CERTIFYING THE FULL YOLO PIPELINE: A PROBABILISTIC VERIFICATION APPROACH

Anonymous authors
Paper under double-blind review

ABSTRACT

Object detection systems are essential in safety-critical applications, but they are vulnerable to object disappearance (OD) threat, in which valid objects become undetected under small input perturbations, creating serious risks. This paper addresses the problem of verifying the robustness of YOLO (You Only Look Once) networks against OD by proposing a three-step probabilistic verification framework: (1) estimating output ranges under a distribution of input perturbations, (2) formally verifying the Non-Maximum Suppression (NMS) process within these ranges, and (3) iteratively refining the results to reduce over-approximation. The framework scales to practical YOLO models. Both theoretical analysis and experimental results demonstrate that our method achieves comparable probabilistic guarantees and provides tighter Intersection-over-Union (IoU) lower bounds while requiring significantly fewer samples than existing methods.

1 INTRODUCTION

Object detection (Zhao et al., 2019; Zou et al., 2023) is a fundamental computer vision task that combines object localization and classification. Neural network architectures, including YOLO (You Only Look Once) (Redmon, 2016; Redmon & Farhadi, 2017; Farhadi & Redmon, 2018; Bochkovskiy et al., 2020a), Fast R-CNN (Girshick, 2015), and SSD (Liu et al., 2016; Li et al., 2017), have achieved significant progress in both accuracy and computational efficiency, enabling their widespread deployment in real-world applications. Despite these advances, neural network-based detection systems remain vulnerable to minute, often imperceptible, input perturbations (Im Choi & Tian, 2022; Lin et al., 2025; Goodfellow et al., 2015; Madry et al., 2018; Dong et al., 2018; Carlini & Wagner, 2017). Of particular concern is the *object disappearance (OD) problem*, in which minor input perturbations suppress the detection of valid objects. Such perturbations pose substantial risks in safety-critical domains, potentially leading to catastrophic consequences due to detection failures. Consequently, verifying the safety of object detection systems is crucial for their reliable deployment.

To measure network robustness, verification methods are commonly employed. For a given network F, an input $\boldsymbol{x}$, and a property function ϕ, verification methods can be grouped into three categories:
Formal Verification. The goal is to find the maximum perturbation radius ε such that $\phi(\mathrm{F}(\boldsymbol{x}')) = \phi(\mathrm{F}(\boldsymbol{x}))$ for all $\boldsymbol{x}' \in \mathcal{B}_p(\boldsymbol{x}, \varepsilon)$, where $\mathcal{B}_p(\boldsymbol{x}, \varepsilon) = \{\boldsymbol{x}' : \|\boldsymbol{x}' - \boldsymbol{x}\|_p \leq \varepsilon\}$ is the p-norm ball of radius ε centered at $\boldsymbol{x}$. Alternatively, for a fixed ε, one can verify whether the property holds for all $\boldsymbol{x}' \in \mathcal{B}_p(\boldsymbol{x}, \varepsilon)$. However, formal verification is NP-complete (Katz et al., 2017), making it infeasible for large-scale networks. Even state-of-the-art tools (Zhang et al., 2022b;a) face challenges in handling networks with millions of neurons (Brix et al., 2023; 2024).
Probabilistic Verification. Given a radius ε and a tolerance α, the goal is to verify whether $\mathrm{P}_{\boldsymbol{x}' \sim \mathcal{D}}(\phi(\mathrm{F}(\boldsymbol{x}')) = \phi(\mathrm{F}(\boldsymbol{x}))) \geq 1 - \alpha$, where $\mathcal{D}$ is a distribution over $\mathcal{B}_p(\boldsymbol{x}, \varepsilon)$. Although this approach leverages probabilistic guarantees to reduce verification time and memory, its reliance on processing internal network nodes prevents it from scaling to larger network architectures. Representative works include (Weng et al., 2019; Boetius et al., 2025).
PAC Verification. Given ε, α, and β, the goal is to verify whether $\mathrm{P}_{\boldsymbol{x}' \sim \mathcal{D}}(\phi(\mathrm{F}(\boldsymbol{x}')) = \phi(\mathrm{F}(\boldsymbol{x}))) \geq 1 - \alpha$ holds with confidence at least $1 - \beta$. PAC methods rely on sampling and do not require access to internal network nodes, which allows them to scale further to larger models and datasets. Representative works include ~~(Tran et al., 2023; Park et al., 2020; Li et al., 2022)~~ (Tran et al., 2023; Park et al., 2020; Li et al., 2022; Blohm et al., 2025).

Verifying object detection networks with these methods, however, presents additional challenges beyond the large parameter scales:
(1) **Post-Processing Stage**: Critical post-processing steps, such as Non-Maximum Suppression (NMS) (Neubeck & Van Gool, 2006), generally fall outside the scope of current formal verification methods (Cohen et al., 2024; Elboher et al., 2024);
(2) **Large Input-Output Spaces**: The dimensionality of the detection inputs and outputs even renders PAC-based methods ~~(Li et al., 2022)~~ (Li et al., 2022; Blohm et al., 2025; Haussler & Welzl, 1987) computationally infeasible.

Due to these limitations, even recent verification methods specifically designed for object detection (Cohen et al., 2024; Elboher et al., 2024) are restricted to simplified models or do not account for complex operations such as NMS. To address this gap, we propose a PAC-based **O**bject **D**etection **P**robabilistic **V**erification (ODPV) framework for YOLO networks under OD threats. To our knowledge, **this is the first framework that effectively verifies the robustness of the original object detection networks at a practical scale**. Although PAC verification cannot provide deterministic guarantees, it currently offers the most practical means to validate YOLO in a reasonable time.

Our methodology includes three main components: (1) estimating output ranges under input perturbations, (2) formally verifying NMS within the estimated output space, and (3) iteratively refining verification results. We implement our approach and evaluate it on standard benchmarks. Our main contributions are as follows.

(1) We formally define the PAC verification problem of the ~~object disappearance~~ OD threat in object detection and propose a novel verification approach to address it.
(2) We implement a complete verification process that includes the NMS step, which has been underexplored in previous work, and provide probabilistic guarantees for each step.
(3) We conduct experiments on widely used networks and datasets to evaluate our proposed method. We demonstrate that our method requires fewer samples to achieve comparable probabilistic guarantees and tighter certified Intersection-over-Union (IoU) bounds.

In summary, we are the first to address the challenges of verifying large-scale detection networks and to provide an efficient probabilistic verification method.

Remark 1. *We emphasize an important distinction: Our work differs from randomized smoothing in the type of guarantee it provides (Cohen et al., 2019; Yang et al., 2020). Randomized smoothing establishes robustness for modified, "smoothed" classifiers, not the original detector. In contrast, we leave the network unchanged and provide statistical guarantees for the original model.*

2 Related Work

Object detection. Early detectors relied on hand-crafted features such as HOG (Dalal & Triggs, 2005) and sliding windows (Viola & Jones, 2001), but lacked adaptability. CNN-based approaches transformed feature extraction; R-CNN variants (Girshick et al., 2014; Ren, 2015) combined region proposals with deep learning methods. More recent approaches such as YOLO (Redmon, 2016; Redmon & Farhadi, 2017; Farhadi & Redmon, 2018; Bochkovskiy et al., 2020b) and SSD (Liu et al., 2016; 2017) achieved real-time detection in complex scenarios.

Verification techniques for Neural Networks. Formal verification determines whether a property holds under given input constraints. State-of-the-art tools (Katz et al., 2017; 2019; Zhang et al., 2022a; 2018) employ Branch-and-Bound, combining relaxations (Singh et al., 2019; Bak, 2021), bound propagation (Wang et al., 2018b; Weng et al., 2018; Wang et al., 2018a; Gowal et al., 2019), and constraint solving (Khedr et al., 2021; Ehlers, 2017; Henriksen & Lomuscio, 2020; Kouvaros & Lomuscio, 2021). However, for large networks such as YOLO (with $640 \times 480 \times 3$ inputs), even basic bound propagation may require more than 5000 GB of memory, rendering formal verification infeasible in practice. To address scalability, probabilistic verification estimates the likelihood of property satisfaction. Sampling-based methods (Webb et al., 2019; Cardelli et al., 2019; Mangal et al., 2019; Anderson & Sojoudi, 2023) provide probabilistic estimates, but may miss rare cases, thereby creating gaps between analysis and actual robustness. DeepPAC (Li et al., 2022) approximates local network behavior with linear equations and high-confidence error bounds, but it requires prohibitively large sample sizes for models such as YOLO. Techniques like median smoothing (Chi-

Figure 1: (First Stage) The network tries to find all boxes that may contain objects. A subset of these boxes is shown here.

Figure 2: (Second Stage) Final output boxes selected by NMS include the corresponding label and its confidence score.

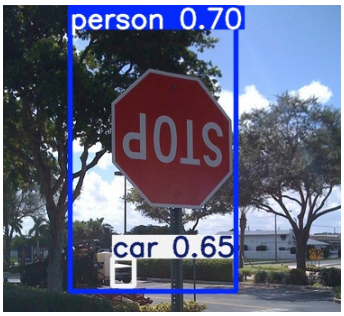


Figure 3: Under imperceptible input perturbations, YOLO can no longer recognize these objects.

ang et al., 2020) certify robustness for a modified, "smoothed" detector, whereas our approach directly verifies the original network.

Verification of Object Detection. Current efforts mainly focus on small or simplified detectors. (Cohen et al., 2024) propagate bounds to certify ~~Intersection-over-Union (IoU)~~IoU, while (Elboher et al., 2024) encode IoU into networks for existing verifiers. Both approaches ignore the NMS step and fail to scale to real-world detectors. Comprehensive verification of complete detection pipelines remains an open problem.

3 PRELIMINARIES

This section outlines the key stages of YOLO object detection, as shown in Fig. 1- 3 with an image from the COCO validation dataset (Lin et al., 2014) and defines the threat of ~~object disappearance~~OD.

3.1 KEY STAGES OF YOLO OBJECT DETECTION

Bounding Box Prediction (First Stage). The YOLO network $\mathrm{F} : \mathbb{R}^{d_0} \to \mathbb{R}^{d_L}$ processes an input $\boldsymbol{x}$ (with dimension d_0) to generate an output $\boldsymbol{y} = \mathrm{F}(\boldsymbol{x})$ (with dimension d_L). The output $\boldsymbol{y}$ can be reformulated as a set of bounding boxes $\{box_i\}_{i=1}^{n_{\boldsymbol{x}}}$, where $n_{\boldsymbol{x}}$ is a constant determined by the fixed input dimension. Each bounding box box_i is represented as $(x_i, y_i, w_i, h_i, c_i, p_{i_1}, p_{i_2}, \ldots, p_{i_n})$. Here, (x_i, y_i) denotes the box's center coordinates, (w_i, h_i) its width and height, c_i its confidence score, and p_{i_j} the probability of the object belonging to class j (for $j \in [n]$, where n is the total number of classes). The class of box_i is assigned as $\mathrm{Class}(box_i) = \arg\max_{j \in [n]} p_{i_j}$. These boxes collectively identify possible object locations in the input image, as Figure 1 illustrates.

Non-Maximum Suppression (Second Stage). Let $\boldsymbol{y} = \mathrm{F}(\boldsymbol{x})$ be the output tensor from the first stage. The second stage processes $\boldsymbol{y}$ by using an operator N to select a subset of bounding boxes $\{box_{i_j}\}_{i_j \in [n_{\boldsymbol{x}}]} \subseteq \boldsymbol{y} = \{box_i\}_{i=1}^{n_{\boldsymbol{x}}}$, forming the final YOLO output (Figure 2). The standard operator N is ~~Non-Maximum Suppression (NMS)~~NMS (Neubeck & Van Gool, 2006) in YOLO, which uses $\boldsymbol{y}$ and predefined thresholds $\eta, \iota \in (0, 1)$ to select the final output. For simplicity, we denote this as $\mathrm{N}(\boldsymbol{y})$, as η and ι are fixed, so we omit them. NMS selects boxes based on the following three rules:
(n1): If $i_j \in [n_{\boldsymbol{x}}]$ and $box_{i_j} \in \mathrm{N}(\boldsymbol{y})$, then it must satisfy $c_{i_j} \geq \iota$;
(n2): If $i_j \in [n_{\boldsymbol{x}}]$ satisfies $box_{i_j} \notin \mathrm{N}(\boldsymbol{y})$ and $c_{i_j} \geq \iota$, then there must exist a $box_{i_k} \in \mathrm{N}(\boldsymbol{y})$ such that $\mathrm{Class}(box_{i_j}) = \mathrm{Class}(box_{i_k})$ and $c_{i_j} \leq c_{i_k}$, $\mathrm{IoU}(box_{i_j}, box_{i_k}) \geq \eta$;
(n3): If $i_j, i_k \in [n_{\boldsymbol{x}}]$ such that $box_{i_j}, box_{i_k} \in \mathrm{N}(\boldsymbol{y})$ and $\mathrm{Class}(box_{i_j}) = \mathrm{Class}(box_{i_k})$, then it must satisfy $\mathrm{IoU}(box_{i_j}, box_{i_k}) < \eta$.

The $\mathrm{IoU}(box_1, box_2) = \frac{\mathrm{Area}(box_1 \cap box_2)}{\mathrm{Area}(box_1 \cup box_2)}$ measures overlap between two boxes, where $\mathrm{Area}(box_1 \cap box_2)$ and $\mathrm{Area}(box_1 \cup box_2)$ denote the ~~intersection and union~~ IoU areas. The NMS-selected subset is unique and we focus on its properties, as implementation details are beyond our scope.

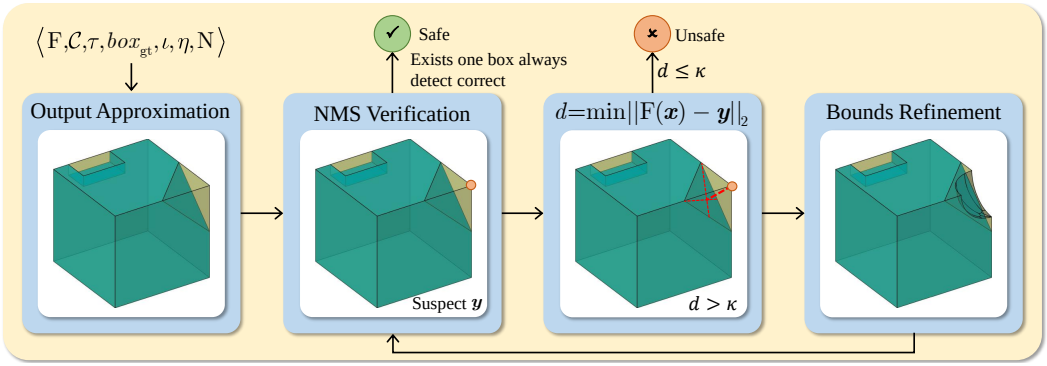


Figure 4: Our verification framework for object detection networks. The green cube represents the network's **true but unknown output space** under input constraints. The yellow cube is the **over-approximated region** calculated by our method, which is probabilistically guaranteed to contain the true output space. Part 3 of our framework (Refinement) progressively shrinks the yellow region by identifying and excluding areas that do not intersect with the true output space, thereby tightening the verification bounds.

3.2 Object Disappearance Threat on Object Detection

An object detection model successfully detects an object O in the image $\boldsymbol{x}$ if there exists at least one $box_i \in \mathrm{N}(\mathrm{F}(\boldsymbol{x}))$ satisfying: $\mathrm{Class}(box_i) = \mathrm{Class}(box_{\mathrm{gt}})$ and $\mathrm{IoU}(box_i, box_{\mathrm{gt}}) \geq \tau$, where τ is a predefined IoU threshold and box_{gt} is O's ground truth bounding box. We define the ~~object disappearance (OD)~~ OD threat as follows:

~~Object Disappearance (~~OD ~~)~~Threat Definition. Given ground truth box box_{gt}, perturbation radius ε, IoU threshold τ, and class $\mathrm{Class}(box_{\mathrm{gt}})$, OD occurs if there exists a perturbation $\boldsymbol{\delta}$ with $\|\boldsymbol{\delta}\|_p \leq \varepsilon$ such that

$$\max_{box_i \in \mathrm{N}(\mathrm{F}(\boldsymbol{x}+\boldsymbol{\delta}))} \left[\mathrm{IoU}(box_i, box_{\mathrm{gt}}) \cdot \mathbb{I}\left(\mathrm{Class}(box_i) = \mathrm{Class}(box_{\mathrm{gt}})\right) \right] < \tau.$$

where $\mathbb{I}(\cdot)$ denotes an indicator function (returns 1 if true, 0 otherwise).

4 Verification Framework for Object Detection

In this section, we introduce the verification target and our verification approach.

First, we formally define the OD PAC-Verification problem.

Definition 1 (OD PAC-Verification Problem)**.** Given input constraints $\mathcal{C}$, IoU threshold τ, error rate $\alpha \in [0, 1]$ and significance level $\beta \in [0, 1]$ and ground truth box box_{gt}, verify whether with confidence at least $1 - \beta$, the following holds:

$$\mathrm{P}_{\boldsymbol{x} \sim \mathcal{C}} \left(\exists box_i \in \mathrm{N}(\mathrm{F}(\boldsymbol{x})) \text{ s.t. } [\mathrm{IoU}(box_i, box_{\mathrm{gt}}) \geq \tau \wedge \mathrm{Class}(box_i) = \mathrm{Class}(box_{\mathrm{gt}})] \right) \geq 1 - \alpha$$

If true, the system is deemed PAC-safe in $\mathcal{C}$ under τ. This definition reduces to the OD Formal Verification Problem when $\alpha = 0, \beta = 0$. Here we use $\boldsymbol{x} \sim \mathcal{C}$ to denote that $\boldsymbol{x}$ is sampled from a distribution over the input constraint set $\mathcal{C}$.

Then, we propose a three-part verification framework (see Alg. 1 and Fig. 4) to solve it:

Part 1: Network Output Approximation. For input $\boldsymbol{x}^{(0)}$ and constraint $\mathcal{C}$, approximate the output set $\{\mathrm{F}(\boldsymbol{x})\}_{\boldsymbol{x} \in \mathcal{C}}$ with a regular region $\mathcal{Z}$ (hyperrectangles/hyperspheres) such that: $\{\mathrm{F}(\boldsymbol{x})\}_{\boldsymbol{x} \in \mathcal{C}} \subseteq \mathcal{Z}$.

Part 2: NMS Verification. Verify whether, for all $\boldsymbol{y} \in \mathcal{Z}$, there exists a $box_i \in \mathrm{N}(\boldsymbol{y})$ that satisfies the OD safety property (Definition 1). If this holds, the detector is safe. Otherwise, identify a $\boldsymbol{y}$ that violates the IoU or class-matching condition.

Part 3: Counterexample Validation and Refinement. Compute $d_{\min} = \min_{\boldsymbol{x} \in \mathcal{C}} \|\mathrm{F}(\boldsymbol{x}) - \boldsymbol{y}\|_2$. If $d_{\min} \leq \kappa$ (with $\kappa \geq 0$ as a tolerance), the system is unsafe. Otherwise, refine $\mathcal{Z}$ by excluding

Algorithm 1 Verification framework for the OD PAC-Verification problem

Require:
The network F; the input constraints $\mathcal{C}$; the threshold in OD verification problem τ ; the threshold for Part Three κ; the number of refinement steps T; the ground truth bounding box box_{gt}.
Ensure:
Whether YOLO is safe under OD attack.
Get $\mathcal{Z}$ over-approximating $\{\text{F}(\boldsymbol{x})\}_{\boldsymbol{x}\in\mathcal{C}}$ ▷ Part One
repeat
  if $\forall \boldsymbol{y} \in \mathcal{Z}, \exists box_i \in \text{N}(\boldsymbol{y})$ such that $\text{IoU}(box_i, box_{\text{gt}}) \geq \tau \wedge \text{Class}(box_i) = q$ **then**
    return Safe. ▷ $q = \text{Class}(box_{\text{gt}})$, Part Two
  else
    Get $\boldsymbol{y}' \in \mathcal{Z}$ violating the specified property ▷ Part Two
    $d_{\min} = \min_{\boldsymbol{x}\in\mathcal{C}} \|\text{F}(\boldsymbol{x}) - \boldsymbol{y}'\|_2$ ▷ Part Three
    $\mathcal{Z} = \mathcal{Z} \setminus \mathcal{B}_2(\boldsymbol{y}', d_{\min})$ ▷ Part Three
  end if
until $d_{\min} \leq \kappa$ or refine T steps
return Unsafe ▷ Part Three

Algorithm 2 ~~Algorithm for Part 1:~~ Network Output Approximation (Part 1)

Require:
The neural network F; the input constraints $\mathcal{C}$, ~~$N_1, N_2 \in \mathbb{Z}^+$~~ $N_1, N_2 \in \mathbb{Z}^+$, a threshold ζ.
Ensure:
The bounding box $\mathcal{Z}$.
$\{\boldsymbol{x}^{(i)}\}_{i=1}^{N_1} \leftarrow$Randomly select N_1 points in $\mathcal{C}$. ▷ Find the $\boldsymbol{v}_{\max}$
for $j \in [d_L]$ **do** ▷ Find the $\boldsymbol{v}_{\max}$
  $(\boldsymbol{v}_{\max})_j \leftarrow \max\{\max_i\{|\text{F}(\boldsymbol{x}^{(i)})_j - \text{F}(\boldsymbol{x}^{(0)})_j|\}, \zeta\}$ ▷ Use ζ to prevent division by zero
end for
$\{\boldsymbol{z}^{(i)}\}_{i=1}^{N_2} \leftarrow$Randomly select N_2 points in $\mathcal{C}$. ▷ Find the c_1
$c_1 \leftarrow \max_{i\in[N_2], j\in[d_L]} \frac{|\text{F}(\boldsymbol{z}^{(i)}) - \text{F}(\boldsymbol{x}^{(0)})|_j}{(\boldsymbol{v}_{\max})_j}$. ▷ Find the c_1
return $\mathcal{Z} \leftarrow \{\text{F}(\boldsymbol{x}^{(0)}) + \boldsymbol{\epsilon} : |\boldsymbol{\epsilon}| \leq c_1 \boldsymbol{v}_{\max}\}$.

$\mathcal{B}_2(\boldsymbol{y}, d_{\min}) = \{\boldsymbol{y}' : \|\boldsymbol{y} - \boldsymbol{y}'\|_2 < d_{\min}\}$ as the regular region we obtained may be larger than the actual output space $\{\text{F}(\boldsymbol{x})\}_{\boldsymbol{x}\in\mathcal{C}}$, and then go back to Part 2. Note that we limit Part 3 iterations for high-dimensional outputs to prevent computational overload.

Remark 2. *Because our goal is PAC verification, each of the three steps is implemented using probabilistic methods with probability guarantees, rather than exact computation, as shown in the next section.*

5 Verification Method for YOLO Object Detection

We illustrate the application of the verification framework from Section 4 to YOLO object detection. Because of YOLO's complexity and scale, formal verification becomes intractable; therefore, we adopt PAC verification, i.e. black-box verification via sampling. Proofs for Propositions, Lemmas, and Theorems are provided in the Appendix.

We define the input constraint as $\mathcal{C} = \{\boldsymbol{x} : \|\boldsymbol{x} - \boldsymbol{x}^{(0)}\|_p \leq \varepsilon\}$ for a given sample $\boldsymbol{x}^{(0)}$, norm $p \in \mathbb{Z}^+ \cup \{\infty\}$, and perturbation radius $\varepsilon \in (0, 1)$. We consider a probability distribution over the input set $\mathcal{C}$, and write $\boldsymbol{x} \sim \mathcal{C}$ to denote that $\boldsymbol{x}$ is a sample drawn from this distribution. For convenience, we define the comparison $\boldsymbol{a} \leq \boldsymbol{b}$ for vectors $\boldsymbol{a}, \boldsymbol{b} \in \mathbb{R}^n$ to mean $\forall j \in [n] : \boldsymbol{a}_j \leq \boldsymbol{b}_j$, where $\boldsymbol{a}_j$ is the j-th component of $\boldsymbol{a}$. Similarly, scalar-vector multiplication is defined element-wise.

