IMPERIAL

IMPERIAL COLLEGE LONDON

DEPARTMENT OF COMPUTING

Improved Ensemble-Based Certified Defenses Against Data Poisoning

Author:
Siu Pei Ooi

Supervisor:
Dr. Matthew Wicker

Second Marker:
Dr. Calvin Tsay

Abstract

The growing success and widespread adoption of machine learning has been accelerated by harnessing data at unprecedented scales. When datasets grow to internet-scale, checking them rigorously becomes impossible, leaving practitioners with the critical task of ensuring their models remain stable even in the face of adversarial perturbations to the training data. Certified defenses against adversarial data poisoning offer provable guarantees by bounding the power of an attacker who can modify a limited number of training examples. The current state-of-the-art bounds rely on *aggregation methods*, where ensembles are trained on disjoint data shards, ensuring that a single corrupted point affects only one model. However, existing approaches typically provide per-sample certificates, which often overestimate the adversary's impact, leading to loose guarantees for attack goals such as targeted and untargeted poisoning. While per-sample certificates are valuable, many adversarial goals require poisoning more than a single test data point. To address this, we propose a general framework for computing *provably optimal* robustness certificates in the *multi-sample setting*. Our method uses a Mixed-Integer Linear Programming (MILP) formulation that flexibly models complex aggregation strategies, such as finite aggregation and run-off elections, and also allows for the integration of intrinsically robust techniques like bound propagation. We evaluate our method on standard benchmarks including MNIST, CIFAR-10, and GTSRB, showing it consistently outperforms state-of-the-art bounds in all tested settings. In particular, it certifies the robustness of up to 14% more test samples than per-sample DPA and 9% more than naive multi-sample certification. Finally, we explore incorporating model-level intrinsic robustness and propose a novel hybrid ensemble strategy, highlighting promising directions for future research.

Acknowledgements

I am deeply grateful to my supervisor, Dr. Matthew Wicker, for his invaluable guidance and support throughout this project. His mentorship has made this project not only a fruitful one, but also a profound learning experience that I will carry forward in my future endeavours.

I would like to sincerely thank Philip Sosnin, for his generous assistance and insightful contributions, which greatly helped with my technical understanding throughout this project.

To my family, thank you for being there for me unconditionally. I am endlessly grateful to Mina, for the mental and academic support for the past 4 years. Finally, a special thank you to my lovely friends, whose constant presence and encouragement made this university journey not just possible, but meaningful.

Contents

Chapter 1

Introduction

As machine learning, particularly deep learning models, becomes increasingly prevalent in critical domains like autonomous driving [1] and large language models, concerns about their potential for harm have grown. These models can produce dangerous outputs, especially under adversarial influence [2]. One major threat is data poisoning, where adversaries manipulate training data to compromise a model's integrity [3]. Since modern models often rely on large, uncurated public datasets, they are vulnerable to such attacks. Empirical research indicates that poisoning as little as 5% of the training data can cause a classifier to perform almost randomly [4], whereas for large-scale foundation models, altering only 0.01% of the data can be enough to induce harmful targeted behaviors [5]. Poisoning can degrade a model's overall accuracy (untargeted attacks) or force incorrect predictions for specific inputs (targeted attacks) [3]. Data poisoning attacks have been studied for over two decades [6], and while many defenses exist [2, 7, 8], heuristic methods lack formal guarantees. In safety-critical fields like medicine and autonomous driving, certifiable robustness, which is a formal assurance that model predictions remain stable despite worst-case data manipulation, is crucial.

A range of methods have been proposed to certify robustness against data poisoning, including approaches based on randomised smoothing [9], bound propagation [10], learning-theoretic approaches [11], and techniques based on differential privacy [12, 13, 14]. Among these, aggregation-based defenses, such as Deep Partition Aggregation (DPA) [15] and Finite Aggregation (FA) [16], offer state-of-the-art guarantees against general data poisoning. These approaches are built on a simple yet powerful idea: the training dataset is split into M disjoint shards, with a separate model trained on each shard. At inference time, predictions are combined via majority vote. Because each training point influences only one model, the effect of any single corrupted point is confined to a single vote. This enables provable robustness guarantees against targeted and untargeted poisoning attacks [15]. For example, if the predicted class receives at least $n + 1$ more votes than any other, the ensemble's decision is provably robust to up to $\lfloor n/2 \rfloor$ arbitrary manipulations to training data.

Despite their strong performance, current aggregation-based certification methods

face a key drawback: they compute robustness guarantees on a per-sample basis, which implicitly assumes that the adversary can selectively choose which models to manipulate *after* observing *each* prediction. This assumption restricts per-sample analysis to a narrow class of poisoning adversarial goals and leads to overly conservative bounds, particularly when certifying robustness over batches of predictions. Batch certification is common both for evaluating model generalisation and for assessing the effectiveness of targeted and untargeted poisoning attacks. Initial work by Chen et al. [17] computed multi-sample certification via a Binary Integer Programming (BIP) formulation and demonstrated that, for simple aggregation methods, relaxing the per-sample assumption yields substantially tighter bounds. However, their framework is limited to the most basic aggregation mechanism and does not incorporate many recent advancements. Moreover, the BIP formulation cannot accommodate intrinsic robustness guarantees of individual ensemble members. This motivates the need for a new, more flexible formulation that can support broader certification methods in aggregation settings.

Contributions

In this thesis, we introduce a general framework for computing optimally tight multi-sample certificates of poisoning robustness for a broad class of aggregation-based defenses. We first define a Mixed-Integer Linear Programming (MILP) formulation that models how an adversary could optimally poison training data to affect a *batch* of predictions. Importantly, our formulation accommodates complex aggregation schemes, such as those with overlapping training datasets (such as FA [16]), and advanced voting procedures (such as ROE [18]), and can incorporate other existing certified defenses against data poisoning [9, 10, 14]. For any given aggregation defense, the MILP solution identifies the worst-case allocation of poisoned training datapoints across models for a batch of predictions, thereby tightly upper-bounding adversarial effectiveness. Across evaluations on MNIST [19], CIFAR-10 [20], and GTSRB [21], we show that our certificates consistently outperform existing approaches, improving the certified robustness rate by up to 14% over per-sample aggregation methods and up to 9% over naive multi-sample bounds [17].

Furthermore, we explore Abstract Gradient Training (AGT) [10] as an intrinsic robustness function applied to individual classifiers. Integrating intrinsic robustness in ensemble mechanisms is an approach that, to the best of our knowledge, absent from existing certification frameworks. We examine its potential to tighten certification bounds and outline directions for future research.

In summary, we make the following contributions:

- We present a general framework for certifying multi-sample poisoning robustness by modeling the adversary's optimal attack as a Mixed-Integer Linear Program (MILP). (Chapter 4)
- Our approach provides optimally tight robustness certificates for a wide class of aggregation-based defenses, including those with overlapping training subsets

(e.g. Finite Aggregation), nonlinear voting rules (e.g. Run-Off Election), and hybrid certified defenses.

- We apply our method across vision benchmarks and demonstrate improvements of up to 14% in certified accuracy over prior state-of-the-art and reveal general trends in designing robust aggregation schemes. (Chapter 5)
- We explore the potential gains of applying AGT to obtain intrinsic robustness at the model level, discuss its limitations, and lay the groundwork for future research directions. (Chapter 6)
- We propose a novel hybrid strategy to better balance the robustness-accuracy trade-off in ensemble methods and demonstrate its potential as a promising direction for future research (Chapter 6).

Chapter 2

Background

2.1 Adversarial Robustness

Adversarial robustness is a crucial aspect of building trustworthy machine learning systems, ensuring resilience against adversarial attacks. Adversarial attacks are small perturbations of the data fed into the machine learning pipeline. Adversaries may modify the data at either the *training stage* or *inference stage*.

Inference stage attacks involve crafting specific test inputs, called *adversarial examples*, to affect the output of a well-trained model. For example, Goodfellow et al. [22] demonstrated that applying small, carefully crafted perturbations to a test input from ImageNet can cause GoogLeNet to misclassify the image.

At the *training stage*, adversaries may manipulate the data used to train the model, compromising its behavior before deployment. This class of attacks is broadly referred to as *data poisoning*, where the adversary manipulates the training dataset to prevent the target model from converging, or perform abnormally on specific inputs. These attacks can have detrimental effects, especially for applications in critical domains such as medicine and autonomous driving.

The battle between attackers and defenders is an endless challenge in the field. In recent years, various research have been conducted to defend machine learning models against such attacks [9, 23, 24]. As part of the broader landscape of adversarial machine learning, this thesis narrows its focus to **data poisoning** and explores methods for certifying and improving robustness against such attacks.

2.2 Data Poisoning

The concept of data poisoning was first proposed by Barreno et al. [25]. The idea of data poisoning is to introduce malicious data into training datasets of target models to hinder model training, as illustrated in Figure 2.1. The original model training protocol is still followed, but the model is now trained on a poisoned dataset, en-

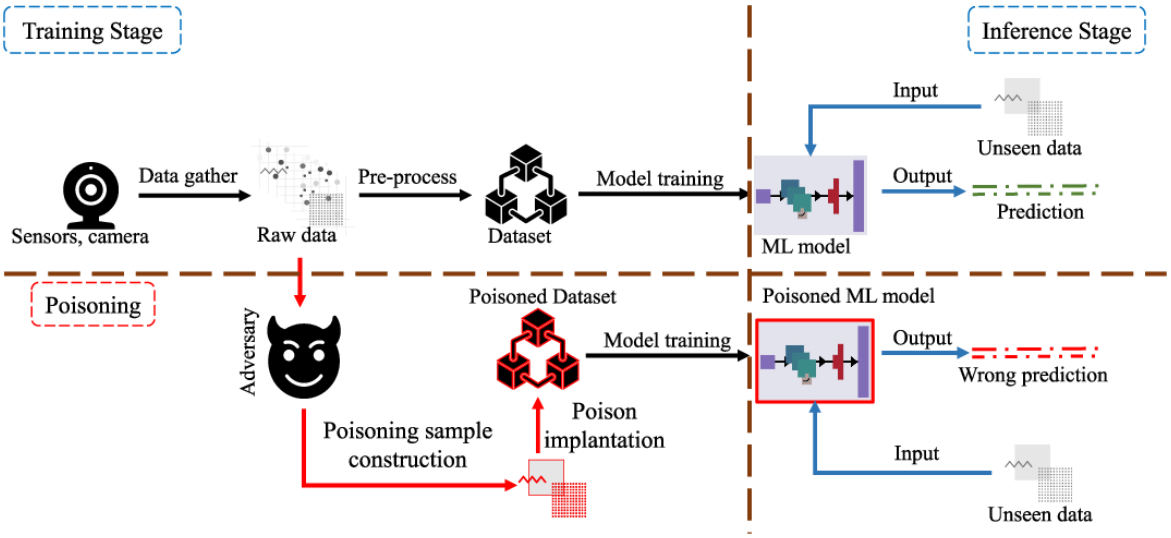


Figure 2.1: Architecture diagram of an adversary manipulating data during the training stage of a machine learning model. (adapted from [2])

abling the adversary to gain control over model behaviour. Poisoned data can be introduced either as raw data, such as sensor inputs like camera data, [26] or as feature files generated after pre-processing [27].

Especially with the rise of generative AI, machine learning models are growing to an extremely large scale, making them increasingly vulnerable to data poisoning. Carlini et al. [5] showed that poisoning just 0.01% of large scale foundation models, achievable by buying expired domains for $60 USD, can induce harmful behaviors. They also demonstrated that 6.5% of Wikipedia documents can be poisoned in the absence of adequate defensive measures, despite Wikipedia being a crowdsourced encyclopedia and one of the most reliable sources on the web. More recently, data poisoning has also been leveraged as a tool to protect intellectual property: Glaze [28] and Nightshade [29] apply subtle, imperceptible perturbations to images to prevent style mimicry by generative models and interfere with their training. In the medical field, Alber et al. [30] demonstrated that replacing 0.001% of training tokens with medical misinformation produces harmful LLMs prone to spreading medical errors.

2.2.1 Adversarial Objectives

The success of an adversarial modification to a dataset depends on the attacker's objective, which typically falls into one of three main categories:

1. **Untargeted Poisoning**: The goal of an adversary in an untargeted poisoning attack is to reduce the overall performance of a machine learning model. The adversary can achieve this by hindering convergence of target model and causing a denial of service, as illustrated in Figure 2.2.

   The challenge is to use the smallest possible poisoning data to affect the legit-

imate function of the target model to the greatest extent. For example, Mei et al. [31] proposed a general framework to optimise poisoning attacks on conventional models using convex optimisation, enabling effective attacks with minimal data perturbations. Untargeted attacks are less stealthy and can be easily detected by users.

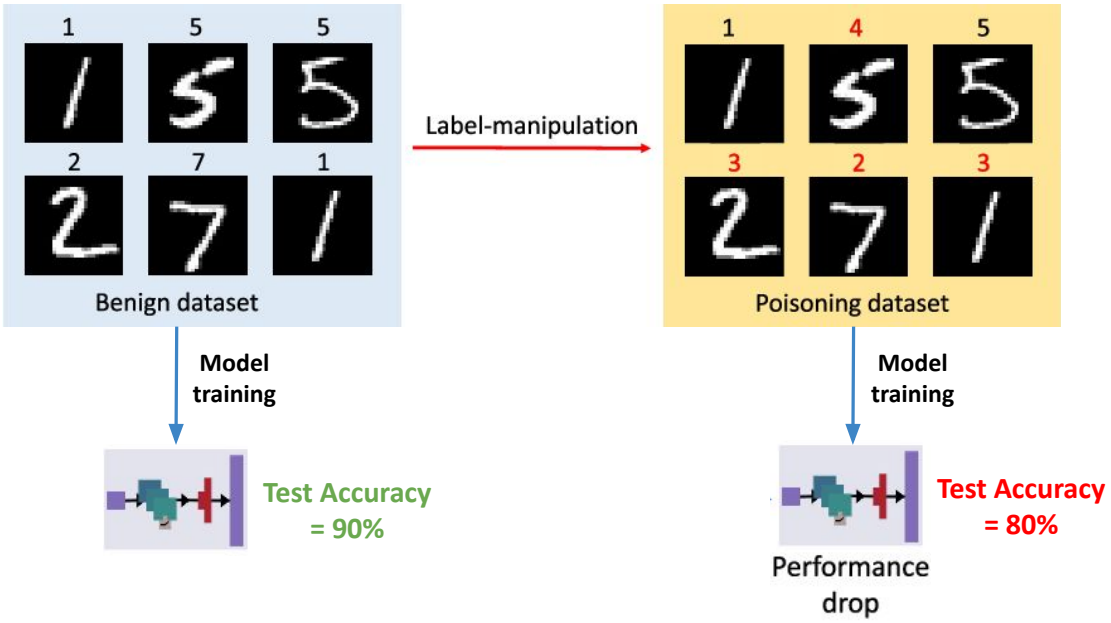


Figure 2.2: Illustration of Label Manipulation for Untargeted Attacks (adapted from [2]). In this illustration, training on a poisoned dataset with label manipulation reduces model accuracy by 10%.

2. **Targeted Poisoning**: Targeted attacks aim to cause the model to make incorrect predictions on specified samples, while ensuring legitimate function on other benign samples. Hence, targeted attacks are stealthier and more sophisticated to carry out. Several targeted poisoning attack have been proposed such as modifying training samples that significantly impact the target loss to reduce the performance of the target model on a designated subpopulation [32, 33].

3. **Backdoor Poisoning**: Backdoor poisoning attacks are a specific form of targeted poisoning in which an adversary embeds patterns, referred to as *triggers*, into poisoning samples. These triggers implant a backdoor into the model during training, allowing the adversary to later manipulate the model's behaviour when the trigger is present at inference time, as illustrated in Figure 2.3. Only test samples containing the same trigger can activate the hidden backdoor. Meanwhile, the model performs normally when inputs do not contain any triggers. This differentiates backdoor attacks from the other types of attacks where backdoor attacks assumes test-time samples can be manipulated to potentially include the implanted trigger. Such triggers can be embedded in various do-

mains such as deep learning networks [26], image classifiers [34] and language models [35].

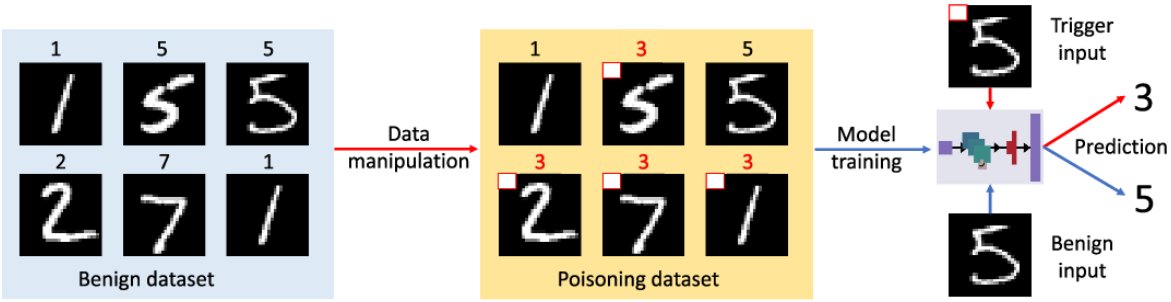


Figure 2.3: Illustration of Backdoor Attacks. We observe that the white box on the top left corner is added to the samples as a trigger, such that the classifier misclassifies the sample 5 as 3 when the white box is present in the test image. (adapted from [2])

2.2.2 Techniques of Poisoning attacks

An adversary can poison a machine learning model by adding carefully crafted malicious poisoned samples, or modifying/removing existing training samples. For example, Gu et al. [36] injects malicious samples by adding white pixels to the corners of training images to perform backdoor attacks, similar to what we observe in Figure 2.3.

A specific poisoning mechanism that has been widely studied is *label manipulation*, where the adversary manipulates only the labels of training data in a supervised/semi-supervised setting. For instance, Zhang et al. [37] demonstrated the effectiveness of label manipulation attacks by showing that a deep neural network can be forced to learn from incorrect features of random sample-label pairs, achieving 0 training error when trained on datasets where the true labels are replaced by random labels. Although label manipulation attacks are more straightforward to execute, these attacks are less stealthy, as they significantly decrease the performance of the model and lack the ability to achieve more sophisticated attacks such as backdoor attacks [2].

Early poisoning attacks relied on manual techniques, such as inserting words frequently found in legitimate emails into spam messages to mislead the SpamBayes filter [38]. Over time, these approaches have evolved into more sophisticated methods based on optimisation and model training. *Optimisation-based* methods generate poisoning samples by optimising an objective function designed to degrade model performance, such as maximising hinge loss in SVMs [27], using bilevel optimisation for deep learning models [7] or using Projected Gradient Ascent (PGA) for label manipulation [39]. These methods allow precise control but are computationally intensive and often converge to suboptimal solutions. To improve stealth, *clean-label* poisoning techniques create poisoned samples that visually resemble benign ones without altering the training data labels, making detection more difficult [40]. *Training-based* methods leverage auxiliary models, including isomorphic models and

generative models like GANs, to efficiently create poisoning samples. These methods simulate target model behavior or generate adversarial examples, improving attack efficiency but often at the expense of stealth. These data poisoning techniques can be applied in various domains such as computer vision, where pixel triggers are embedded [34], and NLP, where subtle textual triggers are introduced [35]. While these advanced methods broaden the scope of poisoning attacks, challenges remain in maintaining computational efficiency and undetectability.

2.2.3 Countermeasures

The dynamic between adversaries and defenders resembles a continuous cycle of adaptation and counter-adaptation, many countermeasures have been continuously developed to defend machine learning models against data poisoning attacks.

