89	61.0	0.588	0.310
	Consistency	192.75	664	**493.6**	88	55.0	0.822	0.260
	SmoothAdv	338.36	378	754.6	89	53.6	0.825	0.162
1.00	Gaussian	**78.93**	**1,622**	519.0	**74**	43.6	0.875	**0.461**
	MACER	127.64	1,003	500.1	88	41.0	1.008	0.327
	ADRE	131.51	973	504.6	78	**48.0**	0.696	0.341
	Consistency	192.03	667	**493.6**	88	41.6	0.982	0.249
	SmoothAdv	362.96	353	754.6	89	40.3	**1.040**	0.149

Table 3.2.: **The key baseline metrics on ImageNet**. An upward pointing arrow ↑ in a column header indicates that higher values are better, and a downward pointing arrow ↓ indicates that lower values are better. Values highlighted in **bold** represent the best value for the metric in that section of the table.

We next extend the evaluation to ImageNet, a substantially larger and more computationally demanding benchmark. The objective is to assess whether the efficiency and robustness trends observed on CIFAR-10 persist at scale. Owing to the cost of continuous certification on ImageNet, the analysis focuses on the aggregate performance metrics and their associated trade-offs, as summarized by the benchmark results and CUI evaluation.

Overall Trade-off

The aggregate ImageNet results, reported in Table 3.2, provide a large-scale evaluation of the efficiency and robustness properties of the considered training methods. Given the increased computational demands of the benchmark, they also serve as an opportunity to examine the scalability of the trade-offs observed previously on CIFAR-10.

Scaling Behavior. The ImageNet results largely reinforce the computational trends previously observed on CIFAR-10. In particular, training overhead remains closely tied to the number of noise samples required by each method, supporting the observation that repeated noise subsampling constitutes a dominant source of computational overhead in certified training. This scaling trend is reflected in the total training time, with Gaussian augmentation requiring 78.18 GPU-h at $\sigma = 0.5$ using a single noise sample, compared to 132.09, 142.39, and 192.75 GPU-h for MACER, ADRE, and Consistency, respectively, all of which rely on two noise samples per update, with the increased computational burden primarily driven by repeated stochastic evaluations.

Therefore, Gaussian augmentation continues to require the least computational costs, while methods employing additional robustness objectives incur longer training times.

Robustness–Efficiency Trade-off. Across methods, robustness improvements come at markedly different efficiency costs. SmoothAdv and MACER achieve the strongest ACR values, but the associated computational burden varies significantly, with SmoothAdv requiring nearly twice the training time of the most efficient regularization-based methods for comparable robustness results. Gaussian augmentation, on the contrary, remains positioned at the other end of the spectrum, achieving fast training at the cost of consistently lower robustness.

In contrast to the CIFAR-10 results in Section 4.1, Consistency does not lead in robustness on ImageNet. Instead, it maintains a broadly competitive profile across both robustness and efficiency. Its ACR remains close to that of MACER and SmoothAdv, while its training time is only moderately higher than that of MACER and ADRE, indicating a balanced but less clearly dominant trade-off behavior at larger scale.

Overall, the ImageNet results confirm that the efficiency–robustness trade-offs observed on CIFAR-10 persist at scale, while also highlighting differences in how methods distribute performance across individual dimensions. We next turn to a trade-off analysis to interpret the differing performance profiles of the selected methods.

Trade-off Interpretation

We next turn to the analysis of efficiency–robustness trade-offs on ImageNet, where the CUI aggregate metric is used to compare the evaluated methods. In

contrast to CIFAR-10, this discussion is restricted to aggregate results, as training-time certification was not used and Pareto-based visualizations are therefore not available.

The ImageNet results reveal a shift in the trade-off structure compared to CIFAR-10. Gaussian augmentation achieves the highest CUI at $\sigma = 0.5$ and $\sigma = 1.0$ (with 0.537 and 0.461, respectively), reflecting that its lower computational cost outweighs its reduced robustness performance at this scale.

A broader pattern emerges among the regularization-based methods, all of which operate at a less favorable trade-off regime on ImageNet compared to Gaussian augmentation, as reflected in their consistently lower CUI values. In contrast to CIFAR-10, where the robustness gains of Consistency were sufficient to offset its higher computational cost, none of the regularization-based approaches achieve a comparable balance at this scale, even when considering the strongest robustness outcomes such as MACER.

SmoothAdv, by comparison, remains positioned at a distinct extreme of the trade-off spectrum: it attains competitive robustness but only at the cost of substantially increased training time and memory consumption, resulting in an unfavorable overall trade-off profile under the CUI evaluation which penalizes each dimension equally.

Overall, the ImageNet results reinforce that a central factor underlying the computational efficiency problem in certified training is the number of noise samples used, which induces an approximately linear increase in computational cost across methods. However, robustness gains are not determined by this factor alone: different methods exhibit substantially different ACR values under comparable sampling budgets, and in some cases methods with fewer samples achieve higher robustness than more expensive alternatives.

Within this setting, regularization-based approaches provide the clearest basis for studying efficiency-oriented modifications: they achieve competitive robustness, and their additional cost is primarily driven by multi-pass sampling, which is amenable to controlled modification. In contrast, adversarial training as exemplified by SmoothAdv occupies a fundamentally less scalable regime, where substantially increased computational overhead is not matched by proportional gains in overall efficiency–robustness balance.

These results identify the dominant empirical trends under the benchmark protocol. Before using them to motivate new efficiency interventions, we state the boundaries of the comparison and clarify how the reported metrics should be interpreted.

Key Insights

- **Adversarial Training is inherently less scalable**: nested optimization structures prioritize robustness over training time and memory requirements, limiting scalability.
- **Efficiency gap becomes more pronounced at scale**: methods that balance robustness and efficiency on CIFAR-10 become less efficient at ImageNet scale, as computational costs dominate.

5. Benchmark Limitations

The purpose of BenchmarkCtrs is controlled comparison, not exhaustive method ranking. The benchmark isolates training logic, standardizes the surrounding infrastructure, and exposes quality–efficiency trade-offs that are otherwise difficult to compare across papers. These strengths also define the limits of the evidence: the conclusions describe the behavior of the evaluated methods under the protocol specified in this chapter. We therefore treat the results as comparative evidence for a shared experimental setting, not as a complete ordering of certified training methods across all possible training regimes.

Method and Architecture Coverage. The evaluated baselines cover the main training families used in randomized smoothing: Gaussian augmentation, adversarial smoothing, and regularization-based objectives. This selection is sufficient to expose the dominant efficiency patterns observed in this chapter, especially the cost of repeated noise sampling and the weaker scalability of adversarial inner-loop optimization. However, the benchmark does not include every recent variant of certified training, nor does it evaluate transformer-based vision models, larger convolutional networks, or foundation-model fine-tuning settings. The conclusions therefore apply most directly to the considered ResNet-style workloads on CIFAR-10 and ImageNet. Methods with different architectural bottlenecks or pretraining assumptions may shift the relative balance between throughput, memory, and certified robustness.

Controlled but Limited Hyperparameter Search. The benchmark fixes a controlled training protocol to isolate algorithmic differences. This decision improves comparability, but it also restricts the extent to which each method can be individually optimized. Certified training methods are sensitive to the smoothing noise σ, the number of noise samples, loss weights, batch size, and learning-rate schedule. An exhaustive search over these parameters would blur the distinction between method quality and tuning budget, while a fully fixed configuration can understate the best achievable result for a particular method. Our protocol chooses the latter trade-off deliberately: it prioritizes controlled comparison over per-method optimality.

Certification and Monitoring Constraints. The distinction between training-time monitoring and final evaluation also limits interpretation. Final results use fixed-sample certification to remain comparable with prior randomized smoothing work, whereas CIFAR-10 training dynamics are monitored with adaptive certification to keep the evaluation cost manageable. As discussed in Section 2.3, adaptive estimates are conservative indicators of robustness progression rather than direct substitutes for fixed-sample certified radii. This affects convergence analysis and Pareto-front visualizations, which rely on validation-time robustness estimates. On ImageNet, the cost of certification prevents the same continuous monitoring, so the large-scale analysis is necessarily restricted to aggregate end-of-training behavior.

Metric Aggregation. The Certified Utility Index (CUI) condenses accuracy, robustness, training time, and memory into a single score. This makes trade-offs easier to compare, but it also imposes a particular weighting structure on the benchmark. Equal treatment of the normalized dimensions is appropriate for a general efficiency study, yet different deployment settings may prioritize certified accuracy at a fixed radius, peak ACR, memory footprint, or wall-clock time differently. For this reason, we use CUI as a diagnostic summary rather than as the sole basis for method selection. The Pareto analyses and raw metric tables remain necessary for interpreting why a method scores well under the aggregate measure.

These limitations qualify the scope of the benchmark rather than the central empirical conclusion. Across the evaluated settings, repeated stochastic evaluation remains the main source of computational overhead, and robustness gains do not scale proportionally with training cost. The next section summarizes these findings and connects them to the method-level interventions studied in Chapter 4.

Key Insights

BenchmarkCtrs improves comparability by enforcing a shared protocol, but its conclusions remain conditional on the selected methods, architectures, hyperparameter budget, certification strategy, and utility weighting.

6. Summary

This chapter establishes a systematic perspective on the efficiency–robustness trade-off in certified training for randomized smoothing through the BenchmarkCtrs framework. The framework decouples training logic from experimental infrastructure and enables controlled, reproducible comparisons across methods. A key component of this protocol is the integration of efficient adaptive certification for training-time monitoring alongside fixed-sample certification for final evaluation,

enabling continuous tracking of robustness during training without incurring prohibitive computational overhead. To interpret the framework results, we define an evaluation protocol that covers the joint trade-off between accuracy, certified robustness, and computational cost.

Our results reveal that multi-pass noise sampling is the primary driver of computational inefficiency in certified training. However, we also observe that increased computational investment does not necessarily translate into proportionally stronger certified performance. While training cost scales predictably with the number of noise samples, the resulting robustness gains vary substantially across methods. This decoupling of cost and outcome shifts attention toward the design of training algorithms themselves, rather than purely increasing computational budget. In particular, the benchmark points to two factors that warrant direct investigation: reliance on repeated stochastic passes and the amount of useful information extracted from each pass.

A second key finding is that the relative attractiveness of certified training methods depends strongly on scale. Methods that exhibit highly favorable trade-offs on CIFAR-10 do not necessarily retain those advantages on ImageNet, where computational overhead becomes increasingly dominant. Nevertheless, regularization-based approaches remain the most informative setting for the subsequent analysis, achieving competitive robustness while exposing opportunities to study the effect of computational strategies. In contrast, adversarial training methods such as SmoothAdv consistently occupy a substantially less favorable efficiency regime, suggesting inherent scalability limitations.

Taken together, these findings suggest that the next step is to test approaches that go beyond conventional regularization strategies and more directly address the structural sources of computational overhead. These observations and insights directly motivate the next chapter. Having identified multi-pass noise sampling and robustness-oriented optimization as the dominant sources of overhead, we turn our attention to two complementary lines of analysis. The first, *Loss Projection*, investigates optimization dynamics through the lens of reducing conflicting gradient interactions, while the second, *Selective Backpropagation*, explores how data-centric sample selection changes unnecessary computation during training.

Chapter 4.

Towards Efficient Certified Training: Analysis and Insights

Building on the analysis presented in the previous chapter, we investigate two hypotheses regarding the sources of inefficiency in certified training and evaluate experiments derived from each. Our objective is not to obtain isolated training performance improvements, but to determine which classes of approaches constitute viable directions for efficiency-oriented certified training methods.

We focus on two hypotheses. (i) Whether structural conflicts between the classification and robustness objectives measurably affect the efficiency–robustness trade-off. (ii) Whether redundant computation during training changes that trade-off when only a subset of samples contributes to gradient updates.

These hypotheses lead to two intervention families that stress different parts of the training process. Section 1 studies *Objective Prioritization with Loss Projection* as an optimization-level strategy that alters the interaction between classification and robustness gradients. Its role is to test whether explicit gradient control changes the accuracy, robustness, and cost profile of certified training. Section 2 studies *Subsampling with Selective Backpropagation* as a data-level approach that changes how often individual samples contribute to gradient computation. Its role is to evaluate how selective computation moves models along the efficiency–robustness trade-off, rather than to claim an unconditional improvement in training efficiency.

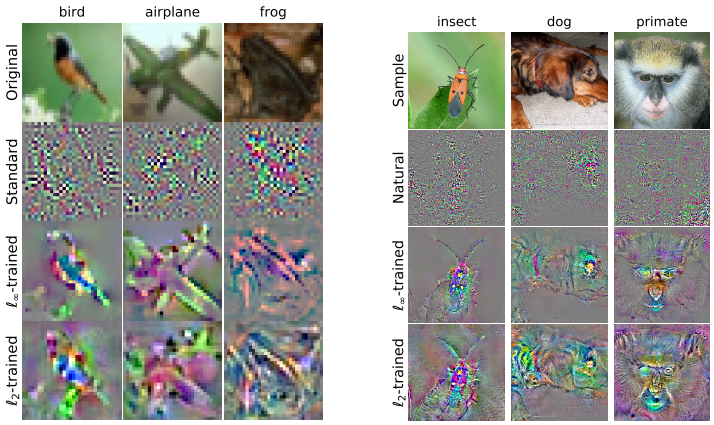
1. Optimization-Level Intervention: Gradient Alignment

This section is concerned with the hypothesis that part of the inefficiency of certified training originates from conflicts between the classification and robustness objectives. Such conflicts can produce competing optimization signals, slow convergence, and change the balance between clean accuracy and certified robustness. We evaluate

that hypothesis using LossProj, a projection-based update rule that modifies the update direction to favor progress on the classification objective. The purpose of this strategy is to measure how explicit gradient control affects the efficiency–robustness trade-off under different smoothing scales and objectives.

In Section 1.1, we highlight the presence of gradient conflict through evidence from both prior work and our own experiments. We then formulate the projection procedure in Section 1.2 and evaluate its empirical behavior in Section 1.3 according to the unified evaluation protocol established in Chapter 3.