Algorithm 3 ~~Algorithm for Part 2:~~ NMS Verification (Part 2)

Require:
$\{\{box_i^k\}_{i=1}^{n_{\boldsymbol{x}}}\}_{k\in\Delta}$ reinterpreted from $\mathcal{Z}$; IoU threshold τ; ground truth bounding box box_{gt}.
Ensure:
Either a non-empty safe set $Q(\mathcal{Z}, \tau, box_{\text{gt}})$, or an unsafe witness $\boldsymbol{z} \in \mathcal{Z}$.
$Q \leftarrow \emptyset$.
for $i \in [n_{\boldsymbol{x}}]$ **do**
  Calculate $\tau_1(i, \mathcal{Z}, box_{\text{gt}})$ and $\tau_2(i, \mathcal{Z}, box_{\text{gt}})$ ▷ Appendix I
  $\tau(i, \mathcal{Z}, box_{\text{gt}}) \leftarrow \min(\tau_1(i, \mathcal{Z}, box_{\text{gt}}), \tau_2(i, \mathcal{Z}, box_{\text{gt}}))$ ▷ Lemma 2
  if $\tau(i, \mathcal{Z}, box_{\text{gt}}) > \tau$ **then** $Q \leftarrow Q \cup \{i\}$ ▷ Lemma 1
  end if
end for
if $Q \neq \emptyset$ **then return** (Safe, Q)
else $i \leftarrow \arg\max_{i\in[n_{\boldsymbol{x}}]}\{\tau(i, \mathcal{Z}, box_{\text{gt}})\}$
end if
return (Unsafe, $\boldsymbol{z}$) ▷ $\boldsymbol{z} \in \mathcal{Z}$ such that its corresponding box i leads to the value $\tau(i, \mathcal{Z}, box_{\text{gt}})$

5.1 Implementation Part 1 on YOLO

Consider a network $\mathrm{F} : \mathbb{R}^{d_0} \to \mathbb{R}^{d_L}$ and an input constraint $\mathcal{C}$. In Part 1 of our approach, we aim to determine the range of $\{\mathrm{F}(\boldsymbol{x})\}_{\boldsymbol{x}\in\mathcal{C}}$ with a probabilistic guarantee. We first find a constant $c_1 \in \mathbb{R}^+$ and a vector $\boldsymbol{v}_{\max} \in \mathbb{R}^{d_L}$ such that $\forall \boldsymbol{x} \in \mathcal{C}, c_1\boldsymbol{v}_{\max} \geq |\mathrm{F}(\boldsymbol{x}) - \mathrm{F}(\boldsymbol{x}^{(0)})|$ holds element-wise. Then let $\mathcal{Z} = \{\mathrm{F}(\boldsymbol{x}^{(0)}) + \boldsymbol{\epsilon} : |\boldsymbol{\epsilon}| \leq c_1\boldsymbol{v}_{\max}\}$, and it is easy to see that $\{\mathrm{F}(\boldsymbol{x})\}_{\boldsymbol{x}\in\mathcal{C}} \subset \mathcal{Z}$.

As shown in Algorithm 2, we first randomly select N_1 samples from $\mathcal{C}$, and define $(\boldsymbol{v}_{\max})_j = \max\{\max_i\{|\mathrm{F}(\boldsymbol{x}^{(i)})_j - \mathrm{F}(\boldsymbol{x}^{(0)})_j|\}, \zeta\}$, where $\zeta > 0$ is a small constant to ensure all components are positive. When finding c_1, directly solving the problem $c_1 = \min_{c\geq 0} c$ s.t. $c \in \bigcap_{\boldsymbol{x}\in\mathcal{C}}\{|\mathrm{F}(\boldsymbol{x}) - \mathrm{F}(\boldsymbol{x}^{(0)})| \leq c\boldsymbol{v}_{\max}\}$ is infeasible. Since each constraint is convex for c, by the RCP_N (Campi et al., 2009), we can get c_1 by randomly selecting N_2 samples $\{\boldsymbol{x}^{(i)}\}_{i=1}^{N_2}$ from $\mathcal{C}$, then we calculate c_1 by the following optimization problem:

$$c_1 = \min_{c\geq 0} c \quad \text{s.t.} \quad |\mathrm{F}(\boldsymbol{x}^{(i)}) - \mathrm{F}(\boldsymbol{x}^{(0)})| \leq c\boldsymbol{v}_{\max}, \quad \forall i \in [N_2]. \tag{1}$$

Proposition 1 (probabilistic guarantee for Part 1)**.** *For any $N_1 > 1$, let $\boldsymbol{v}_{\max}$ be a vector with positive components (e.g., as estimated from N_1 samples in Algorithm 2). If c_1 is computed based on this $\boldsymbol{v}_{\max}$ using $N_2 \geq [\frac{2\ln 1/\beta}{\alpha} + 2 + \frac{2\ln 2/\alpha}{\alpha}]$ samples as described in Algorithm 2, then with probability $1-\beta$, we have: $\mathrm{P}_{\boldsymbol{x}\sim\mathcal{C}}\left(|\mathrm{F}(\boldsymbol{x}) - \mathrm{F}(\boldsymbol{x}^{(0)})| \leq c_1 \cdot \boldsymbol{v}_{\max}\right) \geq 1-\alpha$, which implies $\mathrm{P}_{\boldsymbol{x}\sim\mathcal{C}}\left(\mathrm{F}(\boldsymbol{x}) \in \mathcal{Z}\right) \geq 1-\alpha$.*

Remark 3. *The probabilistic guarantee imposes no special requirements on N_1. We select $\boldsymbol{v}_{\max}$ in this way because a larger N_1 yields a tighter approximation of the true output range (Appendix D).*

5.2 Implementation Part 2 on NMS

To better illustrate the NMS verification, we use an infinite index set Δ to enumerate all possible values in $\mathcal{Z}$, i.e., $\mathcal{Z} = \{\boldsymbol{z}^k\}_{k\in\Delta}$, where each $\boldsymbol{z}^k \in \mathcal{Z}$ is a possible output vector. Each $\boldsymbol{z}^k$ can be interpreted as a set of boxes $\{box_i^k\}_{i=1}^{n_{\boldsymbol{x}}}$ according to the YOLO output format. We assume that box_i^k can be written $box_i^k = (x_i^k, y_i^k, w_i^k, h_i^k, c_i^k, p_{i_1}^k, p_{i_2}^k, \ldots, p_{i_n}^k)$. To soundly verify the NMS, we first define the safe set $Q(\mathcal{Z}, \tau, box_{\text{gt}})$, which contains indices of boxes that satisfy the NMS conditions.

Definition 2 (Safe Set)**.** The safe set $Q(\mathcal{Z}, \tau, box_{\text{gt}}) \subseteq [n_{\boldsymbol{x}}]$ and $i \in Q(\mathcal{Z}, \tau, box_{\text{gt}})$ if and only if:

(1): $\forall k \in \Delta$, $\text{Class}(box_i^k) = \text{Class}(box_{\text{gt}})$, $c_i^k \geq \iota$ and $\text{IoU}(box_i^k, box_{\text{gt}}) \geq \tau$;
(2): $\nexists k \in \Delta, n \in [n_{\boldsymbol{x}}] \setminus \{i\}$ such that $c_n^k \geq \iota$, $\text{Class}(box_n^k) = \text{Class}(box_{\text{gt}})$, $c_n^k \geq c_i^k$, $\text{IoU}(box_i^k, box_n^k) \geq \eta$, and $\text{IoU}(box_{\text{gt}}, box_n^k) < \tau$.

Then we can soundly verify the NMS by checking whether the safe set is empty.

Proposition 2 (NMS Soundness Verification)**.** *For given $\mathcal{Z}$, τ, and box_{gt}, if $Q(\mathcal{Z}, \tau, box_{\text{gt}}) \neq \emptyset$, then for $\forall k \in \Delta : \exists box_i \in \text{N}(\boldsymbol{z}^k)$, $s.t. \text{IoU}(box_i, box_{\text{gt}}) \geq \tau \wedge \text{Class}(box_i) = \text{Class}(box_{\text{gt}})$.*

According to this proposition, verification reduces to calculating the safe set. To calculate the safe set, we need the following key metric:

Definition 3 (Safe IoU Threshold)**.** The Safe IoU Threshold $\tau(i, \mathcal{Z}, box_{\text{gt}}) := \inf\{\tau' \in [0,1] | i \notin Q(\mathcal{Z}, \tau', box_{\text{gt}})\}$, where $\inf$ is the infimum operator.

The following lemmas about $\tau(i, \mathcal{Z}, box_{\text{gt}})$ can help us compute the safe set $Q(\mathcal{Z}, \tau, box_{\text{gt}})$.

Lemma 1 (Threshold Properties)**.** $\tau < \tau(i, \mathcal{Z}, box_{\text{gt}}) \Rightarrow i \in Q(\mathcal{Z}, \tau, box_{\text{gt}})$

We can obtain $\tau(i, \mathcal{Z}, box_{\text{gt}})$ by solving the following optimization problem:

Lemma 2 (Threshold Computation)**.** *The threshold can be calculated as $\tau(i, \mathcal{Z}, box_{\text{gt}}) = \min\{\tau_1(i, \mathcal{Z}, box_{\text{gt}}), \tau_2(i, \mathcal{Z}, box_{\text{gt}})\}$, where:*

$$\tau_1(i, \mathcal{Z}, box_{\text{gt}}) = \min_{k \in \Delta} \text{IoU}(box_i^k, box_{\text{gt}}) \cdot \mathbb{I}(\text{Class}(box_i^k) = q) \cdot \mathbb{I}(c_i^k \geq \iota), q = \text{Class}(box_{\text{gt}})$$

$$\tau_2(i, \mathcal{Z}, box_{\text{gt}}) = \begin{cases} \min\limits_{k \in \Delta, n \neq i} \text{IoU}(box_n^k, box_{\text{gt}}) & \textit{if } \exists (k,n) \textit{ s.t. } \mathcal{C}_{kn} = 1 \\ 1 & \textit{otherwise} \end{cases}$$

where constraint $\mathcal{C}_{kn} \equiv \mathbb{I}(c_n^k \geq \iota) \cdot \mathbb{I}(\text{Class}(box_n^k) = q) \cdot \mathbb{I}(\text{IoU}(box_i^k, box_n^k) \geq \eta) \cdot \mathbb{I}(c_n^k \geq c_i^k)$.

Appendix I shows how we encode the optimization problem in line 3 of Algorithm 3 as a mixed-integer quadratic program (MIQP) and use the Gurobi solver to solve it.

5.3 Implementation Part 3 on YOLO

Part 3 of our framework refines the initial output approximation $\mathcal{Z}$. When Part 2 detects a potential counterexample $\boldsymbol{y} \in \mathcal{Z}$, in part 3, we need to check whether $\boldsymbol{y}$ is actually reachable by F for some $\boldsymbol{x} \in \mathcal{C}$. This is done by computing $d_{\min} = \min_{\boldsymbol{x} \in \mathcal{C}} \|\text{F}(\boldsymbol{x}) - \boldsymbol{y}\|_2$.

Due to the high dimensionality, even if $\boldsymbol{y} \in \{\text{F}(\boldsymbol{x})\}_{\boldsymbol{x} \in \mathcal{C}}$, the $d_{\min}$ derived from the sampled outputs converges to zero very slowly as the sample size increases, so directly estimating $d_{\min}$ simply by taking the minimum distance from a set of sampled outputs $\{\text{F}(\boldsymbol{x}^{(i)})\}$ to $\boldsymbol{y}$ may be unreliable.

To address this, we introduce Algorithm 4, a two-step procedure for estimating $d_{\min}$ with probabilistic guarantees. **Step One (Estimating C):** This step aims to characterize the local variability of the function F within the input constraint set $\mathcal{C}$. It computes a constant C by repeatedly sampling pairs of points and observing the ratio $\frac{B_i'}{A_i' - B_i'}$. **Step Two (Estimating $d_{\min}$ using C):** Using the constant C and a new set of M_2 samples, this step estimates $d_{\min}$ for the specific target vector $\boldsymbol{y}$. The formula $d_{\min} \leftarrow \max\{\frac{B_m - C(A_m - B_m)}{1+2C}, 0\}$ leverages C to provide a more conservative estimate of the minimum distance than B_m (the minimum observed distance from the M_2 samples) alone.

Let $\text{V}(\boldsymbol{y}, d_{\min}) = \text{P}_{\boldsymbol{x} \sim \mathcal{C}}(\text{F}(\boldsymbol{x}) \in \mathcal{B}_2(\boldsymbol{y}, d_{\min}))$, and $\text{V}(\boldsymbol{y}, 0) = 0$. We use $\text{V}(\boldsymbol{y}, d_{\min})$ to measure the intersection between $\mathcal{B}_2(\boldsymbol{y}, d_{\min})$ and $\{\text{F}(\boldsymbol{x})\}_{\boldsymbol{x} \in \mathcal{C}}$. We show that with high probability, the $\text{V}(\boldsymbol{y}, d_{\min})$ is very small.

Theorem 1 (probabilistic guarantee for Part 3)**.** *For any $\alpha, \beta, \delta, \epsilon \in (0,1)$ satisfying $(1-2\epsilon)^M - \delta > 0$ and $N \cdot ((1-2\epsilon)^M - \delta) > \frac{2}{\alpha}\ln(\frac{1}{\beta}) + 2 + \frac{2}{\alpha}\ln(\frac{2}{\alpha})$, with the algorithm 4, for any $\boldsymbol{y}$, with probability at least $1 - e^{-2N\delta^2} - \beta - 2(1-\epsilon)^{M_2}$ over steps one and two, we have $\text{V}(\boldsymbol{y}, d_{\min}) \leq \alpha$.*

Remark 4. *Take $N = 3000, M = 10, M_2 = 2000$, and $\epsilon = 1/200$, $\delta = 0.1$, $\alpha, \beta = 0.0099$, then $1 - e^{-2N\delta^2} - \beta \geq 0.99$ and $1 - \alpha - 2(1-\epsilon)^{M_2} \geq 0.99$.*

We also provide a sound refinement algorithm for small networks, shown in Appendix K.

5.4 The Probabilistic Guarantee for the Entire Algorithm

We prove that the whole algorithm implemented above has a probabilistic guarantee as follows, by combining proposition 1, 2 and theorem 1:

Algorithm 4 ~~Algorithm for Part 3:~~ Counterexample Validation and Refinement (Part 3)

Require:
The neural network F; the input constraints $\mathcal{C}$, ~~$N, M, M_2 \in \mathbb{Z}^+$~~ $N, M, M_2 \in \mathbb{Z}^+$, a vector $\boldsymbol{y}$.
Ensure:
Estimate $d_{\min} = \min_{\boldsymbol{x}\in\mathcal{C}} \|\boldsymbol{y} - \mathrm{F}(\boldsymbol{x})\|_2$.
1: $C \leftarrow 0$; $\{\boldsymbol{x}^{(i)}\}_{i=1}^{N} \leftarrow$ Randomly select N samples from $\mathcal{C}$ ▷ Step One
2: **for** $i \in [N]$ **do**
3: $\{\boldsymbol{x}^{(i,j)}\}_{j=1}^{M} \leftarrow$ Randomly select M samples from $\mathcal{C}$ again ▷ Step One
4: $A'_i \leftarrow \max_{j\in[M]} \|\mathrm{F}(\boldsymbol{x}^{(i,j)}) - \mathrm{F}(\boldsymbol{x}^{(i)})\|_2$, $B'_i \leftarrow \min_{j\in[M]} \|\mathrm{F}(\boldsymbol{x}^{(i,j)}) - \mathrm{F}(\boldsymbol{x}^{(i)})\|_2$
5: $C \leftarrow \max\{C, \frac{B'_i}{A'_i - B'_i}\}$ ▷ Step One
6: **end for**
7: $\{\boldsymbol{x}^{(i)}\}_{i=1}^{M_2} \leftarrow$Select M_2 samples from $\mathcal{C}$ ▷ Step Two
8: Let $A_m \leftarrow \max_{i\in[M_2]} \|\mathrm{F}(\boldsymbol{x}^{(i)}) - \boldsymbol{y}\|_2$, $B_m \leftarrow \min_{i\in[M_2]} \|\mathrm{F}(\boldsymbol{x}^{(i)}) - \boldsymbol{y}\|_2$ ▷ Step Two
9: **return** $d_{\min} \leftarrow \max\{\frac{B_m - C(A_m - B_m)}{1+2C}, 0\}$ ▷ Step Two

Theorem 2. *Using the notation from the three algorithms above. Given $\alpha, \beta, \delta, \epsilon \in (0,1)$ satisfying $(1-2\epsilon)^M - \delta > 0$, $N \cdot ((1-2\epsilon)^M - \delta) > \frac{2}{\alpha}\ln(\frac{1}{\beta}) + 2 + \frac{2}{\alpha}\ln(\frac{2}{\alpha})$ and $N_2 \geq [\frac{2\ln 1/\beta}{\alpha} + 2 + \frac{2\ln 2/\alpha}{\alpha}]$. Then, after executing the algorithms defined above, with any κ in part 3, if for a sample $\boldsymbol{x}$, these algorithms output 'safe' after T refinement turns, then with probability at least $1 - T(e^{-2N\delta^2} + \beta + 2(1-\epsilon)^{M_2}) - \beta$ of parts one and three, we have $\mathrm{P}_{\boldsymbol{x}\sim\mathcal{C}}(\boldsymbol{x}$ is safe$) > 1 - (1+T)\alpha$.*

If we take $N_1 = 30{,}000, N_2 = 5{,}000, N = 3{,}000, M = 10, M_2 = 2{,}000, \alpha = \beta = 0.0099, \epsilon = 1/200, \delta = 0.1$, we can achieve a 98% probabilistic guarantee with 98% confidence using only 37,000 samples, which means with at least 98% confidence, the probability of an OD event occurring under the given perturbation distribution is at most 2%.

Remark 5. *Note all our theoretical guarantees depend only on the i.i.d. assumption and hold for any sampling distribution, not just uniform.*

6 Experiments

Our experiment consists of the evaluations of the bounds accuracy and the safety guarantee. Detailed experimental settings and more experimental results are provided in Appendix N to Appendix T.

Basic setting. Our experiments used the medium and large versions of the YOLOv3, YOLOv5, YOLOv8 and YOLO11 models by Ultralytics (Jocher et al., 2023). We conduct verification on the COCO dataset (Lin et al., 2014), a widely used benchmark for object detection, and randomly select 100 validation images containing more than 520 objects. We use a uniform distribution for sampling, which is a common choice in the literature (Li et al., 2022; Cohen et al., 2024). The IoU threshold $\tau \in \{0.5, 0.7\}$, the constants in NMS are $\eta = 0.45$ and $\iota = 0.25$, which are commonly used in object detection tasks. In Appendix X we also evaluate our method under other perturbation distributions (e.g., Gaussian, Salt and Pepper) on different threat models (e.g., False Appearance). We set $\zeta = 0.001$ (Alg. 2) and $\kappa = 0.01$ (Alg. 4). The perturbation radius is set to $\frac{1}{255}$ or $\frac{2}{255}$. Larger radii make the network overly fragile, enabling counterexamples to be found with very few samples, and thus eliminating meaningful differences between methods.

Baseline Selection. By Theorem 2, our method achieves a 98% probabilistic guarantee with 98% confidence using only **37,000** samples. In contrast, RCP_N requires over ~~560~~**11,000,000** samples, while DeepPAC (Li et al., 2022) requires over **100,000,000** samples and needs to solve LPs with more than $\mathbf{10^{12}}$ variables to achieve the same guarantee (see Appendix N), making both approaches impractical. Formal verification methods are also infeasible: existing tools (Cohen et al., 2024; Elboher et al., 2024) handle only 2-3 convolutional layers with 2-3 linear layers, far below the scale of YOLO, and cannot address its complex architecture or NMS. Therefore, **direct comparisons** with DeepPAC, RCP_N, and formal verification are **not feasible**. Instead, we use RCP_N with 10^6 samples (yielding weaker guarantees) as a baseline.

Table 1: Comparison of our method with RCP_N. Δ_{PGD} denotes the mean absolute difference of IoU lower bounds relative to the PGD attack. Bold values indicate the best performance.

ε	method	model	time	Δ_{PGD} $\tau=0.5$	Δ_{PGD} $\tau=0.7$	model	time	Δ_{PGD} $\tau=0.5$	Δ_{PGD} $\tau=0.7$
$\frac{1}{255}$	Ours	v3spp	**109.0**	**0.49**	**0.45**	v8m	**50.7**	**0.48**	**0.44**
	RCP_N		563.5	0.55	0.53		455.0	0.52	0.52
$\frac{2}{255}$	Ours		**106.3**	**0.48**	**0.41**		**49.6**	**0.53**	**0.45**
	RCP_N		562.9	0.58	0.54		454.7	0.59	0.55
$\frac{1}{255}$	Ours	v3	**108.5**	**0.52**	**0.46**	v8x	**132.3**	**0.49**	**0.46**
	RCP_N		561.0	0.57	0.55		591.3	0.55	0.54
$\frac{2}{255}$	Ours		**105.0**	**0.48**	**0.42**		**129.0**	**0.51**	**0.44**
	RCP_N		560.3	0.60	0.55		590.6	0.61	0.57
$\frac{1}{255}$	Ours	v5m	**43.6**	**0.42**	**0.39**	11m	**59.1**	**0.48**	**0.43**
	RCP_N		445.5	0.47	0.47		468.8	0.53	0.52
$\frac{2}{255}$	Ours		**42.8**	**0.48**	**0.42**		**58.0**	**0.50**	**0.43**
	RCP_N		444.4	0.55	0.50		467.8	0.57	0.53
$\frac{1}{255}$	Ours	v5x	**131.5**	**0.48**	**0.44**	11x	**147.2**	**0.49**	**0.45**
	RCP_N		593.8	0.54	0.54		618.1	0.54	0.54
$\frac{2}{255}$	Ours		**128.4**	**0.52**	**0.45**		**141.9**	**0.50**	**0.44**
	RCP_N		593.2	0.60	0.57		616.5	0.62	0.57

Table 2: Guarantee evaluation of our method with $\tau = 0.5$ and ~~$\varepsilon = \frac{1}{255}$~~ $\varepsilon \in \{\frac{1}{255}, \frac{2}{255}\}$ under 10^6 uniform perturbations. TPR/FPR: True/False Positive Rate. TNR/FNR: True/False Negative Rate. A detection is considered positive if verified robust by our method, and negative otherwise. Certified Robust Accuracy (CRA): percentage of detections verified robust that are indeed robust. Average Bounds Improvement (ABI): average gain in certified IoU lower bounds.

model	ε	TPR (%)	FPR (%)	TNR (%)	FNR (%)	CRA (%)	ABI
yolo11x	1/255	94.9	2.9	97.1	5.1	98.9	0.10
	2/255	85.2	1.4	98.6	14.8	99.4	0.17
yolo11m	1/255	95.0	3.0	97.0	5.0	98.6	0.10
	2/255	93.1	1.1	98.9	6.9	99.4	0.11
yolov8x	1/255	95.1	0.0	100.0	4.9	100.0	0.09
	2/255	89.7	0.6	99.4	10.3	99.7	0.14
yolov8m	1/255	97.3	2.6	97.4	2.7	98.9	0.08
	2/255	93.9	2.4	97.6	6.1	98.8	0.11
yolov5xu	1/255	94.8	0.7	99.3	5.2	99.7	0.09
	2/255	90.6	0.6	99.4	9.4	99.7	0.13
yolov5mu	1/255	96.0	2.9	97.1	4.0	98.6	0.10
	2/255	92.4	0.5	99.5	7.6	99.7	0.11
yolov3-sppu	1/255	95.6	0.0	100.0	4.4	100.0	0.08
	2/255	87.4	0.0	100.0	12.6	100.0	0.15
yolov3u	1/255	95.7	1.3	98.7	4.3	99.4	0.09
	2/255	85.8	0.0	100.0	14.2	100.0	0.16

Bounds Accuracy. Table 1 compares our method with RCP_N, showing that our approach is both faster and more accurate. In particular, it achieves a smaller mean absolute difference between IoU lower bounds and the worst-case input found by the PGD attack (Δ_{PGD}), indicating tighter certified bounds. Figure 5 further confirms this, as our bounds remain consistently closer to those of PGD.

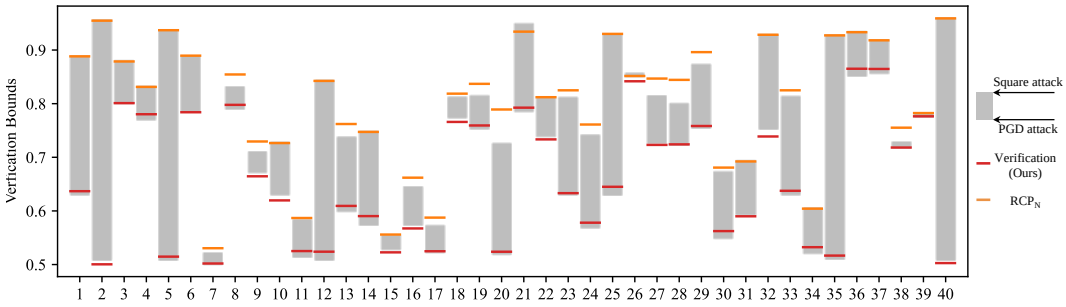


Figure 5: ~~IoU lower~~ Lower bounds of the IoU between detected boxes and their corresponding ~~ground truth~~ ground-truth boxes under our method, RCP$_N$, the square attack, and the PGD attack. Results are computed on YOLO11x with $\varepsilon = \frac{1}{255}$ and $\tau = 0.5$. Each x-axis tick corresponds to an object in the COCO dataset.