Data-driven countermeasures aim to improve model robustness by identifying and removing poisoned samples from the training dataset before model training. Early methods focused on outlier detection and data sanitisation [41], while later techniques measure sample impact [42] or use probabilistic approaches like importance reweighting to increase weighting of clean data [43]. Other approaches detect poisoned samples through optimisation, distance metrics, or differences in feature distributions and neuron behaviors [23, 44, 45, 46, 47]. Some methods detect backdoors by exploiting model sensitivity to perturbations [48]. Additionally, techniques like data augmentation and neuron pruning aim to reduce poisoning effects [49, 50]. However, these methods can risk removing clean data, increase computational cost, and struggle with high-dimensional data.

On the other hand, *model-driven countermeasures* focus on enhancing the robustness of learning algorithms during the training phase, often tailored to specific algorithms. For instance, SloppySVM [24] uses a nonconvex objective to resist label flipping, while other methods modify SVM kernels [51]. More general approaches include unbiased loss estimators and weighted loss functions to counter class-conditional label noise without altering model structures [52].

Certified Defenses

Many of the countermeasures discussed above are heuristic in nature and do not provide formal guarantees. However, in domains where safety is critical, such as healthcare or autonomous systems, it is essential to have ***certifiable robustness***—a formal guarantee that a model's predictions remain reliable even under worst-case data poisoning. Early work by Steinhardt et al. [8] provided robustness guarantees on overall accuracy, but under the assumption that the test data is drawn from the same distribution as the training set.

More recent research on **aggregation-based defenses** such as Deep Partition Aggregation (DPA) [15] and its extensions, Finite Aggregation (FA) [16] and Run-off Election, [18], offer stronger, state-of-the-art guarantees by certifying the robustness of *individual test samples*. The core idea behind these methods is to partition the training dataset into multiple disjoint subsets, each used to train a separate model

within the ensemble. By isolating the influence of any poisoned data point to only one member, the ensemble's final prediction can be aggregated in a way that dampens the impact of the corrupted member. This structure enables pointwise robustness certification by ensuring that, under bounded adversarial influence, a majority of the ensemble remains unaffected, thereby preserving prediction integrity.

Beyond aggregation-based defenses, significant research efforts focus on providing intrinsic robustness certification for individual models. Notable examples include bound propagation methods such as Abstract Gradient Training (AGT) [10], which aim to tightly estimate worst-case perturbations during training. Other prominent approaches include randomised smoothing techniques [9], which certifies robustness by averaging predictions over noisy inputs, and Differential Privacy-based methods [12], which provide robustness guarantees against data poisoning for differentially private learners.

However, the aggregation-based methods discussed above certify each test point *individually*, implicitly assuming that the adversary can selectively target and manipulate models *after* observing each prediction. In practice, adversarial objectives are often defined over a **batch** of test points. As a result, per-sample certificates fall short in offering direct robustness guarantees for batches of test points. Using pointwise certificates to estimate the fraction of certified points in a batch introduces looseness, since the adversary cannot realistically poison all ensemble members that yield worst-case predictions across every test sample in the batch.

An initial effort by Chen et al. [53] addresses multi-sample certification by formulating worst-case predictions over a batch of test samples as a Binary Integer Program (BIP). However, this approach is restricted to the simplest Deep Partition Aggregation (DPA) method and does not support more advanced aggregation mechanisms like Finite Aggregation (FA) [16] or Run-off Election (ROE) [18]. Moreover, the BIP formulation treats each ensemble member as either poisoned or not, without accounting for the intrinsic robustness of individual members. This thesis proposes a new formulation to overcome these limitations. In the next chapter, we provide a detailed review of these exisiting certified defenses, laying the foundation for our formulation in Chapter 4.

Chapter 3

Preliminaries

In this section, we cover the technical foundations of the supervised machine learning setup we consider, as well as the technical details from prior work on which we build our methodology.

3.1 Setup and Definitions

We denote a machine learning model as a function f with feature space $x \in \mathbb{R}^n$ and label space $y \in \mathcal{Y}$. In a supervised learning setting, we denote our training dataset as $\mathcal{D} := \{(x^{(i)}, y^{(i)})\}_{i=1}^n$. We denote $\mathcal{M}$ as the training function that returns the trained parameter θ, given the parameter initialisation θ', model f and data $\mathcal{D}$, i.e. $\theta = \mathcal{M}(f, \theta', \mathcal{D})$.

Finally, we denote the element-wise loss function as $\mathcal{L}(f(x^{(i)}), y^{(i)})$. We also denote $[k]$ as the interval of natural numbers from $0, \cdots, k$. $A \triangle B$ represents the symmetric set difference between A and B: $A \triangle B = (A \backslash B) \cup (B \backslash A)$.

We first introduce general definitions to build an understanding of **certification**. These definitions serve as the foundation for formalising certification against data poisoning, as detailed in the next subsections. Section 3.1.1 defines the perturbation sets used in Definition 1. Section 3.1.2 specifies the adversarial goals, where upper bounding these goals corresponds to properties we aim to certify, as formalised in Definition 2. Lastly, Section 3.1.3 outlines how certification is defined in relation to these adversarial goals.

Definition 1 (Sensitivity)**.** *The sensitivity of the function g with respect to a perturbation set $\mathcal{T}$ is given by:*

$$S(g, \mathcal{T}) := \{g(x') | x' \in \mathcal{T}\}$$

In other words, sensitivity is the set of possible outputs that a function g can attain when its input is perturbed within the set $\mathcal{T}$.

Definition 2 (Property Satisfaction)**.** *Let $P : (\mathcal{T}_p, S_p)$ denote a property defined by an input perturbation set $\mathcal{T}_p$ and an output constraint set S_p. A function g satisfies P if the sensitivity of g with respect to $\mathcal{T}_p$ is a subset of S_p, i.e. $S(g, \mathcal{T}_p) \subseteq S_p$.*

Definition 3 (Certification)**.** *A property $P : (\mathcal{T}_p, S_p)$ can be proven to hold for an objective function g with sensitivity $S(g, \mathcal{T}_P)$, if:*
Given a function h that is sound by construction, i.e. always returns a set C that is a superset of $S(g, \mathcal{T}_P)$, then we can say that C certifies the property P for function g if $S(g, \mathcal{T}_p) \subseteq C \subseteq S_p$.

In other words, certification of a property is achieved by constructing an over-approximation of the objective function's sensitivity that still lies within the allowed output constraints of the property. For example, if we can compute a sound upper bound b for the objective function g w.r.t. input perturbation set $\mathcal{T}$, then the strongest property that we have certified for g is $P : (\mathcal{T}, (-\infty, b])$.

*In the data poisoning settings we consider, the objective function g in the definition corresponds to the **adversarial objectives** (3.3, 3.4, 3.5), while the function h represents the certified defense mechanism.*

3.1.1 Data Poisoning Threat Model

Here we define the perturbation set $\mathcal{T}(\mathcal{D})$, which is the set of all possible poisoned datasets in different poisoning settings. Throughout the work, we will denote this as $\mathcal{T}$ for simplicity. This thesis considers data poisoning attacks in their *most general form*, which encompasses all types of poisoning, including label manipulation, but excluding backdoor attacks. As backdoor poisoning combines both training and inference-stage manipulation, and cannot be fully addressed by aggregation-based defenses alone, we do not provide an in-depth analysis. Nonetheless, it remains an important and promising direction for future research.

General Form: This is the most general poisoning setting considered, where the adversary can modify (including replacements, insertions and deletions) up to K points. We define all possible poisoned datasets as $\mathcal{T}_K(\mathcal{D}) := \{\tilde{\mathcal{D}} \text{ s.t., } |\tilde{\mathcal{D}} \triangle \mathcal{D}| \leq K\}$.

More specific data poisoning settings have also been studied, such as feature poisoning [54] and label poisoning [55]. Commonly analysed attack scenarios include (1) l_p-norm bounded attacks, and (2) unbounded attacks, both of which typically assume adversarial control over features and labels simultaneously [10].

l_p-norm Bounded Attacks: l_p-norm bounded attacks are commonly considered in backdoor attacks. The adversary perturbs a subset of the *training* data up to at most K_f data points in feature space within ϵ distance in l_p-norm, and at most K_l data points in label space within ν distance in l_q-norm. Common label-flipping attacks can be represented by setting $q = 0$ (hence restricting the number of labels flipped). Formally, given dataset $\mathcal{D}$ and an adversary $\langle K_f, \epsilon, p, K_l, \nu, q \rangle$:

$$\mathcal{T}^{K_f,\epsilon,p}_{K_l,u,q}(\mathcal{D}) := \left\{ \widetilde{\mathcal{D}} \;\middle|\; \begin{array}{l} \|\mathcal{D}_x^{(i)} - \widetilde{\mathcal{D}}_x^{(i)}\|_p \leq \epsilon,\ \forall i \in I, \mathcal{D}_x^{(i')} = \widetilde{\mathcal{D}}_x^{(i')},\ \forall i' \notin I,\ \forall I \in \mathcal{S}_{K_f}, \\ \|\mathcal{D}_y^{(j)} - \widetilde{\mathcal{D}}_y^{(j)}\|_q \leq \nu,\ \forall j \in J, \mathcal{D}_y^{(j')} = \widetilde{\mathcal{D}}_y^{(j')},\ \forall j' \notin J,\ \forall J \in \mathcal{S}_{K_l} \end{array} \right\} \tag{3.1}$$

where S_z is the set of all subsets of integers less than the number of training datapoints with cardinality at most z. The index sets I and J refer to the data points that have been poisoned in the feature and label spaces, respectively.

Unbounded Attacks. In most cases, the adversary may inject entirely new poisoned data points to the training set, or modify the feature and labels arbitrarily, in which case we consider unbounded attacks. Here, we consider only a 'paired' modification setting, where an adversary can substitute or manipulate any of the features or labels up to K data points in the training set. Formally, for a dataset $\mathcal{D}$, the set of potentially poisoned dataset is:

$$\mathcal{T}_K(\mathcal{D}) := \left\{ \widetilde{\mathcal{D}} = (\mathcal{D} \setminus \mathcal{D}^r) \cup \mathcal{D}^a \;\middle|\; |\mathcal{D}^a| \leq K,\ |\mathcal{D}^r| \leq K,\ \mathcal{D}^r \subset \mathcal{D} \right\} \tag{3.2}$$

where $|\cdot|$ is the cardinality operator, $\mathcal{D}^a$ is the set of arbitrary added data points, and $\mathcal{D}^r$ is the set of corresponding data points removed from the original dataset.

3.1.2 Adversarial Goals

The different goals an adversary may seek to achieve can be split into 3 main categories: untargeted poisoning, targeted poisoning and backdoor poisoning.

1. **Untargeted Poisoning**: The adversary's objective is to degrade overall performance of the model. Formally, the adversary's objective is to maximise the number of predictions on unseen data that fall outside the 'safe' set of output $S(x^{(i)}, y^{(i)})$ (i.e. the set of output that is considered close to the ground truth). For a test set $\{(x^{(i)}, y^{(i)})\}_{i=1}^N$, the objective can be formulated as:

$$\max_{D' \in \mathcal{T}} \frac{1}{N} \sum_{i=1}^{N} \mathbb{1}\left(f^{\mathcal{M}(f,\theta',D')}(x^{(i)}) \notin S(x^{(i)}, y^{(i)}) \right) \tag{3.3}$$

2. **Targeted Poisoning**: In targeted poisoning, the adversary aims to degrade model performance on a specific subset of test data, such as images of a specific class. Formally, we express the subset of the test set as $\{(x_T^{(i)}, y_T^{(i)})\}_{i=1}^N$ and the adversary's objective as:

$$\max_{D' \in \mathcal{T}} \frac{1}{N} \sum_{i=1}^{N} \mathbb{1}\left(f^{\mathcal{M}(f,\theta',D')}(x^{(i)}) \notin S(x_T^{(i)}, y_T^{(i)}) \right) \tag{3.4}$$

3. **Backdoor Poisoning**: The adversary's objective in a backdoor attack is to maximise the number of predictions on manipulated test-time samples containing the trigger that fall out of the 'safe' set. Assuming the trigger manipulations are bounded to a set $V(x)$, we can formulate the adversary's objective as:

$$\max_{D' \in \mathcal{T}} \frac{1}{N} \sum_{i=1}^{N} \mathbb{1}\left(\exists x^\star \in V(x^{(i)})\ s.t.\ f^{\mathcal{M}(f,\theta',D')}(x^\star) \notin S(x^{(i)}, y^{(i)}) \right) \tag{3.5}$$

Note that these adversarial goals are measured as *averages* over a *set of test points*. This is a key observation that we will leverage in our formulation in Chapter 4.

3.1.3 Certified Robustness

To certify robustness against the poisoning attacks, we evaluate and compute a *sound* **upper bound** for the sensitivity of the adversarial objectives (3.3), (3.4), and (3.5). In other words, we bound the success rate of the adversary by ensuring that these objectives remain within the range $[0, b]$ for an upper bound b given that the adversary can change K points.

3.2 Certified Defenses

Various approaches have been studied to certify robustness against data poisoning. Notable recent defenses include:

1. **Aggregation-based Defenses**
   Aggregation-based defenses combat data poisoning by dividing the training data into multiple subsets and then combining the outputs of models independently trained on each subset during inference time. Several ensemble methods, like Deep Partition Aggregation (DPA) [15], Finite Aggregation [16], and Run-Off Election (ROE) [18], offer strong pointwise certificates against poisoning. They achieve this by splitting the training dataset into *disjoint* subsets, ensuring that a poisoned data point can only affect one sub-model's prediction. While these methods partition the data across training samples, Feature Partition Aggregation (FPA) [56] instead partitions across the feature space, enabling certified robustness in that domain. In contrast, Jia et al. [57] extends the partitioning concept to recommender systems, providing certification over recommendation outputs rather than individual classification predictions. In Section 3.2.1, we discuss DPA, FA and ROE in detail.

2. **Bound Propagation**
   Bound propagation approaches, such as Interval Bound Propagation (IBP) [58] and CROWN [59] were initially developed to certify adversarial examples by computing upper and lower bounds on a model's output in response to perturbations in the input. Abstract Gradient Training (AGT) [10] extends these methods to provide formal guarantees against data poisoning attacks. A detailed discussion of AGT is provided in Section 3.2.2.

3. **Randomised Smoothing**
   Randomised smoothing constructs a robust classifier by averaging the predictions of the original model over noisy versions of the input. It is initially used to certify against adversarial robustness [53], and was extended to providing certifications against label-flipping data poisoning attacks. [9, 13].

4. **Differential Privacy and Poisoning Robustness**
   Differential Privacy (DP) offers formal guarantees for protecting individual

data privacy. In machine learning models, it is commonly implemented using DP-SGD [60], which privatises model parameters and any predictions released [61]. Recent works [12, 14] provides provably certified data poisoning guarantees for differentially private learners.

3.2.1 Aggregation-Based Defenses

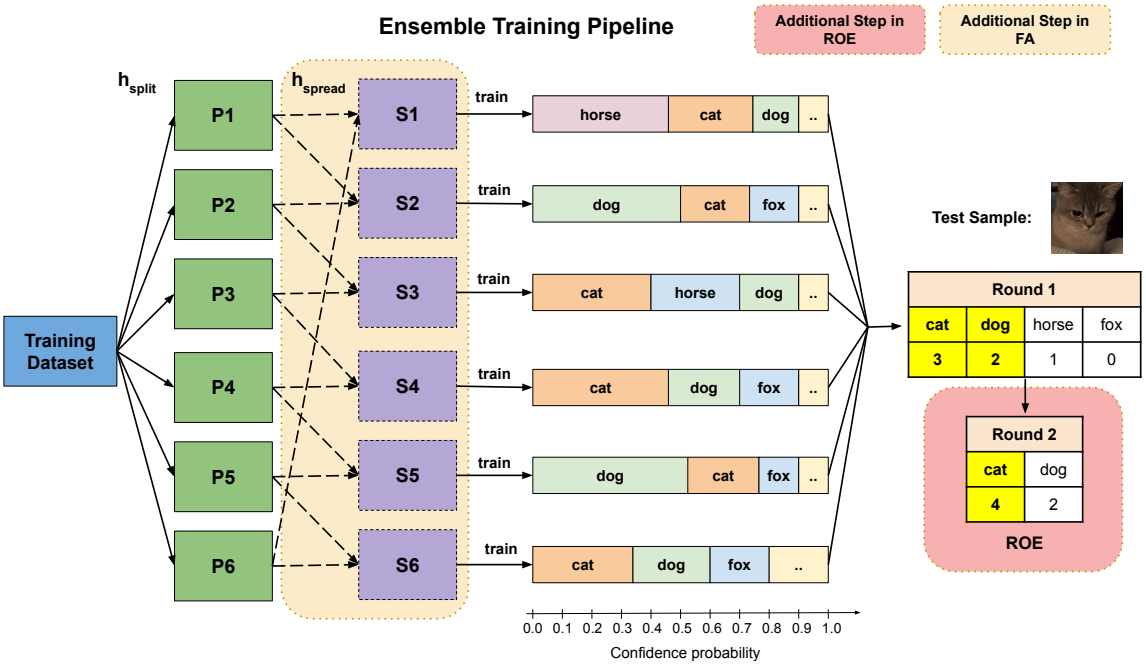


Figure 3.1: An illustration of how DPA, FA, and ROE integrate into an ensemble training pipeline. The initial partitioning step is shared between DPA with $M = 6$ and FA with $M = 3, d = 2$, which produces $P_1, \ldots, P_6$ via h_{split}. However, from this point forward, the two methods diverge. DPA uses these partitions $P_1, \ldots, P_6$ directly to train member classifiers, while FA further spreads these partitions $P_1, \ldots, P_6$ via h_{spread} into new partitions $S_1, \ldots, S_6$ before training. It is worth noting that when $d = 1$ and h_{spread} is injective, i.e. for each S_j, there exists a unique pre-image P_i, FA reduces to the standard DPA case. Finally, ROE is applied independently of DPA and FA at inference time during the aggregation step.

One of the most notable aggregation-based defenses against data poisoning is Deep Partition Aggregation (DPA) [15]. This technique provide certification against general data poisoning attacks, which includes untargeted and targeted attacks. In this section, we discuss in detail DPA and its extensions: Finite Aggregation (FA), DPA* and Run-Off Election (ROE). For these aggregation mechanisms, the maximum possible difference in votes between the top two classes is bounded by M, the number of models in the ensemble. Consequently, the largest certifiable perturbation radius is $\lfloor M/2 \rfloor$. Figure 3.1 illustrates how DPA, FA and ROE fits into the ensemble training pipeline.

Deep Partition Aggregation (DPA)

Deep Partition Aggregation (DPA) [15] offers pointwise certification for each test sample against general poisoning threats, where a bounded number of insertions or deletions occur in feature or label space in the training set. For a given training dataset D, the DPA algorithm is outlined as followed:

1. At training time, define a deterministic hash function $h_{split} : \mathcal{D} \to \mathbb{N}$ to divide the training set into M disjoint partitions $P_1, P_2, \ldots, P_M \subseteq \mathcal{D}$, where M is a hyperparameter, such that

$$P_i := \{\boldsymbol{s} \in \mathcal{D} | h(\boldsymbol{s}) \equiv i \bmod M\}$$

   Ideally, h_{split} should map samples to a domain significantly larger than M so that the partition sizes are roughly equal over $[M]$.

2. The base classifier model is then trained over each partition separately.

3. At inference time, evaluate the input on each base classifier, then count the number of classifiers which return each class.

4. Return the classifier with consensus output of the ensemble.

$$g_{\text{dpa}}(\mathcal{D}, x) := \arg\max_{c \in \mathcal{Y}} v_c(x)$$

   where $v_c(x) = \sum_{j=1}^{M} \mathbb{1}[f_j(x) = c]$.