1.1. Empirical Analysis: Evidence of Objective Conflict



(a) CIFAR-10 (b) Restricted ImageNet

Figure 4.1.: **Optimization objectives shape learned features**. Visualization of the gradient of the classification loss with respect to the input image, highlighting the input features that most strongly influence the model's predictions (adapted from Tsipras et al. [42]). Although all models correctly classify the same images, the features emphasized by naturally and robustly trained models differ substantially. This illustrates that changing the training objective fundamentally alters the representations learned by the model.

As detailed in Chapter 2, regularization-based certified training typically optimizes a composite objective of the form: $\mathcal{L} = \mathcal{L}_{\text{CE}} + \lambda \mathcal{L}_{\text{R}}$, where $\mathcal{L}_{\text{CE}}$ denotes the standard cross-entropy loss and $\mathcal{L}_{\text{R}}$ is a robustness-promoting regularizer. This formulation implicitly assumes that progress on both objectives can be achieved through a weighted combination of their gradients. We argue that the gradients of these two terms often conflict, leading to optimization inefficiencies where progress in one objective comes at the expense of the other.

We motivate this claim through two complementary perspectives. The first views the conflict at *the representational level*, where standard accuracy and robustness

favor different feature representations. The second examines the conflict directly in *the parameter gradient space* through the alignment of their corresponding gradients. Together, these perspectives suggest that standard gradient aggregation may introduce optimization inefficiencies that warrant more explicit handling.

Representational Conflict

Before examining the optimization dynamics directly, it is useful to consider why a conflict between classification and robustness objectives might arise in the first place. One explanation is that the two objectives encourage different feature representations, creating competing pressures throughout training. Tsipras et al. [42] show that these two objectives can be fundamentally misaligned: for certain data distributions, improved robustness can provably require sacrificing standard accuracy. The authors attribute this trade-off to the existence of *brittle features*: patterns that are highly predictive under standard conditions but can be easily manipulated by an adversary to induce misclassification.

Input-space gradient visualizations (Figure 4.1) illustrate how robust training alters the representations learned by a model. Rather than relying on all predictive features, robust models increasingly favor stable features that remain informative under perturbation. This shift improves robustness, but it also suppresses features that would otherwise contribute to standard classification performance.

From this perspective, the accuracy–robustness trade-off arises because the two objectives encourage different feature representations. The classification objective benefits from exploiting all predictive signals available in the data, whereas the robustness objective actively discourages reliance on features that are vulnerable to adversarial manipulation. While this argument is typically framed in terms of representation learning, it also suggests the presence of competing optimization pressures during training. If the two objectives favor different representations, their corresponding gradients may likewise favor different update directions in parameter space.

Gradient Conflict

The representational perspective suggests that classification and robustness objectives may favor different update directions during training. If this is the case, then the corresponding gradients will interfere with one another in parameter space, reducing the effectiveness of each optimization step. In practice, such interference can increase the number of iterations required to reach a given performance level, contributing to the high computational cost associated with certified training. To examine whether such interference occurs in practice, we analyze the alignment between the classification and robustness gradients throughout training. We quantify

this alignment using the cosine similarity,

$$\cos\phi = \frac{\left\langle \tilde{\nabla}\,\mathcal{L}_{\mathrm{CE}}, \tilde{\nabla}\,\mathcal{L}_{\mathrm{R}} \right\rangle}{\|\tilde{\nabla}\,\mathcal{L}_{\mathrm{CE}}\|\|\tilde{\nabla}\,\mathcal{L}_{\mathrm{R}}\|}.$$

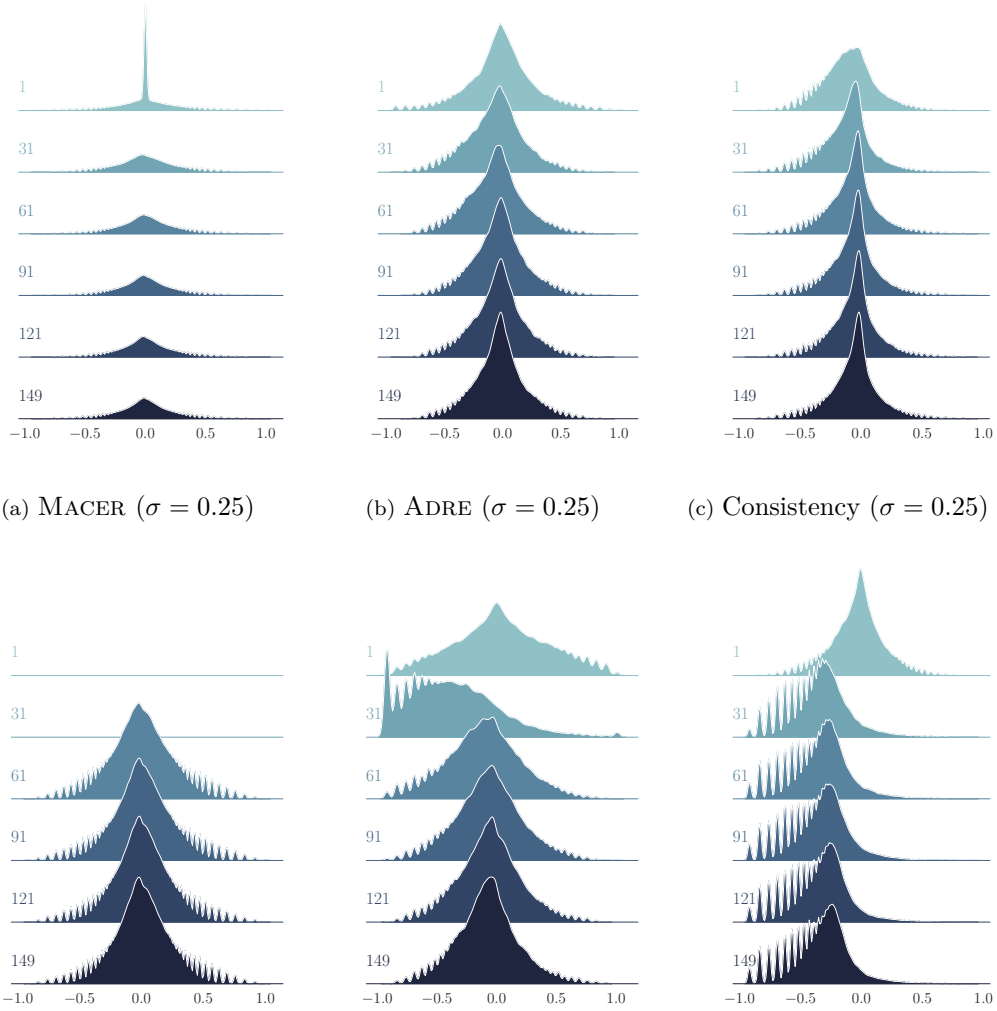
Figure 4.2 tracks the distribution of $\cos\phi$ throughout training for several certified training objectives. For $\mathcal{L}_{\textsc{Macer}}$, the distribution remains centered around zero ($-0.2 \leq \cos\phi \leq 0.2$), indicating that the classification and robustness gradients are largely orthogonal. In contrast, both $\mathcal{L}_{\textsc{Adre}}$ and $\mathcal{L}_{\mathrm{Consistency}}$ exhibit a pronounced negative shift, with the distribution center moving from approximately -0.1 to -0.6 as the noise level increases. These results indicate that the robustness gradient increasingly opposes the classification gradient under stronger robustness constraints.

This observation provides direct evidence that the tension identified at the representational level also manifests in the optimization dynamics.

While the severity of the conflict varies across training objectives, the consistently negative alignment observed for $\mathcal{L}_{\textsc{Adre}}$ and $\mathcal{L}_{\mathrm{Consistency}}$ suggests that simple linear aggregation may require unnecessary optimization effort to balance competing objectives. This motivates optimization strategies that explicitly account for conflicting gradient directions when improving certified training efficiency.

Key Insights

Regularized certified training exhibits a persistent structural gradient conflict in parameter space, where the robustness regularizer gradient actively opposes the classification loss gradient.



(a) MACER ($\sigma = 0.25$) (b) ADRE ($\sigma = 0.25$) (c) Consistency ($\sigma = 0.25$)

(d) MACER ($\sigma = 1.0$) (e) ADRE ($\sigma = 1.0$) (f) Consistency ($\sigma = 1.0$)

Figure 4.2.: **Evolution of Gradient Conflict Between Classification and Robustness Objectives**. The figure illustrates the distribution of cosine similarity between the cross-entropy gradient and different robustness regularization gradients throughout training. Each plot corresponds to a different robustness objective, with the distribution evaluated at regular training intervals to show how gradient alignment evolves over time. The cosine similarity measures the directional agreement between the two objectives: values close to 1 indicate aligned updates, values near 0 indicate independent directions, and negative values indicate conflicting gradients. The evolution of these distributions reveals whether the robustness objective consistently reinforces or opposes the classification objective during optimization.

1.2. Intervention: Loss Projection (LossProj)

To investigate whether gradient conflicts can be addressed through explicit optimization control, we evaluate a projection-based update strategy. Rather than combining the classification and robustness gradients through a direct weighted sum, this approach modifies the robustness update by removing the component that interferes with the classification objective. The underlying hypothesis is that preserving progress on clean accuracy while retaining the remaining robustness signal may change the optimization trajectory in measurable ways.

This approach is related to the method proposed by Zhang et al. [49], which constructs low-distortion adversarial examples by optimizing within the tangent hyperplane of the classification loss in input space. In contrast, our approach applies the same principle in parameter space by modifying the model update direction rather than the input perturbation.

A practical limitation of this approach is the additional computational cost required to obtain the two individual gradients, $\tilde{\nabla}\,\mathcal{L}_{\mathrm{CE}}$ and $\tilde{\nabla}\,\mathcal{L}_{\mathrm{R}}$. Unlike standard training, which requires only the gradient of the combined objective, loss projection requires separate backward passes for each loss component. This approximately doubles the gradient computation cost per iteration, creating a trade-off between optimization control and training efficiency.

Adaptive Directional Scaling. To control the relative contribution of the projected robustness update, we use a learning rate scaling parameter α, such that the robustness-specific learning rate $\eta_R = \alpha\eta$ where η is the base learning rate. This parameter allows the influence of robustness updates to be adjusted independently from the original loss weighting coefficient λ. Consequently, the procedure provides finer control over the convergence behavior of the two objectives without modifying the underlying training objective.

Let the projection of the robustness loss gradient onto the classification loss gradient be:

$$\mathrm{proj}_{\tilde{\nabla}\,\mathcal{L}_{\mathrm{CE}}}\,\tilde{\nabla}\,\mathcal{L}_{\mathrm{R}} = \frac{\left\langle \tilde{\nabla}\,\mathcal{L}_{\mathrm{R}}, \tilde{\nabla}\,\mathcal{L}_{\mathrm{CE}} \right\rangle}{\|\tilde{\nabla}\,\mathcal{L}_{\mathrm{CE}}\|^2}\,\tilde{\nabla}\,\mathcal{L}_{\mathrm{CE}}. \tag{4.1}$$

Algorithm 1 A LossProj Iteration

procedure Train-LossProj($f_{\boldsymbol{\theta}}, \boldsymbol{X}, \mathbf{y}, \eta, \alpha$)
  $\mathbf{g}_{CE} \leftarrow \tilde{\nabla}\,\mathcal{L}_{\mathrm{CE}}\,(f_{\boldsymbol{\theta}}(\boldsymbol{X}), \mathbf{y})$
  $\mathbf{g}_{R} \leftarrow \lambda\,\tilde{\nabla}\,\mathcal{L}_{\mathrm{R}}\,(f_{\boldsymbol{\theta}}(\boldsymbol{X}), \mathbf{y})$
  $\mathbf{g}_{\perp} \leftarrow \mathbf{g}_R - \dfrac{\langle \mathbf{g}_R, \mathbf{g}_{CE} \rangle}{\|\mathbf{g}_{CE}\|^2}\mathbf{g}_{CE}$
  $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - \eta\,(\mathbf{g}_{CE} + \alpha\mathbf{g}_{\perp})$
end procedure

Then, the projected robustness update is obtained by removing this component from the original robustness gradient. The resulting update rule is therefore defined as:

$$\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t - \eta \left(\tilde{\nabla} \mathcal{L}_{\mathrm{CE}} + \alpha\lambda \left(\tilde{\nabla} \mathcal{L}_{\mathrm{R}} - \mathrm{proj}_{\tilde{\nabla} \mathcal{L}_{\mathrm{CE}}} \tilde{\nabla} \mathcal{L}_{\mathrm{R}} \right) \right). \tag{4.2}$$

This formulation preserves the robustness contribution that is orthogonal to the classification objective while preventing robustness updates from directly interfering with classification progress.

Key Insights

We use LossProj to investigate how explicit gradient conflict handling affects the efficiency–robustness trade-off. LossProj removes the component of the robustness gradient that opposes the classification objective and uses an adaptive scaling factor α to control the balance between the two objectives, at the cost of nearly doubling the computational effort per iteration.

1.3. Results: Limited Gains and Scalability Issues

The objective of this evaluation is to determine how explicit gradient-level interventions affect the efficiency–robustness trade-off in certified training. Although the analysis in Section 1.1 identifies interference between classification and robustness objectives during optimization, it remains unclear whether reducing this interference translates into meaningful changes in convergence, accuracy, or robustness. We therefore evaluate LossProj across multiple certified training objectives and noise settings, paying particular attention to the trade-off between optimization behavior and computational cost.

Experimental Setup

Our evaluation seeks to determine whether the optimization effects of LossProj are sufficient to offset its additional computational cost. To isolate the effect of gradient projection, we compare LossProj directly against the corresponding baseline training procedure for each robustness objective under identical training conditions.

We evaluate the method using the Macer, Adre, and Consistency regularizers on CIFAR-10. All experiments are conducted within the BenchmarkCtrs framework using the same system configuration and hyperparameter settings established in Chapter 3. The only additional degree of freedom used by LossProj is the directional scaling parameter α, allowing the influence of the projected robustness update to be adjusted independently of the original loss formulation. All remaining hyperparameters are inherited directly from the corresponding baseline configuration.