Safety Guarantee. Table 2 further shows results under 10^6 uniform perturbations: the certified robust accuracy (CRA) exceeds 98%, and the false positive rate (FPR) remains very low, consistent with theory. The true positive rate (TPR) is lower, as expected since our certification is stricter than empirical robustness. Finally, the average bounds improvement (ABI) confirms that our method yields tighter certified IoU lower bounds.

Additional Experiments. We further evaluate (i) the effect of Part 3 (Appendix P), (ii) an ablation study on hyperparameters (Appendix T), (iii) real-world applications (Appendix R), and (iv) a comparison with median smoothing (Appendix S).

7 Conclusion

This paper presents a novel ~~probabilistic framework for verifying~~ framework to provide provable probabilistic guarantees for YOLO-based object detection systems against ~~object disappearance attacks,~~ specific threats (e.g., Object Disappear, False Appearance) under various perturbation distributions, a key step toward trustworthy deployment. Our contributions are threefold: (i) a formal definition of the OD verification problem, (ii) a practical three-stage methodology that explicitly incorporates formal analysis of ~~Non-Maximum Suppression (NMS)~~NMS, and (iii) strong probabilistic guarantees for the full pipeline. Experiments on multiple YOLO architectures and ~~the COCO dataset~~ distributions show that our approach delivers reliable safety assurances and achieves tighter certified IoU bounds with far greater sample efficiency than prior methods.

Limitations and Future Directions: Our method relies on an assumed distribution of input perturbations, a limitation inherent to the PAC framework. Developing verification methods for other types of attacks remains an important direction for future work. Another valuable direction involves leveraging adversarial attack strategies to further refine Stages 1 and 3, alongside investigating more efficient methods for interval estimation and refinement.

References

Brendon G. Anderson and Somayeh Sojoudi. Data-driven certification of neural networks with random input noise. *IEEE Transactions on Control of Network Systems*, 10(1):249–260, 2023. doi: 10.1109/TCNS.2022.3199148.

Stanley Bak. nnenum: Verification of ReLU neural networks with optimized abstraction refinement. In *Proceedings of the 13th International Symposium on NASA Formal Methods (NFM)*, pp. 19–36. Springer, 2021.

Teodora Baluta, Zheng Leong Chua, Kuldeep S. Meel, and Prateek Saxena. Scalable quantitative verification for deep neural networks. In *2021 IEEE/ACM 43rd International Conference on Software Engineering (ICSE)*, pp. 312–323, 2021. doi: 10.1109/ICSE43902.2021.00039.

Peter Blohm, Patrick Indri, Thomas Gärtner, and SAGAR MALHOTRA. Probably approximately global robustness certification. In *Forty-second International Conference on Machine Learning*, 2025. URL https://openreview.net/forum?id=UKHlXpiFMy.

Alexey Bochkovskiy, Chien-Yao Wang, and Hong-Yuan Mark Liao. Yolov4: Optimal speed and accuracy of object detection. *CoRR*, abs/2004.10934, 2020a. URL https://arxiv.org/abs/2004.10934.

Alexey Bochkovskiy, Chien-Yao Wang, and Hong-Yuan Mark Liao. Yolov4: Optimal speed and accuracy of object detection, 2020b.

David Boetius, Stefan Leue, and Tobias Sutter. Solving probabilistic verification problems of neural networks using branch and bound. In *Forty-second International Conference on Machine Learning*, 2025. URL https://openreview.net/forum?id=suZ1pNdKrV.

Christopher Brix, Stanley Bak, Changliu Liu, and Taylor T. Johnson. The fourth international verification of neural networks competition (vnn-comp 2023): Summary and results, 2023.

Christopher Brix, Stanley Bak, Taylor T. Johnson, and Haoze Wu. The fifth international verification of neural networks competition (vnn-comp 2024): Summary and results, 2024. URL https://arxiv.org/abs/2412.19985.

Marco C Campi, Simone Garatti, and Maria Prandini. The scenario approach for systems and control design. *Annual Reviews in Control*, 33(2):149–157, 2009.

Luca Cardelli, Marta Kwiatkowska, Luca Laurenti, and Andrea Patane. Robustness guarantees for bayesian inference with gaussian processes. In *Proceedings of the AAAI conference on artificial intelligence*, volume 33, pp. 7759–7768, 2019.

Nicholas Carlini and David A. Wagner. Towards evaluating the robustness of neural networks. In *2017 IEEE Symposium on Security and Privacy, SP 2017, San Jose, CA, USA, May 22-26, 2017*, pp. 39–57. IEEE Computer Society, 2017. doi: 10.1109/SP.2017.49. URL https://doi.org/10.1109/SP.2017.49.

Yanbo Chen and Weiwei Liu. A theory of transfer-based black-box attacks: explanation and implications. *Advances in Neural Information Processing Systems*, 36, 2024.

Ping-yeh Chiang, Michael Curry, Ahmed Abdelkader, Aounon Kumar, John Dickerson, and Tom Goldstein. Detection as regression: Certified object detection with median smoothing. *Advances in Neural Information Processing Systems*, 33:1275–1286, 2020.

Jung Im Choi and Qing Tian. Adversarial attack and defense of yolo detectors in autonomous driving scenarios. In *2022 IEEE Intelligent Vehicles Symposium (IV)*, pp. 1011–1017, 2022. doi: 10.1109/IV51971.2022.9827222.

Jeremy Cohen, Elan Rosenfeld, and Zico Kolter. Certified adversarial robustness via randomized smoothing. In *international conference on machine learning*, pp. 1310–1320. PMLR, 2019.

Noémie Cohen, Mélanie Ducoffe, Ryma Boumazouza, Christophe Gabreau, Claire Pagetti, Xavier Pucel, and Audrey Galametz. Verification for object detection–ibp iou. *arXiv preprint arXiv:2403.08788*, 2024.

N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, volume 1, pp. 886–893 vol. 1, 2005. doi: 10.1109/CVPR.2005.177.

Yinpeng Dong, Fangzhou Liao, Tianyu Pang, Hang Su, Jun Zhu, Xiaolin Hu, and Jianguo Li. Boosting adversarial attacks with momentum. In *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018*, pp. 9185–9193. Computer Vision Foundation / IEEE Computer Society, 2018. doi: 10.1109/CVPR.2018.00957.

Ruediger Ehlers. Formal verification of piece-wise linear feed-forward neural networks. In *Automated Technology for Verification and Analysis: 15th International Symposium, ATVA 2017, Pune, India, October 3–6, 2017, Proceedings 15*, pp. 269–286. Springer, 2017.

Yizhak Y. Elboher, Avraham Raviv, Yael Leibovich Weiss, Omer Cohen, Roy Assa, Guy Katz, and Hillel Kugler. Formal verification of deep neural networks for object detection, 2024. URL https://arxiv.org/abs/2407.01295.

Ali Farhadi and Joseph Redmon. Yolov3: An incremental improvement. In *Computer vision and pattern recognition*, volume 1804, pp. 1–6. Springer Berlin/Heidelberg, Germany, 2018.

Ross Girshick. Fast r-cnn. In *2015 IEEE International Conference on Computer Vision (ICCV)*, pp. 1440–1448, 2015. doi: 10.1109/ICCV.2015.169.

Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 580–587, 2014.

Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. In Yoshua Bengio and Yann LeCun (eds.), *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015. URL http://arxiv.org/abs/1412.6572.

Sven Gowal, Krishnamurthy Dj Dvijotham, Robert Stanforth, Rudy Bunel, Chongli Qin, Jonathan Uesato, Relja Arandjelovic, Timothy Mann, and Pushmeet Kohli. Scalable verified training for provably robust image classification. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 4842–4851, 2019.

David Haussler and Emo Welzl. ε-nets and simplex range queries. *Discrete & Computational Geometry*, 2:127–151, 1987. URL https://api.semanticscholar.org/CorpusID:27638326.

Patrick Henriksen and Alessio Lomuscio. Efficient neural network verification via adaptive refinement and adversarial search. In *ECAI 2020*, pp. 2513–2520. IOS Press, 2020.

Jung Im Choi and Qing Tian. Adversarial attack and defense of yolo detectors in autonomous driving scenarios. In *2022 IEEE Intelligent Vehicles Symposium (IV)*, pp. 1011–1017. IEEE, 2022.

Glenn Jocher, Jing Qiu, and Ayush Chaurasia. Ultralytics YOLO. In *github*, January 2023. URL https://github.com/ultralytics/ultralytics.

Guy Katz, Clark Barrett, David L Dill, Kyle Julian, and Mykel J Kochenderfer. Reluplex: An efficient smt solver for verifying deep neural networks. In *Proceedings of the 29th International Conference on Computer Aided Verification (CAV)*, pp. 97–117. Springer, 2017.

Guy Katz, Derek A. Huang, Duligur Ibeling, Kyle Julian, Christopher Lazarus, Rachel Lim, Parth Shah, Shantanu Thakoor, Haoze Wu, Aleksandar Zeljic, David L. Dill, Mykel J. Kochenderfer, and Clark W. Barrett. The marabou framework for verification and analysis of deep neural networks. In Isil Dillig and Serdar Tasiran (eds.), *Computer Aided Verification - 31st International Conference, CAV 2019, New York City, NY, USA, July 15-18, 2019, Proceedings, Part I*, volume 11561 of *Lecture Notes in Computer Science*, pp. 443–452. Springer, 2019. doi: 10.1007/978-3-030-25540-4_26. URL https://doi.org/10.1007/978-3-030-25540-4_26.

Haitham Khedr, James Ferlez, and Yasser Shoukry. Peregrinn: Penalized-relaxation greedy neural network verifier. In *Computer Aided Verification: 33rd International Conference, CAV 2021, Virtual Event, July 20–23, 2021, Proceedings, Part I 33*, pp. 287–300. Springer, 2021.

Suhas Kotha, Christopher Brix, J Zico Kolter, Krishnamurthy Dvijotham, and Huan Zhang. Provably bounding neural network preimages. *Advances in Neural Information Processing Systems*, 36: 80270–80290, 2023.

Panagiotis Kouvaros and Alessio Lomuscio. Towards scalable complete verification of ReLU neural networks via dependency-based branching. In *IJCAI*, pp. 2643–2650, 2021.

Huichen Li, Xiaojun Xu, Xiaolu Zhang, Shuang Yang, and Bo Li. Qeba: Query-efficient boundary-based blackbox attack. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 1221–1230, 2020.

Renjue Li, Pengfei Yang, Cheng-Chao Huang, Youcheng Sun, Bai Xue, and Lijun Zhang. Towards practical robustness analysis for dnns based on pac-model learning. In *Proceedings of the 44th International Conference on Software Engineering*, pp. 2189–2201, 2022.

Zuoxin Li, Lu Yang, and Fuqiang Zhou. Fssd: feature fusion single shot multibox detector. *arXiv preprint arXiv:1712.00960*, 2017.

Tao Lin, Lijia Yu, Gaojie Jin, Renjue Li, Peng Wu, and Lijun Zhang. Out-of-bounding-box triggers: A stealthy approach to cheat object detectors. In Aleš Leonardis, Elisa Ricci, Stefan Roth, Olga Russakovsky, Torsten Sattler, and Gül Varol (eds.), *Computer Vision – ECCV 2024*, pp. 269–287, Cham, 2025. Springer Nature Switzerland. ISBN 978-3-031-72848-8.

Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V 13*, pp. 740–755. Springer, 2014.

Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C Berg. Ssd: Single shot multibox detector. In *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11–14, 2016, Proceedings, Part I 14*, pp. 21–37. Springer, 2016.

Yanpei Liu, Xinyun Chen, Chang Liu, and Dawn Song. Delving into transferable adversarial examples and black-box attacks. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net, 2017. URL https://openreview.net/forum?id=Sys6GJqxl.

Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net, 2018. URL https://openreview.net/forum?id=rJzIBfZAb.

Ravi Mangal, Aditya V. Nori, and Alessandro Orso. Robustness of neural networks: a probabilistic and practical approach. In *Proceedings of the 41st International Conference on Software Engineering: New Ideas and Emerging Results*, ICSE-NIER '19, pp. 93–96. IEEE Press, 2019. doi: 10.1109/ICSE-NIER.2019.00032. URL https://doi.org/10.1109/ICSE-NIER.2019.00032.

P. Massart. The Tight Constant in the Dvoretzky-Kiefer-Wolfowitz Inequality. *The Annals of Probability*, 18(3):1269 – 1283, 1990. doi: 10.1214/aop/1176990746. URL https://doi.org/10.1214/aop/1176990746.

Michael Naaman. On the tight constant in the multivariate dvoretzky-kiefer-wolfowitz inequality. *Statistics & Probability Letters*, 173:109088, 2021. ISSN 0167-7152. doi: https://doi.org/10.1016/j.spl.2021.109088. URL https://www.sciencedirect.com/science/article/pii/S016771522100050X.

Alexander Neubeck and Luc Van Gool. Efficient non-maximum suppression. In *18th international conference on pattern recognition (ICPR'06)*, volume 3, pp. 850–855. IEEE, 2006.

Sangdon Park, Osbert Bastani, Nikolai Matni, and Insup Lee. PAC confidence sets for deep neural networks via calibrated prediction. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020. URL https://openreview.net/forum?id=BJxVI04YvB.

J Redmon. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016.

Joseph Redmon and Ali Farhadi. Yolo9000: better, faster, stronger. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 7263–7271, 2017.

Shaoqing Ren. Faster r-cnn: Towards real-time object detection with region proposal networks. *arXiv preprint arXiv:1506.01497*, 2015.

Gagandeep Singh, Timon Gehr, Markus Püschel, and Martin Vechev. An abstract domain for certifying neural networks. *Proceedings of the ACM on Programming Languages*, 3(POPL):1–30, 2019.

Hoang-Dung Tran, Sungwoo Choi, Hideki Okamoto, Bardh Hoxha, Georgios Fainekos, and Danil Prokhorov. Quantitative verification for neural networks using probstars. In *Proceedings of the 26th ACM International Conference on Hybrid Systems: Computation and Control*, HSCC '23, New York, NY, USA, 2023. Association for Computing Machinery. ISBN 9798400700330. doi: 10.1145/3575870.3587112. URL https://doi.org/10.1145/3575870.3587112.

Paul Viola and Michael Jones. Rapid object detection using a boosted cascade of simple features. In *Proceedings of the 2001 IEEE computer society conference on computer vision and pattern recognition. CVPR 2001*, volume 1, pp. I–I. Ieee, 2001.

Shiqi Wang, Kexin Pei, Justin Whitehouse, Junfeng Yang, and Suman Jana. Efficient formal safety analysis of neural networks. *Advances in neural information processing systems*, 31, 2018a.

Shiqi Wang, Kexin Pei, Justin Whitehouse, Junfeng Yang, and Suman Jana. Formal security analysis of neural networks using symbolic intervals. In *27th USENIX Security Symposium (USENIX Security 18)*, pp. 1599–1614, 2018b.

Stefan Webb, Tom Rainforth, Yee Whye Teh, and M. Pawan Kumar. Statistical verification of neural networks. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=S1xcx3C5FX.

Lily Weng, Huan Zhang, Hongge Chen, Zhao Song, Cho-Jui Hsieh, Luca Daniel, Duane Boning, and Inderjit Dhillon. Towards fast computation of certified robustness for ReLu networks. In *International Conference on Machine Learning*, pp. 5276–5285. PMLR, 2018.

Lily Weng, Pin-Yu Chen, Lam Nguyen, Mark Squillante, Akhilan Boopathy, Ivan Oseledets, and Luca Daniel. Proven: Verifying robustness of neural networks with a probabilistic approach. In *International Conference on Machine Learning*, pp. 6727–6736. PMLR, 2019.

Kaidi Xu, Huan Zhang, Shiqi Wang, Yihan Wang, Suman Jana, Xue Lin, and Cho-Jui Hsieh. Fast and complete: Enabling complete neural network verification with rapid and massively parallel incomplete verifiers. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net, 2021. URL https://openreview.net/forum?id=nVZtXBI6LNn.

Greg Yang, Tony Duan, J Edward Hu, Hadi Salman, Ilya Razenshteyn, and Jerry Li. Randomized smoothing of all shapes and sizes. In *International conference on machine learning*, pp. 10693–10705. PMLR, 2020.

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.

Huan Zhang, Shiqi Wang, Kaidi Xu, Linyi Li, Bo Li, Suman Jana, Cho-Jui Hsieh, and J. Zico Kolter. General cutting planes for bound-propagation-based neural network verification. In *NeurIPS*, 2022a.

Huan Zhang, Shiqi Wang, Kaidi Xu, Yihan Wang, Suman Jana, Cho-Jui Hsieh, and Zico Kolter. A branch and bound framework for stronger adversarial attacks of ReLU networks. In *International Conference on Machine Learning*, pp. 26591–26604. PMLR, 2022b.

Zhong-Qiu Zhao, Peng Zheng, Shou-tao Xu, and Xindong Wu. Object detection with deep learning: A review. *IEEE transactions on neural networks and learning systems*, 30(11):3212–3232, 2019.

Zhengxia Zou, Keyan Chen, Zhenwei Shi, Yuhong Guo, and Jieping Ye. Object detection in 20 years: A survey. *Proceedings of the IEEE*, 111(3):257–276, 2023.

A APPENDIX

This appendix provides more related work, supplementary discussions, proofs, and experiments to support the main text. We organize the appendix as follows:

- **Section B**: Summarizes additional related work.
- **Section C**: Provides the proof of Proposition 1 (probabilistic guarantee for part 1).
- **Section D**: Explains the role of sample size in Part 1 Step 1 and its effect on $\boldsymbol{v}_{\max}$.
- **Section E**: Gives the proof of Proposition 2 (NMS Soundness Verification).
- **Section F**: Proves Lemma 1 (threshold properties).
- **Section G**: Proves Lemma 2 (explicit threshold computation formulas).
- **Section H**: Defines and estimates the verification bound for object detection and provides proofs for Lemmas 4 and 5.
- **Section I**: Details the verification procedure for Non-Maximum Suppression, including abstract box construction and IoU bound computation.
- **Section J**: Proves Theorem 1 (probabilistic guarantee for Part 3).
- **Section K**: We propose a strict sound gradient-based refinement algorithm to implement Part Three on small-scale networks(Theorem 4).
- **Section L**: Provides the proof of Theorem 4 based on dual formulations.
- **Section M**: Gives the proof of Theorem 2, combining probabilistic guarantees across all parts.
- **Section N**: Reports detailed experimental settings, sample number calculations, and server configuration.
- **Section O**: Discuss the efficiency of the NMS verification process.
- **Section P**: Shows the effectiveness of Part 3 refinement for YOLO and CNNs.
- **Section Q**: Shows time-verified box comparison between our method and RCP_N.
- **Section R**: Demonstrates our method's effectiveness on real-world images.
- **Section S**: Compares our method against median smoothing under Gaussian noise.
- **Section T**: Presents an ablation study on parameters η and ι.
- **Section U**: Discusses the broader impact of our verification method.
- **Section V** : LLM Usage Statement.
- **Section W**: Lists and explains all hyperparameters used in our algorithms.
- **Section X**: Provides details on how to adapt our method to other attacks beyond OD attacks.

B ADDITIONAL RELATED WORK

~~**Adversarial attacks.**~~

Adversarial attacks. Adversarial methods induce misclassification through imperceptible perturbations. White-box attacks exploit gradient information (Goodfellow et al., 2015; Madry et al., 2018; Carlini & Wagner, 2017) and can be adapted to OD attacks (Choi & Tian, 2022; Li et al., 2020). Black-box attacks use transferability (Chen & Liu, 2024) or query-based optimization (Li et al., 2020); these are more practical but may be computationally costly. Using adversarial attacks in isolation only demonstrates non-robustness if the attack is successful, whereas our method provides a rigorous probabilistic certificate of robustness. A network that is robust with high probability under common perturbations(e.g. sensor noise) is acceptable for practical deployment, whereas the strict requirement of complete robustness in a neighborhood often leads to a significant drop in network performance. Thus focus is on providing probabilistic robustness under realistic perturbations such as sensor noise rather than adversarial attacks (which often represent worst-case scenarios).

Bound Estimate Methods. There are several classical methods for estimating probabilistic output bounds of neural networks, including DKW-based confidence regions (Massart, 1990; Naaman, 2021), ERM-based hyper-rectangles, and ϵ-nets (Haussler & Welzl, 1987; Blohm et al., 2025). While these methods are theoretically robust, their sample complexity typically scales with the output dimension d_L (e.g., $\tilde{O}(d_L/\epsilon)$ for ϵ-nets). Given the extremely high-dimensional output space of YOLO networks, dimension-dependent bounds like those from ϵ-nets or DKW would be computationally infeasible. We therefore focus on dimension-independent PAC bounds, making estimation feasible even for high-dimensional outputs.

PAC with Attack. There are several PAC verification methods that incorporate adversarial attacks to refine the estimated output bounds (Blohm et al., 2025; Li et al., 2022; Baluta et al., 2021). For example, Blohm et al. (2025) evaluates the robustness of individual points via a local robustness oracle (such as PGD or LiRPA) and leverages ϵ-net sampling methods to provide high-probability statistical guarantees for global robustness. Combining attacks with PAC verification is an interesting direction, we will explore this in future work.

C THE PROOF OF PROPOSITION 1

Proposition (probabilistic guarantee for part 1). *For any $N_1 > 1$, let $\boldsymbol{v}_{\max}$ be a vector with positive components (e.g., as estimated from N_1 samples in Algorithm 2). If c_1 is computed based on this $\boldsymbol{v}_{\max}$ using $N_2 \geq [\frac{2\ln 1/\beta}{\alpha} + 2 + \frac{2\ln 2/\alpha}{\alpha}]$ samples as described in Algorithm 2, then with probability $1-\beta$, we have: $\mathrm{P}_{\boldsymbol{x}\sim\mathcal{C}}\left(|\mathrm{F}(\boldsymbol{x}) - \mathrm{F}(\boldsymbol{x}^{(0)})| \leq c_1 \cdot \boldsymbol{v}_{\max}\right) \geq 1-\alpha$, which implies $\mathrm{P}_{\boldsymbol{x}\sim\mathcal{C}}\left(\mathrm{F}(\boldsymbol{x}) \in \mathcal{Z}\right) \geq 1-\alpha$.*

This proposition can be directly obtained by classic method RCP_N, which is introduced below.

C.1 CLASSIC METHOD FOR PROBABILITY SAMPLING.

We begin by introducing a well-known ~~method,~~ result from the RCP_N method (Campi et al., 2009), which forms the basis of our approach in this section. Consider the following optimization problem with ~~possibly infinite constraints:~~

an infinite number of constraints:

$$\min_{\boldsymbol{x}\in\mathbb{R}^d} \boldsymbol{a}^\top \boldsymbol{x} + b \text{ s.t. } \boldsymbol{x} \in \bigcap_{\delta\in\Delta} \mathcal{X}_\delta, \tag{2}$$

~~Where~~ where Δ is ~~a index set~~an index set, and $\mathcal{X}_\delta \subseteq \mathbb{R}^d$ ~~is the~~ denotes the constraint set corresponding to the index δ ~~-th constraint set~~.