DPA results in a robust classifier with the following guarantees:

For a fixed deterministic base classifier f, hash function h, ensemble size M, training set $\mathcal{D}$, and input x, let:

$$c := g_{\text{dpa}}(\mathcal{D}, x)$$

$$\text{Cert}_{\text{DPA}}(x) := \left\lfloor \frac{v_c - \max_{c' \neq c}(v_{c'}(x) + \mathbb{1}_{c'<c})}{2} \right\rfloor .$$

where g_{dpa} is the prediction obtained from the above DPA algorithm. Then, for any poisoned training set $\mathcal{T}(\mathcal{D})$, if $|\mathcal{D} \triangle \mathcal{T}(\mathcal{D})| \leq \text{Cert}_{\text{DPA}}(x)$, then $g_{\text{dpa}}(\mathcal{T}(\mathcal{D}), x) = c$. $\text{Cert}_{\text{DPA}}(x)$ is half of the gap between the votes of the top 2 classes. In other words, denoting c_{pred} as the predicted class and c_{sec} as the class with the second highest votes, $\text{Cert}_{\text{DPA}}(x)$ is the number of votes required to change from the c_{pred} to c_{sec} before the ensemble classifier no longer predicts c_{pred}. This effectively provides robustness certification against general poisoning attacks, and can also be used in a deep learning setting.

Finite Aggregation (FA)

Finite Aggregation (FA) [16] is an extension of DPA that improves certified robustness by strategically reusing every sample. FA first partitions the dataset into $M \cdot d$ disjoint partitions $P_1, P_2, \ldots, P_{Md} \subseteq \mathcal{D}$ with a deterministic function h_{split}. Each partition P_i is then redistributed to d different destinations with a function h_{spread}, forming $\mathcal{S}_1, \mathcal{S}_2, \ldots \mathcal{S}_{Md}$ subsets. Each classifier f_j is then trained on a subset of the training dataset which is a union of the partitions that spreads to it, $\mathcal{S}_j := \bigcup_{j \in h_{spread}(i)} P_i$. Just as in DPA, the votes are aggregated across classifiers at test time. Since the number of classifiers is d times greater than in DPA, and each classifier is trained on a subset of the same size as those used in DPA, the total training time increases by a factor of d.

FA results in a robust classifier with the following guarantees:

For a fixed deterministic base classifier f, hash functions h_{split}, h_{spread}, ensemble size M, training set $\mathcal{D}$, and input x, let:

$$c := g_{\text{fa}}(\mathcal{D}, x)$$

$$\overline{\Delta^{\text{Cert}_{\text{FA}}}_{\mathcal{D},x}} \leq \underbrace{\sum_{j=1}^{Md} \mathbb{1}[f_j(x) = c] - \sum_{j=1}^{Md} \mathbb{1}[f_j(x) = c'] - \mathbb{1}[c' < c]}_{\text{Margin between } c \text{ and } c'}$$

holds for all $c' \neq c$, where $g_{\text{fa}}(\mathcal{D}, x)$ is the prediction obtained from FA and $\Delta_{\mathcal{D},x}$ is a multiset defined as

$$\left\{ d + \sum_{i \in h_{spread}(j)} \mathbb{1}[f_i(x) = c] - \sum_{i \in h_{spread}(j)} \mathbb{1}[f_i(x) = c'] \right\}_{j \in [Md]}$$

$\overline{\Delta^{\text{Cert}_{\text{FA}}}_{\mathcal{D},x}}$ denotes the sum of the largest Cert_{FA} elements in the multiset $\Delta_{D,x}$. Then, for any poisoned training set $\mathcal{T}(\mathcal{D})$, if $|\mathcal{D} \triangle \mathcal{T}(\mathcal{D})| \leq \text{Cert}_{\text{FA}}$, then $g_{\text{fa}}(\mathcal{T}(\mathcal{D}), x) = c$.

Intuitively, each element of $\Delta_{D,x}$ represent the largest change to the margin between c and c' for any one partition j poisoned. When a single training sample is poisoned, it can affect at most d classfiers. The margin between class c and c' decreases by 2 for every classifer $f_i(x)$ whose prediction changes from c to c', by 1 for each that changes from another class to c', and remains unchanged for classifiers that originally predicts c'. Hence, the maximum change to the margin between c and c' computed by summing up the largest $\text{Cert}_{\text{FA}}(x)$ elements of $\Delta_{\mathcal{D},x}$.

This guarantee for FA presented in Wang et al. [16] has been shown to be tighter than that provided by DPA. However, it can still be loose as it may overcount the influence of poisoned partitions when multiple such partitions affect the same resulting partition via h_{spread}. In Section 4.2.2, we present an alternative formulation of this certificate that avoids such overcounting, resulting in a tighter guarantee even at the per-sample level.

DPA*

DPA* or Boosted DPA [18], is a variation of DPA that trains d classifiers with different random seeds on each partition P_i, resulting in an ensemble size of $M * d$. At test time, the logits of each d classifiers are then averaged to give a single prediction for classifiers trained on partition P_i. This approach enhances the stability of predictions by mitigating the effects of noise introduced during the training process. The robustness certificate is calculated as per that of DPA. Rezaei et al. [18] empirically demonstrated that DPA* outperforms DPA.

Run-Off Election (ROE)

Run-Off Election (ROE) [18] is an improved aggregation mechanism that can be used in conjunction with other aggregation methods like DPA, FA and DPA*. ROE improves robustness certificates through a two-stage voting mechanism that widens the gap between the top two classes. In the first stage of voting, the top two classes with the highest votes are shortlisted. Formally, the top two classes are:

$$c_1 := \arg\max_{c \in \mathcal{Y}} v_c(x) \text{ and } c_2 := \arg\max_{c \neq c_1 \in \mathcal{Y}} v_c(x)$$

In the second round of voting, votes are collected from each model in an election between c_1 and c_2 based on the logits layer of the models. Formally, for an ensemble with size M:

$$g_{\text{roe}}(\mathcal{D}, x) := \arg \max_{\substack{c \in \{c_1, c_2\} \\ \text{s.t. } c' \in \{c_1, c_2\} \neq c}} \sum_{j=1}^{M} \mathbb{1}[f_j^{\text{logits}}(x, c) > f_j^{\text{logits}}(x, c')]$$

We can now calculate the robustness certificate for the prediction. We denote the predicted class as c_{pred} and the class with the second highest votes as c_{sec}. The adversary can change the prediction in one of the two ways:

1. c_{pred} is eliminated in Round 1. This means that the adversary needs to choose 2 classes to have more votes than c_{pred} in Round 1. The corresponding certificate can be formally defined as:

   $$\text{Cert}_{\text{R1}}(x) := \min_{c_1, c_2 \in \mathcal{Y} \setminus c_{\text{pred}}} \text{Cert}_{\text{V2}}(x, c_{\text{pred}}, c_1, c_2)$$

   where $\text{Cert}_{\text{V2}}(x, c_{\text{pred}}, c_1, c_2)$ represents the number of points that needs to be poisoned for both c_1 and c_2 to beat c_{pred}.

2. c_{pred} is eliminated in Round 2. For this to happen, the final predicted class after poisoning, c, must defeat either c_{pred} or c_{sec} in Round 1. Since we have already assumed that c_{pred} advances to Round 2, it follows that c must surpass c_{sec} in Round 1. Next, c needs to beat c_{pred} in Round 2 of voting which is based on the logits layer. Formally:

   $$\text{Cert}_{\text{R2}}(x) := \min_{c \neq c_{\text{pred}}} \max\{\text{Cert}_{\text{V1}}(x, c_{\text{sec}}, c), \text{Cert}_{\text{V1}}^{\text{logits}}(x, c_{\text{pred}}, c)\}$$

where $\text{Cert}_{\text{V1}}(x, c_{\text{sec}}, c)$ is the number of points that needs to be poisoned for c to surpass c_{sec}. $\text{Cert}_{\text{V1}}^{\text{logits}}(x, c_{\text{pred}}, c)$ is the number of points that needs to be poisoned for c to surpass c_{pred} using the logits softmax score $s_j(x, c)$, where each classifier votes for c if $s_j(x, c) > s_j(x, c_{\text{pred}})$ and vice versa, with ties broken arbitrarily.

Finally:

$$\text{Cert}_{\text{ROE}}(x) := \min\{\text{Cert}_{\text{R1}}(x), \text{Cert}_{\text{R2}}(x)\} - 1$$

Note that the -1 term arises from $\text{Cert}_{\text{R1}}(x)$ and $\text{Cert}_{\text{R2}}(x)$ being defined as the *minimum number of points* needed to eliminate c_{pred} in Round 1 and Round 2 respectively.

If $|\mathcal{D} \triangle \mathcal{T}(\mathcal{D})| \leq \text{Cert}_{\text{ROE}}(x)$, then $g_{\text{roe}}(\mathcal{T}(\mathcal{D}), x) = c_{\text{pred}}$. $\text{Cert}_{\text{V1}}(x)$, $\text{Cert}_{\text{V2}}(x)$ and $\text{Cert}_{\text{V1}}^{\text{logits}}(x)$ are defined based on the different aggregation mechanisms used in conjunction with ROE. The ROE formulation in our specific setting will be detailed in the next chapter.

3.2.2 Abstract Gradient Training (AGT)

In this section, we present the AGT framework, beginning with a brief overview of interval arithmetic fundamentals.

Interval Arithmetic

We denote the intervals over matrices in bold as $\boldsymbol{A} := [A_L, A_U] \subseteq \mathbb{R}^{n \times m}$ such that $\forall A \in \boldsymbol{A}, A_L \leq A \leq A_U$ element-wise.

Definition 4 (Interval Matrix Arithmetic)**.** *Let $\boldsymbol{A} = [A_L, A_U]$ and $\boldsymbol{B} = [B_L, B_U]$ be intervals over matrices. Let $\oplus, \otimes, \odot$ represent interval matrix addition, matrix multiplication and element-wise multiplication, such that*

$$\begin{aligned} A + B &\in \boldsymbol{A} \oplus \boldsymbol{B} \quad \forall A \in \boldsymbol{A}, B \in \boldsymbol{B} \\ A \times B &\in \boldsymbol{A} \otimes \boldsymbol{B} \quad \forall A \in \boldsymbol{A}, B \in \boldsymbol{B} \\ A \circ B &\in \boldsymbol{A} \odot \boldsymbol{B} \quad \forall A \in \boldsymbol{A}, B \in \boldsymbol{B} \end{aligned}$$

where $\circ$ represents element-wise multiplication.

These operations can be computed using standard arithmetic techniques and incur at most $4\times$ the cost of standard matrix operation[10].

Abstract Gradient Training

Abstract Gradient Training (AGT) [10] is a sound framework that certifies robustness against untargeted and targeted data poisoning attacks without modifying the model or learning algorithm. AGT bounds the set of all reachable parameters by over-approximation for any gradient-based learning algorithm such as stochastic gradient descent (SGD) or Adam. We will outline the application of this framework on a neural network trained using SGD.

We define a neural network $f^\theta : \mathbb{R}^{n_{in}} \to \mathbb{R}^{n_{out}}$ with η layers and parameters $\theta = \{(W^{(i)}, b^{(i)})\}_{i=1}^{\eta}$ as:

$$\hat{z}^{(k)} = W^{(k)} z^{(k-1)} + b^{(k)}, z^{(k)} = \sigma(\hat{z}^{(k)})$$

where $z^{(0)} = x, f^\theta(x) = \hat{z}^{(\eta)}$ and σ is the activation function that we assume to be monotonic.

The algorithm bounds all reachable parameters with the following steps:

1. Initialise the initial parameter bounds $[\theta^L, \theta^U]$ to the initial parameter $[\theta', \theta']$

2. At each iteration

   (a) Compute the SGD update for each batch $\mathcal{B}$

   $$\Delta\theta \leftarrow \frac{1}{|\mathcal{B}^{(t)}|} \sum_{(x,y)\in\mathcal{B}^{(t)}} \nabla_\theta \mathcal{L}(f^\theta(x), y)$$
   $$\theta^{(t+1)} \leftarrow \theta^{(t)} - \alpha\Delta\theta$$

   (b) Compute the bounds of the set $\Delta\Theta$, which is the set of descent directions possible under all allowed dataset perturbations by the poisoning adversary $\mathcal{T}$, i.e. Find $\Delta\theta^L$, $\Delta\theta^U$ such that $\forall\Delta\theta \in \Delta\Theta, \Delta\theta^L \preceq \Delta\theta \preceq \Delta\theta^U$.

   $$\Delta\Theta^{(t)} \leftarrow \left\{ \frac{1}{|\tilde{\mathcal{B}}|} \sum_{(x,y)\in\tilde{\mathcal{B}}} \nabla_\theta \mathcal{L}(f^{\theta^*}(x), y) | \tilde{\mathcal{B}} \in \mathcal{T}(\mathcal{B}^{(t)}), \theta^* \in [\theta^L, \theta^U] \right\} \quad (3.6)$$

   $\mathcal{T}(\mathcal{B}^{(t)})$ represents the effect of the poisoning adversary on the current batch, and $[\theta^L, \theta^U]$ is the bound obtained on parameter θ, taking into account the effect of the poisoning adversary on all previously seen batches. Computing the bounds $\Delta\theta^L$, $\Delta\theta^U$ is further discussed below. (§3.7)

   (c) Update the reachable parameter bounds for this iteration under adversary $\mathcal{T}$.

   $$[\theta^{(t),L}, \theta^{(t),U}] \leftarrow [\theta^{(t-1),L} - \alpha\Delta\theta^U, \theta^{(t-1),U} - \alpha\Delta\theta^L]$$

3. Finally, after E iterations, we obtain the final learned parameter for this dataset, $\theta^{(E)}$ and its bounds on all reachable parameters $[\theta^{(E),L}, \theta^{(E),U}]$ subject to poisoning adversary $\mathcal{T}$.

4. With this sound bound on all reachable parameters $[\theta^{*L}, \theta^{*U}] = [\theta^{(E),L}, \theta^{(E),U}]$, we can then find an upper bound of the adversarial objective functions (3.3), (3.4), and (3.5) over the parameter interval $[\theta^{*L}, \theta^{*U}]$, effectively upper bounding the success rate of the adversary.

Computing sound bounds on $\Delta\Theta$

In step 2(b), we want to bound the set $\Delta\Theta$ which is the set of all possible descent directions for the current batch $\mathcal{B}$, $\Delta\theta^L, \Delta\theta^U$.

Firstly, we compute the *element-wise sound gradient bound*. For each training sample $(x^{(i)}, y^{(i)}) \in \mathcal{B}$, we want to find $\delta^{(i),L}, \delta^{(i),U}$, which are defined respectively as:

$$\min \& \max \left\{ \frac{\partial}{\partial \theta^\star} \mathcal{L}\left(f^\theta(\tilde{x}), \tilde{y}\right) \;\middle|\; \theta^\star \in [\theta^L, \theta^U],\ \|\tilde{x} - x^{(i)}\|_p \leq \epsilon,\ \|\tilde{y} - y^{(i)}\|_q \leq \nu \right\} \quad (3.7)$$

where $(\tilde{x}, \tilde{y})$ represents the perturbed instance of the training sample $(x^{(i)}, y^{(i)})$.

To compute Equation (3.7), we first need to compute sound bounds for the *output of the neural network* f with interval parameters. That is, $\forall \tilde{x} \in [x^{(i),L}, x^{(i),U}], f^\theta(\tilde{x}) \in [\gamma^L, \gamma^U]$. Given interval bounds on both the parameters θ and input $x^{(i)}$, these can be propagated through the network's layers to yield valid output bounds for $f^\theta(\tilde{x})$ using bound propagation techniques such as IBP[58] or CROWN [59]. Using these forward-pass intervals, the input bounds around the input $x^{(i)}$ in our original objective function (Equation (3.7)) can be replaced with the corresponding forward-pass bounds, resulting in the following relaxed objective:

$$\min \& \max \left\{ \frac{\partial}{\partial \theta^\star} \mathcal{L}\left(f^\theta(\tilde{x}), \tilde{y}\right) \;\middle|\; \theta^\star \in [\theta^L, \theta^U],\ f^\theta(\tilde{x}) \in [\gamma^L, \gamma^U],\ \|\tilde{y} - y^{(i)}\|_q \leq \nu \right\} \quad (3.8)$$

Note that Equations (3.7) and (3.8) are equivalent formulations, differing only in the parts highlighted in blue. Equation (3.8) is the formulation of the original objective (3.7) where we have already bounded the forward pass of the input $\tilde{x}$.

To solve Equation (3.8), we need to back-propagate the forward-pass intervals through the intermediate layers of the neural networks using interval arithmetic. By utilising the differentiation chain rule, we can carry out automatic backward differentiation [62]. We first compute the $\frac{\partial \mathcal{L}}{\partial \tilde{y}}$ bound, which is loss function specific. Combined with the following expressions, the forward-pass intervals can be back-propagated through the network to compute intervals over all gradients:

$$\frac{\partial \mathcal{L}}{\partial \boldsymbol{z}^{(k-1)}} = \left(\boldsymbol{W}^{(k)}\right)^\top \otimes \frac{\partial \mathcal{L}}{\partial \hat{\boldsymbol{z}}^{(k)}}, \quad \frac{\partial \mathcal{L}}{\partial \hat{\boldsymbol{z}}^{(k)}} = \frac{\partial}{\partial \sigma} \odot \frac{\partial \mathcal{L}}{\partial \boldsymbol{z}^{(k)}}$$

$$\frac{\partial \mathcal{L}}{\partial \boldsymbol{W}^{(k)}} = \frac{\partial \mathcal{L}}{\partial \hat{\boldsymbol{z}}^{(k)}} \otimes \left[\left(z^{(k-1),L}\right)^\top, \left(z^{(k-1),U}\right)^\top\right], \quad \frac{\partial \mathcal{L}}{\partial \boldsymbol{b}^{(k)}} = \frac{\partial \mathcal{L}}{\partial \hat{\boldsymbol{z}}^{(k)}}$$

This yields sound element-wise bounds for both the original objective function (3.7) and its relaxation (3.8), from which we can compute $\Delta\theta^L, \Delta\theta^U$.

At each iteration t, we want to update the gradient bounds based on the current batch sampled $\mathcal{B}$. We assume that in every batch, the features of at most m samples

could be manipulated, and up to n labels may be flipped. To find sound updated gradient bounds $\Delta\theta^L, \Delta\theta^U$, we assume that the adversary manipulates the $m+n$ most sensitive samples, which are those that would cause the greatest change in gradient bounds if altered. Here we note that as the batch size $b = |\mathcal{B}|$ tends to ∞, the bounds proposed become infinitely tight, since $m+n$ becomes insignificant with respect to the batch size. Formally, the sound gradient bounds $\Delta\theta^L, \Delta\theta^U$ can be computed by:

$$\Delta\theta^L = \frac{1}{b}\left(\underset{m+n}{\text{SEMin}}\{\tilde{\delta}^{(i),L} - \delta^{(i),L}\}_{i=1}^{b} + \sum_{i=1}^{b}\delta^{(i),L}\right)$$

$$\Delta\theta^U = \frac{1}{b}\left(\underset{m+n}{\text{SEMax}}\{\tilde{\delta}^{(i),U} - \delta^{(i),U}\}_{i=1}^{b} + \sum_{i=1}^{b}\delta^{(i),U}\right)$$

where SEMin and SEMax takes the sum of the a smallest/largest elements in the set. $\overset{a}{\delta}^{(i),L}, \overset{a}{\delta}^{(i),U}$ are sound bounds that accounts for previous adversarial manipulations at each datapoint i, i.e.:

$$\delta^{(i),L} \preceq \delta \preceq \delta^{(i),U} \quad \forall\delta \in \left\{\nabla_\theta \mathcal{L}\left(f^{\theta'}(x^{(i)}), y^{(i)}\right) \mid \theta' \in [\theta^L, \theta^U]\right\}$$

and terms $\tilde{\delta}^{(i),L}, \tilde{\delta}^{(i),U}$ are bounds on the worse-case adversarial manipulation on the i-th datapoint in the current batch obtained by solving (3.7).