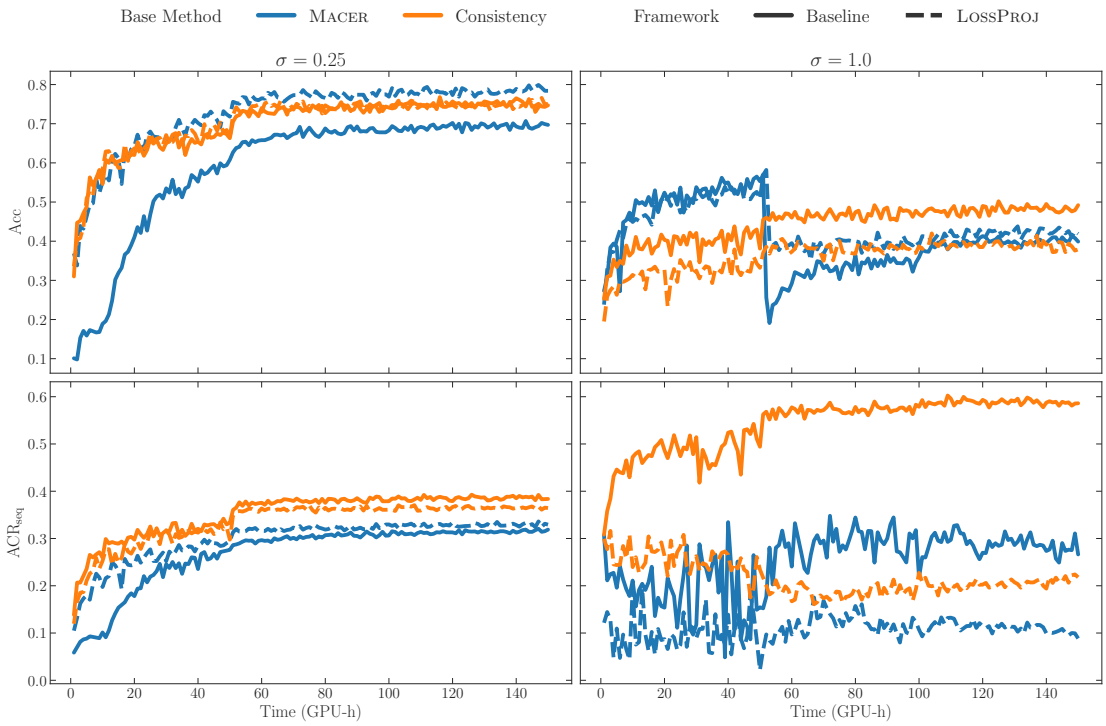


Figure 4.3.: **Training Dynamics under Loss Projection**. Comparison of the optimization dynamics of the baseline methods and the LOSSPROJ framework. The horizontal axis reports training epochs to emphasize data efficiency rather than computational efficiency, as LOSSPROJ incurs a nearly constant increase in per-iteration cost. Solid lines denote the baseline methods, while dashed lines of the corresponding color denote their LOSSPROJ counterparts.

For comparisons involving MACER, we use the reproduced benchmark results described in Chapter 3 rather than the values originally reported by the authors. The selected values of α are reported for each regularizer and noise level: $\alpha = 0.25$ for $\mathcal{L}_{\text{MACER}}$; $\alpha = 1.75$, 0.15, and 0.75 for $\mathcal{L}_{\text{ADRE}}$ at $\sigma = 0.25$, 0.5, and 1.0, respectively; and $\alpha = 0.7$, 0.7, and 0.35 for $\mathcal{L}_{\text{Consistency}}$ at $\sigma = 0.25$, 0.5, and 1.0.

Empirical Results and Analysis

Table 4.1 summarizes the CIFAR-10 evaluation results for LOSSPROJ. The experiments provide limited evidence that explicitly resolving gradient conflicts improves the efficiency–robustness trade-off. While the projection yields measurable changes in optimization stability and clean accuracy under low-noise settings, these effects diminish as the robustness objective becomes stronger and are consistently offset by the additional computational cost required to compute separate gradients.

Effects are concentrated in low-noise regimes. At $\sigma = 0.25$, LOSSPROJ increases the clean accuracy of MACER while maintaining a comparable ACR. The training

trajectory is also noticeably more stable, leading to faster convergence of the robustness objective and reducing the time-to-convergence by approximately 0.35 GPU-h ($\approx$ 21 minutes). These results suggest that, under relatively weak perturbation, mitigating gradient interference can change optimization behavior without sacrificing robustness. In contrast, the same strategy has little measurable effect when applied to Consistency regularization despite incurring the same computational overhead.

Effects do not scale with noise. As the smoothing noise level increases, the advantages of gradient projection deteriorate rapidly. For $\sigma \in \{0.5, 1.0\}$, robustness degrades relative to the baseline and the convergence changes observed at $\sigma = 0.25$ disappear entirely. In several configurations, LossProj requires more training to converge than the original method. These results indicate that the effectiveness of gradient projection is highly sensitive to the strength of the robustness objective and does not generalize across the range of operating regimes considered in this thesis.

Figure 4.3 further illustrates this behavior by visualizing the amount of training data required to reach the final clean accuracy and ACR values obtained by each method. To isolate the data efficiency of the optimization process, the horizontal axis is reported in training epochs rather than wall-clock time. Since LossProj nearly doubles the computational cost of each iteration, using elapsed training time would largely reflect this known overhead and obscure differences in optimization efficiency. For conciseness, the $\sigma = 0.5$ case is omitted. The accuracy plots show that LossProj can substantially increase clean accuracy for Macer at $\sigma = 0.25$. Although this effect becomes smaller at higher noise levels, the training curves remain noticeably smoother, indicating that the projection step succeeds in reducing short-term optimization interference. This stabilization effect, however, does not translate into a consistent robustness benefit.

The ACR trajectories reinforce this conclusion. While LossProj is generally capable of maintaining comparable robustness at lower noise levels, it does not accelerate robustness convergence and becomes increasingly ineffective as the smoothing scale increases. For Consistency regularization, the projection step has no meaningful positive effect and ultimately degrades performance at $\sigma = 1.0$. These observations suggest that reducing local gradient conflict alone is insufficient to meaningfully alter the long-term robustness dynamics of certified training.

Steep increase in computational costs. The central motivation behind LossProj is the hypothesis that gradient conflict contributes to inefficient training dynamics. The empirical results provide partial support for this hypothesis: the projection step often stabilizes training and can increase clean accuracy under favorable conditions. However, the magnitude of these changes remains modest relative to the additional computational cost introduced by the method. Because separate gradients must

be computed for the classification and robustness objectives, iteration time nearly doubles across all configurations. The reductions in the number of epochs required for convergence are therefore insufficient to produce a favorable overall cost-quality trade-off.

Taken together, these results indicate that gradient conflict is not a particularly strong leverage point under the evaluated conditions. While local gradient projection can stabilize optimization in specific low-noise settings, the resulting changes are neither robust across settings nor favorable after accounting for cost. The projection's substantial computational overhead ultimately outweighs the observed quality changes, leaving little justification for extending the evaluation to ImageNet. We therefore conclude that local gradient alignment alone is unlikely to address the primary efficiency bottlenecks of certified training. Consequently, we shift our focus from modifying the fixed-sample optimization process to reducing unnecessary computation within each individual optimization step. The next section evaluates how selective backpropagation changes the efficiency–robustness trade-off when computation is allocated only to selected samples.

Key Insights

- LossProj improves convergence and clean accuracy for Macer under low-noise conditions.
- The effects diminish under stronger robustness constraints, where robustness and convergence degrade.
- The additional cost of separate gradient computations outweighs the observed gains in accutacy and convergence.
- Gradient conflict explains part of the optimization difficulty in certified training, but addressing it alone does not provide a scalable solution.

σ	Method	Time ↓	TP ↑	Mem ↓	Convergence *Acc* ↓	Convergence *ACR* ↓	Acc_{full} ↑	$\text{ACR}_{\text{fixed}}$ ↑	CUI ↑
0.25	MACER	9.12	224	**85.0**	62	96	67.9	0.451	0.268
	LOSSPROJ ($\mathcal{L}_{\text{MACER}}$)	16.48	124	5,209.4	**50**	**50**	74.6	0.452	0.050
	Consistency	**1.19**	**1,713**	87.3	51	51	**76.2**	**0.551**	**0.809**
	LOSSPROJ ($\mathcal{L}_{\text{Consistency}}$)	2.32	880	1,377.5	**50**	**50**	75.6	0.514	0.188
0.50	MACER	9.09	225	**85.0**	97	127	40.9	0.394	0.185
	LOSSPROJ ($\mathcal{L}_{\text{MACER}}$)	16.43	124	5,209.4	**50**	144	22.3	0.213	0.048
	Consistency	**1.20**	**1,706**	87.3	**50**	**50**	**64.1**	**0.719**	**0.819**
	LOSSPROJ ($\mathcal{L}_{\text{Consistency}}$)	2.33	878	1,377.5	147	148	50.0	0.448	0.153
1.00	MACER	9.05	226	**85.0**	109	149	23.0	0.324	0.160
	LOSSPROJ ($\mathcal{L}_{\text{MACER}}$)	13.88	147	5,209.4	**104**	149	8.1	0.061	0.038
	Consistency	**1.19**	**1,722**	87.3	122	**64**	**46.6**	**0.755**	**0.569**
	LOSSPROJ ($\mathcal{L}_{\text{Consistency}}$)	2.31	882	1,377.5	148	148	19.9	0.288	0.143

Table 4.1.: **The key LOSSPROJ metrics on CIFAR-10**. LOSSPROJ is applied with various robustness penalties that corresponds to the projected robustness loss. Robustness is evaluated using fixed-sample certification. The specific setup of each run is described on Section 1.3. Baseline values for MACER correspond to our reproduced metrics in accordance with our dual-reporting approach (see Chapter 3).

σ	Method	$\text{ACR}_{\text{fixed}} \uparrow$	0.0	0.25	0.5	0.75	1.0	1.25	1.5	1.75	2.0	2.25	2.5
0.25	MACER	0.451	67.9	58.7	47.2	34.4	0.0	0.0	0.0	0.0	0.0	0.0	0.0
	LOSSPROJ ($\mathcal{L}_{\text{MACER}}$)	0.452	74.6	61.7	46.5	30.8	0.0	0.0	0.0	0.0	0.0	0.0	0.0
	Consistency	0.551	76.2	67.9	57.9	46.4	0.0	0.0	0.0	0.0	0.0	0.0	0.0
	LOSSPROJ ($\mathcal{L}_{\text{Consistency}}$)	0.514	75.6	65.5	54.0	40.7	0.0	0.0	0.0	0.0	0.0	0.0	0.0
0.50	MACER	0.394	40.9	35.4	29.2	23.9	18.8	14.1	10.1	6.0	0.0	0.0	0.0
	LOSSPROJ ($\mathcal{L}_{\text{MACER}}$)	0.213	22.3	19.1	16.0	13.0	10.1	7.3	5.0	3.6	0.0	0.0	0.0
	Consistency	0.719	64.1	57.2	50.3	43.2	36.3	29.5	22.9	15.9	0.0	0.0	0.0
	LOSSPROJ ($\mathcal{L}_{\text{Consistency}}$)	0.448	50.0	42.2	34.4	27.3	20.8	14.8	9.7	5.5	0.0	0.0	0.0
1.00	MACER	0.324	23.0	20.5	17.9	15.4	13.2	11.1	9.1	7.4	6.1	4.8	3.8
	LOSSPROJ ($\mathcal{L}_{\text{MACER}}$)	0.061	8.1	5.9	3.9	2.6	1.7	1.3	1.1	0.9	0.7	0.6	0.5
	Consistency	0.755	46.6	42.5	38.3	34.0	30.1	26.4	22.9	19.6	16.6	13.7	11.2
	LOSSPROJ ($\mathcal{L}_{\text{Consistency}}$)	0.288	19.9	17.7	15.5	13.4	11.5	9.8	8.3	6.9	5.6	4.6	3.7

Table 4.2.: **The approximate certified accuracy at various thresholds ϵ for LOSSPROJ on CIFAR-10**. The projected loss column indicates the robustness penalty used. Results obtained with fixed-sample certification.

2. Data-Level Intervention: Exploiting Redundancy

This section explores a different intervention point in certified training: reducing unnecessary computation rather than accelerating individual optimization steps. Since robustness estimation relies on repeated stochastic perturbation samples, the computational cost of training grows directly with the number of evaluated samples. We investigate how reducing this cost by identifying and prioritizing selected samples affects the efficiency–robustness trade-off.

To this end, we study existing selective computation methods and their limitations, decouple the selection and optimization objectives of selective backpropagation, and evaluate how this approach changes training cost, clean accuracy, and certified robustness.

2.1. Hypothesis: Can Data Redundancy Be Leveraged?

Selective Backpropagation, introduced by Jiang et al. [17], accelerates training by exploiting redundancy in the training data. Rather than performing a backward pass for every sample, the method first evaluates sample difficulty through a forward pass and selectively backpropagates only the most informative examples. Since gradient computation dominates the cost of training, this selective execution can substantially reduce iteration time while maintaining predictive performance.

The effectiveness of Selective Backpropagation relies on the assumption that loss is a reliable proxy for sample importance. Samples with high loss are more likely to contribute useful gradient information, making them natural candidates for prioritization. This assumption has proven effective in standard supervised learning, where loss-based selection can reduce computation with only minor effects on model quality.

However, subsequent work has shown that this behavior deteriorates in noisy training environments [27]. Samples affected by label noise, ambiguity, corruption, or distributional irregularities often exhibit disproportionately high loss values despite contributing little to generalization. As a result, loss-based prioritization increasingly allocates computation toward uninformative examples, reducing the effectiveness of selective training.

This observation is particularly relevant for certified training, where stochastic perturbation sampling introduces substantial noise into the optimization process. Since conventional loss-based selection is already known to struggle under noisy conditions, directly applying Selective Backpropagation is unlikely to recover the efficiency gains observed in standard training. This raises a more specific question: can robustness-aware indicators provide a more reliable notion of sample importance and thereby change the trade-off induced by selective computation in certified training?