Since ~~the constraints are infinite~~there are infinitely many constraints, we can not solve the problem directly. Thus we ~~consider~~ sample the constraints $\{\delta_i\}_{i=1}^N$ from Δ ~~, and estimate the possibility that the optimal value of the optimization problem equation 3with the constraints $\{\mathcal{X}_{\delta_i}\}_{i=1}^N$ is the optimal value of the problem equation 2 with the constraints $\{\mathcal{X}_\delta\}_{\delta\in\Delta}$.~~

according to a probability distribution $\mathcal{Q}$ on Δ, and consider the sampled problem equation 3. We are interested in quantifying how likely it is that the optimal solution of the sampled problem is (almost) feasible for the original problem.

$$\min_{\boldsymbol{x}\in\mathbb{R}^d} \boldsymbol{a}^\top \boldsymbol{x} + b \text{ s.t. } \boldsymbol{x} \in \bigcap_{\delta_i, i\in[N]} \bigcap_{i\in[N]} \mathcal{X}_{\delta_i}. \tag{3}$$

When ~~X_δ~~ $\mathcal{X}_\delta$ is convex for every $\delta \in \Delta$, we have ~~that:~~ the following result. If $\mathcal{Q}$ is a distribution defined on Δ, and ~~$N \geq [\frac{2\ln(1/\beta)}{\alpha} + 2d + \frac{2d\ln(2/\alpha)}{\alpha}]$, thenwith probability~~

$$N \geq \left\lceil \frac{2\ln(1/\beta)}{\alpha} + 2d + \frac{2d\ln(2/\alpha)}{\alpha} \right\rceil,$$

then, with probability at least $1-\beta$ ~~of $\{\delta_i\}_{i=1}^N \sim \mathcal{Q}$~~over the i.i.d. samples $\{\delta_i\}_{i=1}^N \sim \mathcal{Q}^N$, if the ~~following~~ optimization problem equation 3 has a unique solution $\boldsymbol{x}_{\min}$, ~~such~~ this solution $\boldsymbol{x}_{\min}$ satisfies ~~$\mathrm{P}_{\delta\sim\Delta}(\boldsymbol{x}_{\min} \in \mathcal{X}_\delta) \geq 1-\alpha.$~~

$$\mathbb{P}_{\delta\sim\mathcal{Q}}\left(\boldsymbol{x}_{\min} \in \mathcal{X}_\delta\right) \geq 1-\alpha.$$

A classic application ~~for~~ of RCP_N is to ~~find the minimum (or maximum) value~~ compute a high-probability upper bound of a function ~~f when input in a constraint~~ $f(t)$ over a domain Δ~~, which can be written as :~~. This can be formulated as an optimization problem with a decision variable $x \in \mathbb{R}$ (i.e., dimension $d=1$):

$$\min_{\text{~~x~~}\in\mathbb{R}, x\in\mathbb{R}} x \quad \text{s.t.} \text{~~x~~} \quad x \geq f(t),\ \forall t \in \Delta. \tag{4}$$

~~By~~ By applying the RCP_N result with $d=1$, we know that ~~when $N \geq \lceil \frac{2\ln(1/\beta)}{\alpha} + 2d + \frac{2d\ln(2/\alpha)}{\alpha} \rceil$ and we select~~ if we draw N samples $\{t_i\}_{i=1}^N$ ~~in~~ from Δ ~~, then with probability~~ according to $\mathcal{Q}$, where

$$N \geq \left\lceil \frac{2\ln(1/\beta)}{\alpha} + 2 + \frac{2\ln(2/\alpha)}{\alpha} \right\rceil,$$

then, with probability at least $1-\beta$ ~~, there are $\mathrm{P}_{t\sim\Delta}(f(t) \leq \max_i\{f(t_i)\}) \geq 1-\alpha.$~~ over the sampling, we have $\mathbb{P}_{t\sim\mathcal{Q}}(f(t) \leq \max_{i=1,\ldots,N} f(t_i)) \geq 1-\alpha$.

D ~~About the Effect~~ Analysis of the Sample ~~Number~~ Size Effect in Part 1 Step 1

In this section, we ~~will demonstrate why we choose~~ justify the choice of $\boldsymbol{v}_{\max}$ as ~~show in the algorithm 2. Then main result is that we want~~ presented in Algorithm 2. The main objective is to ensure that the range $\mathcal{Z}$ ~~we obtained by algorithm should not far beyond the real~~ obtained by Algorithm 2 does not significantly exceed the actual range $\{\mathrm{F}(\boldsymbol{x})\}_{\boldsymbol{x}\in\mathcal{C}}$.

~~And algorithm 2 can guarantee such thing under some assumption as shown in the below:~~ Algorithm 2 guarantees this property under certain assumptions, as detailed in the following proposition.

Proposition 3. *Let $|(\mathrm{F}(\boldsymbol{x}))_i - (\mathrm{F}(\boldsymbol{x}^{(0)}))_i| \sim N_i$ when $x \sim \mathcal{C}$, and $v_i \in \mathbb{R}_+$ is the minimum value such that $\mathrm{P}_{x\sim N_i}(x \leq v_i) = 1$.*

If $\alpha_1^i \leq \alpha_2^i \leq 1$ and β_1, β_2 satisfy that: $\mathrm{P}_{x\sim N_i}(x \leq \alpha_1^i v_i) = \beta_1$ and $\mathrm{P}_{x\sim N_i}(x \leq \alpha_2^i v_i) = \beta_2$ for any $i \in [d_L]$.

Let $z_i = (c_1 \boldsymbol{v}_{\max})_i$ where c_1 and $\boldsymbol{v}_{\max}$ are obtained by algorithm 2, then we have that: $\frac{z_j}{v_j} \leq \max_i\{\frac{\alpha_2^i}{\alpha_1^i}\}\alpha_2^j$ for any $j \in [d_L]$ with probability $1 - d_L(1+\beta_1^{N_1} - \beta_2^{N_1}) - d_L(1+\beta_1^{N_2} - \beta_2^{N_2})$.

Proof. ~~Easy to see that , for any $i \in [d_L]$, for the random selected~~ Observe that for any coordinate $i \in [d_L]$, given the N_1 ~~points $\{x_j\}$ in~~ randomly selected points $\{x_k\}$ from $\mathcal{C}$ in ~~algorithm 2, there are $\mathrm{P}(\alpha_2^i v_i \geq \max_j(x_j)_i \geq \alpha_1^i v_i) \geq -\beta_1^{N_1} + \beta_2^{N_1}$ for any $i \in [d_L]$. So with probability $1 - d_L(1+\beta_1^{N_1} - \beta_2^{N_1})$, there are $\alpha_2^i v_i \geq \max_j(x_j)_i \geq \alpha_1^i v_i$ stand for any $i \in [d_L]$~~Algorithm 2, we have:

$$\mathrm{P}\left(\alpha_2^i v_i \geq \max_k (x_k)_i \geq \alpha_1^i v_i\right) \geq \beta_2^{N_1} - \beta_1^{N_1}.$$

Applying the union bound, with probability $1 - d_L(1-(\beta_2^{N_1} - \beta_1^{N_1})) = 1 - d_L(1+\beta_1^{N_1} - \beta_2^{N_1})$, the condition $\alpha_2^i v_i \geq \max_k(x_k)_i \geq \alpha_1^i v_i$ holds for all $i \in [d_L]$.

~~Similar~~Similarly, for the N_2 points ~~$\{x'_j\}$ random selected in the algorithm~~ $\{x'_k\}$ randomly selected in Algorithm 2, with probability ~~$1 - d_L(1+\beta_1^{N_2} - \beta_2^{N_2})$, there are $\alpha_2^i v_i \geq \max_j(x'_j)_i \geq \alpha_1^i v_i$ stand~~

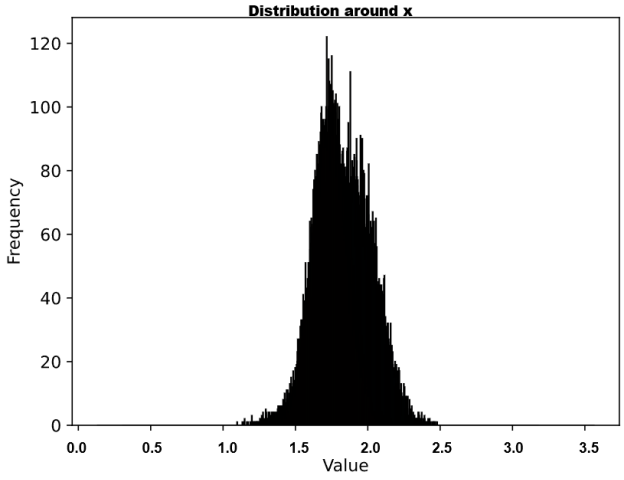


Figure 6: This is a picture about the distribution around a sample x. When we take $\alpha_1 = 2.3/3.5$ and $\alpha_2 = 2.8/3.5$, by such figure, we have that $\beta_1 \leq 0.99$ but $\beta_2 \approx 1$. Thus $(\alpha_2/\alpha_1)\alpha_2 \approx 0.97$. Hence, if there are $N_1 = N_2 = 3000$, then $1 - d_L(1 + \beta_1^{N_1} - \beta_2^{N_1}) - d_L(1 + \beta_1^{N_2} - \beta_2^{N_2}) \geq 0.99$ when $d_L \leq 10^7$.

~~for any $i \in [d_L]$~~$1 - d_L(1 + \beta_1^{N_2} - \beta_2^{N_2})$, the condition $\alpha_2^i v_i \geq \max_k(x'_k)_i \geq \alpha_1^i v_i$ holds for all $i \in [d_L]$.

~~Hence~~Consequently, based on the ~~algorithm 2, we know that $c_1 \leq \max_i\{\alpha_2^i/\alpha_1^i\}$ when the above results hold for any $i \in [d_L]$. Hence, there are $(c_1 \boldsymbol{v}_{\max})_j \leq \max_i\{\frac{\alpha_2^i}{\alpha_1^i}\}\alpha_2^j v_j$ any $j \in [d_L]$, which is what we want~~construction in Algorithm 2, if the above events occur, we have $c_1 \leq \max_i\{\alpha_2^i/\alpha_1^i\}$. Therefore, for any $j \in [d_L]$, implies:

$$(c_1 \boldsymbol{v}_{\max})_j \leq \max_i \left\{ \frac{\alpha_2^i}{\alpha_1^i} \right\} \alpha_2^j v_j,$$

which completes the proof. □

Based on our observations of ~~many~~ numerous neural network outputs, we ~~have found that the outputs of neural networks~~ find that the output values are highly likely to be concentrated ~~in a certain~~ within a specific region. An example is ~~given in the figure~~ provided in Figure 6. Based on this ~~, there can be $\alpha_1 \approx \alpha_2$ but $1 \approx \beta_2$ and $1 > \beta_1$. Hence, because $\alpha_1 \approx \alpha_2$, we know~~ observation, it is possible that $\alpha_1 \approx \alpha_2$ while $\beta_2 \approx 1$ and $\beta_1 < 1$. Consequently, since $\alpha_1 \approx \alpha_2$, we can infer that each dimension of $\mathcal{Z}$ will not ~~far~~ extend significantly beyond v_i~~; because $1 \approx \beta_2$ and $1 > \beta_1$, we know $1 - d_L(1 + \beta_1^{N_1} - \beta_2^{N_1}) - d_L(1 + \beta_1^{N_2} - \beta_2^{N_2}) \approx 1$, which is what we want.~~ . Furthermore, since $\beta_2 \approx 1$ and $\beta_1 < 1$, the probability term satisfies:

$$1 - d_L(1 + \beta_1^{N_1} - \beta_2^{N_1}) - d_L(1 + \beta_1^{N_2} - \beta_2^{N_2}) \approx 1.$$

However, in ~~reality~~practice, we cannot accurately estimate α_i and β_i~~, so~~ . Therefore, the choice of N_1 is ~~mainly determined by experiments. The above theorem is only used to support~~ primarily determined empirically. The theorem above serves to theoretically justify the accuracy of the region $\mathcal{Z}$ ~~we found~~ obtained when N_1 is sufficiently large.

E THE PROOF OF PROPOSITION 2

Proposition (NMS Soundness Verification). *For given $\mathcal{Z}$, τ, and box_{gt}, if $Q(\mathcal{Z}, \tau, box_{\text{gt}}) \neq \emptyset$, then for $\forall k \in \Delta : \exists box_i \in \text{N}(\boldsymbol{z}^k)$, s.t.$\text{IoU}(box_i, box_{\text{gt}}) \geq \tau \wedge \text{Class}(box_i) = \text{Class}(box_{\text{gt}})$.*

We ~~give a lemmaat first~~first present a lemma.

Lemma 3. *For a given $\boldsymbol{z} \in \mathcal{Z}$, if there is no $box_i \in \text{N}(\boldsymbol{z})$ satisfies that $\text{IoU}(box_i, box_{\text{gt}}) \geq \tau$ and ~~$\text{Class}(box_i) = q$, where $q = \text{Class}(box_{\text{gt}})$~~ $\text{Class}(box_i) = \text{Class}(box_{\text{gt}})$ then:*
For any $box_i \in \boldsymbol{z}$ with $c_i \geq \iota$, $\text{Class}(box_i) = q$ and $\text{IoU}(box_i, box_{\text{gt}}) \geq \tau$, there exists another $box_j \in \boldsymbol{z}$ such that $c_j \geq \iota$, $\text{Class}(box_j) = \text{Class}(box_i)$, $c_i \leq c_j$, $\text{IoU}(box_j, box_{\text{gt}}) < \tau$ and $\text{IoU}(box_i, box_j) \geq \eta$.

Proof. Assume that there exists a $box_i \in \boldsymbol{z}$ with $c_i \geq \iota$, $\text{Class}(box_i) = \text{Class}(box_{\text{gt}})$, and $\text{IoU}(box_i, box_{\text{gt}}) \geq \tau$. By ~~assumption, there must be~~ the assumption of the lemma, it implies that $box_i \notin \text{N}(\boldsymbol{z})$. ~~According to~~

According to condition (n2) in Section 3.1, there must exist a $box_j \in \text{N}(\boldsymbol{z})$ such that $\text{IoU}(box_i, box_j) \geq \eta$, $c_i \leq c_j$, and $\text{Class}(box_i) = \text{Class}(box_j) = \text{Class}(box_{\text{gt}})$. By condition (n1) in ~~Sec.~~ Section 3.1, $box_j \in \text{N}(\boldsymbol{z})$ implies that $c_j \geq \iota$.

However, by the assumptions, there is no $box_j \in \text{N}(\boldsymbol{z})$ that satisfies ~~$\text{Class}(box_j) = q$~~ $\text{Class}(box_j) = \text{Class}(box_{\text{gt}})$ and $\text{IoU}(box_j, box_{\text{gt}}) \geq \tau$~~, and~~. Since we have shown that $\text{Class}(box_j) = \text{Class}(box_{\text{gt}})$ above, ~~so we can get~~ it follows that $\text{IoU}(box_j, box_{\text{gt}}) < \tau$. This completes the proof. □

Using ~~such a~~ this lemma, we can directly ~~get the proposition~~ prove Proposition 2.

Proof. Assume that $Q(\mathcal{Z}, \tau, box_{\text{gt}}) \neq \emptyset$, but for some $k \in \Delta$~~such~~, the stated result does not hold.

~~Base on the~~ Based on condition (1) of the definition of $Q(\mathcal{Z}, \tau, box_{\text{gt}})$, let $i \in Q(\mathcal{Z}, \tau, box_{\text{gt}})$~~, then based on the lemma~~. Then, based on Lemma 3, we know that there exists another $box_j \in \boldsymbol{z}^k$ such that $c_j^k \geq \iota$, $\text{Class}(box_j^k) = \text{Class}(box_i^k)$, $c_i^k \leq c_j^k$, $\text{IoU}(box_j, box_{\text{gt}}) < \tau$, and $\text{IoU}(box_i^k, box_j^k) \geq \eta$~~, which is a contradiction with the~~. This contradicts condition (2) of the definition of $Q(\mathcal{Z}, \tau, box_{\text{gt}})$.

~~So the assumption is wrong and we get the result~~ Thus, the assumption leads to a contradiction, and the result follows. □

F THE PROOF OF LEMMA 1

Lemma (Threshold Properties). $\tau < \tau(i, \mathcal{Z}, box_{\text{gt}}) \Rightarrow i \in Q(\mathcal{Z}, \tau, box_{\text{gt}})$

Proof. ~~Firstly, we show that when $\tau' > \tau$, we have $Q(\mathcal{Z}, \tau', box_{\text{gt}}) \subset Q(\mathcal{Z}, \tau, box_{\text{gt}})$. Because if~~

First, we demonstrate that for any $\tau' > \tau$, the inclusion $Q(\mathcal{Z}, \tau', \text{box}_{\text{gt}}) \subseteq Q(\mathcal{Z}, \tau, \text{box}_{\text{gt}})$ holds. Indeed, if an index i ~~satisfied the~~ satisfies Condition (1) in ~~definition 2 with~~ Definition 2 with a threshold τ', ~~then it also satisfied the~~ it necessarily satisfies Condition (1) ~~in definition 2 with any threshold $\tau < \tau'$ under~~ with any lower threshold $\tau < \tau'$, assuming the input constraints and ground truth ~~unchanged. Similar for~~ remain unchanged. A similar argument applies to Condition (2) in ~~definition 2. Then we get the~~ Definition 2. This establishes the monotonicity result.

~~So if $\tau < \tau(i, \mathcal{Z}, box_{\text{gt}})$~~ Now, suppose for the sake of contradiction that $\tau < \tau(i, \mathcal{Z}, \text{box}_{\text{gt}})$ and ~~$i \notin Q(\mathcal{Z}, \tau, box_{\text{gt}})$, then $i \notin Q(\mathcal{Z}, \tau', box_{\text{gt}})$ for any $\tau' > \tau$ by the preceding result, which implies $\tau \geq \tau(i, \mathcal{Z}, box_{\text{gt}})$ according~~ $i \notin Q(\mathcal{Z}, \tau, \text{box}_{\text{gt}})$. By the monotonicity established above, this implies that $i \notin Q(\mathcal{Z}, \tau', \text{box}_{\text{gt}})$ for any $\tau' > \tau$. According to the definition of ~~$\tau(i, \mathcal{Z}, box_{\text{gt}})$~~,

~~but this is contradiction to $\tau < \tau(i, \mathcal{Z}, box_{\text{gt}})$. So the assumption is wrong, and we prove the lemma.~~ $\tau(i, \mathcal{Z}, \text{box}_{\text{gt}})$, this entails $\tau \geq \tau(i, \mathcal{Z}, \text{box}_{\text{gt}})$, which contradicts the initial hypothesis that $\tau < \tau(i, \mathcal{Z}, \text{box}_{\text{gt}})$. Thus, the assumption leads to a contradiction, which completes the proof. □

G THE PROOF OF LEMMA 2

Lemma (Threshold Computation). *The threshold can be calculated as* $\tau(i, \mathcal{Z}, box_{\text{gt}}) = \min\{\tau_1(i, \mathcal{Z}, box_{\text{gt}}), \tau_2(i, \mathcal{Z}, box_{\text{gt}})\}$, *where:*

$$\tau_1(i, \mathcal{Z}, box_{\text{gt}}) = \min_{k \in \Delta} \text{IoU}(box_i^k, box_{\text{gt}}) \cdot \mathbb{I}(\text{Class}(box_i^k) = q) \cdot \mathbb{I}(c_i^k \geq \iota), q = \text{Class}(box_{\text{gt}})$$

$$\tau_2(i, \mathcal{Z}, box_{\text{gt}}) = \begin{cases} \min\limits_{k \in \Delta, n \neq i} \text{IoU}(box_n^k, box_{\text{gt}}) & \text{if } \exists (k, n) \text{ s.t. } \mathcal{C}_{kn} = 1 \\ 1 & \text{otherwise} \end{cases}$$

where constraint $\mathcal{C}_{kn} \equiv \mathbb{I}(c_n^k \geq \iota) \cdot \mathbb{I}(\text{Class}(box_n^k) = q) \cdot \mathbb{I}(\text{IoU}(box_i^k, box_n^k) \geq \eta) \cdot \mathbb{I}(c_n^k \geq c_i^k)$.

Proof. ~~There are $\tau(i, \mathcal{Z}, box_{\text{gt}}) \leq \min\{\tau_1(i, \mathcal{Z}, box_{\text{gt}}), \tau_2(i, \mathcal{Z}, box_{\text{gt}})\}$.~~

~~It is easy to see that when $\tau' > \tau_1(i, \mathcal{Z}, box_{\text{gt}})$,~~

We prove the equality by showing both directions of the inequality.

Part 1: Proof of $\tau(i, \mathcal{Z}, \text{box}_{\text{gt}}) \leq \min\{\tau_1, \tau_2\}$.

Observe that if $\tau' > \tau_1(i, \mathcal{Z}, \text{box}_{\text{gt}})$, index i ~~will not in safe-set $Q(\mathcal{Z}, \tau', box_{\text{gt}})$ because violate~~ is not contained in the safe set $Q(\mathcal{Z}, \tau', \text{box}_{\text{gt}})$ because it violates Condition (1) in ~~definition 2. When $\tau' > \tau_2(i, \mathcal{Z}, box_{\text{gt}})$, even if~~ Definition 2. Similarly, if $\tau' > \tau_2(i, \mathcal{Z}, \text{box}_{\text{gt}})$, index i ~~satisfies (1), i will not in safe-set $Q(\mathcal{Z}, \tau', box_{\text{gt}})$ because violate~~ is excluded from the safe set $Q(\mathcal{Z}, \tau', \text{box}_{\text{gt}})$ because it violates Condition (2)~~in definition 2. So there are $\tau(i, \mathcal{Z}, box_{\text{gt}}) \leq \min\{\tau_1(i, \mathcal{Z}, box_{\text{gt}}), \tau_2(i, \mathcal{Z}, box_{\text{gt}})\}$.~~, even if Condition (1) is satisfied. Consequently, the threshold must satisfy:

$$\tau(i, \mathcal{Z}, \text{box}_{\text{gt}}) \leq \min\{\tau_1(i, \mathcal{Z}, \text{box}_{\text{gt}}), \tau_2(i, \mathcal{Z}, \text{box}_{\text{gt}})\}.$$

~~There are $\tau(i, \mathcal{Z}, box_{\text{gt}}) \geq \min\{\tau_1(i, \mathcal{Z}, box_{\text{gt}}), \tau_2(i, \mathcal{Z}, box_{\text{gt}})\}$.~~**Part 2: Proof of $\tau(i, \mathcal{Z}, \text{box}_{\text{gt}}) \geq \min\{\tau_1, \tau_2\}$.**

~~Easy to see that when $\tau' < \tau_1(i, \mathcal{Z}, box_{\text{gt}})$,~~ Note that when $\tau' < \tau_1(i, \mathcal{Z}, \text{box}_{\text{gt}})$, index i must ~~satisfied the~~ satisfy Condition (1) in ~~definition 2for such τ'; when $\tau' < \tau_2(i, \mathcal{Z}, box_{\text{gt}})$,~~ Definition 2. Likewise, when $\tau' < \tau_2(i, \mathcal{Z}, \text{box}_{\text{gt}})$, index i must ~~satisfied the~~ satisfy Condition (2)~~in definition 2 for such τ'.~~. Therefore, if we choose a threshold $\tau < \min\{\tau_1(i, \mathcal{Z}, \text{box}_{\text{gt}}), \tau_2(i, \mathcal{Z}, \text{box}_{\text{gt}})\}$, index i satisfies both conditions, which implies $i \in Q(\mathcal{Z}, \tau, \text{box}_{\text{gt}})$. By the definition of the threshold $\tau(i, \mathcal{Z}, \text{box}_{\text{gt}})$, this implies:

$$\tau(i, \mathcal{Z}, \text{box}_{\text{gt}}) \geq \min\{\tau_1(i, \mathcal{Z}, \text{box}_{\text{gt}}), \tau_2(i, \mathcal{Z}, \text{box}_{\text{gt}})\}.$$

~~So when $\tau < \min\{\tau_1(i, \mathcal{Z}, box_{\text{gt}}), \tau_2(i, \mathcal{Z}, box_{\text{gt}})\}$, there must be $i \in Q(\mathcal{Z}, \tau, box_{\text{gt}})$, which implies $\tau(i, \mathcal{Z}, box_{\text{gt}}) \geq \min\{\tau_1(i, \mathcal{Z}, box_{\text{gt}}), \tau_2(i, \mathcal{Z}, box_{\text{gt}})\}$.~~

~~So we get the result~~Combining the results from Part 1 and Part 2, the equality holds. □

H GET THE VERIFICATION BOUND

In this section, we show how to **Calculate the verification bound** for NMS.

We first define the verification bounds:

Definition 4 (OD Verification Bounding)**.** For constraints $\mathcal{C}$ and box_{gt}, define:

$$\min_{\boldsymbol{x}\in\mathcal{C}} \max_{box_i\in \text{N}(\text{F}(\boldsymbol{x}))} \text{IoU}(box_i, box_{\text{gt}}) \cdot \mathbb{I}(\text{Class}(box_i) = \text{Class}(box_{\text{gt}})),$$

as the OD verification bounding. This quantifies robustness against OD attacks.

We need to estimate the verification bound ~~$\min_{z\in\mathcal{Z}} \max_{box_i\in \text{N}(z)} \text{IoU}(box_i, box_{\text{gt}})\mathbb{I}(\text{Class}(box_i) = \text{Class}(box_{\text{gt}}))$~~ $\min_{z\in\mathcal{Z}} \max_{box_i\in \text{N}(z)} \text{IoU}(box_i, box_{\text{gt}})\mathbb{I}(\text{Class}(box_i) = \text{Class}(box_{\text{gt}}))$ under the input restriction $\mathcal{Z}$ and ground truth box_{gt}, according to definition 4.

To estimate such bound, firstly, we need the following lemma:

Lemma 4. *For any $\mathcal{Z}, \tau, box_{\text{gt}}$, there is:*

$$\min_{z\in\mathcal{Z}} \max_{box_i\in \text{N}(z)} \text{IoU}(box_i, box_{\text{gt}})\mathbb{I}(\text{Class}(box_i) = \text{Class}(box_{\text{gt}})) \geq \tau \mathbb{I}(\{|Q(\mathcal{Z}, \tau, box_{\text{gt}})| \geq 1\})$$

So we will try to find the maximum τ that makes $Q(\mathcal{Z}, \tau, box_{\text{gt}})$ bigger than 0, but sometimes this maximum value does not exist (can only approach the maximum value arbitrarily), so we look for the following value instead:

$$\min_{\tau\in[0,1]} \tau \text{ s.t. } |Q(\mathcal{Z}, \tau', box_{\text{gt}})| = 0, \ \forall 1 \geq \tau' > \tau \tag{5}$$

We use the following lemma to calculate such minimum value:

Lemma 5. *The solution of problem 5 is equal to:* $\max_{i\in[n_{\boldsymbol{x}}]} \tau(i, \mathcal{Z}, box_{\text{gt}})$.