Practical Limitations of AGT

Since AGT assumes the worst-case poisoning at each parameter index, which is a loose approximation, bound propagation across successive iterations can result in looser bounds in practice. Consequently, AGT faces several limitations: obtaining non-vacuous guarantees often requires training with larger batch sizes and fewer epochs or layers than typical. Additionally, AGT tends to produce relatively weaker robustness guarantees for multi-class problems compared to regression or binary classification settings.

Chapter 4

Methodology

4.1 Multi-sample Analysis

Certification methods based on aggregation mechanisms [15, 16, 18] provide strong pointwise guarantees by certifying robustness for each individual test point through per-sample analysis. A powerful insight, however, is that the adversarial objectives in Equations 3.3 and 3.4 are defined across the entire test set and are computed after training. This implies that the adversary must commit to which models and how much to poison each model prior to evaluation. In contrast, per-sample certification implicitly assumes that the adversary can choose poisoned models individually for each prediction in a post hoc manner. While this assumption enables more tractable analysis, it introduces looseness in the guarantees, as illustrated in Figure 4.1. Chen et al. [17] lays the ground work for multi-sample analysis by using Binary Integer Programming (BIP). However, the formulation only works for the most naive case (DPA). Techniques that yield stronger guarantees have been studied such as FA and ROE. Moreover, the BIP formulation cannot incorporate intrinsic robustness guarantees of individual ensemble members. In this work, we present a general formulation of the adversary's optimal attack as a Mixed-Integer Linear Program (MILP) that can cater for a wide-class of aggregation mechanisms and intrinsic robustness certification methods.

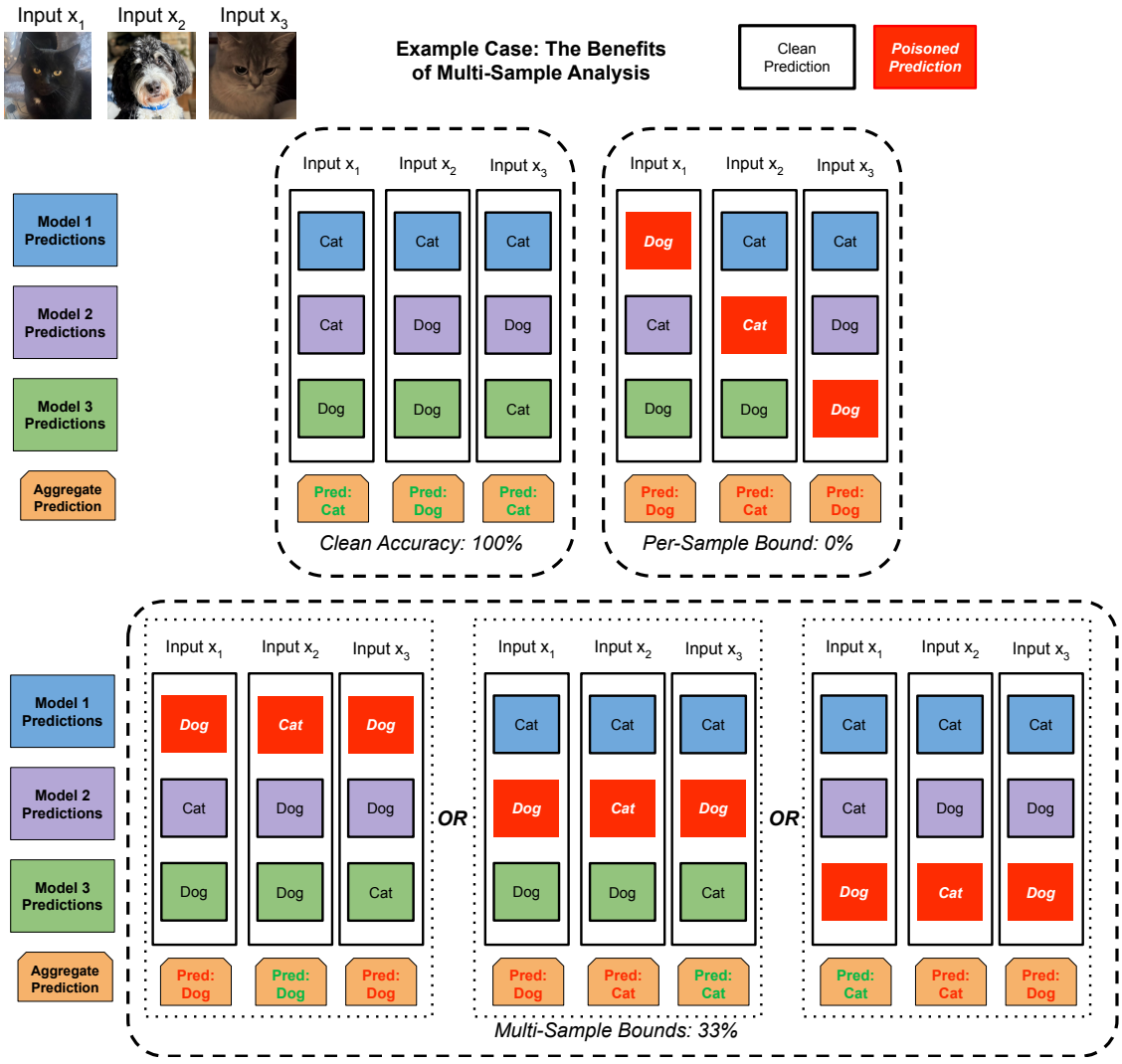


Figure 4.1: A toy example illustrating how multi-sample certification yields tighter robustness guarantees in a general voting-based certification framework. We consider a setting with $M = 3$ models, $N = 3$ test images, where the adversary can poison $K = 1$ model. The top left box shows the clean predictions with 100% clean accuracy. In the top right box, we show that per-sample certification allows the adversary to poison Models 1, 2 and 3 to change the prediction for x_1, x_2 and x_3 individually, resulting in 0% certified accuracy. Finally, in the bottom box, we show that, in multi-sample analysis, the adversary can only poison $K = 1$ of the models but **not all three models**. Poisoning any of the three models cannot change the ensemble predictions for all three images — the true lower bound is 33%, a substantial improvement over per-sample analysis.

4.2 Optimal Certification via MILP

The key insight in the multi-sample setting is that the adversary must commit to a global poisoning strategy, which means deciding in advance which models to poison and by how much. By computing the optimal adversarial strategy, one can certify a tight worst-case bound. We model this adversarial decision using a vector a, which we refer to as the *adversarial allocation vector*. When there are M models and the adversary is allowed to poison K data points in total, the allocation vector satisfies $a \in \mathbb{N}_0^M$ with the constraint $\sum_{i=1}^{M} a_i = K$.

For each input $x^{(i)} \in \mathbb{R}^n$ in a given test set, we define the *aggregation margin function* $G(x^{(i)}) : \mathbb{R}^n \to \mathbb{N}_0$, which represents the number of model votes that can be flipped while still preserving the correct ensemble prediction. This function depends heavily on the specific aggregation strategy used (discussed in detail in Section 4.2.1). When the ensemble's original prediction is incorrect, i.e., $\hat{y}^{(i)} \neq y^{(i)}$, we set $G(x^{(i)}) = -1$ to indicate that the misclassification results from inherent model error rather than adversarial influence. Additionally, we introduce the function $R_j(x^{(i)}) : \mathbb{R}^n \to \mathbb{N}_0$, which quantifies the *intrinsic robustness* of the j-th model at input $x^{(i)}$ — a component that, to the best of our knowledge, is absent from existing aggregation-based defenses.

We are now prepared to formalise this setting as an optimisation problem. For a given testset $\mathcal{D}_{\text{test}} = \{(x^{(i)}, y^{(i)})\}_{i=1}^{N}$, we denote the original ensemble prediction as $\hat{y}^{(i)} = f_{\text{agg}}(x^{(i)})$. The worst-case error for a model trained on a poisoned dataset with a fixed budget K is formulated by:

$$\max_{a \in \mathbb{N}_0^M} \frac{1}{N} \sum_{i=1}^{N} \mathbb{1} \left[\underbrace{G(x^{(i)})}_{\substack{\text{Aggregation} \\ \text{Margin}}} < \sum_{j=1}^{M} \mathbb{1}\{a_j > \underbrace{R_j(x^{(i)})}_{\substack{\text{Intrinsic} \\ \text{Robustness}}}\} \right] \quad s.t. \quad \sum_{j}^{M} a_j = K \tag{4.1}$$

Breaking down the optimisation problem, a sample $x^{(i)}$ is considered *not certified* if the number of flipped predictions exceeds its aggregation margin $G(x^{(i)})$, as indicated by the outer indicator function. Within the inner indicator, the prediction of the j-th model is considered flipped if the number of poisoned points in its corresponding partition, a_j, exceeds the model's intrinsic robustness $R_j(x^{(i)})$.

Solving this optimisation problem can be done by encoding it into a mixed integer linear program (MILP) by evaluating the G and R functions and storing their results in a vector and matrix respectively, with each value incremented by 1. We provide the explicit encoding that can be readily translated into open source [63] or

proprietary solvers [64]:

$$\text{maximise} \quad \frac{1}{N}\sum_{i=1}^{N} e_i \quad \text{subject to} \quad \begin{aligned} &\sum_{j=1}^{M} a_j = K \\ &\textstyle\sum_{j=1}^{M} b_{j,i} \geq G_i \cdot e_i && \forall i \in [N] \\ &a_j - R_{j,i} \geq -M_{\text{big}}(1 - b_{j,i}) && \forall i \in [N],\, j \in [M] \\ &b_{j,i} \in \{0,1\} && \forall i \in [N],\, j \in [M] \\ &e_i \in \{0,1\} && \forall i \in [N] \\ &a_j \in \mathbb{N}_0 && \forall j \in [M] \end{aligned} \tag{4.2}$$

where $b_{j,i}$ and e_i are binary decision variables related to the inner and outer indicator constraints of (4.1), respectively, and M_{big} is any sufficiently large constant. We highlight that the above formulation does not specify the exact form of the aggregation margin function, G, or the known intrinsic robustness R. In the subsequent sections we detail the various approaches to aggregation margin functions (Section 4.2.1) and intrinsic robustness (Section 4.2.2) that allow the above program to compute an optimal certificate for almost any combination of methods in the literature.

4.2.1 Aggregation Functions

In this section, we define the specific formulations of $G(x^{(i)})$ for different aggregation functions.

1. **Deep Partition Aggregation (DPA)**
   As outlined in Section 3.2.1, a prediction for input x is certified robust if an adversary is limited to modifying fewer than half the vote gap between the top two classes, with ties broken arbitrarily. In our framework, this approach is captured by the following formulation:

$$G_{\text{DPA}}\left(x^{(i)}\right) := \begin{cases} \left\lfloor \left(v_{\hat{y}^{(i)}}(x^{(i)}) - \max\limits_{c' \neq \hat{y}^{(i)}} \left(v_{c'}(x^{(i)}) + \mathbb{1}\{c' < c\} \right) \right) / 2 \right\rfloor & \text{if } \hat{y}^{(i)} = y^{(i)} \\ -1 & \text{otherwise} \end{cases}$$

   where $v_c(x^{(i)}) = |\{f_j^{\theta}(x^{(i)}) = c\}_{j=1}^{M}|$ is the number of votes received by a given class $c \in \mathcal{Y}$. The first two terms in the numerator express the margin between the top class and the second highest class, and the third term simply breaks ties by adding 1 if $c' < c$.

2. **DPA***
   The formulation for DPA* is identical to that of DPA, since only the training process of DPA* differs from DPA and not the voting mechanism.

3. **Run-Off Election (ROE)**
   As discussed in Section 3.2.1, ROE is a two-stage election process. To quantify the ensemble's robustness under this scheme, we define a two-stage aggregation margin function $G_{\text{ROE}}(x^{(i)})$ following the ROE certificate provided for DPA

in Rezaei et al. [18]:

$$G_{\text{ROE}}(x^{(i)}) := \begin{cases} \min\left(\text{Cert}_{\text{R1}}(x^{(i)}),\ \text{Cert}_{\text{R2}}(x^{(i)})\right) - 1 & \text{if } \hat{y}^{(i)} = y^{(i)} \\ -1 & \text{otherwise} \end{cases}$$

We now define $\text{Cert}_{\text{R1}}(x^{(i)})$ and $\text{Cert}_{\text{R2}}(x^{(i)})$.

Let $c_{pred} = \hat{y}^{(i)}$ be the nominal prediction of the ensemble for $x^{(i)}$, the Round 1 certificate $\text{Cert}_{\text{R1}}(x^{(i)})$ is the number of votes that needs be flipped such that c_{pred} does not proceed to Round 2:

$$\text{Cert}_{\text{R1}}(x^{(i)}) := \min_{c_1, c_2 \in \mathcal{Y} \setminus c_{\text{pred}}} \text{Cert}_{\text{V2}}(x^{(i)}, c_{\text{pred}}, c_1, c_2)$$

Here, we obtain Cert_{V2} recursively using dynamic programming. Intuitively, the adversary needs to choose its optimal action to change its vote from c_{pred} to either c_1 or c_2. We denote the gap between c_{pred} and c_1, c_2 as gap_1 and gap_2. When the $\text{gap}_1, \text{gap}_2 > 1$, the adversary should optimally decide to reduce one of the gaps and continue in a similar fashion. Finally, when one of the gaps is ≤ 1, the adversary should change $\left\lceil \frac{\max(\text{gap}_1, \text{gap}_2)}{2} \right\rceil$ votes from c to class with the larger remaining gap. Formally, we populate a dynamic programming matrix of size $(M+2)^2$ where if $\min(i, j) \geq 2$ we set:

$$\text{dp}[i, j] = 1 + \min\{\text{dp}[i-1, j-2], \text{dp}[i-2, j-1]\}$$

and when $\min[i, j] \leq 1$, we set $\text{dp}[i, j] := \left\lceil \frac{\max(i,j)}{2} \right\rceil$. As such, we get:

$$\text{Cert}_{\text{V2}}(x^{(i)}, c_{pred}, c_1, c_2) = \text{dp}[\text{gap}_1, \text{gap}_2]$$

where $\text{gap}_i := v_c(x^{(i)}) - v_{c_i}(x^{(i)}) - \mathbb{1}[c_i < c_{pred}]$.

Let c be the final predicted class after poisoning, then the second-round certificate, $\text{Cert}_{\text{R2}}(x^{(i)})$ is defined as:

$$\text{Cert}_{\text{R2}}(x^{(i)}) := \min_{c \neq c_{\text{pred}}} \max\{\text{Cert}_{\text{V1}}(x^{(i)}, c_{\text{sec}}, c), \text{Cert}^{\text{logits}}_{\text{V1}}(x^{(i)}, c_{\text{pred}}, c)\}$$

where $\text{Cert}_{\text{V1}}(x^{(i)}, c_{\text{sec}}, c)$ is the number of votes needed to be flipped for c to defeat c_{sec}:

$$\text{Cert}_{\text{V1}}(x^{(i)}, c_{\text{sec}}, c) := \left\lceil \frac{v_{c_{\text{sec}}}(x^{(i)}) - v_c(x^{(i)}) - \mathbb{1}[c' < c]}{2} \right\rceil$$

and $\text{Cert}^{\text{logits}}_{\text{V1}}(x^{(i)}, c_{\text{pred}}, c)$ lower bounds the number of classifiers that needs to be poisoned in Round 2 using the softmax scores $s_j(x^{(i)}, c)$:

$$\text{Cert}^{\text{logits}}_{\text{V1}}\left(x^{(i)}\right) := \left\lceil \frac{1}{2} \left(\sum_{j=1}^{M} \mathbb{1}\left\{ f_j(x^{(i)}) \in \{c_{\text{pred}}, c\} \right\} \cdot \mathbb{1}\left\{ s_j(x^{(i)}, c_{\text{pred}}) > s_j(x^{(i)}, c) \right\} \right) \right\rceil$$

4.2.2 Intrinsic Robustness

In this section, we outline the different methods that can be applied to improve the intrinsic robustness, i.e. the value $R_j(x^{(i)})$ in our formulation 4.1. Notably, our formulation allows any of the aggregation functions to be used in conjunction with the intrinsic robustness techniques discussed below.

1. **Finite Aggregation**
   To account for the overlapping data in each shard a classifier is trained on, we adapt the adversarial allocation vector a to cater for the spreading mechanism. We now define the effective adversarial allocation vector $\tilde{a}_j := \sum_{t \in h^{-1}_{\text{spread}}(j)} a_t$, where $h^{-1}_{\text{spread}}(j)$ is the set of partitions that are spread to model j. The condition for model j's prediction to change becomes $\mathbb{1}[\tilde{a}_j > R_j(x^{(i)})]$, and the rest of the optimisation remains unchanged. The worst-case error for an ensemble then becomes:

$$\max_{a \in \mathbb{N}_0^M} \frac{1}{N} \sum_{i=1}^{N} \mathbb{1} \left[\underbrace{G(x^{(i)})}_{\substack{\text{Aggregation} \\ \text{Margin}}} < \sum_{j=1}^{M} \mathbb{1}\{\tilde{a}_j > \underbrace{R_j(x^{(i)})}_{\substack{\text{Intrinsic} \\ \text{Robustness}}}\} \right] \quad s.t. \quad \sum_{j}^{M} \tilde{a}_j = K$$

2. **Abstract Gradient Training (AGT)**
   As outlined in Section 3.2.2, AGT certifies robustness by computing parameter bounds under a perturbation budget K. In our formulation, this translates to setting $R_j(x^{(i)}) = K$ if $x^{(i)}$ is K is the maximum number of points that the adversary can poison where AGT certifies robustness. We set $R_j(x^{(i)}) = 0$ for cases where there is no known intrinsic robustness, which collapses to the naive DPA case.

3. **Other Intrinsic Robustness Methods**
   As highlighted above, various intrinsic robustnes methods can be readily incorporated into our formulation by adapting $R_j(x^{(i)})$. Notably, **Randomised Smoothing** [9] certifies robustness against label flipping poisoning by averaging predictions over learners trained on perturbations of the original training dataset. Recent works have also explored providing robustness certificates for **differentially private learners** [12].

Chapter 5

Experiments: Multi-Sample Analysis

5.1 Hypotheses

As introduced in Section 4.1, we hypothesise that performing multi-sample analysis will provide, both from a theoretical standpoint and through empirical evidence, significantly tighter certification bounds compared to pointwise (per-sample) certification methods. In this chapter, we empirically evaluate the effectiveness of our proposed Mixed-Integer Linear Programming (MILP) framework within this multi-sample context. Our experimental design closely follows the setups and benchmarks established in prior foundational works [15, 16, 18] to ensure a fair and direct comparison.

We compare the **certified accuracy** obtained from our multi-sample analysis using test-time batches of size B, with the **certified fraction** used in prior per-sample methods [15, 16, 18]. Both metrics quantify the proportion of test points that are:

1. Nominally predicted correctly when no data is poisoned, and
2. Certified (i.e. predictions remain unchanged under perturbation level K)

but they differ in how certification is computed — multi-sample versus per-sample analysis, respectively. Importantly, our framework generalises previous approaches: when the batch size is set to $B = 1$, our method reduces exactly to the pointwise certification approach used in previous studies. This enables us to empirically validate the advantage of multi-sample analysis by observing improvements as B increases.