2.2. Selection–Optimization Decoupling

In this section, we investigate whether robustness indicators can provide a more reliable notion of sample importance for certified training. The underlying hypothesis is that, unlike the clean classification loss, robustness metrics are derived from multiple stochastic perturbations and may therefore offer a more stable estimate of the samples that most influence robustness performance.

To evaluate this, we decouple the optimization objective from the sample selection criterion in Selective Backpropagation. Rather than selecting samples based solely on the training loss, the approach uses a selection metric $\mathcal{M}$ that can be chosen independently of the optimization objective $\mathcal{L}$. This allows robustness regularization terms, such as the MACER radius estimate or the Consistency variance, to be evaluated directly as indicators of sample importance. Samples with larger values of $\mathcal{M}$ should be selected more frequently. Following the original Selective Backpropagation formulation, we convert metric values into selection probabilities using their empirical rank within a running distribution of recently observed samples.

Let $(\mathbf{x}_i, y_i)$ be a training sample, CDF be the cumulative density function for the selection metric over the training set, β_{select} be the selectivity factor, and f be the base classifier. The probability of backpropagation with the sample is:

$$\mathbb{P}(\text{select} \mid \mathbf{x}_i, y_i) = \left[\text{CDF}\left(\mathcal{M}(f(\mathbf{x}_i), y_i)\right)\right]^{\beta_{select}}. \tag{4.3}$$

Since the true distribution of $\mathcal{M}$ is unknown and evolves throughout training, we estimate the cumulative distribution function using a fixed-length queue $Q_{\mathcal{M}}$ containing the L most recent metric values. The empirical estimate is given by

$$\text{CDF}_R(u) = \frac{|\{t \in Q_{\mathcal{M}} \mid t < u\}|}{L},$$

which corresponds to the fraction of recently observed samples with metric values smaller than u.

2.3. Selective Backpropagation (SB)

The resulting Selective Backpropagation (SB) configuration is summarized in Algorithms 2 to 4. Its primary extension over the original formulation is the decoupling of the optimization objective $\mathcal{L}$ from the sample selection metric $\mathcal{M}$. This allows the model to be optimized using a certified training objective while independently selecting samples according to robustness-aware importance indicators.

The configuration trains a model f on the training set $\mathcal{D}_{\text{train}}$ with noise level σ, using mini-batches of size B. Sample selection is controlled through the selectivity parameter β_{select} and an empirical metric distribution estimated from a memory of

Algorithm 2 A Training Epoch in the SB Framework (see Algorithm 1 in [17])

procedure TRAINSB($B, L, \beta_{select}, p_{min}, b$)
  $Q_{bp} \leftarrow [\,]$
  for all $(\boldsymbol{X}, \mathbf{y}) \in \mathcal{D}_{\text{train}}$ **do**
    SELECTBP($Q_{bp}, \boldsymbol{X}, \mathbf{y}$) ▷ See Algorithm 3
    if $|Q_{bp}| \geq B$ **then**
      $(\boldsymbol{X}_{bp}, \mathbf{y}_{bp}) \leftarrow Q_{bp}[: B]$
      TRAINOBJECTIVE($f_{\boldsymbol{\theta}}, \boldsymbol{X}_{bp}, \mathbf{y}_{bp}, \mathcal{L}$) ▷ See Algorithm 4
      $Q_{bp} \leftarrow Q_{bp}[B + 1 :]$
    end if
  end for
end procedure

the L most recent metric values. To avoid overly aggressive prioritization during the early stages of training, we additionally use a warmup period of T_w iterations and a minimum selection probability p_{min}.

Distributed Training. In distributed settings, the stochastic number of selected samples per iteration can lead to poor hardware utilization and excessive synchronization overhead. To preserve the regularization effect of randomized sample selection while maintaining efficient parallel execution, we enforce a fixed number of selected samples b in each iteration.

Let n denote the number of samples selected by the stochastic procedure. If $n > b$, excess samples are discarded uniformly from the selected set. Conversely, if $n < b$, the remaining $b - n$ samples are filled using the highest-ranked unselected samples according to the selection metric. This modification preserves the stochastic nature of Selective Backpropagation while ensuring predictable computational workload across distributed workers.

Key Insights

Selective Backpropagation (SB) lets us evaluate how robustness-aware sample prioritization changes training time, clean accuracy, and certified robustness.

Algorithm 3 Sample Selection Algorithm

$Q_{\mathcal{M}} \leftarrow \text{deque}(L)$
procedure SELECTBP($Q_{bp}, \boldsymbol{X}, \mathbf{y}, \beta_{select}, B, p_{min}, b$)
    $\tilde{\boldsymbol{X}} \sim \mathcal{N}(\boldsymbol{X}, \sigma^2 \boldsymbol{I})$
    $\boldsymbol{M} \leftarrow \mathcal{M}\left(f_{\boldsymbol{\theta}}(\tilde{\boldsymbol{X}}), \mathbf{y}\right)$ ▷ Forward pass
    $i \leftarrow 1$
    $n \leftarrow 0$
    while $i \leq B$ **and** $n < b$ **do**
        $q \sim \text{Uniform}(0, 1)$
        $p_{select} \leftarrow \text{CDF}_{\text{R}}(\boldsymbol{M}_i; Q_{\mathcal{M}})^{\beta_{select}}$
        if $q \leq \max(p_{select}, p_{min})$ **then**
            APPEND($Q_{bp}, (\boldsymbol{X}_i, \mathbf{y}_i)$)
            $n \leftarrow n + 1$
        end if
        PUSH($Q_{\mathcal{M}}, m$)
    end while
    if distributed **and** $n < b$ **then** ▷ Add top $b - n$ samples
        $\boldsymbol{M}_{unused} \leftarrow \{\boldsymbol{M}_i \mid 1 \leq i \leq B, \boldsymbol{X}_i \notin Q_{bp}\}$
        SORTDESC($\boldsymbol{M}_{unused}$)
        for all $\boldsymbol{M}_i \in \boldsymbol{M}_{unused}[: b - n]$ **do**
            APPEND($Q_{bp}, (\boldsymbol{X}_i, \mathbf{y}_i)$)
        end for
    end if
end procedure

Algorithm 4 Training using the Optimization Objective

procedure TRAINOBJECTIVE($f_{\boldsymbol{\theta}}, \boldsymbol{X}, \mathbf{y}, \mathcal{L}$)
    $\tilde{\boldsymbol{X}} \sim \mathcal{N}(\boldsymbol{X}, \sigma^2 \boldsymbol{I})$
    $l \leftarrow \mathcal{L}(f_{\boldsymbol{\theta}}(\tilde{\boldsymbol{X}}), \mathbf{y})$ ▷ Forward pass
    $\tilde{\nabla}\,\mathcal{L} \leftarrow \text{Backpropagate}(l)$ ▷ Backward pass
    UPDATEPARAMETERS($f_{\boldsymbol{\theta}}, \tilde{\nabla}\,\mathcal{L}$)
end procedure

2.4. Results: Scalable Efficiency Gains

The objective of this evaluation is to determine how robustness-aware sample selection affects the efficiency–robustness trade-off in certified training. Since the approach decouples the optimization objective from the sample selection criterion, we first investigate which selection signals provide the most effective notion of sample importance. In particular, we compare both classification- and robustness-based metrics, as well as the conventional setting where the same objective is used for optimization and sample selection.

Having identified the most informative selection strategies, we evaluate their impact on training cost and model quality on CIFAR-10. We then extend the strongest configuration to ImageNet to assess whether the observed trade-off persists at larger scale. Finally, we compare the results across datasets to identify which conclusions generalize beyond a particular benchmark or robustness objective.

All experiments are conducted within the BenchmarkCtrs framework using the same baseline configurations and evaluation protocol established in Chapter 3. Unless otherwise stated, the reported efficiency metrics account for both training-time reductions and any resulting changes in robustness or clean accuracy.

Selection Metric Analysis

	$\mathcal{M} \coloneqq \mathcal{L}_{\text{CE}} + \lambda\mathcal{L}_{\text{R}}$		$\mathcal{M} \coloneqq \lambda\mathcal{L}_{\text{R}}$	
$\mathcal{L}_{\text{R}}$	Accuracy	ACR_{seq}	Accuracy	ACR_{seq}
$\mathcal{L}_{\text{R}} \coloneqq 0$	$\mathbf{0.56 \pm 0.05}$	0.316 ± 0.030	—	—
$\mathcal{L}_{\text{ADRE}}$	$\mathbf{0.56 \pm 0.05}$	0.312 ± 0.029	0.54 ± 0.05	0.311 ± 0.032
$\mathcal{L}_{\text{MACER}}$	0.55 ± 0.05	$\mathbf{0.342 \pm 0.034}$	0.50 ± 0.04	0.339 ± 0.032
$\mathcal{L}_{\text{Consistency}}$	$\mathbf{0.56 \pm 0.05}$	0.323 ± 0.033	$\mathbf{0.55 \pm 0.05}$	$\mathbf{0.346 \pm 0.037}$

Table 4.3.: **Selection Metric Performance**. A comparison of the validation accuracy and ACR of various selection metrics $\mathcal{M}$ on CIFAR-10. The metrics considered consist of regularization terms ($\mathcal{L}_{\text{R}}$) optionally combined with the cross-entropy loss ($\mathcal{L}_{\text{CE}}$). Reported values are the average of 3 trained models per metric.

Based on the benchmark results, we select Consistency regularization as the optimization objective for the selective backpropagation experiments. It reaches comparable robustness performance to more computationally expensive approaches while giving the selective-computation experiments a more favorable baseline cost profile. Therefore, all subsequent experiments optimize the objective

$$\mathcal{L} \coloneqq \mathcal{L}_{\text{CE}} + \mathcal{L}_{\text{Consistency}}.$$

With the optimization objective fixed, the remaining design choice is the selection metric $\mathcal{M}$. Since SB decouples sample selection from optimization, we investigate whether robustness-specific indicators provide a better estimate of sample importance than the conventional classification loss. We evaluate three robustness-based metrics derived from the regularization terms $\mathcal{L}_{\text{MACER}}$, $\mathcal{L}_{\text{ADRE}}$, and $\mathcal{L}_{\text{Consistency}}$, both independently and combined with the cross-entropy loss $\mathcal{L}_{\text{CE}}$. As an additional baseline, we evaluate selection based solely on $\mathcal{L}_{\text{CE}}$.

Each configuration is evaluated using three random seeds with $\sigma = 0.5$, and the mean and standard deviation of clean accuracy and ACR_{seq} on the CIFAR-10 validation set are reported in Table 4.3. The results show that selection based on the Consistency regularization term provides the strongest robustness performance without introducing a measurable accuracy degradation. This suggests that, under stochastic robustness training, the robustness objective itself provides a more informative selection signal than the classification loss on its own.

Consequently, we use $\mathcal{M} := \mathcal{L}_{\text{Consistency}}$ as the selection metric for the remaining evaluation of the selective backpropagation framework.

Experimental Setup

All following experiments use the Consistency regularization objective and inherit the corresponding baseline configuration established in Chapter 3. The SB configuration uses five additional hyperparameters: (i) the number of warmup epochs T_w, (ii) the selection metric memory length L, (iii) the selectivity parameter β_{select}, (iv) the minimum selection probability p_{min}, and (v) the fixed selected batch size b.

To evaluate the trade-off between computational savings and sample coverage, we report two operating points on CIFAR-10: a *high-selectivity* configuration and a *low-selectivity* configuration. The high-selectivity configuration uses $T_w = 5$ epochs, $L = |\mathcal{D}_{\text{train}}|$, $\beta_{select} = 3.0$, and $p_{min} = 0.05$. The low-selectivity configuration uses $T_w = 15$ epochs, $L = 2|\mathcal{D}_{\text{train}}|$, $\beta_{select} = 2.0$, and $p_{min} = 0.2$.

The high-selectivity configuration prioritizes computational savings by aggressively filtering samples, whereas the low-selectivity configuration retains a larger fraction of the training data throughout optimization. Unless otherwise specified, we refer to these configurations as *SB (high)* and *SB (low)*, respectively.

The CIFAR-10 experiments are conducted in a non-distributed setting and therefore do not use the fixed-batch parameter b. For ImageNet, we evaluate only the high-selectivity configuration and enable distributed SB with $b = 40$ selected samples per iteration (10 per GPU).

Results on CIFAR-10

We first evaluate how selective backpropagation changes training cost, robustness, and accuracy on CIFAR-10. Table 4.4 and Section 2.4 summarize the results of the SB configurations compared with the baseline Consistency method, reporting both

σ	Method	Time ↓	TP ↑	Mem ↓	Conv_{Acc} ↓	Acc_{full} ↑	$\text{ACR}_{\text{fixed}}$ ↑	CUI ↑
0.25	Consistency	1.19	1,713	**87.3**	51	**76.2**	**0.551**	0.809
	SB (high)	1.04	1,969	91.2	**50**	72.4	0.520	0.803
	SB (low)	**0.82**	**2,490**	91.4	**50**	70.2	0.494	**0.815**
0.50	Consistency	1.20	1,706	**87.3**	**50**	**64.1**	**0.719**	**0.819**
	SB (high)	1.03	1,977	91.2	52	61.2	0.671	0.806
	SB (low)	**0.82**	**2,476**	91.4	52	58.7	0.632	0.815
1.00	Consistency	1.19	1,722	**87.3**	**122**	**46.6**	**0.755**	0.569
	SB (high)	1.03	1,981	91.2	146	44.1	0.706	0.543
	SB (low)	**0.83**	**2,471**	91.4	130	43.2	0.673	**0.604**

Table 4.4.: **The key SB metrics on CIFAR-10**. Direct comparison of the performance of the SB framework with $\mathcal{M} = \mathcal{L}_{\text{Consistency}}$ as a selection metric in contrast with the best performing baselines. An upward pointing arrow ↑ in a column header indicates that higher values are better, and a downward pointing arrow ↓ indicates that lower values are better. Values highlighted in **bold** represent the best value for the metric in that section of the table.

aggregate efficiency metrics and certified accuracy values. We evaluate two selection configurations: *SB (low)*, which selects approximately 39% of training samples per epoch, and *SB (high)*, which selects approximately 26%. These correspond to performing a backward pass in roughly one out of three and one out of four iterations, respectively.