Use such lemma, we just to need calculate $\tau(i, \mathcal{Z}, box_{\text{gt}})$ as said before, and then we can estimate the verification bounding.

H.1 The Proof of Lemma 4

Proof. ~~By the proposition~~ By Proposition 2, if ~~$|Q(\mathcal{Z}, \tau, box_{\text{gt}})| \geq 1$~~ $|Q(\mathcal{Z}, \tau, \text{box}_{\text{gt}})| \geq 1$, then for any ~~$z \in \mathcal{Z}$, there is a $box_i \in \text{N}(z)$ such that $\text{IoU}(box_i, box_{\text{gt}}) \geq \tau$ and $\text{Class}(box_i) = \text{Class}(box_{\text{gt}})$, which implies~~ $z \in \mathcal{Z}$, there exists a $\text{box}_i \in \text{N}(z)$ such that $\text{IoU}(\text{box}_i, \text{box}_{\text{gt}}) \geq \tau$ and $\text{Class}(\text{box}_i) = \text{Class}(\text{box}_{\text{gt}})$. This implies that the value of ~~$\min_{z\in\mathcal{Z}} \max_{box_i\in \text{N}(z)} \text{IoU}(box_i, box_{\text{gt}})\mathbb{I}(\text{Class}(box_i) = \text{Class}(box_{\text{gt}}))$~~ $\min_{z\in\mathcal{Z}} \max_{\text{box}_i\in \text{N}(z)} \text{IoU}(\text{box}_i, \text{box}_{\text{gt}}) \cdot \mathbb{I}(\text{Class}(\text{box}_i) = \text{Class}(\text{box}_{\text{gt}}))$ is greater than or equal to τ, ~~so we get the~~ which yields the desired result. □

H.2 The Proof of Lemma 5

Proof. We have shown that when $\tau' > \tau$, we have $Q(\mathcal{Z}, \tau', box_{\text{gt}}) \subset Q(\mathcal{Z}, \tau, box_{\text{gt}})$ in Lemma 1.

By the definition of $\tau(i, \mathcal{Z}, box_{\text{gt}})$ and above result, we know that the safe set $Q(\mathcal{Z}, \tau', box_{\text{gt}})$ is $\emptyset$ for any $\tau' > \max_{i\in[n_{\boldsymbol{x}}]} \tau(i, \mathcal{Z}, box_{\text{gt}})$, so the solution of problem 5 is not more than $\max_{i\in[n_{\boldsymbol{x}}]} \tau(i, \mathcal{Z}, box_{\text{gt}})$.

If τ is the solution of problem 5, then for any $i \in [n_{\boldsymbol{x}}]$, there must be $i \notin Q(\mathcal{Z}, \tau', box_{\text{gt}})$ for any $\tau' \geq \tau$, so $\tau \geq \max_{i\in[n_{\boldsymbol{x}}]} \tau(i, \mathcal{Z}, box_{\text{gt}})$.

So we prove the lemma.

□

I Verification Process of NMS

To illustrate Non-Maximum Suppression (NMS) verification, we define constraints as $\mathcal{Z} = \mathcal{H} \setminus \mathcal{S}$. Here, $\mathcal{H}$ constrains the neural network output $\boldsymbol{y}$, and $\mathcal{S}$ constrains bounding box parameters. This formulation is equivalent to the original. Algorithm 5 details the NMS verification process.

Algorithm 5 Soundness Object Disappear Thread Verification for NMS

Require: Constraints $\mathcal{Z} = \mathcal{H} \setminus \mathcal{S}$; input $\boldsymbol{x}$; output $\boldsymbol{y}$; ground truth box_{gt}; confidence threshold ι; IoU threshold τ; upper bound η.
Ensure: Calculate τ_1, τ_2.

$\mathcal{B}_{\text{cand}}, \mathcal{B}_{\text{other}} \leftarrow \emptyset, \emptyset$
$\{\underline{\overline{box}}_i\}_{i=1}^{n_{\boldsymbol{x}}}$ = CONSTRUCT_ABSTRACT_BOX($\mathcal{Z}$)
for all $\underline{\overline{box}}_i \in \{\underline{\overline{box}}_k\}_{k=1}^{n_{\boldsymbol{x}}}$ **do**
  $\tau_1(i) \leftarrow 0, \tau_2(i) \leftarrow 1$
  if $\forall k \in [n] \setminus \{\text{Class}(box_{\text{gt}})\}, \overline{p}_{i,\text{Class}(box_{\text{gt}})} \leq \underline{p}_{i,k}$ **then**
    continue ▷ Skip boxes must not match box_{gt} class
  end if
  if $\underline{c}_i \geq \iota$ **then** ▷ Ensure indicator $\mathbb{I}(c_i \geq \iota) = 1$
    $\tau_1(i) \leftarrow$ IOU_LOWER_BOUNDS($\underline{\overline{box}}_i$, box_{gt})
    if $\forall k \in [n] \setminus \{\text{Class}(box_{\text{gt}})\}, \underline{p}_{i,\text{Class}(box_{\text{gt}})} \geq \overline{p}_{i,k}$ **then**
      $\mathcal{B}_{\text{cand}} \leftarrow \mathcal{B}_{\text{cand}} \cup \{\underline{\overline{box}}_i\}$ ▷ Box must match box_{gt} class
    else
      $\mathcal{B}_{\text{other}} \leftarrow \mathcal{B}_{\text{other}} \cup \{\underline{\overline{box}}_i\}$ ▷ Box may match box_{gt} class
    end if
  end if
end for
for all $\underline{\overline{box}}_i \in \mathcal{B}_{\text{cand}}$ **do**
  for all $\underline{\overline{box}}_j \in \mathcal{B}_{\text{other}}$ **do**
    if $\overline{c}_j < \underline{c}_i$ **then** ▷ Ensure box_j may suppress box_i
      continue ▷ Skip boxes that cannot suppress box_i
    end if
    $ub \leftarrow$ IOU_UPPER_BOUNDS($\underline{\overline{box}}_i$, $\underline{\overline{box}}_j$)
    if $ub \geq \eta$ **then** ▷ Ensure box_i may suppressed by box_j
      $lb \leftarrow$ IOU_LOWER_BOUNDS($\underline{\overline{box}}_j$, box_{gt})
      $\tau_2(i) \leftarrow \min(\tau_2(i), lb)$
    end if
  end for
end for
return $\{\tau_1(i)\}_{i=1}^{n_{\boldsymbol{x}}}, \{\tau_2(i)\}_{i=1}^{n_{\boldsymbol{x}}}$

Next we show how to construct the abstract box $\underline{\overline{box}}$ and how to calculate the lower and upper bounds of IoU.

Let $\mathbb{B}$ be the 'box space' (space of individual box structures). An interpretation function $\mathrm{G} : \mathbb{R}^{d_L} \to \{\mathcal{S} \subseteq \mathbb{B} \mid |\mathcal{S}| = n_{\boldsymbol{x}}\}$ maps $\boldsymbol{y}$ to a set $\{box_i\}_{i=1}^{n_{\boldsymbol{x}}}$ of $n_{\boldsymbol{x}}$ bounding boxes, where $n_{\boldsymbol{x}}$ is a constant determined by the fixed input dimension. A function $\mathrm{S} : \mathbb{B} \to \mathcal{P}(\{1, \ldots, d_L\})$ (where $\mathcal{P}$ is the power set) maps each distinct $box_k \in \mathbb{B}$ (that could form part of an output set) to its source indices in $\boldsymbol{y}$. We then define the regular region $\mathcal{Z}$ as follows:

Definition 5 (Regular Region)**.** A regular region is a subset $\mathcal{Z} \subseteq \mathbb{R}^{d_L}$ defined as $\mathcal{Z} = \mathcal{H} \setminus \mathcal{S}$, where:

- $\mathcal{H}$ is a hyperrectangle (axis-aligned rectangular region) centered at $\mathbf{F}(\boldsymbol{x}^{(0)})$ with component-wise perturbations bounded by $c_1 \boldsymbol{v}_{\max}$:

$$\mathcal{H} = \left\{ \mathbf{F}(\boldsymbol{x}^{(0)}) + \boldsymbol{\epsilon} \in \mathbb{R}^{d_L} \mid \forall j \in \{1, \ldots, d_L\},\ |\epsilon_j| \leq (c_1 \boldsymbol{v}_{\max})_j \right\}$$

- $\mathcal{S}$ is a union of k hyperspherical zones. Each zone $\mathcal{S}_i = \mathrm{B}(\boldsymbol{c}_i, d_i)$ is defined by a center $\boldsymbol{c}_i$ and radius d_i, where $\boldsymbol{c}_i$ is the center of the exclusion zone and d_i is the radius. The union of these zones is given by:

$$\mathcal{S} = \bigcup_{i=1}^{k} \mathcal{S}_i = \bigcup_{i=1}^{k} \mathrm{B}(\boldsymbol{c}_i, d_i), \mathrm{B}(\boldsymbol{c}_i, d_i) = \left\{ \boldsymbol{y} \in \mathbb{R}^{d_L} \;\middle|\; \|\boldsymbol{y} - \boldsymbol{c}_i\|_2^2 \leq d_i^2 \right\}$$

Note, sometimes we just need part dimension of $\mathcal{S}_i$. Thus we can extend B to $\mathrm{B}(\boldsymbol{c}_i, d_i, \mathcal{I}_i)$, where $\mathcal{I}_i$ is the index set of dimensions in $\mathcal{S}_i$. Then we have:

$$\mathrm{B}(\boldsymbol{c}_i, d_i, \mathcal{I}_i) = \left\{ \boldsymbol{y} \in \mathbb{R}^{d_L} \;\middle|\; \|\boldsymbol{y}_{\mathcal{I}_i} - \boldsymbol{c}_{i,\mathcal{I}_i}\|_2^2 \leq d_i^2 \right\}$$

where $\boldsymbol{c}_{i,\mathcal{I}_i}$ is the component of $\boldsymbol{c}_i$ for index set $\mathcal{I}_i$.

I.1 Abstract Bounding Box Construction and IoU Bound Computation

We use $\tilde{x}$ to represent x is a Gurobi variable, $\tilde{\mathbb{R}}$ to represent real number Gurobi variable space. Let $\tilde{\boldsymbol{y}} \in \tilde{\mathbb{R}}^{d_L}$ be the Gurobi variable vector.

Then we encode the regular region $\mathcal{Z}$ as a set of constraints. To encode $\mathcal{H}$, we need to get the lower and upper bounds of each parameter of each bounding box. Suppose $\mathcal{H} = \left\{ \mathbf{F}(\boldsymbol{x}^{(0)}) + \boldsymbol{\epsilon} \in \mathbb{R}^{d_L} \mid \forall j \in \{1, \ldots, d_L\},\ |\epsilon_j| \leq (c_1 \boldsymbol{v}_{\max})_j \right\}$, where c_1 and $\boldsymbol{v}_{\max}$ (Note $\boldsymbol{v}_{\max} > 0$) are obtained from Part 1. Let $\overline{\boldsymbol{y}} = \mathbf{F}(\boldsymbol{x}^{(0)}) + c_1 \boldsymbol{v}_{\max}$ and $\underline{\boldsymbol{y}} = \mathbf{F}(\boldsymbol{x}^{(0)}) - c_1 \boldsymbol{v}_{\max}$. Then we add the following constraints to the Gurobi model:

$$\underline{\boldsymbol{y}} \leq \tilde{\boldsymbol{y}} \leq \overline{\boldsymbol{y}}$$

Here, $\leq$ is the component-wise less than or equal to operator.

Next, we need to add the exclusion zone constraints. For each exclusion zone $\mathcal{S}_i$, we need to add the following constraints:

$$\sum_{j \in \mathcal{I}_i} (\tilde{\boldsymbol{y}}_j - (\boldsymbol{c}_i)_j)^2 \geq d_i^2$$

where $(\boldsymbol{c}_i)_j$ is the component of $\boldsymbol{c}_i$ for index j.

CONSTRUCT_ABSTRACT_BOX($\mathcal{Z}$) adds these constraints to the Gurobi model. And then reorganizes $\tilde{\boldsymbol{y}}$ as an abstract bounding box set $\{\underline{\overline{box}}_i\}_{i=1}^{n_{\boldsymbol{x}}}$, where $n_{\boldsymbol{x}}$ is the number of bounding boxes. Each bounding box $\underline{\overline{box}}_i = (\tilde{x}_i, \tilde{y}_i, \tilde{w}_i, \tilde{h}_i, \tilde{c}_i, \{\tilde{p}_{i,j}\}_{j=1}^{n_{\boldsymbol{x}}})$. Note all the parameters of $\underline{\overline{box}}_i$ are from $\tilde{\boldsymbol{y}}$ thus have constraints on them.

IoU bounds for abstract boxes involve:

1. **Geometric Constraints**: Box i coordinates are:

$$\tilde{x}^i_{\min} = \tilde{x}_i - \frac{\tilde{w}_i}{2}, \quad \tilde{x}^i_{\max} = \tilde{x}_i + \frac{\tilde{w}_i}{2},$$
$$\tilde{y}^i_{\min} = \tilde{y}_i - \frac{\tilde{h}_i}{2}, \quad \tilde{y}^i_{\max} = \tilde{y}_i + \frac{\tilde{h}_i}{2}.$$

2. **Intersection/Union Area**

$$\begin{aligned}
I_w &= \max(0, \min(\tilde{x}^i_{\max}, \tilde{x}^j_{\max}) - \max(\tilde{x}^i_{\min}, \tilde{x}^j_{\min})), \\
I_h &= \max(0, \min(\tilde{y}^i_{\max}, \tilde{y}^j_{\max}) - \max(\tilde{y}^i_{\min}, \tilde{y}^j_{\min})), \\
A_{\text{int}} &= I_w \cdot I_h, \\
A_{\text{union}} &= (\tilde{x}^i_{\max} - \tilde{x}^i_{\min}) \cdot (\tilde{y}^i_{\max} - \tilde{y}^i_{\min}) + (\tilde{x}^j_{\max} - \tilde{x}^j_{\min}) \cdot (\tilde{y}^j_{\max} - \tilde{y}^j_{\min}) - A_{\text{int}}.
\end{aligned}$$

We use big-M constraints to encode $\max(0, \cdot)$ and $\min(\cdot, \cdot)$ operations in Gurobi. For example, to encode $\tilde{A}_{\text{expr}} = \max(0, E)$, where E is an expression, we introduce an auxiliary binary variable $\tilde{b} \in \{0, 1\}$ and a sufficiently large constant M. The variable $\tilde{A}_{\text{expr}}$ is then constrained by:

$$\begin{aligned}
&\tilde{A}_{\text{expr}} \geq E \\
&\tilde{A}_{\text{expr}} \geq 0 \\
&\tilde{A}_{\text{expr}} \leq E + M \cdot \tilde{b} \\
&\tilde{A}_{\text{expr}} \leq M \cdot (1 - \tilde{b})
\end{aligned}$$

where $\tilde{A}_{\text{expr}}$ is a Gurobi variable representing the maximum. The inner terms of I_w and I_h, such as $\min(\tilde{x}^i_{\max}, \tilde{x}^j_{\max})$, are handled similarly using appropriate big-M formulations or Gurobi's built-in functions. The constant M must be chosen such that $M \geq \max(U_E, -L_E)$, where U_E and L_E are known upper and lower bounds for the expression E.

- If $E \geq 0$, the objective of minimizing $\tilde{A}_{\text{int}}$ (or other constraints) will force $\tilde{b} = 0$. The constraints become $\tilde{A}_{\text{int}} \geq E$, $\tilde{A}_{\text{int}} \geq 0$, $\tilde{A}_{\text{int}} \leq E$, and $\tilde{A}_{\text{int}} \leq M$. This correctly sets $\tilde{A}_{\text{int}} = E$.
- If $E < 0$, the objective will force $\tilde{b} = 1$. The constraints become $\tilde{A}_{\text{int}} \geq E$, $\tilde{A}_{\text{int}} \geq 0$, $\tilde{A}_{\text{int}} \leq E + M$, and $\tilde{A}_{\text{int}} \leq 0$. This correctly sets $\tilde{A}_{\text{int}} = 0$.

3. **Binary Search for IoU bounds**: To find IoU (lower bound of IoU):

1: $\tau_{lb} \leftarrow 0, \tau_{ub} \leftarrow 1, \epsilon_{\text{search}} \leftarrow 10^{-5}$
2: **while** $|\tau_{ub} - \tau_{lb}| > \epsilon_{\text{search}}$ **do**
3: $\quad \tau_{\text{mid}} \leftarrow (\tau_{lb} + \tau_{ub})/2$
4: $\quad$ Solve $L_{\text{check}} = \min(A_{\text{int}} - \tau_{\text{mid}} A_{\text{union}})$ by Gurobi
5: $\quad$ **if** $L_{\text{check}} \geq 0$ **then** $\tau_{lb} \leftarrow \tau_{\text{mid}}$
6: $\quad$ **else**$\tau_{ub} \leftarrow \tau_{\text{mid}}$
7: $\quad$ **end if**
8: **end while**
9: **return** τ_{lb}

Checking IoU bounds against a value τ is often done via optimizing $A_{\text{int}} - \tau A_{\text{union}}$, as direct interval ratio optimization $A_{\text{int}}/A_{\text{union}}$ is complex for solvers (e.g., Gurobi) without reformulation. For instance, to check if $\text{IoU}(box_i, box_{\text{gt}}) \geq \tau$ is possible, one can check if $\max(A_{\text{int}} - \tau A_{\text{union}}) \geq 0$.

It's trivial to extend the above algorithm to find τ_{ub} (upper bound of IoU).

The Gurobi solver, when minimizing or maximizing an objective subject to these constraints, returns an assignment for $\tilde{\boldsymbol{y}}$ (which corresponds to a specific point $\boldsymbol{z} \in \mathcal{Z}$). If a particular optimization (e.g., a step in the binary search showing $L_{\text{check}} < 0$) demonstrates a property violation, the returned $\tilde{\boldsymbol{y}}$ is the concrete instance $\boldsymbol{z}$ that exhibits this violation.

J THE PROOF OF THEOREM 1

Theorem (probabilistic guarantee for Part 3). *For any $\alpha, \beta, \delta, \epsilon \in (0,1)$ satisfying $(1-2\epsilon)^M - \delta > 0$ and $N \cdot ((1-2\epsilon)^M - \delta) > \frac{2}{\alpha}\ln(\frac{1}{\beta}) + 2 + \frac{2}{\alpha}\ln(\frac{2}{\alpha})$, with this algorithm, for any $\boldsymbol{y}$, with probability exceeding $1 - e^{-2N\delta^2} - \beta - 2(1-\epsilon)^{M_2}$ over step 1 and step 2, we have $\text{V}(\boldsymbol{y}, d_{\min}) \leq \alpha$.*

For convenience, ~~name the step one in algorithm 4 as P1, the step two in algorithm 4 as P2.~~ denote Step 1 in Algorithm 4 by **P1**, and Step 2 by **P2**.

J.1 ~~SITUATION 1~~

~~Firstly, we show the situation that when $\boldsymbol{y} \in \{\text{F}(\boldsymbol{x})\}_{\boldsymbol{x}\in\mathcal{C}}$. We~~ Theorem 1 aims to estimate a constant C such that, with high probability over the sampling in Part 3,

$$\|\text{F}(\boldsymbol{x}) - \boldsymbol{y}\|_2 \leq C \quad \text{for most } \boldsymbol{x} \sim \mathcal{C},$$

whenever $\boldsymbol{y}$ lies in the region $\mathcal{Z}$. The proof proceeds in three steps:

1. For each sampled point $\boldsymbol{x}$, we estimate a local interval $[B_{\text{F}(\boldsymbol{x})}, A_{\text{F}(\boldsymbol{x})}]$ that covers, with probability at least $1 - 2\epsilon$, the distances $\|\text{F}(\boldsymbol{x}') - \text{F}(\boldsymbol{x})\|_2$ to its M neighbors.

2. Using Hoeffding's inequality, we show that with high probability at least $N((1-2\epsilon)^M - \delta)$ base points have all M neighbors lying within their respective intervals.

3. On this subset, we apply an RCP_N-style scenario bound to the function $S_{\text{F}(\boldsymbol{x})}$, yielding an upper bound C satisfying $\text{P}_{\boldsymbol{x}\sim\mathcal{C}}(S_{\text{F}(\boldsymbol{x})} \leq C) \geq 1 - \alpha$.

J.1 SITUATION 1

We first consider the case where $\boldsymbol{y}$ is itself realizable as $\mathrm{F}(\boldsymbol{x})$ for some $\boldsymbol{x} \in \mathcal{C}$. We have the following theorem, which can directly get the theorem 1 when $\boldsymbol{y} \in \{\mathrm{F}(\boldsymbol{x})\}_{x\in\mathcal{C}}$:

Theorem 3. *For any $\alpha, \beta, \delta, \epsilon \in (0,1)$ satisfied $(1-2\epsilon)^M - \delta > 0$ and $\mathrm{N}((1-2\epsilon)^M - \delta) > \frac{2}{\alpha}\ln(\frac{1}{\beta}) + 2 + \frac{2}{\alpha}\ln(\frac{2}{\alpha})$, by this algorithm, if $\boldsymbol{y} \sim \mathrm{F}(\mathcal{C})$ (that is, the distribution of the network output when the input obeys $\mathcal{C}$), then with probability $1 - e^{-2N\delta^2} - \beta - 2(1-\epsilon)^{M_2}$ of $P_{1\cup 2}$, ~~there is~~ we have $\mathrm{P}_{\boldsymbol{y}}(B_m - C(A_m - B_m) < 0) \geq 1 - \alpha$.*

Proof. ~~For any given $\boldsymbol{y}$, let $A_{\boldsymbol{y}}$ satisfy that $\mathrm{P}_{\boldsymbol{x}\sim\mathcal{C}}(\|\mathrm{F}(\boldsymbol{x}) - \boldsymbol{y}\|_2 \leq A_{\boldsymbol{y}}) = 1-\epsilon$, let $B_{\boldsymbol{y}}$ satisfy that $\mathrm{P}_{\boldsymbol{x}\sim\mathcal{C}}(\|\mathrm{F}(\boldsymbol{x}) - \boldsymbol{y}\|_2 \geq B_{\boldsymbol{y}}) = 1-\epsilon$. Let $S_{\boldsymbol{y}} = \frac{B_{\boldsymbol{y}}}{A_{\boldsymbol{y}} - B_{\boldsymbol{y}}}$.~~

~~The proof is into four parts:~~

Part One: For any i, with probability $(1-2\epsilon)^M$ of P1, ~~there are~~ we havee $\|\mathrm{F}(\boldsymbol{x}^{(i,j)}) - \mathrm{F}(\boldsymbol{x}^{(i)})\|_2 \in [B_{\mathrm{F}(\boldsymbol{x}^{(i)})}, A_{\mathrm{F}(\boldsymbol{x}^{(i)})}]$ for all $j \in [M]$.

~~This is obvious, based on the definition of $B_{\mathrm{F}(\boldsymbol{x}^{(i)})}$ and $A_{\mathrm{F}(\boldsymbol{x}^{(i)})}$, we know that $\mathrm{P}_{\boldsymbol{x}^{(i,j)}\sim\mathcal{C}}(\|\mathrm{F}(\boldsymbol{x}^{(i,j)}) - \mathrm{F}(\boldsymbol{x}^{(i)})\|_2 \in [B_{\mathrm{F}(\boldsymbol{x}^{(i)})}, A_{\mathrm{F}(\boldsymbol{x}^{(i)})}]) = 1-2\epsilon$. Because each selection of~~ For any i, since the samples $\boldsymbol{x}^{(i,j)}$ ~~is independent, so we get the result~~are i.i.d.,

$$\mathrm{P}\Big(\|\mathrm{F}(\boldsymbol{x}^{(i,j)}) - \mathrm{F}(\boldsymbol{x}^{(i)})\|_2 \in [B_{\mathrm{F}(\boldsymbol{x}^{(i)})}, A_{\mathrm{F}(\boldsymbol{x}^{(i)})}]\Big) = 1-2\epsilon.$$

Thus all M neighbors lie in the interval with probability $(1-2\epsilon)^M$.