However, while the theoretical benefits of multi-sample certification are promising, practical challenges remain. A key concern is the scalability of the MILP formulation as the ensemble size and test dataset size grow, since the complexity and computational demand of solving MILPs can increase sharply. Predicting the runtime required for MILP solvers remains an active area of research [65]. While we do not explicitly investigate how batch size impacts solver runtime, we remain conscious of its computational implications when designing our experiments. In particular, we select

batch sizes that balance certification tightness with solver efficiency, ensuring that our experimental setup remains computationally tractable while still demonstrating the empirical benefits of multi-sample certification.

5.2 Settings

We adopt the same experimental settings and model architectures as those used in prior work [15, 16, 18]. Specifically, we employ the Network-In-Network (NiN) architecture [66], trained using the hyperparameter configuration proposed by Gidaris et al. [67]. We evaluate our method on three benchmark datasets: MNIST [19], CIFAR-10 [20], and GTSRB [21], each on a test set of 10,000 images. The choice of 10,000 samples enables straightforward selection of evenly divisible batch sizes, which is important for systematic experimentation across different batch configurations. In all experiments, per-sample and multi-sample analyses are conducted on a fixed test set within each setting, ensuring a fair and controlled comparison between certification methods. While the specific evaluation set may differ across settings (e.g., across perturbation budget), maintaining consistency within each setting eliminates variability due to test set selection. This allows us to systematically investigate the effects of batch configurations and certification methods under stable and reproducible conditions.

5.2.1 Datasets

1. **MNIST**
   The **MNIST** (Modified National Institute of Standards and Technology database) [19] is a widely used benchmark in machine learning and computer vision, consisting of 70,000 grayscale images of handwritten digits from 0 to 9. Each image is 28×28 pixels in size, resulting in a 784-dimensional input vector. The dataset is divided into 60,000 training samples and 10,000 test samples.

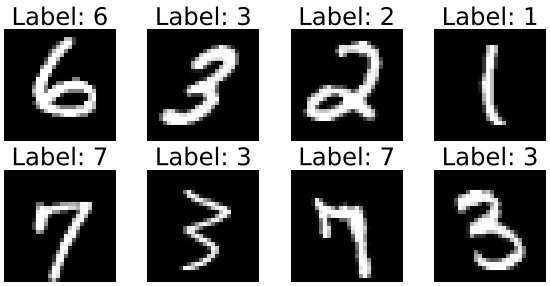


Figure 5.1: Example images from the MNIST dataset.

2. **CIFAR-10**
   CIFAR-10 [20] is a widely used image classification benchmark consisting of 60,000 color images of size 32×32 pixels, divided evenly across 10 object classes such as airplanes, cars, and animals. The dataset is split into 50,000 training images and 10,000 test images, and is commonly used to evaluate

supervised and unsupervised learning algorithms in computer vision.

Figure 5.2: Example images from the CIFAR-10 dataset.

3. **GTSRB**
   The German Traffic Sign Recognition Benchmark (GTSRB) is a dataset for multi-class classification of traffic signs, containing over 50,000 real-world images of 43 traffic sign classes. The images vary in size, lighting, and background conditions, reflecting challenging real-world driving scenarios.

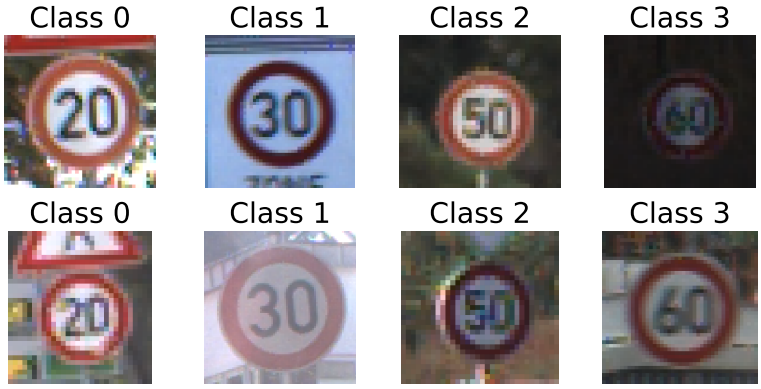


Figure 5.3: Example images from the GTSRB dataset.

5.2.2 Model Architecture

Network-in-Network (NiN)

The Network-in-Network (NiN) architecture [66] enhances traditional convolutional neural networks by replacing simple linear convolutional filters with multilayer perceptrons (MLPs) within each convolutional layer. This design enables more complex and abstract feature extraction at a local level, improving the model's representational power without significantly increasing computational cost. The NiN architecture has been shown to improve performance in various visual recognition tasks by effectively capturing nonlinear feature interactions.

Hyperparameters

Gidaris et al. [67] introduced a self-supervised learning framework that trains convolutional neural networks to predict image rotations, enabling the model to learn robust and invariant feature representations without labeled data. They employ the Network-in-Network (NiN) architecture with carefully designed augmentations, such as rotation prediction, random cropping, and horizontal flipping, to promote generalisable representations. Although Gidaris et al. primarily focus on unsupervised representation learning through rotation prediction, the hyperparameters they employ are largely applicable and effective for supervised learning tasks as well. In our experiments, we follow Levine et al. [15] and adopt the training parameters directly from Gidaris et al. [67], with minor modifications to data augmentation: specifically, we omit random horizontal flips for MNIST and GTSRB due to their semantic sensitivity to orientation, while applying the full augmentation pipeline for CIFAR-10.

5.2.3 Implementation Details

We use Gurobi 12.0.1 [64] to solve the MILP described in 4.2. The solver either returns the optimal objective value or the best (sound) bound on the objective function found within a predefined time limit. We select batch sizes for each setting such that the optimisation problem solves to optimality–or near-optimality–within a time limit of 1800 seconds. The selected batch sizes for each setting and their corresponding average run-times per batch are summarised in Table 5.1. All experiments are conducted on GPU (two NVIDIA L40).

Table 5.1: Batch sizes and average runtimes per batch for each setting. *Values marked $\star$ are evaluated on a subset of 100 test data points.*

dataset	k	method	d	batch sizes (avg time per batch)			
CIFAR-10	50			$K \leq 3$	$K \leq 5$	$K \leq 10$	$K \leq 20$
		DPA		1000 (694s)	1000 (1439s)	400 (547s)	200 (285s)
		DPA+ROE		1000 (547s)	1000 (1155s)	400 (143s)	200 (512s)
		DPA*+ROE	16	1000 (120s)	1000 (268s)	500 (95s)	200 (110s)
			32	1000 (117s)	1000 (303s)	500 (82s)	200 (93s)
		FA	16	100 (96s)	100 (295s)	40 (356s)	10 (1441s) $\star$
		FA+ROE		100 (55s)	100 (228s)	40 (263s)	10 (1104s) $\star$
GTSRB	50			$K \leq 5$	$K \leq 10$	$K \leq 15$	$K \leq 20$
		DPA		2000 (315s)	1000 (1498s)	400 (79s)	250 (32s)
		DPA+ROE		2000 (198s)	1000 (443s)	400 (54s)	250 (43s)
		DPA*+ROE	16	2000 (158s)	1000 (542s)	500 (217s)	250 (79s)
		FA		100 (184s)	50 (212s)	25 (1046s) $\star$	10 (1402s) $\star$
		FA+ROE		100 (123s)	50 (116s)	25 (730s) $\star$	10 (1624s) $\star$
MNIST	1200			$K \leq 100$	$K \leq 200$	$K \leq 300$	$K \leq 500$
		DPA		200 (187s)	200 (138s)	200 (694s)	100 (472s)
		DPA+ROE		200 (187s)	200 (129s)	200 (468s)	100 (836s)
		DPA*+ROE	16	200 (91s)	200 (173s)	200 (255s)	100 (354s)

5.3 Results

5.3.1 Beating State of the Art

We demonstrate on all tested benchmarks that our approach substantially improves state-of-the-art analysis. In Figure 5.4 we plot the best improvement that different approaches offer. We start with the standard per-sample analysis with DPA in blue. Using better per-sample analysis one is able to typically improve this bound by 3% (visualised in green). Using the simplest multi-sample version of DPA (visualised in yellow), we find that one is *typically but not always* able to beat the best aggregation methods. Finally, considering the best bounds on the setting using our method we find that in every case we are able to beat state of the art certified accuracy by 3% – doubling the improvement of prior approaches.

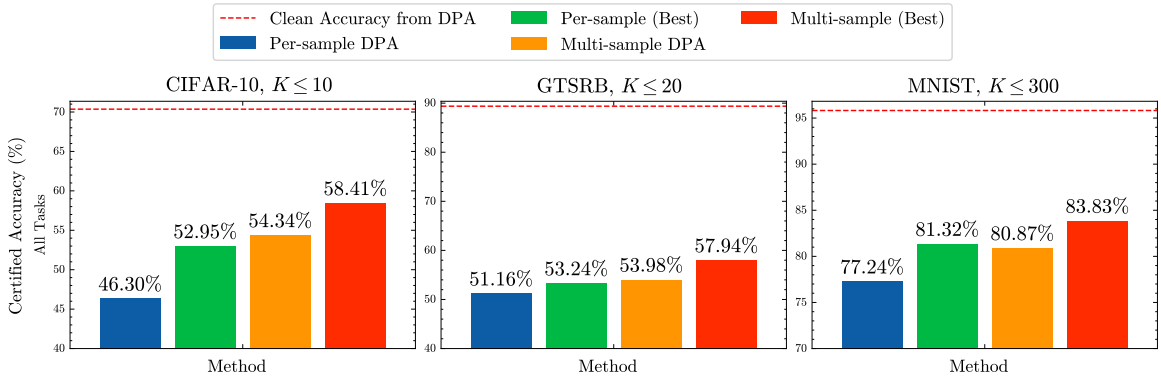


Figure 5.4: On CIFAR-10 ($K \leq 10, M = 50$), our best multi-sample certification (DPA*+ROE) outperforms the best per-sample method (FA) by 5.46% and naive multi-sample DPA by 4.07%. On GTSRB ($K \leq 20, M = 50$), it improves over the best per-sample method by 4.70% and naive multi-sample DPA by 3.96%. On MNIST ($K \leq 300, M = 1200$), it achieves gains of 2.51% and 2.96%, respectively.

5.3.2 Systematic Certification Improvement

While previous results indicate substantial improvements that our approach can offer, in Tables 5.2, 5.3, and 5.4 we perform a systematic exploration of the improvements of our certification approach. In particular, we apply 5 different aggregation functions: DPA, DPA+ROE, DPA*+ROE, FA and FA+ROE on both CIFAR-10 and GTSRB with $M = 50$ as shown in Tables 5.2 and 5.3 and compute all but the FA version for MNIST (Table 5.4) due to the infeasible computational complexity. *In these tables, we find that across every tested combination of methods, our framework improves the certified bounds*. In the setting that exhibits the largest improvement (CIFAR-10, $K \leq 20$), highlighted in yellow in Table 5.2, our multi-sample analysis with the optimal aggregation method outperforms naive multi-sample DPA by **9.91%**, and improves upon the naive DPA per-sample bounds by **14.44%**.

In general, we find DPA and DPA+ROE exhibit the most significant gains. This is

largely attributed to the lower complexity of the associated MILP, which enables certificate computation over larger batch sizes and thus yields stronger robustness guarantees. This enables us to achieve stronger robustness guarantees using simpler aggregation methods such as DPA and DPA+ROE, compared to more computationally intensive methods like FA and FA+ROE.

Table 5.2: A comparison of certified fraction (per-sample analysis) and certified accuracy (multi-sample analysis) across different settings evaluated on CIFAR-10 with $M = 50$. *Values marked $\star$ are evaluated on a subset of 100 test data points.*
Cells highlighted in yellow indicate the largest improvement achieved with multi-sample analysis. Specifically, for $K \leq 20$, the best aggregation mechanism using our multi-sample bounds (DPA*+ROE) gives a certified accuracy of 33.90%, outperforming the naive DPA multi-sample and per-sample case by 9.91% and 14.44% respectively.

method	d	certified fraction (pointwise accuracy) $\rightarrow$ certified accuracy (batch accuracy)			
		$K \leq 3$	$K \leq 5$	$K \leq 10$	$K \leq 20$
DPA		63.41% $\rightarrow$ 66.38% (+2.97%)	58.71% $\rightarrow$ 63.79% (+5.08%)	46.30% $\rightarrow$ 54.34% (+8.04%)	19.46% $\rightarrow$ 23.99% (+4.53%)
DPA+ROE		64.06% $\rightarrow$ 66.78% (+2.72%)	59.82% $\rightarrow$ 64.36% (+4.54%)	48.99% $\rightarrow$ 56.24% (+7.25%)	21.36% $\rightarrow$ 29.05% (+7.69%)
DPA*+ROE	16	65.08% $\rightarrow$ 66.88% (+1.80%)	61.67% $\rightarrow$ 64.91% (+3.24%)	52.79% $\rightarrow$ 58.41% (+5.62%)	27.43% $\rightarrow$ 33.90% (+6.47%)
	32	65.27% $\rightarrow$ 67.05% (+1.78%)	61.72% $\rightarrow$ 64.97% (+3.25%)	52.95% $\rightarrow$ 58.37% (+5.42%)	27.56% $\rightarrow$ 33.81% (+6.25%)
FA	16	64.56% $\rightarrow$ 65.94% (+1.38%)	60.00% $\rightarrow$ 62.90% (+2.90%)	48.16% $\rightarrow$ 52.86% (+4.70%)	17.00% $\rightarrow$ 19.00% (+2.00%) $\star$
FA+ROE		65.37% $\rightarrow$ 66.56% (+1.19%)	61.37% $\rightarrow$ 64.05% (+2.68%)	50.74% $\rightarrow$ 55.61% (+4.87%)	21.50% $\rightarrow$ 27.50% (+6.00%) $\star$

Table 5.3: A comparison of certified fraction (per-sample analysis) and certified accuracy (multi-sample analysis) across different settings evaluated on GTSRB with $M = 50$. *Values marked $\star$ are evaluated on a subset of 100 test data points.*

method	d	certified fraction (pointwise accuracy) $\rightarrow$ certified accuracy (batch accuracy)			
		$K \leq 5$	$K \leq 10$	$K \leq 15$	$K \leq 20$
DPA		82.62% $\rightarrow$ 85.19% (+2.57%)	75.40% $\rightarrow$ 79.43% (+4.03%)	65.65% $\rightarrow$ 70.13% (+4.48%)	51.16% $\rightarrow$ 53.98% (+2.82%)
DPA+ROE		83.03% $\rightarrow$ 84.93% (+1.90%)	76.36% $\rightarrow$ 80.10% (+3.74%)	67.85% $\rightarrow$ 72.38% (+4.53%)	53.24% $\rightarrow$ 57.94% (+4.70%)
DPA*+ROE	16	83.03% $\rightarrow$ 84.93% (+1.90%)	76.36% $\rightarrow$ 80.07% (+3.71%)	67.85% $\rightarrow$ 72.64% (+4.79%)	53.24% $\rightarrow$ 57.94% (+4.70%)
FA		82.27% $\rightarrow$ 84.00% (+1.73%)	74.42% $\rightarrow$ 77.36% (+2.94%)	62.00% $\rightarrow$ 68.00% (+6.00%)$\star$	43.00% $\rightarrow$ 46.50% (+3.50%)$\star$
FA+ROE		82.38% $\rightarrow$ 83.83% (+1.45%)	75.40% $\rightarrow$ 78.65% (+3.25%)	64.00% $\rightarrow$ 69.50% (+5.50%)$\star$	47.50% $\rightarrow$ 50.5% (+3.00%)$\star$

Table 5.4: A comparison of certified fraction (per-sample analysis) and certified accuracy (multi-sample analysis) across different settings evaluated on MNIST with $M = 1200$.

method	d	certified fraction (pointwise accuracy) $\rightarrow$ certified accuracy (batch accuracy)			
		$K \leq 100$	$K \leq 200$	$K \leq 300$	$K \leq 500$
DPA		92.09% $\rightarrow$ 92.30% (+0.21%)	86.35% $\rightarrow$ 87.79% (+1.44%)	77.24% $\rightarrow$ 80.87% (+3.63%)	32.44% $\rightarrow$ 33.89% (+1.45%)
DPA+ROE		92.38% $\rightarrow$ 92.48% (+0.10%)	87.45% $\rightarrow$ 88.61% (+1.16%)	79.50% $\rightarrow$ 82.97% (+3.47%)	37.03% $\rightarrow$ 41.18% (+4.15%)
DPA*+ROE	16	92.45% $\rightarrow$ 92.49% (+0.04%)	88.33% $\rightarrow$ 89.09% (+0.76%)	81.32% $\rightarrow$ 83.83% (+2.51%)	43.87% $\rightarrow$ 46.64% (+2.77%)

5.3.3 Effect of Test Batch Size

As shown in Table 5.5, certified accuracy improves with larger test batch sizes. Intuitively, increasing the test batch size reduces the adversary's ability to tailor perturbations to individual test points, thereby enhancing the robustness of the certification.

Table 5.5: Investigate effect of batch size with different methods for CIFAR-10 for $M = 50$ and $K \leq 10$.

dataset	M	method	d	certified accuracy with batchsize (B)		
CIFAR-10	50			$B = 1$	$B = 100$	$B = 400$
		DPA		46.30%	50.50%	54.34%
		DPA+ROE		48.99%	52.80%	56.24%
				$B = 1$	$B = 250$	$B = 500$
		DPA*+ROE	16	52.79%	57.09%	58.41%
			32	52.95%	57.00%	58.37%
				$B = 1$	$B = 20$	$B = 40$
		FA	16	48.16%	51.95%	52.86%
		FA+ROE		50.74%	54.88%	55.61%

5.3.4 Scalability of MILP

The MILP formulation remains efficiently solvable for ensemble sizes up to around 1200; however, it scales poorly in scenarios involving larger configurations (e.g. 1200×16 ensembles), where the combinatorial complexity renders the approach computationally infeasible. The overall complexity grows with the perturbation budget K, the ensemble size and the batch size. As the ensemble size increases, maintaining computational feasibility often requires reducing the test batch size. As a result, the effectiveness of multi-sample certification at larger ensemble sizes is reduced, as evidenced by the smaller gains observed when applying multi-sample analysis to FA and FA+ROE. As the perturbation budget K increases, the number of possible adversarial allocation vectors a grows, further increasing the complexity of the optimisation problem. In our experiments, FA/FA+ROE with larger K values ($K \leq 10, 20$) become too computationally expensive to evaluate over the entire dataset under the resource limitations. Therefore, we compute results over a subset of 100 test data points, where each batch is solved either to or close to optimality. Despite this limitation, evaluating on 100 test points still yields improvements over the per-sample analysis.

5.3.5 Experimental Variance Analysis

In this section, we investigate the variance of certified accuracies across the batches shown in Tables 5.6, 5.7 and 5.8. The variance arises from two main factors:

1. Natural variance in vulnerability of points from the data distribution when using **small batch sizes**, which amplifies variability across runs. This is evident in the higher standard deviation of certified accuracies across batches in FA/FA+ROE runs, as shown in Tables 5.6 and 5.7, where smaller batch sizes were used.

2. Some batches exceeding the time limit due to variations in the optimisation

landscape when solving Equation (4.2) with Gurobi. As Gurobi's runtimes are highly sensitive to the problem structure, they can also be difficult to predict. A small number of batches exceeding the time limit suggests that, while care was taken during selection, runtime unpredictability remains due to the complexity of the optimisation process.