Overall Trends. The results reveal a consistent efficiency-quality trade-off. The *SB (low)* configuration remains closer to the baseline in terms of robustness, with an ACR reduction of approximately 6– 7% and certified accuracy reductions below 4% across all noise levels. However, these improvements in quality retention come at the cost of more limited efficiency gains, reducing training time by only 0.16 GPU-h on average (13%). In contrast, the *SB (high)* configuration achieves a substantially larger reduction of (0.37) GPU-h (31%) while maintaining comparable clean accuracy. Although the ACR reduction increases to approximately 10–12%, the degradation in certified accuracy remains limited, with maximum reductions of 6%, 5%, and 3% for $\sigma = 0.25$, 0.5, and 1.0, respectively.

This difference emphasizes that the primary effect of selective computation is not improving the robustness frontier, but reducing the computational cost required to reach comparable performance. In particular, the robustness degradation measured by ACR does not translate proportionally to certified accuracy degradation. Since

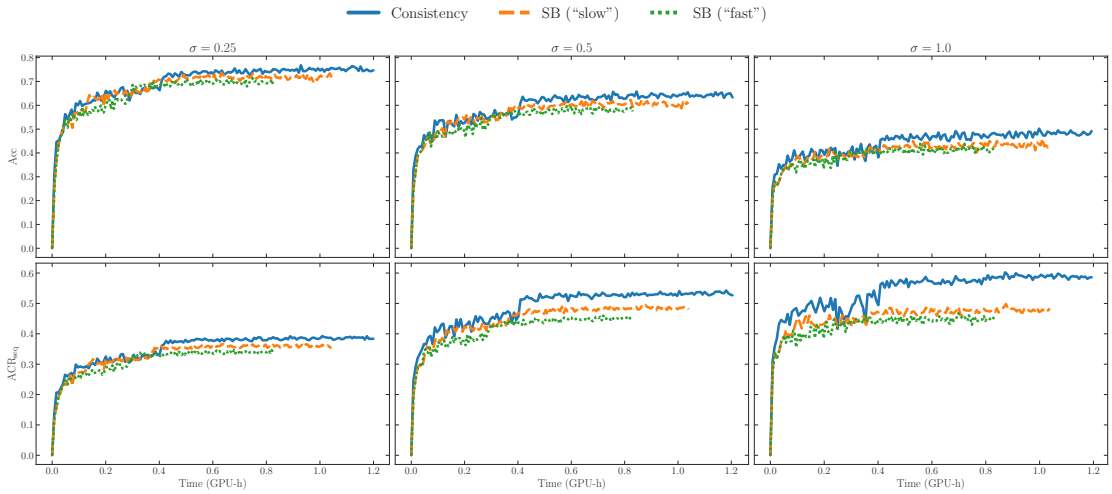


Figure 4.4.: **Training Dynamics and Time-to-Result with SB**. Comparison of the validation accuracy and sequential average certified radius ($\mathrm{ACR}_{\mathrm{seq}}$) over training time for the baseline Consistency method and the two Selective Backpropagation (SB) configurations. The curves illustrate how selectively skipping backward passes affects optimization throughout training while accounting for the actual computational time required. Similar trajectories indicate that SB preserves the optimization behavior of the baseline, while the leftward shift of the curves reflects the reduction in training time achieved through selective computation.

ACR is an average robustness measure, it is sensitive to changes in the tail of the certified radius distribution. This effect is amplified when using CertifySeq, which tends to underestimate certified radii due to its more conservative estimations. Appendix B documents this effect by comparing $\mathrm{Acc}_{\mathrm{seq}}$ and $\mathrm{Acc}_{\mathrm{fixed}}$ under identical certification parameters.

Training Dynamics. Figure 4.4 further illustrates the training dynamics of both SB configurations compared with the baseline. The training trajectories show that selective computation preserves a similar optimization behavior throughout training, with only minor deviations from the baseline at lower noise levels. Although the *SB (low)* configuration generally maintains slightly better robustness, the difference compared with the *SB (high)* configuration remains limited, suggesting that more aggressive selection does not substantially alter the optimization trajectory.

Trade-off Analysis. The Pareto analysis in Figure 4.5 provides a broader view of these trade-offs. At lower noise levels ($\sigma \in 0.25, 0.5$), both SB configurations shift the efficiency-quality frontier because they retain comparable accuracy and robustness to the Consistency and SmoothAdv baselines at reduced training cost. However, the approach does not improve the inherent accuracy-robustness trade-off, as its effects are concentrated along the efficiency dimension. At $\sigma = 1.0$, the shift

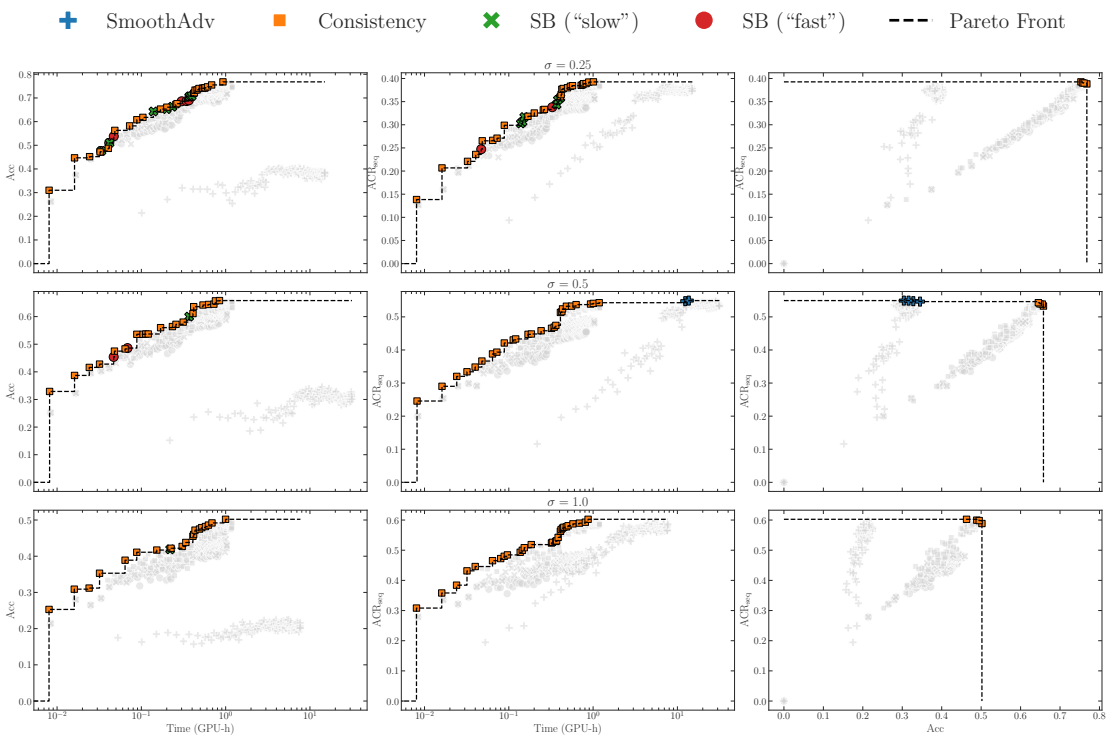


Figure 4.5.: **Training-Time Efficiency–Robustness Trade-offs**. The plots illustrate the trade-offs during training, the metrics are evaluated on the CIFAR-10 validation set. Each method from Table 4.4 adds 150 points for its result at the end of each training epoch. The rows correspond to σ values: 0.25, 0.5 and 1.0 in order. The training time in the first two columns is represented on the x-axis in log-scale. The Pareto front line connects the methods that offer dominating performance on at least one metric.

in the efficiency frontier becomes less pronounced, indicating that aggressive sample selection is less effective under highly stochastic training conditions.

The CUI analysis provides a clearer comparison of these efficiency-quality trade-offs. For the *SB (high)* configuration, the reduction in robustness is compensated by the substantial decrease in training cost, resulting in a more favorable balance between the competing objectives compared with the baseline. In contrast, the *SB (low)* configuration preserves more of the baseline robustness but achieves minimal computational savings to justify the remaining degradation. This highlights that the value of selective computation depends not only on preserving model quality, but on whether the cost reductions are large enough to offset the associated performance changes.

Overall, the CIFAR-10 results show that selective backpropagation mainly affects the cost dimension of the efficiency–robustness trade-off. Unlike optimization-centric approaches, which require additional computation to modify the training

trajectory, selective computation achieves efficiency improvements by reducing unnecessary gradient evaluations while maintaining competitive model quality.

Key Insights

- Selective backpropagation reduces certified training cost with limited quality degradation.
- Higher selectivity provides larger efficiency gains at the cost of robustness.
- ACR changes can overstate the practical impact of selective computation.

Results on ImageNet

Following the CIFAR-10 analysis, we evaluate whether the efficiency improvements from selective computation extend to large-scale certified training. Table 4.6 reports the main evaluation metrics for ImageNet, while Table 4.7 provides the corresponding certified accuracy values across different thresholds.

We evaluate only the distributed high-selectivity configuration. This choice follows the CIFAR-10 analysis, where the low-selectivity configuration reduced cost only modestly despite retaining slightly higher robustness. In contrast, the high-selectivity configuration produced a more favorable balance by obtaining substantial computational savings with comparatively small reductions in model quality. Therefore, it represents the most informative configuration for evaluating selective computation at larger scale.

Overall Trends. The results show that the same cost-quality pattern persists at larger scale. Selective backpropagation reduces the total training cost by approximately 30%, corresponding to a saving of 60.4 GPU-h, while maintaining competitive accuracy and robustness. The ACR decreases by approximately 10%, but this reduction is not reflected proportionally in certified accuracy, which degrades by a maximum of 4% across evaluated thresholds. As observed in the CIFAR-10 experiments, this discrepancy highlights the sensitivity of average robustness measures to changes in the tail of the certified radius distribution.

Trade-off Analysis. The CUI comparison shows that the computational savings justify the corresponding quality trade-offs. Rather than improving the accuracy-robustness frontier, selective backpropagation shifts the efficiency frontier by retaining comparable robustness at substantially lower computational cost. This indicates that the observed trade-off is not limited to small-scale settings, but also appears in large-scale certified training where redundant sample evaluations represent a dominant computational burden.

Overall, the results across CIFAR-10 and ImageNet show that data redundancy is a more impactful target for improving certified training efficiency than optimization refinement. While optimization-centric strategies can alleviate specific training challenges, their benefits remain limited by additional computational overhead and

sensitivity to the training regime. In contrast, selective backpropagation directly addresses the dominant source of computational cost by eliminating unnecessary gradient evaluations while preserving the underlying optimization objective. These results position data-level approaches as a critical direction for scaling certified training, particularly as noise sampling and dataset complexity continue to increase.

Key Insights

- Selective backpropagation scales to ImageNet while maintaining competitive robustness.
- Efficiency gains persist across dataset scales, validating the scalability of data-efficient training.
- Computational savings justify moderate robustness trade-offs in certified training.
- Data redundancy is a more effective scalability target than optimization refinement under the evaluated settings.

σ	Method	$\text{ACR}_{\text{fixed}}$ ↑	0.0	0.25	0.5	0.75	1.0	1.25	1.5	1.75	2.0	2.25	2.5
0.25	Consistency	0.551	76.2	67.9	57.9	46.4	0.0	0.0	0.0	0.0	0.0	0.0	0.0
	SB (high)	0.520	72.4	64.3	54.4	44.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
	SB (low)	0.494	70.2	61.3	51.8	40.7	0.0	0.0	0.0	0.0	0.0	0.0	0.0
0.50	Consistency	0.719	64.1	57.2	50.3	43.2	36.3	29.5	22.9	15.9	0.0	0.0	0.0
	SB (high)	0.671	61.2	54.3	47.0	40.3	33.4	26.9	20.7	14.6	0.0	0.0	0.0
	SB (low)	0.632	58.7	51.9	44.7	38.0	31.3	25.2	19.1	13.5	0.0	0.0	0.0
1.00	Consistency	0.755	46.6	42.5	38.3	34.0	30.1	26.4	22.9	19.6	16.6	13.7	11.2
	SB (high)	0.706	44.1	40.3	35.5	31.4	27.4	24.2	20.9	18.1	15.1	12.5	10.5
	SB (low)	0.673	43.2	38.9	34.6	30.6	26.5	23.1	19.7	16.7	14.0	11.7	9.5

Table 4.5.: **The approximate certified accuracy at various thresholds ϵ for SB on CIFAR-10**. Results are obtained with fixed-sample certification.

σ	Method	Time ↓	TP ↑	Mem ↓	Conv_{Acc} ↓	Acc_{full} ↑	$\text{ACR}_{\text{fixed}}$ ↑	CUI ↑
0.50	Consistency	192.75	664	**493.6**	88	**55.0**	**0.822**	0.260
	SB (low)	**132.33**	**967**	514.9	**79**	51.2	0.733	**0.346**
1.00	Consistency	192.03	667	**493.6**	88	**41.6**	**0.982**	0.249
	SB (low)	**136.66**	**937**	514.9	**81**	37.0	0.881	**0.318**

Table 4.6.: **The key SB metrics on ImageNet**. Direct comparison of the performance of the SB framework with $\mathcal{M} = \mathcal{L}_{\text{Consistency}}$ as a selection metric in contrast with the best performing baselines. An upward pointing arrow ↑ in a column header indicates that higher values are better, and a downward pointing arrow ↓ indicates that lower values are better. Values highlighted in **bold** represent the best value for the metric in that section of the table.