Part Two: With probability $1 - e^{-2N\delta^2}$ of P1, ~~there are~~ we have at least $\mathrm{N}((1-2\epsilon)^M - \delta)$ numbers of $i \in [N]$ satisfied that $\|\mathrm{F}(\boldsymbol{x}^{(i,j)}) - \mathrm{F}(\boldsymbol{x}^{(i)})\|_2 \in [B_{\mathrm{F}(\boldsymbol{x}^{(i)})}, A_{\mathrm{F}(\boldsymbol{x}^{(i)})}]$ for all $j \in [M]$.

~~To proof that, we need use Hoeffding inequality. Let~~

Define indicator variables

$$X_i = \mathbb{I}\Big\{\|\mathrm{F}(\boldsymbol{x}^{(i,j)}) - \mathrm{F}(\boldsymbol{x}^{(i)})\|_2 \in [B_{\mathrm{F}(\boldsymbol{x}^{(i)})}, A_{\mathrm{F}(\boldsymbol{x}^{(i)})}]\ \forall j \in [M]\Big\}.$$

Then X_i ~~is the random variable defined as: $X_i = 1$ if $\|\mathrm{F}(\boldsymbol{x}^{(i,j)}) - \mathrm{F}(\boldsymbol{x}^{(i)})\|_2 \in [B_{\mathrm{F}(\boldsymbol{x}^{(i)})}, A_{\mathrm{F}(\boldsymbol{x}^{(i)})}]$ for all $j \in [M]$, or $X_i = 0$. Without a doubt, $\{X_i\}_{i=1}^N$~~ are i.i.d~~, and by Part One, there are $\mathrm{E}(X_i) = (1-2\epsilon)^M$.~~

~~Then by using Hoeffdinginequality, we have that: $\mathrm{P}(\sum_{i=1}^N X_i/N - (1-2\epsilon)^M \leq -\delta) \leq e^{-2N\delta^2}$, this is what we want~~. with $\mathbb{E}[X_i] = (1-2\epsilon)^M$. Applying Hoeffding's inequality yields

$$\mathrm{P}\left(\frac{1}{N}\sum_{i=1}^N X_i - (1-2\epsilon)^M \leq -\delta\right) \leq e^{-2N\delta^2},$$

implying the desired bound on the number of "good" indices. Then the complement event gives:

$$\mathrm{P}\left(\frac{1}{N}\sum_{i=1}^N X_i - (1-2\epsilon)^M > -\delta\right) > 1 - e^{-2N\delta^2},$$

i.e., at least $N((1-2\epsilon)^M - \delta)$ indices i satisfy the desired property with probability at least $1 - e^{-2N\delta^2}$.

Part Three: With probability $1 - e^{2N\delta^2} - \beta$ of P1, there are $\mathrm{P}_{\boldsymbol{x}\sim\mathcal{C}}(S_{\mathrm{F}(\boldsymbol{x})} \leq C) \geq 1-\alpha$.

We have two simple facts:

Firstly, if $i \in [N]$ such that $\|\mathrm{F}(\boldsymbol{x}^{(i,j)}) - \mathrm{F}(\boldsymbol{x}^{(i)})\|_2 \in [B_{\mathrm{F}(\boldsymbol{x}^{(i)})}, A_{\mathrm{F}(\boldsymbol{x}^{(i)})}]$ for all $j \in [M]$, then there must be $C \geq \frac{B_i'}{A_i' - B_i'} \geq S_{\mathrm{F}(\boldsymbol{x}^{(i)})}$.

Secondly, for any $N_1 \geq \mathrm{N}((1-2\epsilon)^M - \delta)$, by using RCP_N to find the maximum of function $S_{\mathrm{F}(\boldsymbol{x})}$, we know that with probability $1-\beta$ for i.i.d selected N_1 samples $\{\boldsymbol{x}^{(i)}\}_{i=1}^{N_1}$ in $\mathcal{C}$, then $\mathrm{P}_{\boldsymbol{x}\sim\mathcal{C}}(S_{\mathrm{F}(\boldsymbol{x})} \leq \max_{i\in[N_1]}\{S_{\mathrm{F}(\boldsymbol{x}^{(i)})}\}) \geq 1-\alpha$.

By the two facts, we can deduce that:

~~For any $N_1 \in \mathbb{Z}^+$, let $\mathcal{Q}_{N_1} \subset 2^{\mathcal{C}}$ and a $q \in 2^{\mathcal{C}} \cap \mathcal{Q}_{N_1}$ if and only if $|q| = N_1$ and $\mathrm{P}_{\boldsymbol{x}\sim\mathcal{C}}(S_{\mathrm{F}(\boldsymbol{x})} \leq \max\{S_{\mathrm{F}(\boldsymbol{x}^{(p)})}\}_{\boldsymbol{x}^{(p)}\in q}) \geq 1-\alpha$~~Let $\mathcal{Q}_k = \{q \subset \mathcal{C} : |q| = k, \mathrm{P}_{\boldsymbol{x}\sim\mathcal{C}}(S_{\mathrm{F}(\boldsymbol{x})} \leq \max\{S_{\mathrm{F}(\boldsymbol{x}^{(p)})}\}_{\boldsymbol{x}^{(p)}\in q}) \geq 1-\alpha\}$. According to the second fact, when $N_1 \geq \mathrm{N}((1-2\epsilon)^M - \delta)$, there are $\mathrm{P}_{\boldsymbol{x}^{(i)}\sim\mathcal{C}, i\in[N_1]}(\{\boldsymbol{x}^{(i)}\} \in \mathcal{Q}_{N_1}) \geq 1-\beta$.

~~Then let $\mathcal{T} = \{\boldsymbol{x}^{(i)}\}$ mean the set contained all $\boldsymbol{x}^{(i)}$ such that $\|\mathrm{F}(\boldsymbol{x}^{(i,j)}) - \mathrm{F}(\boldsymbol{x}^{(i)})\|_2 \in [B_{\mathrm{F}(\boldsymbol{x}^{(i)})}, A_{\mathrm{F}(\boldsymbol{x}^{(i)})}]$ for all $j \in [M]$~~Let $\mathcal{T} = \{\boldsymbol{x}^{(i)} \in \mathcal{C} : \forall j \in [M], \|\mathrm{F}(\boldsymbol{x}^{(i,j)}) - \mathrm{F}(\boldsymbol{x}^{(i)})\|_2 \in [B_{\mathrm{F}(\boldsymbol{x}^{(i)})}, A_{\mathrm{F}(\boldsymbol{x}^{(i)})}]\}$. If $\mathcal{T} \in \cup_{k=1}^{\infty}\mathcal{Q}_k$, then by the first fact we have $C > \max_{\boldsymbol{x}^{(i)}\in\mathcal{T}} S_{\mathrm{F}(\boldsymbol{x}^{(i)})}$, hence we have $\mathrm{P}_{\boldsymbol{x}\sim\mathcal{C}}(S_{\mathrm{F}(\boldsymbol{x})} \leq C) \geq \mathrm{P}_{\boldsymbol{x}\sim\mathcal{C}}(S_{\mathrm{F}(\boldsymbol{x})} \leq \max\{S_{\mathrm{F}(\boldsymbol{x}^{(p)})}\}_{\boldsymbol{x}^{(p)}\in\mathcal{T}}) \geq 1-\alpha$, so we only need to calculate $\mathrm{P}_{\mathrm{P1}}(\mathcal{T} \in \cup_{k=1}^{\infty}\mathcal{Q}_k)$.

We have that:

$$\begin{aligned} & \mathrm{P}_{\mathrm{P1}}(\mathcal{T} \in \cup_{k=1}^{\infty}\mathcal{Q}_k) \\ = & \sum_{k=1}^{\infty} \mathrm{P}_{\mathrm{P1}}(\mathcal{T} \in \mathcal{Q}_k) \\ = & \sum_{k=1}^{N} \mathrm{P}_{\mathrm{P1}}(\mathcal{T} \in \mathcal{Q}_k) \end{aligned}$$

According to the process of algorithm, which k index i satisfy the condition $\|\mathrm{F}(\boldsymbol{x}^{(i,j)}) - \mathrm{F}(\boldsymbol{x}^{(i)})\|_2 \in [B_{\mathrm{F}(\boldsymbol{x}^{(i)})}, A_{\mathrm{F}(\boldsymbol{x}^{(i)})}]$ for all $j \in [M]$ is determined by the second time randomize samples in P1, and whether these samples are in the set $\mathcal{Q}_k$ is determined by first time randomize samples in P1. These two processes are completely independent. So we have that:

$$\begin{aligned} & \mathrm{P}_{\mathrm{P1}}(\mathcal{T} \in \mathcal{Q}_k) \\ = & \sum_{\{t_i\}_{i=1}^k \subset [N]} \mathrm{P}_{\mathrm{P1}}(\{\boldsymbol{x}^{(t_i)}\}_{i=1}^k = \mathcal{T}, \{\boldsymbol{x}^{(t_i)}\}_{i=1}^k \in \mathcal{Q}_k) \\ = & \sum_{\{t_i\}_{i=1}^k \subset [N]} \mathrm{P}_{\mathrm{P1}}(\{\boldsymbol{x}^{(t_i)}\}_{i=1}^k \in \mathcal{Q}_k) \mathrm{P}_{\mathrm{P1}}(\{\boldsymbol{x}^{(t_i)}\}_{i=1}^k = \mathcal{T}) \end{aligned}$$

Then, ~~there are~~we have that:

$$\begin{aligned} & \mathrm{P}_{\mathrm{P1}}(\mathcal{T} \subset \cup_{k=1}^{\infty}\mathcal{Q}_k) \\ = & \sum_{k=1}^{N} \mathrm{P}_{\mathrm{P1}}(\mathcal{T} \subset \mathcal{Q}_k) \\ = & \sum_{k=1}^{N} \sum_{\{t_i\}_{i=1}^k \subset [N]} \mathrm{P}_{\mathrm{P1}}\left(\{\boldsymbol{x}^{(t_i)}\}_{i=1}^k \in \mathcal{Q}_k\right) \cdot \mathrm{P}_{\mathrm{P1}}\left(\{\boldsymbol{x}^{(t_i)}\}_{i=1}^k = \mathcal{T}\right) \\ \geq & \sum_{k=\mathrm{N}((1-2\epsilon)^M-\delta)}^{N} \sum_{\{t_i\}_{i=1}^k \subset [N]} \mathrm{P}_{\mathrm{P1}}\left(\{\boldsymbol{x}^{(t_i)}\}_{i=1}^k \in \mathcal{Q}_k\right) \cdot \mathrm{P}_{\mathrm{P1}}\left(\{\boldsymbol{x}^{(t_i)}\}_{i=1}^k = \mathcal{T}\right) \\ \geq & (1-\beta) \sum_{k=\mathrm{N}((1-2\epsilon)^M-\delta)}^{N} \sum_{\{t_i\}_{i=1}^k \subset [N]} \mathrm{P}_{\mathrm{P1}}(\{\boldsymbol{x}^{(t_i)}\}_{i=1}^k = \mathcal{T}) \\ = & (1-\beta)\mathrm{P}_{\mathrm{P1}}\left(|\mathcal{T}| \geq \mathrm{N}((1-2\epsilon)^M - \delta)\right) \\ \geq & (1-\beta)(1-e^{-2N\delta^2}) \\ \geq & 1-\beta-e^{-2N\delta^2} \end{aligned}$$

This is what we want.

Part Four: For any $\boldsymbol{x} \in \mathcal{C}$, with probability at least $1-2(1-\epsilon)^{M_2}$ of i.i.d selected M_2 samples form $\mathcal{C}$, there are two points z_1, z_2 such that $z_1 \leq B_{\mathrm{F}(\boldsymbol{x})}, z_2 \geq A_{\mathrm{F}(\boldsymbol{x})}$.

~~The probability of there is no points z_1 such that $z_1 \leq B_{\mathrm{F}(\boldsymbol{x})}$~~

In P2, among M_2 i.i.d. samples, the probability that no point lies at distance $\leq B_{\mathrm{F}(\boldsymbol{x})}$ is $(1-\epsilon)^{M_2}$, ~~similar as z_2. So,~~ and similarly for $\geq A_{\mathrm{F}(\boldsymbol{x})}$. Thus both exist simultaneously with probability at ~~most $2(1-\epsilon)^{M_2}$, these two conditions do not hold simultaneously, this is what we want~~least $1-2(1-\epsilon)^{M_2}$.

~~Get the result~~**Conclusion.**

For a $\boldsymbol{x} \sim \mathcal{C}$, by part three, with probability $1-e^{-2N\delta^2}-\beta$ of P1, there are $P_{x\sim\mathcal{C}}(S_{\mathrm{F}(\boldsymbol{x})} \leq C) \geq 1-\alpha$.

When x satisfied $S_{\mathrm{F}(\boldsymbol{x})} \leq C$, and the Part Four is stand in the P2, then there are $B_m - C(A_m - B_m) \leq B_{\mathrm{F}(\boldsymbol{x})} - C(A_{\mathrm{F}(\boldsymbol{x})} - B_{\mathrm{F}(\boldsymbol{x})}) \leq 0$.

Combine them, this is what we want.

□

J.2 SITUATION 2

Now, we show how to proof the theorem 1 when $\boldsymbol{y} \notin \{\mathrm{F}(\boldsymbol{x})\}_{x\in\mathcal{C}}$.

Proof. The proof is a continuation of the previous proof of theorem 3.

Let p satisfy that $\mathrm{P}_{\boldsymbol{x}\sim\mathcal{C}}(S_{\mathrm{F}(\boldsymbol{x})} \leq p) = 1-\alpha$ and $\mathcal{Q} = \{\boldsymbol{x} \in \mathcal{C} : S_{\mathrm{F}(\boldsymbol{x})} \leq p\}$.

Then for a given $\boldsymbol{y}$, let $\boldsymbol{x}^{(q)} = \arg\min_{\boldsymbol{x}\in\mathcal{Q}} \|\boldsymbol{y} - \mathrm{F}(\boldsymbol{x})\|_2$, and $\|\boldsymbol{y} - F(\boldsymbol{x}^{(q)})\|_2 = t$.

For any randomly selected $\{\boldsymbol{x}^{(i)}\}_{i=1}^{M_2}$ samples in P2, let $A'_m = \max_{i\in[M_2]} \|\mathrm{F}(\boldsymbol{x}^{(q)}) - \mathrm{F}(\boldsymbol{x}^{(i)})\|_2$ and $B'_m = \min_{i\in[M_2]} \|\mathrm{F}(\boldsymbol{x}^{(q)}) - \mathrm{F}(\boldsymbol{x}^{(i)})\|_2$.

Then, there are $A_m = \max_{i\in[M_2]} \|y - \mathrm{F}(\boldsymbol{x}^{(i)})\|_2 \geq \max_{i\in[M_2]} \|\mathrm{F}(\boldsymbol{x}^{(q)}) - \mathrm{F}(\boldsymbol{x}^{(i)})\|_2 - \|y - \mathrm{F}(\boldsymbol{x}^{(q)})\|_2 = A'_m - t$, similar, there are $B_m \leq B'_m + t$. So, we have $B_m - C(A_m - B_m) \leq B'_m - C(A'_m - B'_m) + t(1+2C)$.

As shown in the proof of theorem 3, with probability $1 - e^{-2N\delta^2} - \beta$ of P1, there is $C \geq p$, and then, using $\boldsymbol{x}^{(q)} \in \mathcal{Q}$, we have $\mathrm{P}_{\mathrm{P2}}(B'_m - C(A'_m - B'_m) \leq 0) \geq 1 - 2(1-\epsilon)^{M_2}$, which can imply that $\mathrm{P}_{\mathrm{P2}}(B_m - C(A_m - B_m) \leq t(1+2C)) \geq 1 - 2(1-\epsilon)^{M_2}$ by the above result.

And when $B_m - C(A_m - B_m) \leq t(1+2C)$ holds, there are $t \geq \frac{B_m - C(A_m - B_m)}{(1+2C)} = d_{\min}$, and hence $B(\boldsymbol{y}, d_{\min}) \cap \mathcal{Q} = \emptyset$ by the definition of t, which implies that $\mathrm{V}(\boldsymbol{y}, d_{\min}) \leq \alpha$, this is what we want. □

K THE PART THREE FOR SMALL NETWORKS

To demonstrate the superiority of our method in Section 4 and compare it with existing approaches, we show how we can apply Part Three on small-scale object detection based on the feed-forward ReLU neural networks. In this case, the problem in Part Three can be formulated as follows:

$$\begin{aligned} \min_{\boldsymbol{x},\hat{\boldsymbol{x}}} \quad & ||\boldsymbol{y} - \boldsymbol{x}^{(L)}||_2 \\ \text{s.t. (i)} \quad & \boldsymbol{x} \in \mathcal{C}, \quad \hat{\boldsymbol{x}}^{(0)} = \boldsymbol{x}; \\ \text{(ii)} \quad & \boldsymbol{x}^{(i)} = \mathbf{W}^{(i)}\hat{\boldsymbol{x}}^{(i-1)} + \mathbf{b}^{(i)}, \quad i \in [L]; \\ \text{(iii)} \quad & \hat{\boldsymbol{x}}^{(i)} = \max\left(0, \boldsymbol{x}^{(i)}\right), \quad i \in [L-1] \end{aligned}$$

Where $\mathbf{W}^{(i)}$ is the transition matrix of i-th layer in network, $\mathbf{b}^{(i)}$ is the bias vector of i-th layer in network. Unfortunately, finding an exact solution to this problem is NP-complete(Katz et al., 2017). So, we consider bounds on the outputs of hidden-layers: $l_j^{(i)} \leq x_j^{(i)} \leq u_j^{(i)}$. Based on these bounds, let $\mathcal{I}^{-(i)} = \{j : u_j^{(i)} \leq 0\}$, $\mathcal{I}^{+(i)} = \{j : l_j^{(i)} \geq 0\}$ and $\mathcal{I}^{\pm(i)} = \{j : l_j^{(i)} < 0 < u_j^{(i)}\}$, the condition (iii) can be replaced by the following constraints: $\hat{x}_j^{(i)} \geq 0, \hat{x}_j^{(i)} \geq x_j^{(i)}, (u_j^{(i)} - l_j^{(i)})\hat{x}_j^{(i)} \leq u_j^{(i)}x_j^{(i)} - l_j^{(i)}u_j^{(i)},\ j \in \mathcal{I}^{\pm(i)}; \hat{x}_j^{(i)} = x_j^{(i)},\ j \in \mathcal{I}^{+(i)}; \hat{x}_j^{(i)} = 0,\ j \in \mathcal{I}^{-(i)}$. To solve this problem, we prove that such minimum value is equal to the maximum value of the square root of $d(\boldsymbol{\alpha}, \boldsymbol{\nu})$ which is defined in the following theorem. So, we can use the gradient to estimate the maximum of $d(\boldsymbol{\alpha}, \boldsymbol{\nu})$, and use it to be the minimum distance in the part three.

Theorem 4. *Under the setting in this section, we have that:*

$$\min_{\boldsymbol{x}\in\mathcal{C}} ||\mathrm{F}(\boldsymbol{x}) - \boldsymbol{y}||_2 = \max_{\boldsymbol{\alpha}\in[0,1]} \sqrt{d(\boldsymbol{\alpha}, \boldsymbol{\nu})}$$

, which is defined as below:

$$
\begin{aligned}
d(\boldsymbol{\alpha}, \boldsymbol{\nu}) =& \boldsymbol{y}^\top \boldsymbol{\nu}^{(L)} - \frac{\boldsymbol{\nu}^{(L)\top}\boldsymbol{\nu}^{(L)}}{4} - [\boldsymbol{\nu}^{(1)\top}\mathbf{W}^{(1)}]_+ \boldsymbol{u}^{(0)} + [\boldsymbol{\nu}^{(1)\top}\mathbf{W}^{(1)}]_- \boldsymbol{l}^{(0)} \\
& - \sum_{i=1}^{L} \boldsymbol{\nu}^{(i)\top}\boldsymbol{b}^{(i)} + \sum_{i=1}^{L-1} \sum_{j \in \mathcal{I}^{\pm(i)}} \left[\frac{u_j^{(i)} l_j^{(i)} [\hat{\nu}_j^{(i)}]_+}{u_j^{(i)} - l_j^{(i)}} \right]
\end{aligned} \tag{6}
$$

where

$$
\begin{aligned}
&\hat{\nu}_j^{(i)} = \boldsymbol{\nu}^{(i+1)\top}\mathbf{W}_{:,j}^{(i+1)}, \hat{\nu}_j^{(i)} = [\hat{\nu}_j^{(i)}]_+ - [\hat{\nu}_j^{(i)}]_-, \forall i \in \{1, 2, ..., L-1\} \\
&\nu_j^{(i)} = \begin{cases} \hat{\nu}_j^{(i)} & , j \in \mathcal{I}^{+(i)} \\ 0 & , j \in \mathcal{I}^{-(i)} \\ \frac{u_j^{(i)}}{u_j^{(i)} - l_j^{(i)}} [\hat{\nu}_j^{(i)}]_+ - \alpha_j^{(i)} [\hat{\nu}_j^{(i)}]_- & , j \in \mathcal{I}^{\pm(i)} \end{cases} \quad \forall i \in \{1, 2, ..., L-1\}
\end{aligned}
$$

L THE PROOF OF THEOREM 4

We mainly follow the proof idea in (Kotha et al., 2023).

Proof. Consider the $\mathcal{I}^{-(i)} = \{j : u_j^{(i)} \leq 0\}$, $\mathcal{I}^{+(i)} = \{j : l_j^{(i)} \geq 0\}$ and $\mathcal{I}^{\pm(i)} = \{j : l_j^{(i)} < 0 < u_j^{(i)}\}$, such optimization problem tends to the:

$$
\begin{aligned}
\min_{\boldsymbol{x}, \hat{\boldsymbol{x}}} \quad & (\boldsymbol{x}^{(L)} - \boldsymbol{y})^\top (\boldsymbol{x}^{(L)} - \boldsymbol{y}) \\
\text{s.t.} \quad & \boldsymbol{l}^{(0)} \leq \hat{\boldsymbol{x}}^{(0)} \leq \boldsymbol{u}^{(0)} \\
& \boldsymbol{x}^{(i)} = \mathbf{W}^{(i)} \hat{\boldsymbol{x}}^{(i-1)} + \boldsymbol{b}^{(i)}; i = \{1, 2, 3, ..., L\} \\
& \hat{x}_j^{(i)} \geq 0; \ j \in \mathcal{I}^{\pm(i)} \\
& \hat{x}_j^{(i)} \geq x_j^{(i)}; \ j \in \mathcal{I}^{\pm(i)} \\
& (u_j^{(i)} - l_j^{(i)}) \hat{x}_j^{(i)} \leq u_j^{(i)} x_j^{(i)} - l_j^{(i)} u_j^{(i)}, \ j \in \mathcal{I}^{\pm(i)} \\
& \hat{x}_j^{(i)} = x_j^{(i)}, \ j \in \mathcal{I}^{+(i)} \\
& \hat{x}_j^{(i)} = 0, \ j \in \mathcal{I}^{-(i)}
\end{aligned} \tag{7}
$$

Then we use the Lagrange dual of the optimization problem to solve it, the Lagrange dual of the optimization problem is given by:

$$
\begin{aligned}
\min_{\boldsymbol{x},\hat{\boldsymbol{x}}} \max_{\boldsymbol{\lambda},\boldsymbol{\mu},\boldsymbol{\nu},\boldsymbol{\tau}} \quad & (\boldsymbol{x}^{(L)} - \boldsymbol{y})^\top (\boldsymbol{x}^{(L)} - \boldsymbol{y}) \\
& + \sum_{i=1}^{L} \boldsymbol{\nu}^{(i)\top} \left(\boldsymbol{x}^{(i)} - \mathbf{W}^{(i)} \hat{\boldsymbol{x}}^{(i-1)} - \boldsymbol{b}^{(i)} \right) \\
& + \sum_{i=1}^{L-1} \sum_{j \in \mathcal{I}^{\pm(i)}} \mu_j^{(i)} (-\hat{x}_j^{(i)}) \\
& + \sum_{i=1}^{L-1} \sum_{j \in \mathcal{I}^{\pm(i)}} \tau_j^{(i)} (x_j^{(i)} - \hat{x}_j^{(i)}) \\
& + \sum_{i=1}^{L-1} \sum_{j \in \mathcal{I}^{\pm(i)}} \left[\lambda_j^{(i)} \left((u_j^{(i)} - l_j^{(i)}) \hat{x}_j^{(i)} - u_j^{(i)} x_j^{(i)} + l_j^{(i)} u_j^{(i)} \right) \right] \\
\text{s.t.} \quad & \boldsymbol{l}^{(0)} \leq \hat{\boldsymbol{x}}^{(0)} \leq \boldsymbol{u}^{(0)}; \\
& \hat{x}_j^{(i)} = 0, j \in \mathcal{I}^{-(i)}; \hat{x}_j^{(i)} = x_j^{(i)}, j \in \mathcal{I}^{+(i)} \\
& \boldsymbol{\mu} \geq 0;\ \boldsymbol{\tau} \geq 0;\ \boldsymbol{\lambda} \geq 0;
\end{aligned}
\tag{8}
$$