Table 5.6: Standard Deviation of certified accuracy (multi-sample analysis) across batches for CIFAR-10 with $M = 50$, corresponding to accuracies reported in table 5.2. *Values marked $\star$ are evaluated on a subset of 100 test data points.*

method	d	standard deviation			
		$K \leq 3$	$K \leq 5$	$K \leq 10$	$K \leq 20$
DPA		1.00%	1.17%	2.47%	2.97%
DPA+ROE		1.00%	1.09%	2.42%	3.15%
DPA*+ROE	16	1.01%	1.23%	1.56%	3.36%
	32	1.04%	1.23%	1.23%	3.4%
FA	16	4.56%	4.82%	8.04%	28.93% $\star$
FA+ROE		4.58%	4.74%	8.10%	20.93% $\star$

Table 5.7: Standard Deviation of certified accuracy (multi-sample analysis) across batches for GTSRB with $M = 50$, corresponding to accuracies reported in table 5.3. *Values marked $\star$ are evaluated on a subset of 100 test data points.*

method	d	standard deviation			
		$K \leq 5$	$K \leq 10$	$K \leq 15$	$K \leq 20$
DPA		0.61%	1.37%	1.76%	3.08%
DPA+ROE		0.49%	1.04%	1.61%	2.63%
DPA*+ROE	16	0.49%	1.05%	1.43%	2.65%
FA		3.38%	5.52%	8.12% $\star$	11.84%$\star$
FA+ROE		3.34%	5.53%	9.17%$\star$	13.96% $\star$

Table 5.8: Standard Deviation of certified accuracy (multi-sample analysis) across batches for MNIST with $M = 1200$, corresponding to accuracies reported in table 5.4.

method	d	standard deviation			
		$K \leq 100$	$K \leq 200$	$K \leq 300$	$K \leq 500$
DPA		3.82%	5.22%	6.93%	12.03%
DPA+ROE		3.73%	4.90%	6.29%	11.81%
DPA*+ROE	16	2.82%	4.74%	6.01%	12.08%

We note that the solution to our optimisation problem is deterministic when the base classifiers are fixed. To estimate experimental variance, we perform multiple runs of batch certification by training the base classifiers with different random seeds. We then compute the standard deviation across these runs for a single batch. As shown in Tables A.1, A.2 and A.3 (Appendix A), the observed variance is low.

5.3.6 Discussion

Our experimental results show that the MILP approach yields the most substantial improvement with smaller ensemble sizes. With smaller ensemble sizes, larger batch sizes can be used while keeping the MILP tractable, enabling more substantial gains in certified accuracy through multi-sample analysis.

Across all experimental settings, we find that state-of-the-art certification performance can be attributed to two key factors: (i) the strength of the aggregation mechanism itself, as reflected in higher values of $G(x^{(i)})$ and correspondingly higher certified fraction (i.e. pointwise certified accuracy), and (ii) the effectiveness of multi-sample analysis, which is strongly influenced by the chosen batch size.

Notably, DPA*+ROE consistently achieves the strongest improvements under multi-sample analysis across a wide range of experimental settings, outperforming other aggregation mechanisms and establishing a new state of the art in certified robustness against general data poisoning. Conveniently, the effectiveness of DPA*+ROE introduces no additional complexity for the MILP. Although DPA* has an ensemble size of $d \cdot M$, and its training time is d times that of DPA, its multi-sample certification remains efficient. This is because logits are averaged before voting, resulting in M votes at inference time. Consequently, the MILP scales with M and is not affected by d. As shown in Tables 5.2, 5.3 and 5.4, DPA*+ROE matches or outperforms FA and FA+ROE in per-sample certification. This demonstrates that DPA* effectively benefits from its more expressive aggregation mechanism, improving certified accuracy without increasing certification complexity. This observation is particularly valuable, as it enables efficient certification under larger perturbation budgets K, which is important given that the complexity of the MILP formulation increases with K.

On the other hand, in settings where FA and FA+ROE outperform DPA and DPA+ROE in per-sample analysis, we observe that the multi-sample analysis of DPA and DPA+ROE yields better results than that of FA and FA+ROE. This suggests that the benefits from multi-sample analysis can outweigh the strength of the aggregation mechanism alone. This effect is particularly evident in experiments carried out on CIFAR-10 (Table 5.2). This finding highlights a promising direction for future work to design more powerful aggregation mechanisms that preserve MILP tractability, as demonstrated by the effectiveness of DPA*+ROE.

The experiments conducted so far focus on the impact of the aggregation function, $G(x^{(i)})$, on certified accuracy, without leveraging any intrinsic robustness information from the individual classifiers—that is, all $R_j(x^{(i)}) = 0$. In the next chapter, we extend this analysis by exploring how AGT [10] can be used to obtain intrinsic robustness information for each member classifier, with the aim of further improving certified accuracy.

Chapter 6

Experiments: Intrinsic Robustness

6.1 Hypotheses

Up to this point, our experiments have assumed that member classifiers possess no known intrinsic robustness, i.e., $R_j(x^{(i)}) = 0, \forall i, j$. Intuitively, by obtaining intrinsic robustness guarantees for each model, i.e. setting $R_j(x^{(i)}) > 0$, we can improve certification bounds immensely. When each classifier is individually certified, we no longer assume that poisoning any one partition can always alter the prediction of the corresponding classifier. Instead, for each test sample, the worst-case scenario now corresponds to the adversary targeting the weakest classifiers, which are those with the lowest intrinsic robustness.

As outlined in Section 4.2.2, there are various methods to achieve intrinsic robustness guarantees. In this chapter, we focus on using AGT [10] to provide intrinsic robustness guarantees. While AGT is theoretically expected to improve certification bounds, practical limitations must be acknowledged. As outlined in Section 3.2.2, a key limitation of AGT is that its certification bounds become looser as batch size decreases or model complexity increases. The practical limitations of AGT mean applying it in an ensemble setting or increasing ensemble size may not yield additional benefits. Moreover, obtaining tight bounds with AGT often requires using simpler models, which can lead to a reduction in accuracy. We note that this accuracy–robustness trade-off is also observed in other certification techniques such as DP-based certification methods [12]. By addressing this trade-off in the context of AGT, our research hope to offer insights that extend beyond AGT itself and can inform the design and improvement of other intrinsic certification approaches. To this end, we explore the idea of applying AGT to only a subset of classifiers in the ensemble. By doing so, we aim to retain the robustness benefits of AGT, while maintaining the overall model capacity and accuracy. We analyse how the proportion of AGT-trained classifiers affects the balance between nominal accuracy and certified robustness. To enable a fair exploration of this trade-off, we introduce two new metrics:

1. **Certified percentage**: The proportion of the test space where the ensemble is certifiably robust, regardless of its predicted label.
2. **Nominal accuracy**: The clean accuracy of the ensemble.

To examine these trade-offs in a controlled setting, we conduct our analysis on the halfmoons dataset, which allows for a clearer interpretation of the effects and limitations. As ongoing research explores more scalable variants of AGT, the insights from our analysis may help inform future work aiming to apply general certification methods in an ensemble setting.

6.2 Settings

6.2.1 Dataset

The half-moons dataset is a synthetic, two-dimensional dataset consisting of two interleaving half-circle shapes that create a nonlinear binary classification problem, visualised in Figure 6.1. In our experiments, we apply a cubic basis expansion to the input features prior to training to better capture the nonlinear decision boundary inherent in this dataset. In all the figures presented in this chapter, the black line indicates the decision boundary. Test points are shown in blue or red based on their class. The shaded blue and red regions mark areas where predictions are certified (prediction is invariant to perturbations), while the white regions correspond to uncertified areas.

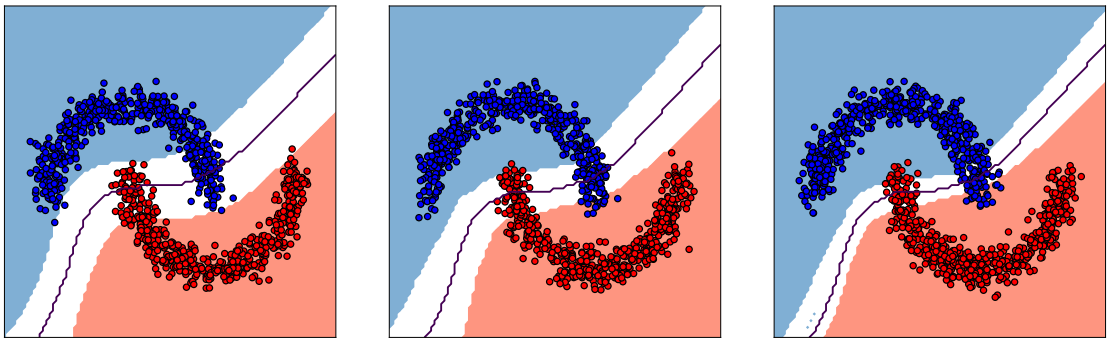
6.2.2 Implementation Details

To obtain intrinsic robustness guarantees, in our experiments, we train each member classifier with AGT, using the public repository of AGT [10], with values of k_poison up to K, incremented at intervals of s. AGT certifies whether a model remains robust under a given perturbation budget k_poison. This procedure allows us to determine the largest value of k_poison for which each member classifier remains certifiably robust. It is important to note that this approach increases training time by approximately a factor of $4 \cdot K/s$, as each k_poison value requires a separate training pass, and AGT itself introduces roughly a $4\times$ overhead in training time [10].

6.3 Results

6.3.1 Multi-sample analysis improves on AGT ensemble

Consistent with the findings in the previous chapter, Figure 6.1 illustrates that multi-sample bounds outperform per-sample bounds when intrinsic robustness is incorporated. When $M = 5$ and $K \leq 20$, the multi-sample analysis yields a 9.35% increase in certified percentage compared to the per-sample analysis. Compared to AGT without any aggregation mechanism ($M = 1$), per-sample analysis and multi-sample analysis with $M = 5$ increase the certified percentage by 4.26% and 13.51%, respectively.



(a) Naive AGT with $M = 1$ – 74.32%

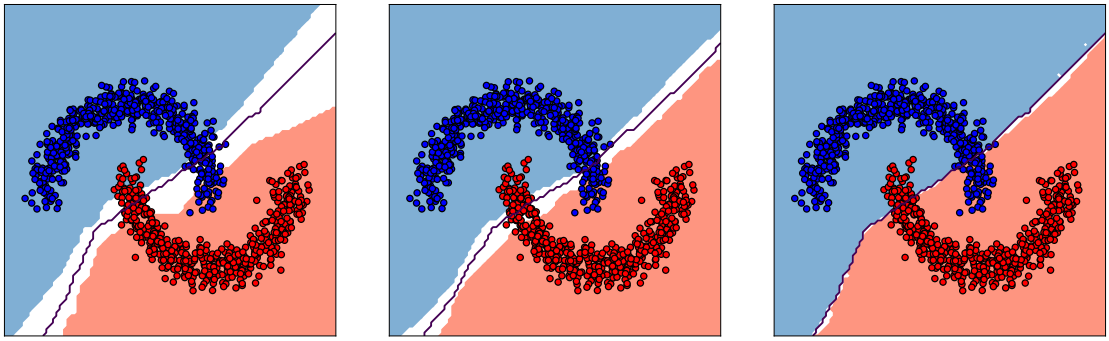
(b) Per-sample Analysis with AGT Bounds, $M = 5$ – 78.58%

(c) Multi-sample Analysis with AGT bounds, $M = 5$ – 87.93%

Figure 6.1: When intrinsic robustness is incorporated, multi-sample analysis yields tighter certification bounds than per-sample analysis for $M = 5$ and $K \leq 20$, and outperforms AGT without any aggregation mechanism ($M = 1$) when each ensemble member is trained on 10,000 data points.

6.3.2 Improvements with Intrinsic Robustness

To illustrate the potential gains from incorporating intrinsic robustness information, we first run DPA, FA and AGT with multi-sample bounds, with each classifier consisting of a simple MLP with a single linear layer. In Figure 6.2 we plot the result of running three methods on the standard half-moons dataset. We observe that multi-sample analysis of naive DPA leads to a certified percentage of 85%. Multi-sample analysis with an improved aggregation method (FA) improves this significantly to 92% highlighting the effectiveness of a better aggregation function. Finally, we run multi-sample analysis with individual models certified using AGT. We find that if models in the ensemble possess intrinsic robustness (Figure 6.2(c)), this leads to an even more substantial increase in certified percentage reaching 99%.



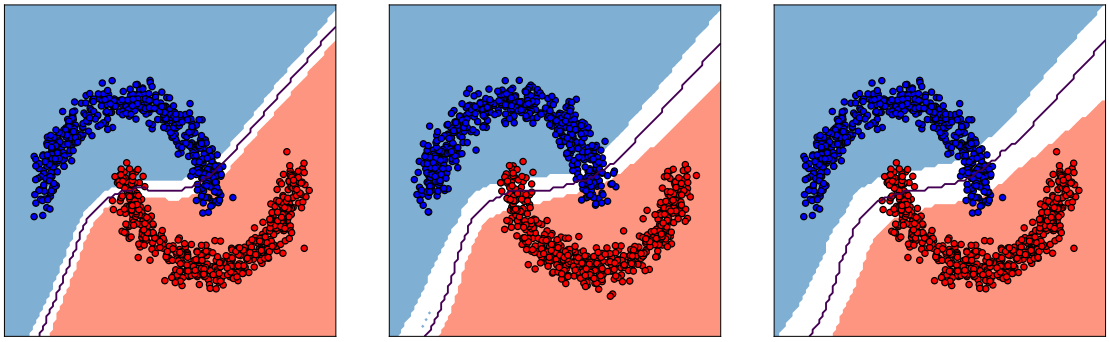
(a) DPA, 85.11% (b) FA, 92.92% (c) AGT, 98.96%

Figure 6.2: Figures (a), (b), and (c) show results for DPA, FA, and AGT respectively, all evaluated using multi-sample bounds. DPA and FA are run without intrinsic robustness, while in Figure (c), each ensemble member is trained with AGT. All methods use $K \leq 5$, $M = 20$, and $d = 2$ for FA. We observe that incorporating intrinsic robustness into individual models leads to substantial improvements in certified accuracy.

6.3.3 Impact of Ensemble Size M on Certification Bounds

To investigate the effect of ensemble size M on ensembles trained with AGT, we run experiments using MLPs with two linear layers as member classifiers, a total training dataset size of $50{,}000$, and a maximum perturbation budget of $K \leq 20$.

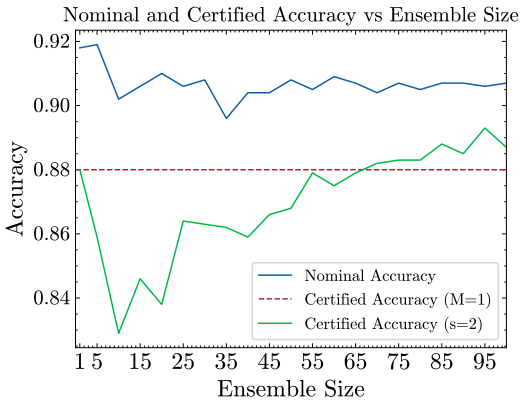
As shown in Figure 6.3, certification bounds degrade as ensemble size increases under a *fixed total dataset size*. Specifically, the certified percentage drops from 93% to 87% to 82% as M increases from 1 (no ensemble) to 5 and then to 10. This degradation is primarily due to the limitations of AGT. As the dataset is partitioned among more member classifiers, each classifier receives fewer samples, reducing its ability to obtain strong intrinsic robustness guarantees.



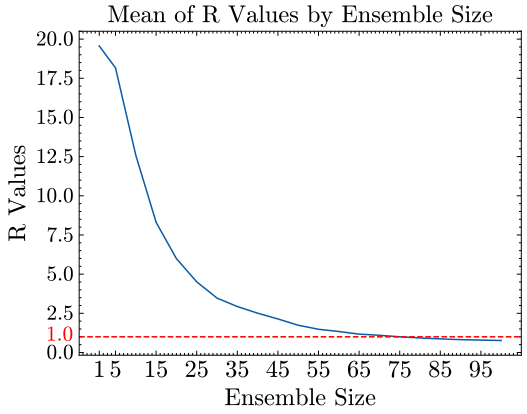
(a) $M = 1$, 93.41% (b) $M = 5$, 87.39% (c) $M = 10$, 82.06%

Figure 6.3: Certified percentage with multi-sample analysis decreases as ensemble size increases from $M = 1$ to $M = 10$ when the *total dataset size* is fixed at 50,000 with $K \leq 20, s = 5$.

To systematically explore the relationship between ensemble size M and both the accuracy and robustness of the model, we plot the nominal accuracy, certified accuracy, and the mean of $R_j(x^{(i)})$ values against ensemble size in Figure 6.4. Figure 6.4(a) shows that nominal accuracy decreases slightly but remains relatively stable as ensemble size increases.



(a) Nominal and Certified Accuracy (Multi-sample)



(b) Mean of $R_j(x^{(i)})$

Figure 6.4: Trends of Nominal Accuracy, Certified Accuracy and Mean of $R_j(x^{(i)})$ with ensemble size when *total dataset size* is fixed at 50,000 and with parameters $K \leq 20, M = 20, s = 2$.

Impact of Certified Accuracy with M

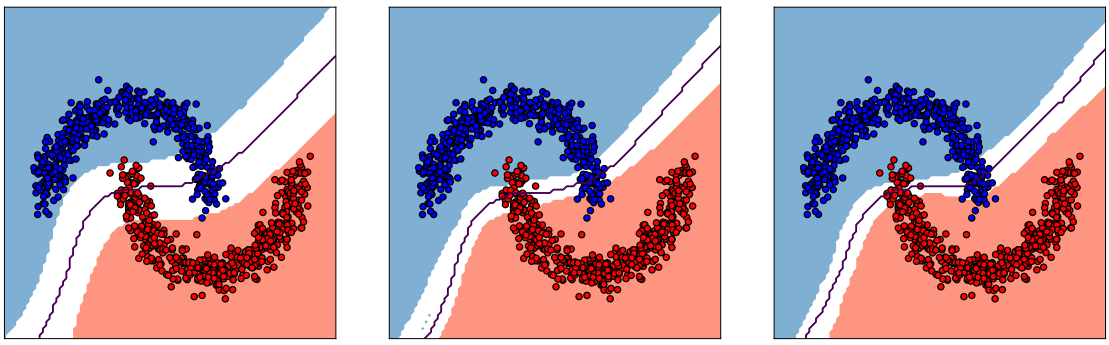
Interestingly, certified accuracy initially decreases for $M \leq 20$, then increases for $M > 20$. When $M \leq K = 20$, an adversary can poison all partitions in the worst case, so as intrinsic robustness weakens, certification suffers. However, for larger ensembles ($M > 20$), the adversary cannot poison all members, and thus attacks target the weakest classifiers. In this situation, gains from increasing aggregation margins $G(x^{(i)})$ outweigh the losses from diminishing intrinsic robustness, resulting in improved certified accuracy. It is noteworthy that in Figure 6.4(a), certified accuracy only exceeds that of the single model case ($M = 1$) when the ensemble size reaches as high as $M \geq 65$. This highlights the effectiveness of incorporating intrinsic robustness information: similar certification guarantees can be achieved with significantly smaller ensembles when intrinsic robustness is leveraged, compared to much larger ensembles without it.

Impact of $R_j(x^{(i)})$ Values with M

As shown in Figure 6.4(b), the mean of $R_j(x^{(i)})$ values decrease with increasing M, eventually plateauing around ≈ 1, indicating a weakening of the intrinsic robustness guarantees provided by AGT. For $M \geq 65$, $R_j(x^{(i)}) \approx 1$, suggesting that AGT provides little to no meaningful robustness information. Consequently, the certification bounds effectively reduce to those of a naive DPA method that lacks intrinsic robustness information.