σ	Method	$\text{ACR}_{\text{fixed}} \uparrow$	0.0	0.5	1.0	1.5	2.0	2.5	3.0	3.5
0.50	Consistency	0.822	55.0	50.2	44.0	34.8	0.0	0.0	0.0	0.0
	SB (low)	0.733	51.2	44.0	38.6	29.4	0.0	0.0	0.0	0.0
1.00	Consistency	0.982	41.6	37.4	32.6	28.0	24.2	21.0	17.4	14.2
	SB (low)	0.881	37.0	33.8	29.2	24.0	21.0	18.2	16.4	15.0

Table 4.7.: **The approximate certified accuracy at various thresholds ϵ for SB on ImageNet**. Results are obtained with fixed-sample certification.

3. Summary

Taken together, our findings reveal a clear asymmetry between optimization-centric and data-centric approaches to improving certified training efficiency. The two frameworks explored in this chapter target distinct sources of inefficiency: gradient alignment during optimization and redundancy in the training data.

Gradient-level strategies, such as Loss Projection, show that accuracy-robustness conflicts can be partially mitigated under specific conditions. However, these benefits are limited by substantial computational overhead and diminishing effectiveness as stochasticity increases. The results suggest that simple parameter-space projection is insufficient to consistently overcome the structural conflict between classification and robustness objectives.

In contrast, data-centric strategies, exemplified by Selective Backpropagation, directly address the dominant source of computational cost in certified training: redundant computation caused by repeated noise sampling. By identifying and prioritizing informative training samples, SB achieves substantial efficiency improvements while maintaining competitive accuracy and robustness across both CIFAR-10 and ImageNet.

These findings indicate that scalable certified training requires a shift in focus from solely refining optimization dynamics toward improving computational allocation. While optimization improvements remain valuable, exploiting redundancy in the training process is a more effective direction for achieving practical efficiency gains.

Key Insights

Inefficiency in certified training is primarily driven by redundant computation rather than optimization conflicts.

Chapter 5.

Conclusion and Future Work

The practical adoption of certified deep learning depends on obtaining strong robustness guarantees efficiently. Randomized smoothing is one of the most scalable certification frameworks for modern neural networks, but certified training remains expensive enough to impede deployment. Addressing this challenge requires understanding where this cost originates, not only how robust different training methods are.

This thesis approached certified training from an efficiency-first perspective. We developed a systematic benchmarking framework and used it to investigate two complementary sources of computational overhead: structural optimization conflicts and redundant computation during training. Together, these contributions clarify the efficiency–robustness trade-off in randomized smoothing and identify directions for improving practical scalability.

1. Summary of Contributions

This thesis investigated the computational efficiency of certified training under randomized smoothing from two complementary perspectives: systematic evaluation and hypothesis-driven algorithmic analysis.

1. **Systematic evaluation of certified training efficiency**. We established a unified framework for benchmarking certified training methods and used it to characterize the efficiency–robustness trade-off under a standardized experimental protocol.

   (i) We developed the BenchmarkCtrs framework, which jointly evaluates classification accuracy, certified robustness, GPU time, throughput, memory consumption, and convergence behavior while isolating algorithmic differences from implementation and hardware effects.

(ii) Using this framework, we conducted a five-method empirical comparison covering Gaussian Augmentation, SmoothAdv, Macer, Adre, and Consistency. The comparison revealed repeated noise sampling as the dominant contributor to computational cost and showed that robustness gains do not necessarily justify the added overhead.

2. **Investigation of computational bottlenecks**. Building on the benchmarking results, we investigated two complementary hypotheses regarding the sources of computational inefficiency in certified training.

   (i) *Optimization conflicts.* We evaluated parameter-space gradient projection (LossProj) to determine whether structural conflicts between classification and robustness objectives limit optimization efficiency. Gradient projection improved optimization behavior in specific low-noise settings, but the gains did not generalize across methods or outweigh the added cost, suggesting that optimization conflicts alone are unlikely to be the primary source of inefficiency.

   (ii) *Redundant computation.* We adapted Selective Backpropagation to robustness-aware sample selection to investigate whether certified training performs unnecessary computation. Consistency variance provided an effective measure of sample importance, allowing up to 30% training-time savings on ImageNet with only minor reductions in certified robustness under the evaluated configuration. These findings identify redundant computation as the more promising efficiency target.

Taken together, these contributions establish an efficiency-first perspective on certified training under randomized smoothing. Rather than treating computational overhead as an unavoidable cost of robustness, this thesis shows that it can be evaluated, understood, and, in some cases, substantially reduced through targeted algorithmic design.

2. Reflections on Methodological Challenges

Several methodological observations extend beyond the individual methods evaluated in this thesis.

First, the observed *reproduction gap* across certified training baselines reinforces the need for standardized benchmarking protocols. Differences in implementations, execution environments, and reporting practices make it difficult to distinguish algorithmic improvements from engineering choices. This motivated the unified benchmarking framework and highlights the importance of reproducible evaluation when studying certified training efficiency.

Second, the contrasting outcomes of the two hypotheses suggest that *not all sources of computational overhead contribute equally to the practical efficiency of certified training.* The limited effectiveness of LossProj indicates that mitigating optimization conflicts alone is insufficient to substantially improve the efficiency–robustness trade-off. In contrast, the success of robustness-aware Selective Backpropagation shows that reducing redundant computation offers a more promising avenue for improving training efficiency.

These observations suggest that the dominant computational challenges of certified training arise less from individual robustness objectives than from the shared computational structure of randomized smoothing. Future advances therefore depend on stronger robustness objectives and on training algorithms that use computational resources more effectively.

3. Future Work

The findings of this thesis suggest three directions for future research: validating generality, exploring alternative optimization strategies, and further developing data-centric approaches to efficient certified training.

Benchmark Generalization. The proposed BenchmarkCtrs framework should be extended to a broader range of architectures and datasets. This work focused on the ResNet family and the standard CIFAR-10 and ImageNet benchmarks, while modern certified learning increasingly considers Wide ResNets, Vision Transformers, and datasets with different image resolutions and class distributions. Evaluating certified training under these settings would test whether the efficiency–robustness trends observed here generalize beyond the current experimental configurations.

Optimization-Centric Training. The limited success of optimization-centric interventions leaves alternative optimization strategies open for study. Curriculum schedules that progressively increase smoothing noise or robustness regularization may reduce optimization conflicts by allowing the model to establish stable classification boundaries before emphasizing certified robustness. Adaptive strategies that track the evolving interaction between classification and robustness objectives may be more effective than static formulations.

Data-Centric Training. The results obtained with robustness-aware Selective Backpropagation identify data-centric optimization as a promising direction for improving certified training efficiency. Future work should investigate richer notions of sample importance that combine historical learning dynamics, robustness estimates, and uncertainty measures to distinguish samples that require continued optimization from those whose contribution has saturated. Recent work on "Metadata Archaeology" [37] shows that learning dynamics can identify informative

subsets in conventional supervised learning, suggesting that similar ideas could reduce redundant computation in randomized smoothing.

Efficiency-Aware Objectives. More broadly, this thesis motivates treating computational efficiency as an explicit design objective in certified training. Rather than evaluating efficiency only after training, future methods should jointly optimize robustness and computational cost throughout training. Objectives that explicitly balance these goals would move certified robustness closer to large-scale and continually evolving deep learning systems.

These directions suggest that the next generation of certified training methods will emerge from the integration of principled evaluation, adaptive optimization, and efficient use of computational resources. Pursuing them is essential for translating certified robustness into practical, scalable learning systems.

4. Closing Statement

The future of safety-critical machine learning depends on both the strength of its robustness guarantees and the practicality with which those guarantees can be achieved. Formal certification has limited value if its computational cost prevents deployment, maintenance, or continual adaptation in real-world systems.

This thesis argues that computational efficiency should no longer be regarded as a secondary engineering concern, but as a fundamental objective of certified training. By benchmarking existing methods and investigating complementary sources of overhead, this work shows that certified training efficiency can be studied rigorously rather than accepted as an unavoidable cost of robustness.

Certified robustness has matured from a theoretical possibility into a practical learning paradigm. Its next stage of progress depends on stronger robustness guarantees and on making those guarantees computationally accessible. Only then can certified learning become a practical foundation for the trustworthy and scalable deployment of deep learning systems.

Bibliography

[1] Jason Ansel et al. "PyTorch 2: Faster Machine Learning Through Dynamic Python Bytecode Transformation and Graph Compilation". In: *Proceedings of the 29th ACM International Conference on Architectural Support for Programming Languages and Operating Systems, Volume 2*. Vol. 2. ASPLOS '24. New York, NY, USA: Association for Computing Machinery, Apr. 27, 2024, pp. 929–947. ISBN: 979-8-4007-0385-0. DOI: 10.1145/3620665.3640366. URL: https://dl.acm.org/doi/10.1145/3620665.3640366.

[2] Anish Athalye, Nicholas Carlini, and David Wagner. "Obfuscated Gradients Give a False Sense of Security: Circumventing Defenses to Adversarial Examples". In: *Proceedings of the 35th International Conference on Machine Learning*. International Conference on Machine Learning. PMLR, July 3, 2018, pp. 274–283. URL: https://proceedings.mlr.press/v80/athalye18a.html.

[3] Brian R. Bartoldson, Bhavya Kailkhura, and Davis Blalock. "Compute-Efficient Deep Learning: Algorithmic Trends and Opportunities". In: *Journal of Machine Learning Research* 24.122 (2023), pp. 1–77. ISSN: 1533-7928. URL: http://jmlr.org/papers/v24/22-1208.html.

[4] Yoshua Bengio et al. "Curriculum learning". In: *Proceedings of the 26th Annual International Conference on Machine Learning*. ICML '09. New York, NY, USA: Association for Computing Machinery, June 14, 2009, pp. 41–48. ISBN: 978-1-60558-516-1. DOI: 10.1145/1553374.1553380. URL: https://dl.acm.org/doi/10.1145/1553374.1553380.

[5] Ruoxin Chen et al. "Input-Specific Robustness Certification for Randomized Smoothing". In: *Proceedings of the AAAI Conference on Artificial Intelligence* 36.6 (June 28, 2022). Number: 6, pp. 6295–6303. ISSN: 2374-3468. DOI: 10.1609/aaai.v36i6.20579. URL: https://ojs.aaai.org/index.php/AAAI/article/view/20579.

[6] Jeremy Cohen, Elan Rosenfeld, and Zico Kolter. "Certified Adversarial Robustness via Randomized Smoothing". In: *Proceedings of the 36th International Conference on Machine Learning*. International Conference on Machine Learn-

ing. PMLR, May 24, 2019, pp. 1310–1320. URL: https://proceedings.mlr.press/v97/cohen19c.html.

[7] George E. Dahl et al. *Benchmarking Neural Network Training Algorithms*. June 18, 2025. DOI: 10.48550/arXiv.2306.07179. arXiv: 2306.07179[cs]. URL: http://arxiv.org/abs/2306.07179.

[8] William Falcon and The PyTorch Lightning team. *PyTorch Lightning*. Jan. 30, 2026. DOI: 10.5281/zenodo.18432694. URL: https://zenodo.org/records/18432694.

[9] Huijie Feng et al. "Regularized Training and Tight Certification for Randomized Smoothed Classifier with Provable Robustness". In: *Proceedings of the AAAI Conference on Artificial Intelligence* 34.4 (Apr. 3, 2020), pp. 3858–3865. ISSN: 2374-3468, 2159-5399. DOI: 10.1609/aaai.v34i04.5798. URL: https://ojs.aaai.org/index.php/AAAI/article/view/5798.

[10] Timon Gehr et al. "AI2: Safety and Robustness Certification of Neural Networks with Abstract Interpretation". In: *2018 IEEE Symposium on Security and Privacy (SP)*. 2018 IEEE Symposium on Security and Privacy (SP). May 2018, pp. 3–18. DOI: 10.1109/SP.2018.00058. URL: https://ieeexplore.ieee.org/document/8418593.

[11] Sven Gowal et al. "Scalable Verified Training for Provably Robust Image Classification". In: *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*. 2019 IEEE/CVF International Conference on Computer Vision (ICCV). Oct. 2019, pp. 4841–4850. DOI: 10.1109/ICCV.2019.00494. URL: https://ieeexplore.ieee.org/document/9010971.

[12] Kaiming He et al. "Deep Residual Learning for Image Recognition". In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2016, pp. 770–778. URL: https://openaccess.thecvf.com/content_cvpr_2016/html/He_Deep_Residual_Learning_CVPR_2016_paper.html.

[13] Xiaowei Huang et al. "A survey of safety and trustworthiness of deep neural networks: Verification, testing, adversarial attack and defence, and interpretability". In: *Computer Science Review* 37 (Aug. 1, 2020), p. 100270. ISSN: 1574-0137. DOI: 10.1016/j.cosrev.2020.100270. URL: https://www.sciencedirect.com/science/article/pii/S1574013719302527.

[14] Jongheon Jeong, Seojin Kim, and Jinwoo Shin. "Confidence-Aware Training of Smoothed Classifiers for Certified Robustness". In: *Proceedings of the AAAI Conference on Artificial Intelligence* 37.7 (June 26, 2023). Number: 7, pp. 8005–8013. ISSN: 2374-3468. DOI: 10.1609/aaai.v37i7.25968. URL: https://ojs.aaai.org/index.php/AAAI/article/view/25968.

[15] Jongheon Jeong and Jinwoo Shin. "Consistency Regularization for Certified Robustness of Smoothed Classifiers". In: *Advances in Neural Information Processing Systems*. Vol. 33. Curran Associates, Inc., 2020, pp. 10558–10570. URL: https://proceedings.neurips.cc/paper/2020/hash/77330e1330ae2b086e5bfcae50d9ffae-Abstract.html.

[16] Jongheon Jeong et al. "SmoothMix: Training Confidence-calibrated Smoothed Classifiers for Certified Robustness". In: Advances in Neural Information Processing Systems. Nov. 9, 2021. URL: https://openreview.net/forum?id=nlEQMVBD359.

[17] Angela H. Jiang et al. *Accelerating Deep Learning by Focusing on the Biggest Losers*. Oct. 2, 2019. DOI: 10.48550/arXiv.1910.00762. arXiv: 1910.00762[cs]. URL: http://arxiv.org/abs/1910.00762.