According to it's strong duality, the solution of the dual problem is the same as the primal problem. The dual problem is given by:

$$
\begin{aligned}
\max_{\boldsymbol{\lambda},\boldsymbol{\mu},\boldsymbol{\nu},\boldsymbol{\tau}} \min_{\boldsymbol{x},\hat{\boldsymbol{x}}} \quad & (\boldsymbol{x}^{(L)} - 2\boldsymbol{y} + \boldsymbol{\nu}^{(L)})^\top \boldsymbol{x}^{(L)} + \boldsymbol{y}^\top \boldsymbol{y} - (\boldsymbol{\nu}^{(1)})^\top \mathbf{W}^{(1)} \hat{x}^{(0)} \\
& + \sum_{i=1}^{L-1} \sum_{j \in \mathcal{I}^{+(i)}} (\nu_j^{(i)} - \boldsymbol{\nu}^{(i+1)\top} W_{i,j}^{(i+1)}) x_j^{(i)} \\
& + \sum_{i=1}^{L-1} \sum_{j \in \mathcal{I}^{-(i)}} \nu_j^{(i)} x_j^{(i)} \\
& + \sum_{i=1}^{L-1} \sum_{j \in \mathcal{I}^{\pm(i)}} \Big[\left(\nu_j^{(i)} + \tau_j^{(i)} - \lambda_j^{(i)} u_j^{(i)} \right) x_j^{(i)} \\
& + (-\nu^{(i+1)\top} \mathbf{W}_{:,j}^{(i+1)} - \mu 1_j^{(i)} - \tau_j^{(i)} + (u_j^{(i)} - l_j^{(i)}) \lambda_j^{(i)}) \hat{x}_j^{(i)} \Big] \\
& - \sum_{i=1}^{L} \nu_i^{(i)\top} b_i^{(i)} + \sum_{i=1}^{L-1} \sum_{j \in \mathcal{I}^{(i)}} \lambda_j^{(i)} u_j^{(i)} l_j^{(i)} \\
\text{s.t.} \quad & \boldsymbol{l}^{(0)} \leq \hat{\boldsymbol{x}}^{(0)} \leq \boldsymbol{u}^{(0)};\ \hat{x}_j^{(i)} = 0, j \in \mathcal{I}^{-(i)}; \hat{x}_j^{(i)} = x_j^{(i)}, j \in \mathcal{I}^{+(i)} \\
& \boldsymbol{\mu} \geq 0;\ \boldsymbol{\tau} \geq 0;\ \boldsymbol{\lambda} \geq 0;
\end{aligned}
\tag{9}
$$

Here, we adjust the order of each term in the formula and directly incorporate the constraints $\hat{x}_j^{(i)} = 0, j \in \mathcal{I}^{-(i)}; \hat{x}_j^{(i)} = x_j^{(i)}, j \in \mathcal{I}^{+(i)}$ into the objective function. Then we can minimize $-(\boldsymbol{\nu}^{(1)})^\top \mathbf{W}^{(1)} \hat{x}^{(0)}$ subject to $\boldsymbol{l}^{(0)} \leq \hat{\boldsymbol{x}}^{(0)} \leq \boldsymbol{u}^{(0)}$ according to the each dimension of $\boldsymbol{\nu}^{(1)\top} \mathbf{W}^{(1)}$. If $(\boldsymbol{\nu}^{(1)\top} \mathbf{W}^{(1)})_j > 0$, then $\hat{x}_j^{(0)} = \boldsymbol{u}_j^{(0)}$; Otherwise, $\hat{x}_j^{(0)} = \boldsymbol{l}_j^{(0)}$. Since no additional constraints applied to $\boldsymbol{x}^{(L)}$, we can minimize $(\boldsymbol{x}^{(L)} - 2\boldsymbol{y} + \boldsymbol{\nu}^{(L)})^\top \boldsymbol{x}^{(L)}$ by setting $\boldsymbol{x}^{(L)} = \boldsymbol{y} - \frac{1}{2}\boldsymbol{\nu}^{(L)}$. Then

we get:

$$\begin{aligned}
\max_{\boldsymbol{\lambda},\boldsymbol{\mu},\boldsymbol{\nu},\boldsymbol{\tau}} \min_{\boldsymbol{x},\hat{\boldsymbol{x}}} \quad & \boldsymbol{y}^\top \boldsymbol{\nu}^{(L)} - \frac{\boldsymbol{\nu}^{(L)\top}\boldsymbol{\nu}^{(L)}}{4} - [\boldsymbol{\nu}^{(1)\top}\mathbf{W}^{(1)}]_+ \boldsymbol{u}^{(0)} + [\boldsymbol{\nu}^{(1)\top}\mathbf{W}^{(1)}]_- \boldsymbol{l}^{(0)} \\
& + \sum_{i=1}^{L-1} \sum_{j\in\mathcal{I}^{+(i)}} (\nu_j^{(i)} - \boldsymbol{\nu}^{(i+1)\top} W_{i,j}^{(i+1)}) x_j^{(i)} \\
& + \sum_{i=1}^{L-1} \sum_{j\in\mathcal{I}^{-(i)}} \nu_j^{(i)} x_j^{(i)} \\
& + \sum_{i=1}^{L-1} \sum_{j\in\mathcal{I}^{\pm(i)}} \Big[\left(\nu_j^{(i)} + \tau_j^{(i)} - \lambda_j^{(i)} u_j^{(i)}\right) x_j^{(i)} \\
& + \left(-\nu^{(i+1)\top} \mathbf{W}_{:,j}^{(i+1)} - \mu_j^{(i)} - \tau_j^{(i)} + (u_j^{(i)} - l_j^{(i)})\lambda_j^{(i)}\right) \hat{x}_j^{(i)} \Big] \\
& - \sum_{i=1}^{L} \nu^{(i)\top} b^{(i)} + \sum_{i=1}^{L-1} \sum_{j\in\mathcal{I}^{\pm(i)}} \lambda_j^{(i)} u_j^{(i)} l_j^{(i)} \\
\text{s.t.} \quad & \boldsymbol{\mu} \geq 0;\ \boldsymbol{\tau} \geq 0;\ \boldsymbol{\lambda} \geq 0;
\end{aligned} \tag{10}$$

Since variables $\boldsymbol{x}, \hat{\boldsymbol{x}}$ are unconstrained, any of thier coefficients are nonzero would make thier value to $-\infty$ by the inner minimization. Therefore, the outer maximization require setting all the coefficients of $\boldsymbol{x}, \hat{\boldsymbol{x}}$ to zero. Then we get:

$$\begin{aligned}
\max_{\boldsymbol{\lambda},\boldsymbol{\mu},\boldsymbol{\tau},\boldsymbol{\nu}} \quad & \boldsymbol{y}^\top \boldsymbol{\nu}^{(L)} - \frac{\boldsymbol{\nu}^{(L)\top}\boldsymbol{\nu}^{(L)}}{4} \\
& - [\boldsymbol{\nu}^{(1)\top}\mathbf{W}^{(1)}]_+ \boldsymbol{u}^{(0)} + [\boldsymbol{\nu}^{(1)\top}\mathbf{W}^{(1)}]_- \boldsymbol{l}^{(0)} \\
& - \sum_{i=1}^{L} \boldsymbol{\nu}^{(i)\top} \boldsymbol{b}^{(i)} + \sum_{i=1}^{L-1} \sum_{j\in\mathcal{I}^{\pm(i)}} \lambda_j^{(i)} u_j^{(i)} l_j^{(i)} \\
\text{s.t.} \quad & \boldsymbol{\mu} \geq 0; \boldsymbol{\tau} \geq 0; \boldsymbol{\lambda} \geq 0; \\
& \nu_j^{(i)} - \boldsymbol{\nu}^{(i+1)\top} \mathbf{W}_{:,j}^{(i+1)} = 0, j \in \mathcal{I}^{+(i)} \\
& \nu_j^{(i)} = 0, j \in \mathcal{I}^{-(i)} \\
& \nu_j^{(i)} + \boldsymbol{\tau}_j^{(i)} - \lambda_j^{(i)} u_j^{(i)} = 0; j \in \mathcal{I}^{\pm(i)} \\
& \boldsymbol{\nu}^{(i+1)\top} \mathbf{W}_{:,j}^{(i+1)} = (u_j^{(i)} - l_j^{(i)})\lambda_j^{(i)} - (\mu_j^{(i)} + \tau_j^{(i)}), j \in \mathcal{I}^{\pm(i)}
\end{aligned} \tag{11}$$

Define $\hat{\nu}_j^{(i)} = \boldsymbol{\nu}^{(i+1)\top} \mathbf{W}_{:,j}^{(i+1)}$. Then, we define $(u_j^{(i)} - l_j^{(i)})\lambda_j^{(i)} = [\hat{\nu}_j^{(i)}]_+$ and $\mu_j^{(i)} + \tau_j^{(i)} = [\hat{\nu}_j^{(i)}]_-$, we get the following bound propagation procedure:

$$\begin{aligned}
\max_{\boldsymbol{\alpha},\boldsymbol{\nu}} \quad & \boldsymbol{y}^\top \boldsymbol{\nu}^{(L)} - \frac{\boldsymbol{\nu}^{(L)\top}\boldsymbol{\nu}^{(L)}}{4} - [\boldsymbol{\nu}^{(1)\top}\mathbf{W}^{(1)}]_+ \boldsymbol{u}^{(0)} + [\boldsymbol{\nu}^{(1)\top}\mathbf{W}^{(1)}]_- \boldsymbol{l}^{(0)} \\
& - \sum_{i=1}^{L} \boldsymbol{\nu}^{(i)\top} \boldsymbol{b}^{(i)} + \sum_{i=1}^{L-1} \sum_{j\in\mathcal{I}^{\pm(i)}} \left[\frac{u_j^{(i)} l_j^{(i)} [\hat{\nu}_j^{(i)}]_+}{u_j^{(i)} - l_j^{(i)}} \right] \\
\text{s.t.} \quad & \hat{\nu}_j^{(i)} = \boldsymbol{\nu}^{(i+1)\top} \mathbf{W}_{:,j}^{(i+1)}, \hat{\nu}_j^{(i)} = [\hat{\nu}_j^{(i)}]_+ - [\hat{\nu}_j^{(i)}]_-, \forall i \in \{1, 2, ..., L-1\} \\
& \nu_j^{(i)} = \begin{cases} \hat{\nu}_j^{(i)} & , j \in \mathcal{I}^{+(i)} \\ 0 & , j \in \mathcal{I}^{-(i)} \\ \frac{u_j^{(i)}}{u_j^{(i)} - l_j^{(i)}} [\hat{\nu}_j^{(i)}]_+ - \alpha_j^{(i)} [\hat{\nu}_j^{(i)}]_- & , j \in \mathcal{I}^{\pm(i)} \end{cases} \quad \forall i \in \{1, 2, ..., L-1\} \\
& 0 \leq \alpha_j^{(i)} \leq 1, j \in \mathcal{I}^{\pm(i)}
\end{aligned} \tag{12}$$

Here $\alpha_j^{(i)}$ is the optimizable parameter controlling the relaxation of neuron j in layer i introduced in (Xu et al., 2021). This is what we want. □

M THE PROOF OF THEOREM 2

Theorem. *Using the notation from the three algorithms above. Given* $\alpha, \beta, \delta, \epsilon \in (0,1)$ *satisfying* $(1-2\epsilon)^M - \delta > 0$, $N \cdot ((1-2\epsilon)^M - \delta) > \frac{2}{\alpha}\ln(\frac{1}{\beta}) + 2 + \frac{2}{\alpha}\ln(\frac{2}{\alpha})$ *and* $N_2 \geq [\frac{2\ln 1/\beta}{\alpha} + 2 + \frac{2\ln 2/\alpha}{\alpha}]$. *Then, after executing the algorithms defined above, if, for a sample* $\boldsymbol{x}$, *these algorithms output the 'safe' after* T *refinement turns, then with probability* $1 - T(e^{-2N\delta^2} + \beta + 2(1-\epsilon)^{M_2}) - \beta$ *of part one and three, there are* $\mathrm{P}_{\boldsymbol{x}\sim\mathcal{C}}(\boldsymbol{x} \text{ issafe}) > 1 - (1+T)\alpha$.

Proof. We use the sample fact that $\mathrm{P}(A \cap B) \geq \mathrm{P}(A) + \mathrm{P}(B) - 1$ to get the following result:

From Theorem 1, after T refinement steps, the reduction in the probability of the system being certifiably safe due to these refinements is at most $T\alpha$. This statement holds with a confidence of at least $T\left(1 - (e^{-2N\delta^2} + \beta + 2(1-\epsilon)^{M_2})\right) - (T-1) = 1 - T\left(e^{-2N\delta^2} + \beta + 2(1-\epsilon)^{M_2}\right)$.

According to Proposition 1, we can get that the probability of the box is safe is $1 - T\alpha - \alpha$ with probability $1 - T(e^{-2N\delta^2} + \beta + 2(1-\epsilon)^{M_2}) + (1-\beta) - 1 = 1 - T(e^{-2N\delta^2} + \beta + 2(1-\epsilon)^{M_2}) - \beta$.

Then we can get that the probability $\mathrm{P}(\boldsymbol{x} \text{ is safe}) > 1 - (1+T)\alpha$ with probability $1 - T(e^{-2N\delta^2} + \beta + 2(1-\epsilon)^{M_2}) - \beta$.

Then we can get the final result. □

Remark 6. *Note we do not assume that the Part 1 and Part 3 are independent, so we can reuse the samples in Part 1 to Part 3.*

N EXPERIMENT DETAILS

Detailed Setting: We use $N_1 = 30000$, $N_2 = 5000$, $N = 3000$, $M = 10$, $M_2 = 2000$, $\alpha = 0.0099$, $\beta = 0.0099$, $\epsilon = 1/200$, $\delta = 0.1$ as our default setting. As we do not assume that the Part 1 and Part 3 are independent, samples from Part 1(Algorithm 2) can be also used in Part 3(Algorithm 4), so we only need to sample $N_1 + N_2 + M_2 = 37000$ samples in total, where $N_1 = N \cdot M$. We use one turn refinement($T = 1$) for all the experiments.

Attack Setting: We use square attack with 5000 iterations for each box. For PGD, we use 20 iterations with step size 1/255 and 2/255. Each iteration, We find the box with the highest IoU and same class with the ground truth box, then use the GIoU of them as the loss function. Note each turn may attack a different box. We found that this is a strong attack for YOLO networks.

Server Setting: We use a server with 8 NVIDIA V100 32G GPUs, 40 Intel(R) Xeon(R) Gold 5215 CPUs at 2.50GHz and 503GB memory. The code is implemented in Python with Gurobi and PyTorch.

Sample Calculation: According to the theorem 2, we can get that the sample number is $N_1 + N_2 + M_2 = 37,000$ for each box. Note this is the sample number of full algorithm. ~~For RCP$_N$ methods, from Appendix C.1, we need:~~

~~$$N \geq [\frac{2\ln 1/\beta}{\alpha} + 2d_0 + \frac{2d_0\ln 2/\alpha}{\alpha}]$$~~

~~to make sure thatwith $1-\beta$ confidence, the range is over-approximate the real range with probability with probability $1-\alpha$. Here we use $d_0 = 640 \times 640 \times 3, \alpha = 0.02, \beta = 0.02$ to achieve comparable results, then we have $N \geq 568341303$. Note that this is the sample number for Part~~

Our work (Part 1~~only.~~) aims to find a tight hyper-rectangle bound $\mathcal{Z}$ for the d_L-dimensional vector output space $\{\mathrm{F}(x)\}_{x\in\mathcal{C}}$. In this context, a more appropriate application of RCP$_N$ as a baseline for

this task would be to directly estimate an upper bound for each dimension in this d_L-dimensional space. The decision-variable dimension for this problem would be $d = d_L$ (i.e., the network output dimension), corresponding to finding a vector $\boldsymbol{u} \in \mathbb{R}^{d_L}$ such that, for all inputs $\boldsymbol{x} \in \mathcal{C}$, the network output $\mathbf{F}(\boldsymbol{x}) \leq \boldsymbol{u}$ element-wise. Based on this, the required sample count would be:

$$N \geq \left\lceil \frac{2\ln 1/\beta}{\alpha} + 2d_L + \frac{2d_L \ln(2/\alpha)}{\alpha} \right\rceil \approx 10^9$$

Alternatively, if one were to use the RCP_N method to directly verify the entire problem, this could be framed as computing an IoU lower bound for each predicted box against its corresponding ground-truth (GT) box. The decision variable dimension in this case would be $d = (80 \times 80 + 40 \times 40 + 20 \times 20) \times 3 = 25,200$ (i.e., the number of bounding boxes). Based on this, the required sample count would be:

$$N \geq \left\lceil \frac{2\ln 1/\beta}{\alpha} + 2d + \frac{2d \ln(2/\alpha)}{\alpha} \right\rceil \approx 10^7$$

The order of magnitude of the sample size under either estimation is still prohibitively large for any practical application.

For the PAC-based methods, according to (Li et al., 2022), to achieve $1-\alpha$ probability with $1-\beta$ confidence, we need N satisfies:

$$N \geq \frac{2}{\beta} \times \left(\log(\frac{1}{\alpha}) + d_0 \right) \geq 122880391.$$

Here we also use $\alpha = 0.02, \beta = 0.02$.

In terms of randomized smoothing(Cohen et al., 2019), the number of samples required is strongly correlated with the standard deviation (σ) and the certified radius. For example, to certify a radius of 2/255 against noise with a standard deviation of $\sigma = \frac{1}{255\times 3}$, the randomized smoothing method would require an impractical number of samples (approximately 3.9×10^9). This severely limits its application in real-world scenarios.

Our method, by introducing $\boldsymbol{v}_{\max}$ and the scalar c_1, reduces the entire verification problem to a $d = 1$ scalar optimization (for c_1) plus a constrained optimization (MIQP). Thus the required sample size is significantly reduced to a manageable level, making it feasible for practical applications.

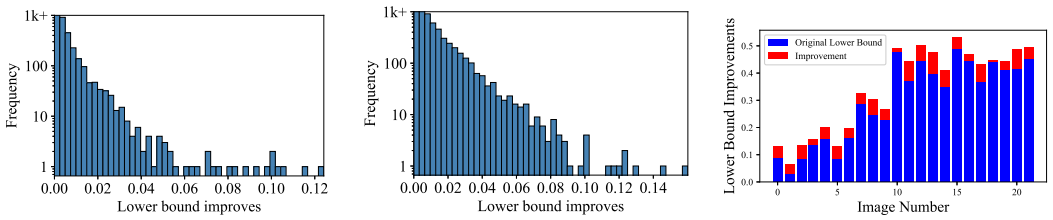
Remark 7. *Note that without refinement (only Part 1 and Part 2), our method can achieve 99% confidence and 99% probability with* $\alpha = 0.0099, \beta = 0.0099$. *After refinement, the true confidence and probability are both 98%. For other two methods, we use error rate* 0.02 *and significance level* 0.02*, to get the same confidence and probability.*

Clarification on Sample Complexity There are two distinct sampling procedures used in our experiments, which serve different purposes:

- **Verification Samples (37,000):** This is the sample budget used by our algorithm to issue the certificate. For each ground-truth box, we use 37,000 samples ($N_1 + N_2 + M_2$) to construct $\mathcal{Z}$ and the refinement constant C, as prescribed by Theorem 2.
- RCP_N **Baseline Samples (10^6 in Table 1):** For the RCP_N baseline in Table 1, we also use 10^6 samples to compute empirical robustness, but this does not yield comparable theoretical guarantees. To achieve our target (98% robustness, 98% confidence), RCP_N would theoretically require about 11.6 million samples. With only 10^6 samples, RCP_N can only provide an 86% robustness guarantee at 98% confidence.
- **Evaluation Samples (10^6 in Table 2):** Independently of the verification algorithm, we draw 10^6 additional uniform perturbations to empirically estimate the "ground truth" robustness. These samples are only used to calculate evaluation metrics (TPR, FPR, TNR, FNR, and CRA) and do not inform the certificate itself.

Table 3: Ground Truth Information for Table 2. #RB: Number of robust boxes; #NRB: Number of non-robust boxes

Model	ε	#RB ($\tau = 0.5$)	#NRB($\tau = 0.5$)	#RB ($\tau = 0.7$)	#NRB($\tau = 0.7$)
yolo11m	1/255	363	164	335	192
yolo11m	2/255	346	181	315	212
yolo11x	1/255	390	137	348	179
yolo11x	2/255	379	148	335	192
yolov3-sppu	1/255	364	163	332	195
yolov3-sppu	2/255	356	171	320	207
yolov3u	1/255	374	153	338	189
yolov3u	2/255	365	162	327	200
yolov5mu	1/255	354	173	316	211
yolov5mu	2/255	344	183	301	226
yolov5xu	1/255	387	140	354	173
yolov5xu	2/255	372	155	337	190
yolov8m	1/255	376	151	342	185
yolov8m	2/255	361	166	327	200
yolov8x	1/255	385	142	348	179
yolov8x	2/255	370	157	337	190



(a) Improvements with $\varepsilon = \frac{1}{255}$ (b) Improvements with $\varepsilon = \frac{2}{255}$ (c) Improvements in CNNs (Cohen et al., 2024)

Figure 7: Verification bound improvement after Part 3

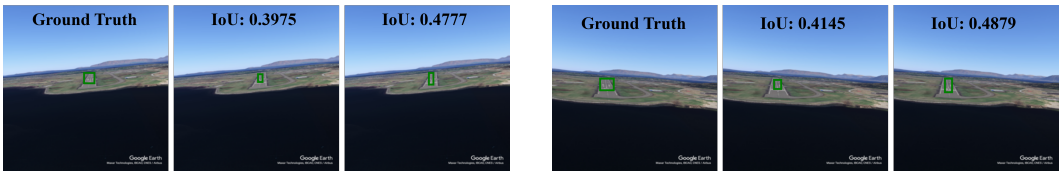
Ground Truth information of Table 2: Table 3 shows the ground truth information of Table 2. Here, #RB means the number of robust boxes, and #NRB means the number of non-robust boxes. The ground truth is calculated by 10^6 uniform perturbations for each box.

O The Effectiveness of NMS Verification

The NMS optimization step is highly efficient. Verifying one ground-truth box against all candidate boxes takes an average of only 4.9 seconds. This efficiency arises because we only perform sound verification: boxes that could potentially cause unsafe behavior are filtered out in advance, leaving only a small number of boxes to be verified.

P The Effect of Part 3

This section shows that Part 3 in Section 4 and Appendix K is effective for both large-scale networks (YOLO) and small-scale CNNs. For the LARD dataset, we use a six-layer CNN network provided in (Cohen et al., 2024). As seen in Figure 7c, with Part 3, the IoU lower bound we obtain is higher than the bound obtained without using Part 3. This improvement occurs because Part 3 reduces the over-approximation of the network's output, implying our bound is closer to the true bound. Note that we use sound formal verification on small object detection, instead of probabilistic verification, so that the network can indeed achieve such a bound. For YOLO, we also show the effect of Part 3



(a) Image 378 (b) Image 397

Figure 8: The effect of Part 3 in small object detection. The middle panel shows the bound from prior methods; the right panel shows our improved bound.

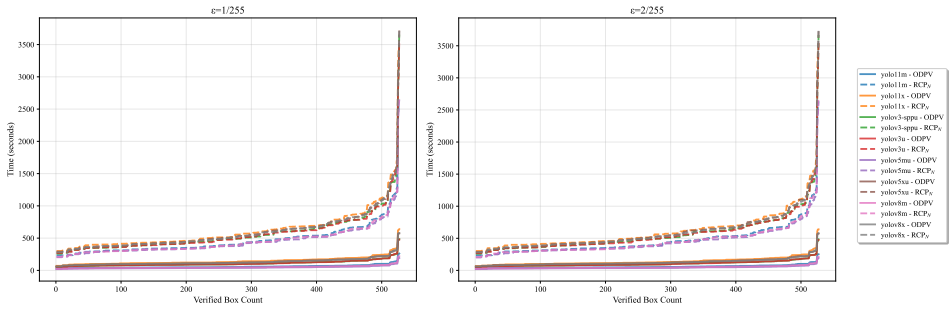


Figure 9: Verification time comparison

in Fig. 7a and Fig. 7b. After refinement, most of the lower bounds are improved, which means Part 3 is also helpful in YOLO networks.

We observe that the first iteration yields the most significant gains. Since further refinement linearly accumulates confidence loss as described in Theorem 2, employing $T = 1$ represents a favorable trade-off in most cases.

Q TIME COMPARISON

Fig. 9 shows the detailed time comparison of our method and RCP_N on different YOLO models with $\varepsilon = \frac{1}{255}$ and $\varepsilon = \frac{2}{255}$. The results show that our method is significantly faster than RCP_N in all images.

R REAL-WORLD EXAMPLES.