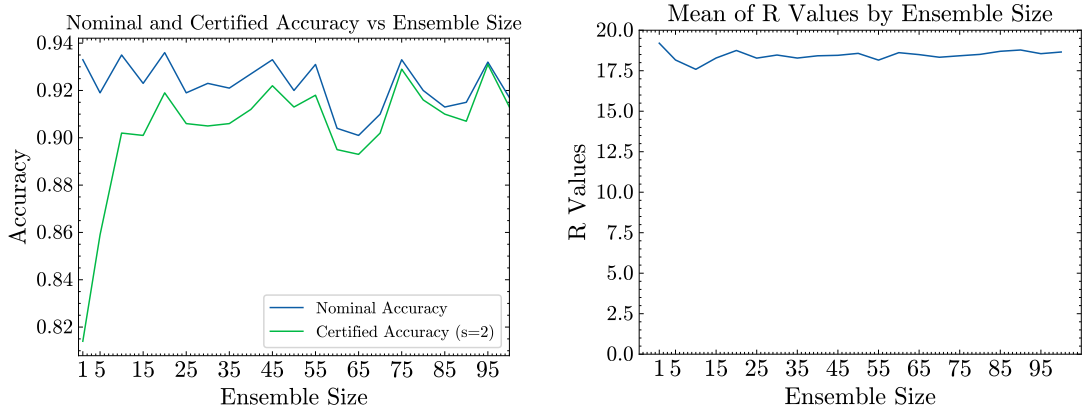
Intrinsic Robustness Under Fixed Dataset Size per Member

To isolate the effect of ensemble size and the assess the effectiveness of intrinsic robustness guarantees, we repeat the same experiments under the simplifying assumption that increasing the ensemble size does not reduce the number of samples per classifier. While this may not strictly hold in practice, it provides useful insights in a controlled setting. Figure 6.5 illustrates how certification bounds improve as ensemble size increases when the *dataset size per ensemble member* is fixed at 10,000. The certified percentage rises from 75% to 87% and then to 89% as M increases from 1 to 5 and 10, respectively. In contrast to the earlier setting where the *total dataset size* was fixed, here the amount of data available to each model does not diminish with increasing ensemble size. As shown in Figure 6.6(b), the values of $R_j(x^{(i)})$ remain relatively stable when ensemble size increases, since the **AGT bounds do not degrade** when each model continues to receive the same amount of training data. Consequently, **certified accuracy improves with ensemble size**, as visualised in Figure 6.6(a).



(a) $M = 1$, 75.32% (b) $M = 5$, 87.39% (c) $M = 10$, 88.75%

Figure 6.5: Certified percentage with multi-sample analysis increases as ensemble size increases from $M = 1$ to $M = 10$ when the *dataset size per member* is fixed at 10,000 with $K \leq 20, s = 5$.

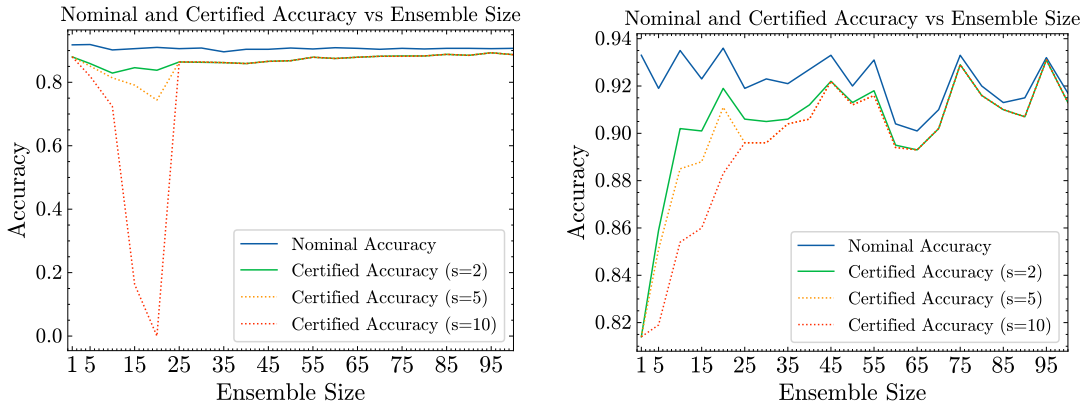


(a) Nominal and Certified Accuracy (Multi-sample) (b) Mean of $R_j(x^{(i)})$

Figure 6.6: Trends of Nominal Accuracy, Certified Accuracy and Mean of $R_j(x^{(i)})$ with ensemble size when *dataset size per member* is fixed at 10,000 and with parameters $K \leq 20, M = 20, s = 2$.

Effect of s

We repeat the experiments using different values of s, which determines the interval of K values at which AGT runs are performed to estimate intrinsic robustness (Section 6.2.2). Intuitively, smaller values of s yield finer-grained robustness guarantees. For instance, if a model is certifiable up to $K = 12$, setting $s = 5$ results in AGT runs at $K = [5, 10, 15, 20]$, so the model is certified only up to $K = 10$. In contrast, with $s = 2$, the AGT runs occur at $K = [2, 4, 6, 8, 10, 12, \dots]$, allowing certification up to $K = 12$ and thereby providing a stronger robustness guarantee than the $s = 5$ case. The choice of s reflects a trade-off between robustness and computational cost: smaller s values yield stronger intrinsic robustness guarantees, but require proportionally more training time.



(a) When *total dataset size* is fixed at 50,000

(b) When *dataset size per member* is fixed at 10,000

Figure 6.7: Certified Accuracy with different values of s with parameters $K \leq 20, M = 20$.

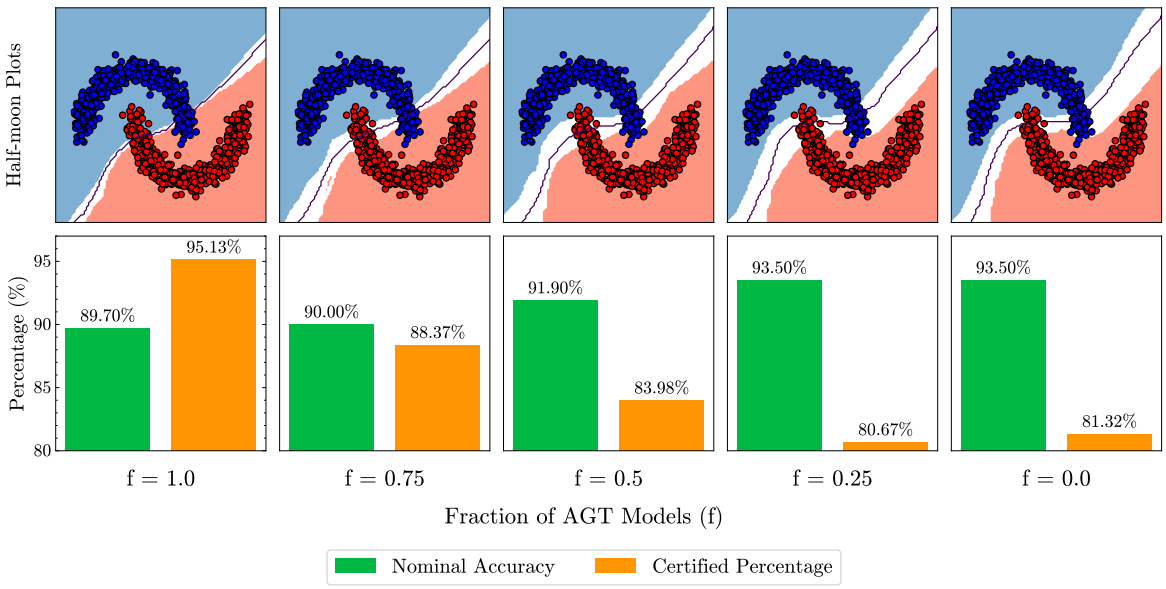
As shown in Figure 6.7, in both experiments, across all ensemble sizes M, smaller values of s consistently yield higher (or at least as high) certified accuracy. In Figures 6.7(a) and 6.7(b), this effect is especially pronounced when $M \leq 20$ and $M \leq 30$ respectively, aligning with our earlier observation in Figure 6.4 that certified accuracy is more sensitive to intrinsic robustness guarantees when M is small.

6.3.4 Robustness-Accuracy Trade-off

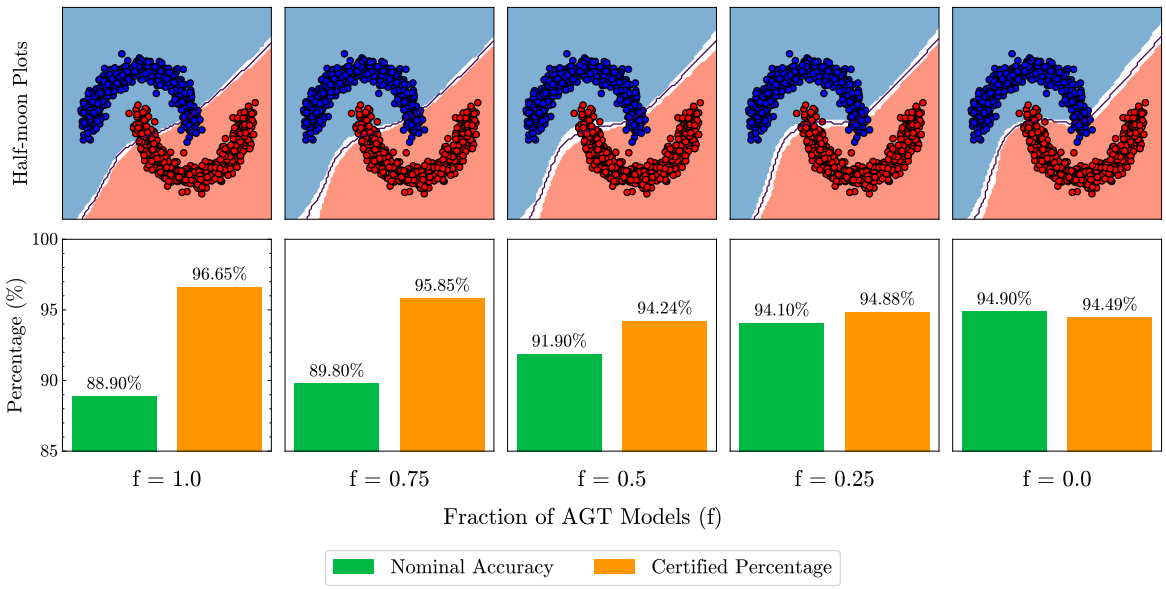
While increasing ensemble size tends to degrade AGT bounds, this can be mitigated by using simpler models. In light of the limitations of AGT, we explore a hybrid approach to balance accuracy and robustness. The intuition behind this is that smaller models trained with AGT contribute intrinsic robustness information, whereas the normally trained classifiers, which can have more expressive architectures, help maintain the ensemble accuracy.

Out of M classifiers, only a fraction f are trained with AGT and use simple MLPs with a single linear layer. The remaining $(1 - f)M$ classifiers follow the standard DPA training procedure and use more expressive MLPs with three linear layers. In this setup, we fix $M = 20, s = 5$ and set $K \leq 20$.

As shown in Figure 6.8, decreasing f leads to higher nominal accuracy but lower certified percentage. Repeating the experiment using FA instead of DPA as the aggregation mechanism produces the same trend, though with significantly tighter certification bounds.



(a) DPA with $M = 20$



(b) FA with $M = 20, d = 2$

Figure 6.8: We run DPA and FA with varying fractions f of models trained with AGT, decreasing from left to right, with $K \leq 20$ in all cases. In both figures, from left to right on the top row, we observe that the decision boundary becomes closer to the ground truth, but the uncertified (white) region increases. The corresponding nominal accuracies and certified percentages (multi-sample) are plotted in the second row of each figure. We observe that the nominal accuracies increase and certified percentages decrease as f decreases.

6.4 Discussion

In this chapter, we first showed that multi-sample analysis improves AGT ensemble bounds. Using these improved bounds, we demonstrated that intrinsic robustness offers substantial potential gains. The ability to obtain tight certification bounds with smaller ensemble sizes by leveraging intrinsic robustness highlights a promising direction for reducing training cost and improving scalability in future research.

We investigated the effect of ensemble size M on certification bounds, showing in Figures 6.5 and 6.6 that bounds improve as M increases, provided each partition's size remains constant. This highlights the strength of intrinsic robustness guarantees when batch size limitations, like those in AGT, are absent. Notably, intrinsic robustness methods like DP-based approaches [12] do not suffer from batch size constraints and represent a promising direction for future work, where, similar to DPA, increasing ensemble size could yield further improvements.

In Section 6.3.3, we observed that ensembles with smaller ensemble sizes are more sensitive to intrinsic robustness guarantees, as shown by the variation in certified accuracy with $R_j(x^{(i)})$ and s in Figures 6.4, 6.6 and 6.7. This indicates that improvements in intrinsic robustness can have a disproportionately large impact on the certified robustness of smaller ensembles. Enhancing intrinsic robustness guarantees could therefore enable comparably tight certification bounds while using much smaller ensemble sizes. This is an important insight given the accuracy–robustness trade-off in ensemble methods, where reducing M can help preserve higher nominal accuracy.

Finally, acknowledging the inherent trade-off between robustness and accuracy found in intrinsic robustness approaches such as AGT [10] and DP-based methods [12], we propose a novel hybrid strategy to better balance this trade-off. While AGT currently scales only to simpler datasets like half-moons, the strong performance observed suggests this may be a promising direction for future research. Furthermore, the inherent diversity of models in the hybrid strategy may enhance ensemble robustness since adversarially poisoned points affect each model differently. This aspect could be a valuable direction for future investigation.

Chapter 7

Conclusion

The aim of this thesis was to advance the state of the art in certified defenses against data poisoning by extending existing ensemble-based methods. We explored two orthogonal yet complementary directions for certification: aggregation-based defenses (such as DPA, FA and ROE) and model-level intrinsic robustness techniques based on bound propagation, such as AGT. To unify these approaches, we proposed a novel and principled framework that formulates the adversary's optimal strategy as a Mixed-Integer Linear Program (MILP), enabling the computation of tight multi-sample certificates that overcome a key limitation of prior works that rely on overly conservative per-sample bounds.

Our MILP-based approach flexibly supports a wide range of aggregation mechanisms and integrates model-level certified robustness into a single unified formulation. This generality allows it to subsume existing methods and go beyond their limitations. Empirical results on standard benchmarks demonstrate that our framework achieves substantial improvements—certifying up to 14% more predictions than prior state-of-the-art methods on CIFAR-10. Furthermore, the framework remains effective under practical constraints on adversarial budgets and batch sizes, highlighting its robustness and scalability in real-world settings.

In the final part of the thesis, we explored how intrinsic robustness, specifically through AGT, can be incorporated into our MILP formulation. While AGT provides meaningful robustness guarantees, we identified key limitations, such as its sensitivity to model complexity and batch size. Nonetheless, our analysis shows that intrinsic robustness holds great potential when paired with ensemble-based certification. This motivates future research into more scalable and effective intrinsic certification methods that can be tightly integrated with aggregation-based frameworks.

Overall, this work lays a strong foundation for developing more robust, scalable, and certifiable ensemble defenses against data poisoning. By integrating aggregation-based strategies with intrinsic robustness techniques, this work highlights a promising direction for advancing certified defenses and lays the groundwork for future improvements in robustness certification against data poisoning. We hope this paves

the way toward a new generation of certifiably robust training pipelines grounded in optimal, principled analysis.

7.1 Future Work

7.1.1 Other Intrinsic Robustness Certification Methods

In our experiments, we only explore AGT as a certification method for intrinsic robustness. Future work may explore other methods such as differential-privacy based models [12, 14], randomised smoothing [9] and influence function-based methods [68]. These alternatives could potentially address some of the limitations inherent to AGT. Additionally, as current work is already underway to improve the scalability of AGT, further efforts in this direction could make it more broadly applicable to larger datasets and complex architectures.

7.1.2 Other Machine Learning Architectures and Settings

A natural extension is to investigate how the proposed certification bounds apply to other model classes and learning paradigms, such as large language models or reinforcement learning agents. These alternative settings introduce distinct challenges and adversarial objectives, which may require adapting the certification framework to accommodate different threat models and training dynamics. Further work could even extend these bounds to adaptive adversaries where attackers obtain information about the learning algorithm and adapt their poisoning strategies accordingly, as discussed in Bose et al. [11].

7.1.3 Hybrid Ensemble Strategies for Tighter Certification Bounds

In Chapter 6, we proposed a hybrid ensemble certification method that aims to balance robustness and accuracy by mixing robustly trained and standard classifiers within an ensemble. This design allows the ensemble to benefit from intrinsic robustness while maintaining strong nominal accuracy. A key advantage of this approach is that the adversary cannot easily determine which models have been trained with robustness guarantees, introducing uncertainty that weakens targeted attacks. Future work can dive deeper into this robustness-accuracy trade-off, exploring how to optimally allocate model capacity or training strategies across ensemble members to further tighten certification bounds in practical settings. Additionally, the diversity among models in the hybrid ensemble may itself improve robustness, as adversarially poisoned points impact each model differently, making this an important direction for future research.