[18] Priya Kasimbeg et al. "Accelerating neural network training: An analysis of the AlgoPerf competition". In: The Thirteenth International Conference on Learning Representations. Oct. 4, 2024. URL: https://openreview.net/forum?id=CtM5xjRSfm.

[19] Guy Katz et al. "Reluplex: An Efficient SMT Solver for Verifying Deep Neural Networks". In: *Computer Aided Verification*. Ed. by Rupak Majumdar and Viktor Kunčak. Cham: Springer International Publishing, 2017, pp. 97–117. ISBN: 978-3-319-63387-9. DOI: 10.1007/978-3-319-63387-9_5.

[20] Guy Katz et al. "The Marabou Framework for Verification and Analysis of Deep Neural Networks". In: *Computer Aided Verification - 31st International Conference, CAV 2019, New York City, NY, USA, July 15-18, 2019, Proceedings, Part I*. Ed. by Isil Dillig and Serdar Tasiran. Vol. 11561. Lecture Notes in Computer Science. Springer, 2019, pp. 443–452. DOI: 10.1007/978-3-030-25540-4_26.

[21] A. Krizhevsky. "Learning Multiple Layers of Features from Tiny Images". In: 2009. URL: https://www.semanticscholar.org/paper/Learning-Multiple-Layers-of-Features-from-Tiny-Krizhevsky/5d90f06bb70a0a3dced62413346235c

[22] Jiawen Li, Kun Fang, and Jie Yang. "Adaptive distillation on hard samples benefits consistency for certified robustness". In: *International Journal of Machine Learning and Cybernetics* (Mar. 6, 2025). ISSN: 1868-808X. DOI: 10.1007/s13042-025-02568-2. URL: https://doi.org/10.1007/s13042-025-02568-2.

[23] Linyi Li, Tao Xie, and Bo Li. "SoK: Certified Robustness for Deep Neural Networks". In: *2023 IEEE Symposium on Security and Privacy (SP)*. 2023 IEEE Symposium on Security and Privacy (SP). May 2023, pp. 1289–1310. DOI: 10.1109/SP46215.2023.10179303. URL: https://ieeexplore.ieee.

org/abstract/document/10179303?casa_token=qJMdTno-uK4AAAAA:0ImSeSG7A_ossHPw2tIsjfvKZIRm24X8uj_R3NIEoIoV8SxaGxMrI5OA9myaI0oIFeW2p2L8zxu

[24] Aleksander Madry et al. "Towards Deep Learning Models Resistant to Adversarial Attacks". In: International Conference on Learning Representations. Feb. 15, 2018. URL: https://openreview.net/forum?id=rJzIBfZAb.

[25] Gaurav Menghani. "Efficient Deep Learning: A Survey on Making Deep Learning Models Smaller, Faster, and Better". In: *ACM Comput. Surv.* 55.12 (2023), 259:1–259:37. ISSN: 0360-0300. DOI: 10.1145/3578938. URL: https://doi.org/10.1145/3578938.

[26] Paulius Micikevicius et al. "Mixed Precision Training". In: International Conference on Learning Representations. Feb. 15, 2018. URL: https://openreview.net/forum?id=r1gs9JgRZ.

[27] Sören Mindermann et al. "Prioritized Training on Points that are Learnable, Worth Learning, and not yet Learnt". In: *Proceedings of the 39th International Conference on Machine Learning*. International Conference on Machine Learning. PMLR, June 28, 2022, pp. 15630–15649. URL: https://proceedings.mlr.press/v162/mindermann22a.html.

[28] Nicolas Papernot et al. "Distillation as a Defense to Adversarial Perturbations Against Deep Neural Networks". In: *2016 IEEE Symposium on Security and Privacy (SP)*. 2016 IEEE Symposium on Security and Privacy (SP). ISSN: 2375-1207. May 2016, pp. 582–597. DOI: 10.1109/SP.2016.41. URL: https://ieeexplore.ieee.org/abstract/document/7546524.

[29] Mansheej Paul, Surya Ganguli, and Gintare Karolina Dziugaite. "Deep Learning on a Data Diet: Finding Important Examples Early in Training". In: *Advances in Neural Information Processing Systems*. Vol. 34. Curran Associates, Inc., 2021, pp. 20596–20607. URL: https://proceedings.neurips.cc/paper/2021/hash/ac56f8fe9eea3e4a365f29f0f1957c55-Abstract.html.

[30] Olga Russakovsky et al. "ImageNet Large Scale Visual Recognition Challenge". In: *International Journal of Computer Vision* 115.3 (Dec. 2015), pp. 211–252. ISSN: 0920-5691, 1573-1405. DOI: 10.1007/s11263-015-0816-y. URL: http://link.springer.com/10.1007/s11263-015-0816-y.

[31] Hadi Salman et al. "Provably Robust Deep Learning via Adversarially Trained Smoothed Classifiers". In: *Advances in Neural Information Processing Systems*. Vol. 32. Curran Associates, Inc., 2019. URL: https://proceedings.neurips.cc/paper/2019/hash/3a24b25a7b092a252166a1641ae953e7-Abstract.html.

[32] Chandramouli S. Sastry, Sri H. Dumpala, and Sageev Oore. "DiffAug: A Diffuse-and-Denoise Augmentation for Training Robust Classifiers". In: *Advances in Neural Information Processing Systems* 37 (Dec. 16, 2024), pp. 20745–20785. URL: https://proceedings.neurips.cc/paper_files/paper/2024/hash/24e8b46430df965674221665816a4964-Abstract-Conference.html.

[33] Jan Schuchardt and Stephan Günnemann. "Invariance-Aware Randomized Smoothing Certificates". In: *Advances in Neural Information Processing Systems* 35 (Dec. 6, 2022), pp. 34302–34320. URL: https://papers.nips.cc/paper_files/paper/2022/hash/ddd45979547a35db2471e69cbf3bca54-Abstract-Conference.html.

[34] Roy Schwartz et al. "Green AI". In: *Commun. ACM* 63.12 (Nov. 17, 2020), pp. 54–63. ISSN: 0001-0782. DOI: 10.1145/3381831. URL: https://dl.acm.org/doi/10.1145/3381831.

[35] Emmanouil Seferis, Simon Burton, and Stefanos Kollias. "Randomized Smoothing (almost) in Real Time?" In: ICML 2023 Workshop The Many Facets of Preference-Based Learning. Oct. 4, 2023. URL: https://openreview.net/forum?id=hkhfOSkMKQ#all.

[36] Jaime Sevilla et al. "Compute Trends Across Three Eras of Machine Learning". In: *2022 International Joint Conference on Neural Networks (IJCNN)*. 2022 International Joint Conference on Neural Networks (IJCNN). ISSN: 2161-4407. July 2022, pp. 1–8. DOI: 10.1109/IJCNN55064.2022.9891914. URL: https://ieeexplore.ieee.org/abstract/document/9891914.

[37] Shoaib Ahmed Siddiqui et al. "Metadata Archaeology: Unearthing Data Subsets by Leveraging Training Dynamics". In: The Eleventh International Conference on Learning Representations. Sept. 29, 2022. URL: https://openreview.net/forum?id=PvLnIaJbt9.

[38] Ben Sorscher et al. "Beyond neural scaling laws: beating power law scaling via data pruning". In: *Advances in Neural Information Processing Systems* 35 (Dec. 6, 2022), pp. 19523–19536. URL: https://proceedings.neurips.cc/paper_files/paper/2022/hash/7b75da9b61eda40fa35453ee5d077df6-Abstract-Conference.html.

[39] Chenhao Sun et al. *Average Certified Radius is a Poor Metric for Randomized Smoothing*. Jan. 31, 2025. DOI: 10.48550/arXiv.2410.06895. arXiv: 2410.06895[cs]. URL: http://arxiv.org/abs/2410.06895.

[40] Christian Szegedy et al. “Rethinking the Inception Architecture for Computer Vision”. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2016, pp. 2818–2826. URL: https://www.cv-foundation.org/openaccess/content_cvpr_2016/html/Szegedy_Rethinking_the_Inception_CVPR_2016_paper.html.

[41] Vincent Tjeng, Kai Y. Xiao, and Russ Tedrake. “Evaluating Robustness of Neural Networks with Mixed Integer Programming”. In: International Conference on Learning Representations. Sept. 27, 2018. URL: https://openreview.net/forum?id=HyGIdiRqtm.

[42] Dimitris Tsipras et al. “Robustness May Be at Odds with Accuracy”. In: International Conference on Learning Representations. Sept. 27, 2018. URL: https://openreview.net/forum?id=SyxAb30cY7.

[43] Shubham Ugare et al. “Incremental Randomized Smoothing Certification”. In: The Twelfth International Conference on Learning Representations. Oct. 13, 2023. URL: https://openreview.net/forum?id=SdeAPV1irk.

[44] Pratik Vaishnavi, Kevin Eykholt, and Amir Rahmati. “Accelerating Certified Robustness Training via Knowledge Transfer”. In: *Advances in Neural Information Processing Systems* 35 (Dec. 6, 2022), pp. 5269–5281. URL: https://proceedings.neurips.cc/paper_files/paper/2022/hash/22bf0634985f4e6dbb1fb40e247d1478-Abstract-Conference.html.

[45] Václav Voráček. “Treatment of Statistical Estimation Problems in Randomized Smoothing for Adversarial Robustness”. In: *Advances in Neural Information Processing Systems* 37 (Dec. 16, 2024), pp. 133464–133486. DOI: 10.5555/3737916.3742158. URL: https://papers.nips.cc/paper/2024/hash/f11394cdd377aab9ff5e2a4e9f27367f-Abstract-Conference.html.

[46] Eric Wong and Zico Kolter. “Provable Defenses against Adversarial Examples via the Convex Outer Adversarial Polytope”. In: *Proceedings of the 35th International Conference on Machine Learning*. International Conference on Machine Learning. PMLR, July 3, 2018, pp. 5286–5295. URL: https://proceedings.mlr.press/v80/wong18a.html.

[47] Xiaoxia Wu, Ethan Dyer, and Behnam Neyshabur. “When Do Curricula Work?” In: International Conference on Learning Representations. Oct. 2, 2020. URL: https://openreview.net/forum?id=tW4QEInpni.

[48] Runtian Zhai et al. “MACER: Attack-free and Scalable Robust Training via Maximizing Certified Radius”. In: International Conference on Learning Representations. Sept. 25, 2019. URL: https://openreview.net/forum?id=rJx1Na4Fwr.

[49] Hanwei Zhang et al. “Walking on the Edge: Fast, Low-Distortion Adversarial Examples”. In: *IEEE Transactions on Information Forensics and Security* 16 (2021), pp. 701–713. ISSN: 1556-6021. DOI: 10.1109/TIFS.2020.3021899. URL: https://ieeexplore.ieee.org/abstract/document/9186644.

[50] Huan Zhang et al. “Efficient Neural Network Robustness Certification with General Activation Functions”. In: *Advances in Neural Information Processing Systems*. Vol. 31. Curran Associates, Inc., 2018. URL: https://proceedings.neurips.cc/paper/2018/hash/d04863f100d59b3eb688a11f95b0ae60-Abstract.html.

[51] Huan Zhang et al. “Towards Stable and Efficient Training of Verifiably Robust Neural Networks”. In: International Conference on Learning Representations. Sept. 25, 2019. URL: https://openreview.net/forum?id=Skxuk1rFwB.

Appendix A.

Algorithmic Details for Training and Certification Methods

This appendix collects the algorithmic forms of established randomized smoothing methods discussed in Chapter 2. These procedures are not introduced as contributions of this thesis; they define the certification and training baselines against which the later efficiency analysis is conducted. The main text discusses their conceptual role, while this appendix isolates the operational steps that determine their sampling structure, optimization cost, and evaluation behavior.

1. Certification Algorithms

Certification under randomized smoothing is governed by a common statistical task: estimate the probability assigned to the predicted class under Gaussian perturbations, then convert a valid lower bound on that probability into a certified radius. The two procedures below differ in how they allocate the Monte Carlo budget. Fixed-sample certification uses a predetermined number of samples and therefore serves as the standard protocol for final evaluation. Adaptive certification monitors the estimate during sampling and stops early once the bound has stabilized, making it more suitable for repeated training-time measurements.

1.1. Fixed-Sample Certification (CERTIFY)

Algorithm 5 makes the two-stage structure of the standard protocol explicit. The first stage selects the candidate class using a comparatively small sample budget n_0, while the second stage estimates a confidence bound for that fixed class using n additional samples. This design gives a clean final guarantee, but its cost is fixed per certified input regardless of whether the sample is easy, difficult, or ultimately abstained. For this reason, fixed-sample certification is appropriate

Algorithm 5 Fixed-Sample Certification (CERTIFY Algorithm [6])

function CERTIFY($f, \mathbf{x}, \sigma, n_0, n, \alpha$)
  Draw noise samples $\boldsymbol{\rho}_i \sim \mathcal{N}(\mathbf{0}, \sigma^2 \boldsymbol{I})$ for $1 \leq i \leq n_0$ ▷ Selection Phase
  $\hat{c}_A \leftarrow \arg\max_c \sum_{i=1}^{n_0} \mathbb{1}_{f(\mathbf{x}+\boldsymbol{\rho}_i)=c}$
  Draw noise samples $\boldsymbol{\delta}_j \sim \mathcal{N}(\mathbf{0}, \sigma^2 \boldsymbol{I})$ for $1 \leq j \leq n$ ▷ Certification Phase
  $k \leftarrow \sum_{j=1}^{n} \mathbb{1}_{f(\mathbf{x}+\boldsymbol{\delta}_j)=\hat{c}_A}$
  $\underline{p_A} \leftarrow \text{LowerConfBound}(k, n, \alpha)$
  if $\underline{p_A} > 0.5$ **then**
    $\hat{CR} \leftarrow \sigma \Phi^{-1}(\underline{p_A})$
    return prediction $\hat{c}_A$ and radius $\hat{CR}$
  else
    return ABSTAIN
  end if
end function

for final reported metrics, but inefficient as a continuous monitoring tool during training.