We take 40 images from sensor cameras of our autonomous vehicles and annotated ground truth by ourselves. Table 4 shows the results of our method on these images. The results show that our method can also work well in real-world scenarios. Our method still uses less time and achieve better bounds than RCP_N. Both CRA are high in these images, which means most boxes verified as robust by our method are reasonably robust.

S COMPARED WITH MEDIAN SMOOTHING

Table 5 compares our method with median smoothing (MS) (Cohen et al., 2019) under normal distribution on 50 images from the COCO dataset. We set the standard deviation of normal distribution as ~~$\sigma = \frac{1}{255*3}$ and $\sigma = \frac{2}{255*3}$~~ $\sigma = \frac{1}{255\times3}$ and $\sigma = \frac{2}{255\times3}$. The results show that our method significantly achieves a smaller mean absolute difference of IoU lower bounds relative to the worst-case input found by the PGD attack(Δ_{PGD}), indicating more precise IoU lower bounds. Besides, in most cases, the CRA of our method is higher than median smoothing. This also prove that our method works well in different distributions.

Table 4: Real world examples of our method.

model	ε	method	time	Δ_{PGD}		CRA	
				$\tau = 0.5$	$\tau = 0.7$	$\tau = 0.5$	$\tau = 0.7$
yolo11x	1/(3 * 255)	RCP_N	571.0	0.53	0.52	**1.00**	**1.00**
		ODPV	**32.1**	**0.47**	**0.44**	**1.00**	**1.00**
	2/(3 * 255)	RCP_N	569.6	0.57	0.55	**1.00**	**1.00**
		ODPV	**32.1**	**0.43**	**0.37**	**1.00**	**1.00**

Table 5: Comparison of our method with median smoothing. Δ_{PGD} denotes the mean absolute difference of IoU lower bounds relative to the PGD attack. Bold values indicate the best performance.

model	ε	method	Δ_{PGD}		CRA	
			$\tau = 0.5$	$\tau = 0.7$	$\tau = 0.5$	$\tau = 0.7$
yolo11x	1/255	RCP_N	0.45	0.47	**1.00**	**1.00**
		ODPV	**0.42**	**0.40**	0.99	0.98
		MS	0.44	0.45	0.99	0.98
	2/255	RCP_N	0.59	0.56	**1.00**	**1.00**
		ODPV	**0.54**	**0.47**	**1.00**	**1.00**
		MS	0.59	0.53	0.96	0.97

T Ablation Study of Parameters

Fig. 10 shows the ablation study of parameters η and ι. The results show that our method still maintains more accurate bounds than RCP_N under different parameters. Besides, the results show that a larger η and a larger ι will lead the bounds to be closer to the worst-case bound found by PGD attack. This is because a larger η and a larger ι will results in a stricter NMS condition, fewer boxes can remain and the resulting boxes will be more robust.

U Broader Impact

Our method focuses on verifying the safety of the object detection model, which may help people to better understand the model and give a safety metric. We do not think there is a negative social impact of our method as our method is not used to attack the model.

V LLM Usage Statement

In the preparation of this manuscript, a Large Language Model (LLM) was utilized as a writing assistance tool. Its use was strictly confined to language polishing, which includes proofreading for grammatical errors, improving sentence structure, and enhancing the overall clarity and readability of the text.

All core intellectual contributions—including the research ideation, paper structure, and the initial drafting of the content—are the original work of the authors. The LLM did not contribute to the formulation of any hypotheses, experimental results, or conclusions presented herein. The authors have reviewed all AI-generated suggestions and take full responsibility for the final content of this paper.

W Description of hyperprameters

Our framework consists of three parts (Algorithms 1-4): Part 1 (Output Approximation), Part 2 (NMS Verification), and Part 3 (Counterexample-Based Refinement). Only Parts 1 and 3 are probabilistic; Part 2 is based on sound MIQP constraints and introduces no additional probabilistic error.

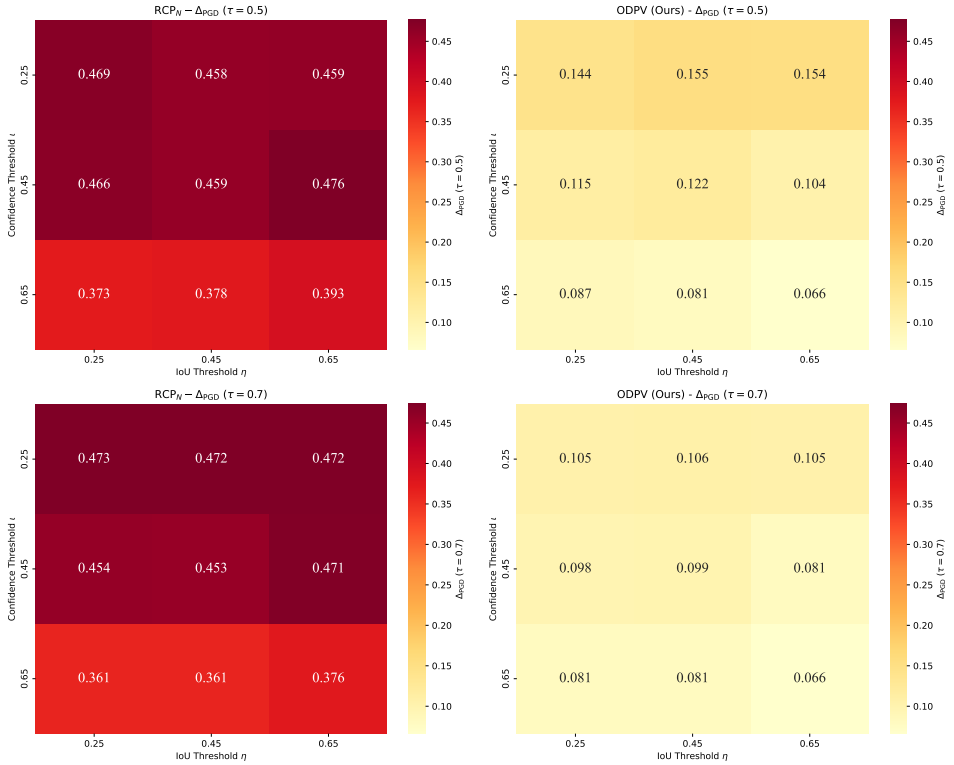


Figure 10: Ablation study of parameters η, ι

W.1 Role of Parameters and Failure Events

We now describe the role of the four key parameters $\alpha, \beta, \delta, \epsilon$ used in our theoretical guarantees and experiments.

- α **(Error Rate)**
  - Appears in Definition 1 (OD PAC-Verification), Proposition 1 (Part 1), Theorem 1 (Part 3), and Theorem 2 (Overall Guarantee).
  - Controls the *allowable violation probability* of the OD property under the input distribution: In Theorem 2, if the algorithm returns "safe" after at most T refinement steps, we guarantee
    $$\mathrm{P}_{\boldsymbol{x}\sim\mathcal{C}}(\boldsymbol{x} \text{ is safe}) > 1 - (1+T)\,\alpha.$$
  - In our experiments, we set $\alpha = 0.0099$, so with $T = 1$ we obtain a lower bound on the safety probability for each certified box of approximately $1 - 0.0198 \approx 0.98$.
- β **(Significance / Confidence)**
  - Used for the scenario-type bounds in Part 1 (Prop. 1) and Part 3 (Thm. 1), and combined in Theorem 2.
  - Controls the *confidence* with which the above probabilistic claim holds, i.e., the probability that the inequality regarding $\mathrm{P}(\boldsymbol{x} \text{ is safe})$ is valid under the randomness of our sampling process. Theorem 2 gives the overall confidence:
    $$\mathrm{P}\left[\mathrm{P}_{x\sim\mathcal{C}}(\boldsymbol{x} \text{ is safe}) > 1 - (1+T)\alpha\right] \geq 1 - T\left(e^{-2N\delta^2} + \beta + 2(1-\epsilon)M_2\right) - \beta.$$
  - We use $\beta = 0.0099$; with our default N, δ, ϵ, M_2, and $T = 1$, this lower bound is ≈ 0.98 (reported in Appendix N).
- δ **(Concentration slack in Part 3)**

  - Appears only in Theorem 1 / Theorem 2. It is the slack term in the Hoeffding bound, controlling how many of the N sampled points in Part 3 have "good" local neighborhoods (i.e., have sufficient neighbors within the interval $[B_{\mathrm{E}(\boldsymbol{x})}, A_{\mathrm{E}(\boldsymbol{x})}]$).
  - The condition

$$N((1-2\epsilon)M - \delta) \;>\; \frac{2}{\alpha}\ln\frac{1}{\beta} + 2 + \frac{2}{\alpha}\ln\frac{2}{\alpha}$$

    ensures that with probability at least $1-\beta$, enough sample points are "good" to apply the RCP_N-style scenario bound to the refinement constant C.
  - We fix $\delta = 0.1$ in all experiments (Appendix N).
- ϵ **(width of the local uncertainty interval in Part 3)**
  - Used in Theorem 1 to define the probability mass of the local distance interval $[B_{\mathrm{E}(\boldsymbol{x})}, A_{\mathrm{E}(\boldsymbol{x})}]$. For each $\boldsymbol{x}$, at least a $1-2\epsilon$ fraction of the M local perturbations fall into this interval.
  - The term $2(1-\epsilon)M_2$ in Theorem 2 bounds the probability that the empirical estimation of this interval fails.
  - We set $\epsilon = 1/200$ in all experiments.

In practice, one needs to select appropriate $\alpha, \beta, \delta, \epsilon$ based on the desired probability thresholds and then calculate the sample sizes N_1, N_2, N, M, M_2. Below we discuss the relationship between sample sizes and these parameters.

W.2 Sample Size Relationships

The main sample size constraints appear in Theorem 2:

- **Part 1 Sampling:**

$$N_2 \geq \frac{2}{\alpha}\ln\frac{1}{\beta} + 2 + \frac{2}{\alpha}\ln\frac{2}{\alpha}.$$

  For fixed α, β, we have $N_2 = O(\frac{1}{\alpha}\log\frac{1}{\beta})$. Importantly, there is **no explicit dependency on the output dimension** d_L in this bound.
- **Part 3 Sampling:** With fixed δ, ϵ, M, the bound on N can be written as

$$N \;\geq\; \frac{1}{(1-2\epsilon)M - \delta}\Big(\frac{2}{\alpha}\ln\frac{1}{\beta} + 2 + \frac{2}{\alpha}\ln\frac{2}{\alpha}\Big),$$

  Thus, for fixed δ, ϵ, M, we again have $N = O(\frac{1}{\alpha}\log\frac{1}{\beta})$.

In the experiments, we instantiate these quantities with the following values:

$$N_1 = 30{,}000, \quad N_2 = 5{,}000, \quad N = 3{,}000, \quad M = 10, \quad M_2 = 2{,}000,$$

and $\alpha = \beta = 0.0099$, $\epsilon = 1/200$, $\delta = 0.1$ (Appendix N). Because the samples from Part 1 are reused in Part 3 (Remark 6), the total number of samples per box is:

$$N_{\text{total}} = N_1 + N_2 + M_2 = 37{,}000$$

This quantity is independent of d_L and forms the basis for the claim "achieving 98% guarantee at 98% confidence with 37,000 samples" in Section 6.

W.3 Notation for Sampling Parameters

Throughout Proposition 1, Theorem 1 and 2, the symbols $N_1, N_2, N, M, M_2, A'_i, B'_i, A_m, B_m$ and C are inherited from Algorithms 2 and 4 as follows. In Algorithm 2, N_1 denotes the number of samples used to estimate the per-coordinate scale vector $\boldsymbol{v}_{\max}$, and N_2 is

the number of samples used to compute the smallest scaling factor c_1 in the optimization problem equation 1, which yields the output hyper-rectangle $\mathcal{Z}$. In Algorithm 4, N is the number of reference points $\{\boldsymbol{x}^{(i)}\}_{i=1}^{N}$ drawn from $\mathcal{C}$ in Step One, and M is the number of auxiliary samples $\{\boldsymbol{x}^{(i,j)}\}_{j=1}^{M}$ drawn for each reference point. For each $i \in [N]$, we define $A'_i = \max_{j\in[M]} \|\mathrm{F}(\boldsymbol{x}^{(i,j)}) - \mathrm{F}(\boldsymbol{x}^{(i)})\|_2$ and $B'_i = \min_{j\in[M]} \|\mathrm{F}(\boldsymbol{x}^{(i,j)}) - \mathrm{F}(\boldsymbol{x}^{(i)})\|_2$, and the constant C is updated as $C \leftarrow \max\{C, B'_i/(A'_i - B'_i)\}$. In Step Two of Algorithm 4, M_2 denotes the number of additional samples $\{\boldsymbol{x}^{(i)}\}_{i=1}^{M_2}$ drawn from $\mathcal{C}$ to estimate the distance from a candidate output $\boldsymbol{y}$ to the true output set $\{\mathrm{F}(\boldsymbol{x})\}_{\boldsymbol{x}\in\mathcal{C}}$, with $A_m = \max_{i\in[M_2]} \|\mathrm{F}(\boldsymbol{x}^{(i)}) - \boldsymbol{y}\|_2$ and $B_m = \min_{i\in[M_2]} \|\mathrm{F}(\boldsymbol{x}^{(i)}) - \boldsymbol{y}\|_2$. These quantities are then combined to form the estimator $d_{\min} = \max\{(B_m - C(A_m - B_m))/(1 + 2C), 0\}$ used in Part 3.

X EXTENDING OUR METHOD

We acknowledge that different architectures and threat models may necessitate distinct verification strategies, making a universal solution challenging. Nonetheless, for many common threats, our proposed method can be adapted by modifying only the encoding in Part 2, while keeping Parts 1 and 3 unchanged. The following describes how to adapt our method to verify two additional threats: class misidentification and false appearances.

X.1 EXTENDING TO CLASS MISIDENTIFICATION

Algorithm 6 Soundness Class Misidentification Verification for NMS

Require: Constraints $\mathcal{Z} = \mathcal{H} \setminus \mathcal{S}$; input $\boldsymbol{x}$; output $\boldsymbol{y}$; ground truth box_{gt}; confidence threshold ι.
Ensure: Calculate per-box upper bounds $\{\tau_{\mathrm{mis}}(i)\}_{i=1}^{n_{\boldsymbol{x}}}$ for class misidentification.

```
{box̄_i}_{i=1}^{n_x} ← CONSTRUCT_ABSTRACT_BOX(Z)
for all box̄_i ∈ {box̄_k}_{k=1}^{n_x} do
    τ_mis(i) ← 0                                   ▷ Initialize upper bound for misidentification IoU of box i
    if ∀k ∈ [n] \ {Class(box_gt)}, p_{i,Class(box_gt)} ≥ p̄_{i,k} then
        continue   ▷ Skip boxes that must match box_gt class (never misclassified w.r.t. this GT)
    end if
    if c̄_i ≥ ι then                                ▷ Box i may pass the confidence threshold in some realization
        τ_mis(i) ← IOU_UPPER_BOUNDS(box̄_i, box_gt)  ▷ Worst-case IoU to box_gt when box_i is potentially of a wrong class
    end if
end for
return {τ_mis(i)}_{i=1}^{n_x}
```

Formalization of the property (bad Event): Given an input constraint set $\mathcal{C}$, if there exists an input $\boldsymbol{x} \in \mathcal{C}$ such that after processing by the network F and NMS module, the output set $\mathrm{N}(\mathrm{F}(\boldsymbol{x}))$ contains a predicted box box_i that has an IoU $\geq \tau$ with some GT box box_{gt}, but their predicted and GT classes do not match, we consider this a class misidentification. Formally:

$$\exists \boldsymbol{x} \in \mathcal{C},\ \exists box_i \in \mathrm{N}(\mathrm{F}(\boldsymbol{x})),\ \exists box_{\mathrm{gt}} \in \mathcal{G} :\ \mathbb{I}(\mathrm{class}(box_i) \neq \mathrm{class}(box_{\mathrm{gt}})) \cdot \mathrm{IoU}(box_i, box_{\mathrm{gt}}) \geq \tau.$$

We want to prove that class misidentification does **not** occur, which is the negation of the bad event, equivalently written as:

$$\forall \boldsymbol{x} \in \mathcal{C},\ \forall box_i \in \mathrm{N}(\mathrm{F}(\boldsymbol{x})),\ \forall box_{\mathrm{gt}} \in \mathcal{G} :\ \mathbb{I}(\mathrm{class}(box_i) \neq \mathrm{class}(box_{\mathrm{gt}})) \cdot \mathrm{IoU}(box_i, box_{\mathrm{gt}}) < \tau.$$

To this end, we can define a worst-case function:

$$\Phi_{\mathrm{cls}}(\boldsymbol{x}) = \max_{box_i \in \mathrm{N}(\mathrm{F}(\boldsymbol{x})),\, box_{\mathrm{gt}} \in \mathcal{G}} \left(\mathbb{I}(\mathrm{class}(box_i) \neq \mathrm{class}(box_{\mathrm{gt}})) \cdot \mathrm{IoU}(box_i, box_{\mathrm{gt}})\right),$$

The property holds if and only if: $\sup_{\boldsymbol{x}\in\mathcal{C}} \Phi_{\text{cls}}(\boldsymbol{x}) < \tau$.

Let $\mathcal{Z}$ be the over-approximation of the pre-NMS network output range $\{\mathrm{F}(\boldsymbol{x})\}_{\boldsymbol{x}\in\mathcal{C}}$ obtained in Part 1.Since our NMS analysis in Part 2 is applied uniformly over all $y \in \mathcal{Z}$, we can compute an upper bound of $\mathbb{I}(\text{class}(box_i) \neq \text{class}(box_{\text{gt}})) \cdot \text{IoU}(box_i, box_{\text{gt}})$ on $\mathcal{Z}$ over all possible predicted boxes and GT pairs:

$$U_{\text{cls}} = \sup_{box_i \in \mathcal{Z}, box_{\text{gt}} \in \mathcal{G}} \left(\mathbb{I}(\text{class}(box_i) \neq \text{class}(box_{\text{gt}})) \cdot \text{IoU}(box_i, box_{\text{gt}})\right).$$

If we can prove $U_{\text{cls}} < \tau$, then it is impossible for any $\boldsymbol{x} \in \mathcal{C}$ and any prediction/GT pair to have an IoU $\geq \tau$ and a class mismatch. Thus, we can certify that no class misidentification occurs within $\mathcal{C}$, and the property is verified.

Algorithm 6 shows how to compute per-box upper bounds on the misidentification IoU in Part 2.

X.2 Extending to False Appearances

Algorithm 7 Soundness False Appearance Verification for NMS

Require: Constraints $\mathcal{Z} = \mathcal{H} \setminus \mathcal{S}$; input $\boldsymbol{x}$; output $\boldsymbol{y}$; set of ground truth boxes $\mathcal{G}$; confidence threshold ι.
Ensure: Calculate per-box lower bounds $\{\tau_{\text{FA}}(i)\}_{i=1}^{n_{\boldsymbol{x}}}$ on $\max_{box_{\text{gt}}\in\mathcal{G}} \text{IoU}(box_i, box_{\text{gt}})$.

```
{box̄_i}_{i=1}^{n_x} ← CONSTRUCT_ABSTRACT_BOX(Z)
for all box̄_i ∈ {box̄_k}_{k=1}^{n_x} do
    τ_FA(i) ← 0                          ▷ Initialize lower bound for maximal IoU to GTs of box i
    if c̄_i < ι then
        continue      ▷ Box i can never become a high-confidence detection, ignore it for False Appearance
    end if
    lb ← 0                               ▷ Lower bound on max_{box_gt∈G} IoU(box_i, box_gt)
    for all box_gt ∈ G do
        lb_gt ← IOU_LOWER_BOUNDS(box̄_i, box_gt)
        lb ← max(lb, lb_gt)              ▷ Aggregate lower bounds to over-approximate max_{box_gt} IoU
    end for
    τ_FA(i) ← lb
end for
return {τ_FA(i)}_{i=1}^{n_x}
```

Formalization of the property (bad Event): Given an input constraint set $\mathcal{C}$, if there exists an input $\boldsymbol{x} \in \mathcal{C}$ and a predicted box $box_i \in \mathrm{N}(\mathrm{F}(\boldsymbol{x}))$ whose IoU with all GT boxes is less than τ, we consider this a False Appearance:

$$\exists \boldsymbol{x} \in \mathcal{C},\ \exists box_i \in \mathrm{N}(\mathrm{F}(\boldsymbol{x})),\ \forall box_{\text{gt}} \in \mathcal{G},\ \text{IoU}(box_i, box_{\text{gt}}) < \tau.$$

Equivalently, define the **maximum IoU** for each predicted box with all GT boxes:

$$\text{IoU}_{\max}(box_i, \boldsymbol{x}) = \max_{box_{\text{gt}}} \text{IoU}(box_i, box_{\text{gt}}),$$

The bad event can be written as:

$$\exists \boldsymbol{x} \in \mathcal{C},\ \exists box_i \in \mathrm{N}(\mathrm{F}(\boldsymbol{x})) :\ \text{IoU}_{\max}(box_i, \boldsymbol{x}) < \tau.$$

We want to prove "no false appearances occur", which is the negation of the bad event:

$$\forall \boldsymbol{x} \in \mathcal{C},\ \forall box_i \in \mathrm{N}(\mathrm{F}(\boldsymbol{x})) :\ \text{IoU}_{\max}(box_i, \boldsymbol{x}) \geq \tau.$$

We can further define:

$$\Phi_{\mathrm{FA}}(\boldsymbol{x}) = \min_{box_i \in \mathrm{N}(\mathrm{F}(\boldsymbol{x}))} \mathrm{IoU}_{\max}(box_i, \boldsymbol{x}),$$

and the property holds if and only if $\inf_{\boldsymbol{x} \in \mathcal{C}} \Phi_{\mathrm{FA}}(\boldsymbol{x}) \geq \tau$.

Algorithm 7 shows how to compute per-box lower bounds on the maximum IoU to GT boxes in Part 2.

For any given property, we first formalize the attack and verification objective as described in Section 3 and the process above. Then, as in Section 4 and above, we adapt Part 2 of the algorithm (e.g., by adjusting the MIQP constraints) based on the specific verification objective. We will add a discussion of these and other potential extensions in the revised manuscript, and specify how Part 2 of the algorithm should be modified for these two threats.

X.3 Evaluation

To assess the effectiveness of our proposed method under diverse noise conditions and threat types, we conduct experiments using four distinct noise models and verify the False Appearance (FA) detection performance of our YOLO11x model. We define a noise tensor $\mathbf{N}_x$ and set the perturbation magnitude to $\varepsilon = 1/255$. The specific noise distributions and their corresponding real-world motivations are outlined below:

- **Uniform**: $\mathbf{N}_x \sim \mathcal{U}(-\varepsilon, \varepsilon)$ (Quantization/Uncertainty).
- **Gaussian**: $\mathbf{N}_x \sim \mathcal{N}(0, (\varepsilon/3)^2)$ (Sensor Readout Noise).
- **Salt-and-Pepper**: $\pm\varepsilon$ impulse noise with $p = 0.05$ (Transmission Faults).

We randomly select 10 images from the COCO validation set and apply each noise model with $\varepsilon = 1/255$ to generate noisy inputs. We then evaluate the FA detection verification performance of our YOLO11x model on these perturbed images. For each input, we draw 10^6 samples from the corresponding noise distribution to approximate the ground truth. The results are summarized in the table below:

Table 6: FA Detection Verification Performance under Diverse Noise Models

Model	Noise type	$\mathrm{CAR}_{\mathrm{FA}}$	$\mathrm{TPR}_{\mathrm{FA}}$	$\mathrm{TNR}_{\mathrm{FA}}$	$\mathrm{FPR}_{\mathrm{FA}}$	$\mathrm{FNR}_{\mathrm{FA}}$
yolo11x	gaussian	100.00%	92.31%	100.00%	0.00%	7.69%
yolo11x	salt and pepper	100.00%	85.71%	100.00%	0.00%	14.29%
yolo11x	uniform	100.00%	92.31%	100.00%	0.00%	7.69%

A detection is considered positive if it remains robust under the corresponding noise model and negative otherwise. The results indicate that our method sustains a high Certified Accuracy Rate (CAR) across different noise types, demonstrating that it provides reliable guarantees under diverse real-world noise conditions and threat types. Moreover, the True Positive Rate (TPR) and True Negative Rate (TNR) remain consistently high, while the False Positive Rate (FPR) and False Negative Rate (FNR) stay low, underscoring the method's effectiveness in distinguishing between robust and non-robust detections.

Overall, these results demonstrate that our method remains reliable across heterogeneous noise distributions and diverse threat types. This confirms that the proposed framework is broadly applicable and provides trustworthy robustness guarantees under a wide range of real-world noise conditions. the algorithm (e.g., by adjusting the MIQP constraints) based on the specific verification objective. We will add a discussion of these and other potential