Bibliography

[1] Naman Patel, Prashanth Krishnamurthy, Siddharth Garg, and Farshad Khorrami. Bait and switch: Online training data poisoning of autonomous driving systems. *arXiv preprint arXiv:2011.04065*, 2020. pages 2

[2] Zhiyi Tian, Lei Cui, Jie Liang, and Shui Yu. A comprehensive survey on poisoning attacks and countermeasures in machine learning. *ACM Computing Surveys*, 55(8):1–35, 2022. pages 2, 6, 7, 8

[3] Mohammad Aljanabi, Alaa Hamza, Maad Mijwil, Mostafa Abotaleb, El-Sayed El-kenawy, Sahar Mohmmed, and Abdelhameed Ibrahim. Data poisoning: issues, challenges, and needs. *IET Conference Proceedings*, 2023:359–363, 2024. pages 2

[4] Avi Schwarzschild, Micah Goldblum, Arjun Gupta, John P Dickerson, and Tom Goldstein. Just how toxic is data poisoning? a unified benchmark for backdoor and data poisoning attacks. In *International Conference on Machine Learning*, pages 9389–9398. PMLR, 2021. pages 2

[5] Nicholas Carlini, Matthew Jagielski, Christopher A Choquette-Choo, Daniel Paleka, Will Pearce, Hyrum Anderson, Andreas Terzis, Kurt Thomas, and Florian Tramèr. Poisoning web-scale training datasets is practical, 2023. pages 2, 6

[6] Nilesh Dalvi, Pedro Domingos, Mausam, Sumit Sanghai, and Deepak Verma. Adversarial classification. In *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 99–108, 2004. pages 2

[7] Matthew Jagielski, Alina Oprea, Battista Biggio, Chang Liu, Cristina Nita-Rotaru, and Bo Li. Manipulating machine learning: Poisoning attacks and countermeasures for regression learning. In *2018 IEEE Symposium on Security and Privacy (SP)*, pages 19–35, 2018. doi: 10.1109/SP.2018.00057. pages 2, 8

[8] Jacob Steinhardt, Pang Wei W Koh, and Percy S Liang. Certified defenses for data poisoning attacks, 2017. pages 2, 9

[9] Elan Rosenfeld, Ezra Winston, Pradeep Ravikumar, and Zico Kolter. Certified robustness to label-flipping attacks via randomized smoothing. In *International*

Conference on Machine Learning, pages 8230–8241. PMLR, 2020. pages 2, 3, 5, 10, 14, 28, 49

[10] Philip Sosnin, Mark N Müller, Maximilian Baader, Calvin Tsay, and Matthew Wicker. Certified robustness to data poisoning in gradient-based training. *arXiv preprint arXiv:2406.05670*, 2024. pages 2, 3, 10, 12, 14, 19, 37, 38, 39, 47

[11] Avinandan Bose, Laurent Lessard, Maryam Fazel, and Krishnamurthy Dj Dvijotham. Keeping up with dynamic attackers: Certifying robustness to adaptive online data poisoning. *arXiv preprint arXiv:2502.16737*, 2025. pages 2, 49

[12] Yuzhe Ma, Xiaojin Zhu, and Justin Hsu. Data poisoning against differentially-private learners: Attacks and defenses. In *Proceedings of the 28th International Joint Conference on Artificial Intelligence*, IJCAI'19, page 4732–4738. AAAI Press, 2019. pages 2, 10, 15, 28, 38, 47, 49

[13] Yuhao Zhang, Aws Albarghouthi, and Loris D'Antoni. Bagflip: A certified defense against data poisoning. *Advances in Neural Information Processing Systems*, 35:31474–31483, 2022. pages 2, 14

[14] Shijie Liu, Andrew C Cullen, Paul Montague, Sarah M Erfani, and Benjamin IP Rubinstein. Enhancing the antidote: Improved pointwise certifications against poisoning attacks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pages 8861–8869, 2023. pages 2, 3, 15, 49

[15] Alexander Levine and Soheil Feizi. Deep partition aggregation: Provable defense against general poisoning attacks. *arXiv preprint arXiv:2006.14768*, 2020. pages 2, 9, 14, 15, 16, 23, 29, 30, 32

[16] Wenxiao Wang, Alexander J Levine, and Soheil Feizi. Improved certified defenses against data poisoning with (deterministic) finite aggregation. In *International Conference on Machine Learning*, pages 22769–22783. PMLR, 2022. pages 2, 3, 9, 10, 14, 17, 23, 29, 30

[17] Ruoxin Chen, Zenan Li, Jie Li, Junchi Yan, and Chentao Wu. On collective robustness of bagging against data poisoning. In *International Conference on Machine Learning*, pages 3299–3319. PMLR, 2022. pages 3, 23

[18] Keivan Rezaei, Kiarash Banihashem, Atoosa Chegini, and Soheil Feizi. Run-off election: Improved provable defense against data poisoning attacks. In *International Conference on Machine Learning*, pages 29030–29050. PMLR, 2023. pages 3, 9, 10, 14, 18, 23, 27, 29, 30

[19] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11): 2278–2324, 1998. pages 3, 30

[20] Alex Krizhevsky. Learning multiple layers of features from tiny images. Technical report, University of Toronto, 2009. Technical Report. pages 3, 30

[21] Johannes Stallkamp, Marc Schlipsing, Jan Salmen, and Christian Igel. The german traffic sign recognition benchmark: A multi-class classification competition. In *Proceedings of the IEEE International Joint Conference on Neural Networks (IJCNN)*, pages 1453–1460. IEEE, 2011. pages 3, 30

[22] Jonathon Shlens Christian Szegedy Ian J. Goodfellow. Explaining and harnessing adversarial examples. *CoRR*, abs/1412.6572, 2014. pages 5

[23] Andrea Paudice, Luis Muñoz-González, Andras Gyorgy, and Emil C. Lupu. Detection of adversarial training examples in poisoning attacks through anomaly detection. *arXiv preprint arXiv:1802.03041*, 2018. pages 5, 9

[24] Guillaume Stempfel and Liva Ralaivola. Learning svms from sloppily labeled data. In *Proceedings of the 19th International Conference on Artificial Neural Networks: Part I*, page 884–893. Springer Berlin Heidelberg, 2009. pages 5, 9

[25] Marco Barreno, Blaine Nelson, Russell Sears, Anthony D. Joseph, and J. D. Tygar. Can machine learning be secure? ASIACCS '06, page 16–25. Association for Computing Machinery, 2006. pages 5

[26] Xinyun Chen, Chang Liu, Bo Li, Kimberly Lu, and Dawn Song. Targeted backdoor attacks on deep learning systems using data poisoning. *arXiv preprint arXiv:1712.05526*, 2017. pages 6, 8

[27] Battista Biggio, Blaine Nelson, and Pavel Laskov. Poisoning attacks against support vector machines. In *Proceedings of the 29th International Coference on International Conference on Machine Learning*, ICML'12, page 1467–1474. Omnipress, 2012. pages 6, 8

[28] Shawn Shan, Jenna Cryan, Emily Wenger, Haitao Zheng, Rana Hanocka, and Ben Y. Zhao. Glaze: Protecting artists from style mimicry by text-to-image models. In *Proceedings of the 32nd USENIX Conference on Security Symposium*, SEC '23. USENIX Association, 2023. pages 6

[29] Shawn Shan, Wenxin Ding, Josephine Passananti, Stanley Wu, Haitao Zheng, and Ben Y. Zhao. Nightshade: Prompt-specific poisoning attacks on text-to-image generative models. In *2024 IEEE Symposium on Security and Privacy (SP)*, pages 807–825, 2024. doi: 10.1109/SP54263.2024.00207. pages 6

[30] Daniel Alber, Zihao Yang, Anton Alyakin, Eunice Yang, Sumedha Rai, Aly Valliani, Jeff Zhang, Gabriel Rosenbaum, Ashley Amend-Thomas, David Kurland, Caroline Kremer, Alexander Eremiev, Bruck Negash, Daniel Wiggan, Michelle Nakatsuka, Karl Sangwon, Sean Neifert, Hammad Khan, Akshay Save, and Eric Oermann. Medical large language models are vulnerable to data-poisoning attacks. *Nature Medicine*, 31:618–626, 01 2025. doi: 10.1038/s41591-024-03445-1. pages 6

[31] Shike Mei and Xiaojin Zhu. Using machine teaching to identify optimal training-set attacks on machine learners. In *Proceedings of the Twenty-Ninth*

AAAI Conference on Artificial Intelligence, AAAI'15, page 2871–2877. AAAI Press, 2015. pages 7

[32] Wei Pang and Percy Koh. Understanding black-box predictions via influence functions. In *Proceedings of the 34th International Conference on Machine Learning - Volume 70*, ICML'17, page 1885–1894. JMLR.org, 2017. pages 7

[33] Matthew Jagielski and Giorgio Severi. Subpopulation data poisoning attacks. In *Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security*, CCS '21, page 3104–3122. Association for Computing Machinery, 2020. pages 7

[34] Aniruddha Saha, Akshayvarun Subramanya, and Hamed Pirsiavash. Hidden trigger backdoor attacks. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 11957–11965, 2020. pages 8, 9

[35] Shaofeng Li, Hui Liu, Tian Dong, Benjamin Zi Hao Zhao, Minhui Xue, Haojin Zhu, and Jialiang Lu. Hidden backdoors in human-centric language models. In *Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security*, CCS '21, page 3123–3140. Association for Computing Machinery, 2021. pages 8, 9

[36] Tianyu Gu, Brendan Dolan-Gavitt, and Siddharth Garg. Badnets: Identifying vulnerabilities in the machine learning model supply chain. *arXiv preprint arXiv:1708.06733*, 2019. pages 8

[37] Moritz Hardt Benjamin Recht Oriol Vinyals Chiyuan Zhang, Samy Bengio. Understanding deep learning requires rethinking generalization. *Commun. ACM*, 64:107–115, 2021. doi: 10.1145/3446776. pages 8

[38] Tony Meyer and Brendon Whateley. Spambayes: Effective open-source, bayesian based, email classification system. *CEAS*, pages 1–8, 2004. pages 8

[39] Mengchen Zhao, Bo An, Wei Gao, and Teng Zhang. Efficient label contamination attacks against black-box learning models. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI-17*, pages 3945–3951, 2017. doi: 10.24963/ijcai.2017/551. pages 8

[40] Ali Shafahi, W. Ronny Huang, Mahyar Najibi, Octavian Suciu, Christoph Studer, Tudor Dumitras, and Tom Goldstein. Poison frogs! targeted clean-label poisoning attacks on neural networks. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, NIPS'18, page 6106–6116. Curran Associates Inc., 2018. pages 8

[41] Gabriela F. Cretu, Angelos Stavrou, Michael E. Locasto, Salvatore J. Stolfo, and Angelos D. Keromytis. Casting out demons: Sanitizing training data for anomaly sensors. In *2008 IEEE Symposium on Security and Privacy (sp 2008)*, pages 81–95, 2008. doi: 10.1109/SP.2008.11. pages 9

[42] Marco Barreno, Blaine Nelson, Anthony D. Joseph, and J. D. Tygar. The security of machine learning. *Machine Learning*, 81(2):121–148, 2010. doi: 10.1007/s10994-010-5188-5. pages 9

[43] Tongliang Liu and Dacheng Tao. Classification with noisy labels by importance reweighting. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 38 (3):447–461, 2016. pages 9

[44] J. Feng, H. Xu, S. Mannor, and S. Yan. Robust logistic regression and classification. *Advances in Neural Information Processing Systems*, 1:253–261, 2014. pages 9

[45] Nealgupta Neeharperi and W Ronnyhuang. Deep k-NN defense against clean-label data poisoning attacks. In *Computer Vision - ECCV 2020 Workshops*, volume 12535, pages 55–70. Springer-Verlag, 2020. pages 9

[46] Brandon Tran, Jerry Li, and Aleksander Madry. Spectral signatures in backdoor attacks. *Advances in neural information processing systems*, 31, 2018. pages 9

[47] Ren Wang, Gaoyuan Zhang, Sijia Liu, Pin-Yu Chen, Jinjun Xiong, and Meng Wang. Practical detection of trojan neural networks: Data-limited and data-free cases. In *Computer Vision - ECCV 2020 - 16th European Conference*, volume 12368, pages 222–238. Springer-Verlag, 2020. pages 9

[48] Yuanshunyao Bolunwang. Neural cleanse: Identifying and mitigating backdoor attacks in neural networks. In *2019 IEEE Symposium on Security and Privacy, SP 2019*, pages 707–723, 2019. doi: 10.1109/SP.2019.00031. pages 9

[49] Eitan Borgnia, Valeriia Cherepanova, Liam Fowl, Amin Ghiasi, Jonas Geiping, Micah Goldblum, Tom Goldstein, and Arjun Gupta. Strong data augmentation sanitizes poisoning and backdoor attacks without an accuracy tradeoff. In *IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 3855–3859, 2021. doi: 10.1109/ICASSP39728.2021.9414862. pages 9

[50] Kang Liu, Brendan Dolan-Gavitt, and Siddharth Garg. Fine-pruning: Defending against backdooring attacks on deep neural networks. In *Research in Attacks, Intrusions, and Defenses - 21st International Symposium*. Springer International Publishing, 2018. pages 9

[51] Huang Xiao, Battista Biggio, Blaine Nelson, Han Xiao, Claudia Eckert, and Fabio Roli. Support vector machines under adversarial label contamination. *Neurocomputing*, 160:53–62, July 2015. pages 9

[52] Nagarajan Natarajan, Inderjit S Dhillon, Pradeep K Ravikumar, and Ambuj Tewari. Learning with noisy labels. In *Advances in Neural Information Processing Systems*, volume 26. Curran Associates, Inc., 2013. pages 9

[53] Jeremy M Cohen, Elan Rosenfeld, and J Zico Kolter. Certified adversarial robustness via randomized smoothing. In *International Conference on Machine Learning*, pages 1310–1320. PMLR, 2019. pages 10, 14

[54] Chen Zhu, Warren He Huang, Hui Li, Gavin Taylor, Christoph Studer, and Tom Goldstein. Transferable clean-label poisoning attacks on deep neural nets. In *Proceedings of the 36th International Conference on Machine Learning*, pages 7614–7623. PMLR, 2019. pages 12

[55] Rishi Jha, Jonathan Hayase, and Sewoong Oh. Label poisoning is all you need. *Advances in Neural Information Processing Systems*, 36:71029–71052, 2023. pages 12

[56] Zayd Hammoudeh and Daniel Lowd. Feature partition aggregation: A fast certified defense against a union of ℓ_0 attacks. *arXiv preprint arXiv:2302.11628*, 2023. pages 14

[57] Jinyuan Jia, Yupei Liu, Yuepeng Hu, and Neil Zhenqiang Gong. {PORE}: Provably robust recommender systems against data poisoning attacks. In *32nd USENIX Security Symposium (USENIX Security 23)*, pages 1703–1720, 2023. pages 14

[58] Sven Gowal, Krishnamurthy Dvijotham, Robert Stanforth, Rudy Bunel, Chongli Qin, Jonathan Uesato, Relja Arandjelovic, Timothy Mann, and Pushmeet Kohli. On the effectiveness of interval bound propagation for training verifiably robust models. *arXiv preprint arXiv:1810.12715*, 2018. pages 14, 21

[59] Huan Zhang, Tsui-Wei Weng, Pin-Yu Chen, Cho-Jui Hsieh, and Luca Daniel. Efficient neural network robustness certification with general activation functions. *Advances in neural information processing systems*, 31, 2018. pages 14, 21

[60] Martin Abadi, Andy Chu, Ian Goodfellow, H Brendan McMahan, Ilya Mironov, Kunal Talwar, and Li Zhang. Deep learning with differential privacy. In *Proceedings of the 2016 ACM SIGSAC conference on computer and communications security*, pages 308–318, 2016. pages 15

[61] Cynthia Dwork, Aaron Roth, et al. The algorithmic foundations of differential privacy. *Foundations and Trends® in Theoretical Computer Science*, 9(3–4):211–407, 2014. pages 15

[62] David E. Rumelhart, Geoffrey E. Hinton, and Ronald J. Williams. Learning representations by back-propagating errors. *Nature*, 323:533–536, 1986. pages 21

[63] Steven Diamond and Stephen Boyd. Cvxpy: A python-embedded modeling language for convex optimization. *Journal of Machine Learning Research*, 17 (83):1–5, 2016. pages 25

[64] Gurobi Optimization, LLC. *Gurobi Optimizer Reference Manual*, 2024. URL `https://www.gurobi.com`. pages 26, 32

[65] S. Pouya, O. Toragay, and M. Mohammadi. Predicting the solution time for optimization problems using machine learning. In *Optimization, Learning Algorithms and Applications*, Communications in Computer and Information Science, pages 450–465. Springer Nature Switzerland, 2024. pages 29

[66] Min Lin, Qiang Chen, and Shuicheng Yan. Network in network. *arXiv preprint arXiv:1312.4400*, 2013. pages 30, 31

[67] Spyros Gidaris, Praveer Singh, and Nikos Komodakis. Unsupervised representation learning by predicting image rotations. *arXiv preprint arXiv:1803.07728*, 2018. pages 30, 32

[68] Pang Wei Koh and Percy Liang. Understanding black-box predictions via influence functions. In *Proceedings of the 34th International Conference on Machine Learning - Volume 70*, ICML'17, page 1885–1894. JMLR.org, 2017. pages 49

Chapter 8

Declarations

8.1 Use of Generative AI

I acknowledge the use of ChatGPT (version: GPT-4o, by OpenAI, `https://chat.openai.com/`) to assist with providing language suggestions and proofreading support for improved clarity. I confirm that no content generated by AI has been presented as my own work.

8.2 Ethical Considerations

Our experiments are conducted on publicly available datasets (CIFAR-10, GTSRB and MNIST), and do not involve any private or sensitive data. We do not engage with real-world deployment or personal user data, and no human subjects are involved. This mitigates any direct negative societal impacts of the publication of numerical results.

Indirect negative societal impacts are always possible when works publish security protocols, as publicly improving defenses reveals defense strategies to adversaries who may leverage them to effect negative outcomes. Given that this work focuses particularly on provably robust defenses, we argue this work has very limited potential for negative societal impacts.

8.3 Sustainability

All experiments were conducted using two NVIDIA L40 GPUs. To promote computational efficiency and limit energy consumption, each MILP solve was restricted to a maximum time of 30 minutes per batch. This time limit was set to balance thoroughness with practical resource usage, avoiding excessively long runs that yield diminishing returns. Instead of performing exhaustive grid searches over hyperparameters such as ensemble size and perturbation budget, values were selected iteratively by running individual experiments and observing their effects before proceeding.

By enforcing these constraints and utilising shared GPU resources, the experiments aimed to maintain reasonable energy use while achieving reliable results. Although the scope of sustainability measures is limited, these practices reflect an effort to manage computational resources responsibly.

8.4 Availability of Data and Materials

The code for this thesis can be found in this public repository: `https://github.com/S-Pei/multi-sample-certification`.

Appendix A

Experimental Variance

Table A.1: Average certified accuracy of 1 batch over 3 runs of DPA, DPA+ROE, FA and FA+ROE on CIFAR-10 with $M = 50$. The certified accuracies are reported as (mean $\pm$ std). *(The batch sizes used are consistent with those reported in Table 5.1.)*

method	d	avg certified accuracy ($\pm$ std)			
		$K \leq 3$	$K \leq 5$	$K \leq 10$	$K \leq 20$
DPA		$66.30\%(\pm 0.29\%)$	$64.00\%(\pm 0.29\%)$	$53.35\%(\pm 0.98\%)$	$21.50\%(\pm 0.41\%)$
DPA+ROE		$67.20\%(\pm 0.22\%)$	$64.87\%(\pm 0.12\%)$	$53.75\%(\pm 0.74\%)$	$29.17\%(\pm 1.03\%)$
FA	16	$69.67\%(\pm 0.47\%)$	$67.67\%(\pm 0.47\%)$	$68.33\%(\pm 1.18\%)$	$10.00\%(\pm 0.00\%)$
FA+ROE		$70.00\%(\pm 0.00\%)$	$69.00\%(\pm 0.00\%)$	$72.50\%(\pm 0.00\%)$	$16.67\%(\pm 4.71\%)$

Table A.2: Average certified accuracy of 1 batch over 3 runs of DPA, DPA+ROE, FA and FA+ROE on GTSRB with $M = 50$. The certified accuracies are reported as (mean $\pm$ std). *(The batch sizes used are consistent with those reported in Table 5.1.)*

method	d	avg certified accuracy ($\pm$ std)			
		$K \leq 5$	$K \leq 10$	$K \leq 15$	$K \leq 20$
DPA		$85.22\%(\pm 0.22\%)$	$79.00\%(\pm 0.45\%)$	$69.42\%(\pm 0.59\%)$	$51.07\%(\pm 0.82\%)$
DPA+ROE		$85.08\%(\pm 0.17\%)$	$79.87\%(\pm 0.42\%)$	$71.58\%(\pm 0.24\%)$	$55.47\%(\pm 0.50\%)$
FA	16	$88.33\%(\pm 0.47\%)$	$76.00\%(\pm 0.00\%)$	$70.67\%(\pm 1.89\%)$	$60.00\%(\pm 0.00\%)$
FA+ROE		$89.33\%(\pm 0.47\%)$	$75.33\%(\pm 0.94\%)$	$72.00\%(\pm 0.00\%)$	$80.00\%(\pm 0.00\%)$

Table A.3: Average certified accuracy of 1 batch over 3 runs of DPA and DPA+ROE on MNIST with $M = 1200$. The certified accuracies are reported as (mean $\pm$ std). *(The batch sizes used are consistent with those reported in Table 5.1.)*

method	avg certified accuracy ($\pm$ std)			
	$K \leq 100$	$K \leq 200$	$K \leq 300$	$K \leq 500$
DPA	$93.00\%(\pm 0.00\%)$	$86.50\%(\pm 0.00\%)$	$81.00\%(\pm 0.00\%)$	$33.67\%(\pm 1.25\%)$
DPA+ROE	$92.83\%(\pm 0.24\%)$	$86.50\%(\pm 0.00\%)$	$82.17\%(\pm 0.24\%)$	$43.33\%(\pm 0.94\%)$