1.2. Adaptive Certification (CertifySeq)

Algorithm 6 Adaptive Certification

function CertifySeq($f, \mathbf{x}, \sigma, n, \texttt{patience}, \delta_p, \alpha$)
  $\hat{c}_A \leftarrow$ Selection Phase as in Algorithm 5
  Initialize betting capital $K_0 = 1$, $\underline{p_A} = 0$
  for $1 \leq t \leq n$ **do**
    Sample $\boldsymbol{\delta}_t \sim \mathcal{N}(\mathbf{0}, \sigma^2 \boldsymbol{I})$
    Update K_t based on $\mathbb{1}_{f(\mathbf{x}+\boldsymbol{\delta}_t)=\hat{c}_A}$
    $\underline{p_A}^{(t)} \leftarrow$ UpdatedLowerBound(K_t, t, α)
    if $\underline{p_A}^{(t)}$ has stabilized for `patience` steps within δ_p **then**
      $\underline{p_A}^{(T)} \leftarrow \underline{p_A}^{(t)}$
      break
    end if
  end for
  if $\underline{p_A}^{(T)} > 0.5$ **then**
    $\hat{CR} \leftarrow \sigma \Phi^{-1}(\underline{p_A}^{(T)})$
    return $\hat{c}_A$ and $\hat{CR}$
  else
    return ABSTAIN and $\hat{CR} = 0$
  end if
end function

Algorithm 6 preserves the same certification target but changes the sampling policy. Instead of consuming the full budget unconditionally, it updates a valid lower confidence bound after each noisy evaluation and stops once the estimate remains stable for the specified patience window. The resulting radius estimates are more conservative than the fixed-sample values used for final comparison, yet they reduce the cost of repeated evaluation enough to support convergence analysis in Chapter 3 and Chapter 4.

2. Certified Training Algorithms

The certified training methods considered in this thesis differ in objective design, but they share the same basic computational pattern: each update evaluates the model on Gaussian perturbations of the training data. The algorithms below therefore emphasize where the additional cost enters the optimization step. Gaussian augmentation introduces the minimal randomized smoothing training loop,

SmoothAdv adds an adversarial inner loop, and regularized training represents the shared template used by MACER, ADRE, and Consistency.

2.1. Gaussian Augmentation Training (Gaussian)

Algorithm 7 A Gaussian Augmentation Training Step

procedure TRAINGAUSSIAN($(\mathbf{x}_i, \mathbf{y}_i)_{i=1}^{B}, \sigma$)
  Draw noise samples $\boldsymbol{\delta}_i \sim \mathcal{N}(\mathbf{0}, \sigma^2 \boldsymbol{I})$ for $1 \le i \le B$
  $\tilde{\mathcal{L}} \leftarrow \dfrac{1}{B} \displaystyle\sum_{1 \le i \le B} \mathcal{L}_{\mathrm{CE}}\left(f(\mathbf{x}_i + \boldsymbol{\delta}_i), \mathbf{y}_i\right)$
  Run backpropagation on $\tilde{\mathcal{L}}$ and update $\boldsymbol{\theta}$
end procedure

Algorithm 7 is the most lightweight certified training baseline. Each input contributes one noisy realization to the supervised loss, so the method aligns the training distribution with the smoothing distribution without adding an adversarial search or robustness regularization. Its simplicity makes it useful as an efficiency reference point: any stronger certified training method needs to justify its additional sampling or optimization cost through improved robustness.

2.2. Certified Adversarial Training (SmoothAdv)

Algorithm 8 shows why SmoothAdv is computationally demanding. The method estimates the smoothed prediction inside each PGD step, so the number of forward and backward passes scales with the batch size, the number of noise samples m, and the attack length T. This inner maximization directly targets difficult neighborhoods around each training point, but it also moves the method into a substantially higher-cost regime than Gaussian augmentation.

Algorithm 8 A SmoothAdv Training Step with the PGD Attack

procedure TrainSmoothAdv($(\mathbf{x}_i, y_i)_{i=1}^B, \sigma, m, T, \epsilon$)
  Draw noise samples $\boldsymbol{\delta}_i^{(j)} \sim \mathcal{N}(\mathbf{0}, \sigma^2 \boldsymbol{I})$ for $1 \le i \le B, 1 \le j \le m$
  for $1 \le i \le B$ **do**
    $\hat{\mathbf{x}}_i \leftarrow \mathbf{x}_i$
    for $1 \le t \le T$ **do**
      $\tilde{\mathcal{L}}_{adv} \leftarrow \frac{1}{m} \sum_{1 \le j \le m} [\mathcal{L}_{\mathrm{CE}}(f(\hat{\mathbf{x}}_i + \boldsymbol{\delta}_i^{(j)}), y_i)]$
      $\hat{\mathbf{x}}_i \leftarrow \mathrm{proj}_{\mathcal{B}(\mathbf{x}_i, \epsilon)} \left(\hat{\mathbf{x}}_i + \eta \nabla_{\mathbf{x}_i} \tilde{\mathcal{L}}_{adv} \right)$
    end for
  end for
  $\tilde{\mathcal{L}} \leftarrow \frac{1}{mB} \sum_{\substack{1 \le i \le B \\ 1 \le j \le m}} [\mathcal{L}_{\mathrm{CE}}(f(\hat{\mathbf{x}}_i + \boldsymbol{\delta}_i^{(j)}), y_i)]$
  Run backpropagation on $\tilde{\mathcal{L}}$ and update $\boldsymbol{\theta}$
end procedure

2.3. Regularized Certified Training (Macer, Adre, Consistency)

Algorithm 9 A Regularized Training Step

procedure TrainRegularized($(\mathbf{x}_i, y_i)_{i=1}^B, \sigma, m, \lambda$)
  Draw noise samples $\boldsymbol{\delta}_i^{(j)} \sim \mathcal{N}(\mathbf{0}, \sigma^2 \boldsymbol{I})$ for $1 \le i \le B, 1 \le j \le m$
  $\tilde{\mathcal{L}_{\mathrm{CE}}} \leftarrow \frac{1}{mB} \sum_{\substack{1 \le i \le B \\ 1 \le j \le m}} \mathcal{L}_{\mathrm{CE}} \left(f(\mathbf{x}_i + \boldsymbol{\delta}_i^{(j)}), y_i \right)$
  $\tilde{\mathcal{L}_{\mathrm{R}}} \leftarrow \frac{\lambda}{B} \sum_{1 \le i \le B} \mathcal{L}_{\mathrm{R}} \left((f_{\boldsymbol{\theta}}(\mathbf{x}_i + \boldsymbol{\delta}_i^{(j)}))_{j=1}^m, y_i \right)$
  Run backpropagation on $\tilde{\mathcal{L}} = \tilde{\mathcal{L}_{\mathrm{CE}}} + \tilde{\mathcal{L}_{\mathrm{R}}}$ and update $\boldsymbol{\theta}$
end procedure

Algorithm 9 abstracts the common structure of robustness-regularized training. The classification term preserves standard supervised learning under Gaussian perturbations, while $\mathcal{L}_{\mathrm{R}}$ denotes a method-specific robustness penalty computed from the same noisy predictions. Macer, Adre, and Consistency instantiate this term differently, but all three depend on repeated noise sampling to estimate smoothed statistics. This shared structure is central to the efficiency analysis in the main text: improvements can target the objective, the data used in each update, or the sampling process itself, but the computational bottleneck originates from the same repeated perturbation evaluations.

Collectively, these algorithms define the operational baseline for the thesis. The certification procedures determine how robustness is measured, while the training procedures determine where the cost of certified robustness enters optimization. This distinction is important for interpreting the benchmark results: fixed-sample certification provides the reference quality metrics, adaptive certification enables repeated monitoring, and the training algorithms expose the sampling-heavy structure that motivates the efficiency interventions studied later.

Appendix B.

Fixed-Sample and Adaptive Certification Comparison

Certification	Method	ACR	0.0	0.25	0.5	0.75	1.0	1.25	1.5	1.75	2.0	2.25	2.5
Fixed-Sample	Standard	0.103	8.3	7.6	6.8	5.9	4.9	3.9	3.0	2.2	1.4	0.8	0.4
	Gaussian	0.511	47.1	40.9	33.8	27.7	22.1	17.2	13.3	9.7	6.6	4.3	2.7
	Consistency	0.756	46.3	42.2	38.1	34.3	30.0	26.3	22.9	19.7	16.6	13.8	11.3
	SmoothAdv	0.790	43.7	40.3	36.9	33.8	30.5	27.0	24.0	21.4	18.4	15.9	13.4
Adaptive	Standard	0.102	8.3	7.6	6.8	5.9	4.9	3.9	3.0	2.2	1.3	0.7	0.3
	Gaussian	0.429	46.0	39.4	32.1	26.0	20.6	15.3	12.7	7.7	3.9	2.5	1.0
	Consistency	0.627	43.4	40.5	33.3	31.9	28.9	22.2	20.6	18.2	14.0	11.8	10.0
	SmoothAdv	0.664	41.0	38.0	34.8	31.4	28.1	24.8	21.6	18.8	16.0	13.0	10.4

Table B.1.: Reports the approximate certified accuracy of selected methods at various values of the threshold ϵ and the ACR of the baseline methods on *CIFAR-10* using both fixed-sample and adaptive certification at $\sigma = 1.0$.

This appendix compares the two certification procedures used throughout the empirical chapters. Fixed-sample certification with CERTIFY provides the reference quality metrics for final evaluation, while adaptive certification with CERTIFYSEQ provides the cheaper monitoring signal used during training. Both procedures use the same certification parameters $n = 100{,}000$, $n_0 = 100$, and $\alpha = 0.001$, so the comparison isolates the effect of the certification protocol rather than a change in evaluation budget or confidence level.

Table B.1 reports the complete certified accuracy values $\text{Acc}(\epsilon)$ and the average certified radius (ACR) for the baseline methods on the *CIFAR-10* test set. The

first half uses fixed-sample certification (Algorithm 5); the second uses adaptive certification (Algorithm 6).

The comparison shows why the two quantities are used for different purposes in the thesis. Adaptive certification broadly preserves the qualitative ordering of the stronger baseline methods, but it yields more conservative robustness estimates across all robustness thresholds. This in turn is strongly reflected in the ACR which is disproportionately affected by the decrease in extreme values.

This comparison clarifies why adaptive certification is suitable for monitoring but not for replacing fixed-sample certification in final quality reports. For Gaussian augmentation, adaptive certification reduces $\mathrm{Acc}(\epsilon)$ by only 1.1 to 1.7 percentage points for $\epsilon \in [0.0, 0.75]$, corresponding to relative changes between 2.3% and 6.1%. In this range, the adaptive estimates preserve the practical interpretation of the fixed-sample results. At larger thresholds, $\epsilon \in [2.0, 2.5]$, similarly sized absolute reductions correspond to much larger relative changes, between 60.3% and 40.1%, because certified accuracy is already close to zero. These relative values help illustrate why the ACR gap can appear larger than the threshold-wise accuracy differences suggest: ACR averages certified radii, so changes near the tail of the radius distribution still affect the mean even when they involve only a small fraction of test samples. The high relative changes at large ϵ therefore indicate that adaptive certification mainly compresses the upper tail of the certified radius distribution, rather than uniformly degrading robustness across all thresholds.

This justifies our use of adaptive certification as a training-time signal: it preserves the main robustness trends at substantially lower cost, while fixed-sample certification remains necessary for final claims about certified accuracy and ACR. For this reason, $\mathrm{ACR}_{\mathrm{seq}}$ and $\mathrm{Acc}_{\mathrm{seq}}$ should be interpreted as training-time indicators rather than direct substitutes for fixed-sample certification. They are useful for comparing convergence behavior and relative robustness trends during training, where repeated fixed-sample certification would be prohibitively expensive. Final quality claims, however, remain tied to fixed-sample certification.

Appendix C.

MACER Reproduction Comparison

σ	Method	$\text{ACR}_{\text{fixed}} \uparrow$	0.0	0.25	0.5	0.75	1.0	1.25	1.5	1.75	2.0	2.25	2.5
0.25	Original	0.531	79.5	69.0	55.8	40.6	0.0	0.0	0.0	0.0	0.0	0.0	0.0
	Reproduced	0.451	67.9	58.7	47.2	34.4	0.0	0.0	0.0	0.0	0.0	0.0	0.0
0.50	Original	0.691	64.2	57.5	49.9	42.3	34.8	27.6	20.2	12.6	0.0	0.0	0.0
	Reproduced	0.394	40.9	35.4	29.2	23.9	18.8	14.1	10.1	6.0	0.0	0.0	0.0
1.00	Original	0.744	41.4	38.5	35.2	32.3	29.3	26.4	23.4	20.2	17.4	14.5	12.1
	Reproduced	0.324	23.0	20.5	17.9	15.4	13.2	11.1	9.1	7.4	6.1	4.8	3.8

Table C.1.: Comparison between the originally reported MACER results and our reproduced MACER results on CIFAR-10.

This appendix documents the discrepancy between the MACER results reported by Zhai et al. [48] and the results obtained in our BENCHMARKCTRS reproduction. The comparison is separated from the main benchmark discussion because it concerns reproducibility rather than the certification-protocol comparison in Appendix B. Throughout the thesis, MACER denotes the originally reported values, while MACER† denotes our reproduced values.

Table C.1 shows that the reproduced MACER models obtain substantially lower certified robustness than the originally reported results, while following the public implementation and reported hyperparameters described in Section 3.4. The gap is present at $\sigma = 0.25$, $\sigma = 0.5$ and $\sigma = 1.0$, where both ACR and threshold-wise certified accuracy decrease sharply. This discrepancy is indicative of the broader reproducibility difficulties in certified training, where implementation details, training stability, and environmental factors can materially affect final robustness.

We therefore use the dual-reporting strategy defined in Chapter 3. Namely, when comparing established baseline methods in Chapter 3, we include the originally reported MACER values to preserve comparability with the literature. When evaluating the methods introduced in Chapter 4, we compare against our reproduced MACER results, because those experiments must be interpreted relative to baselines trained in the same experimental environment.