IoUCert: Robustness Verification for Anchor-based Object Detectors

Benedikt Brückner*,[1], Alejandro J. Mercado*,†,[2], Yanghao Zhang[1], Panagiotis Kouvaros[1], and Alessio Lomuscio[1,2]

[1] Safe Intelligence
{benedikt,yanghao,panagiotis,alessio}@safeintelligence.ai
[2] Imperial College London
a.mercado24@imperial.ac.uk

Abstract. While formal robustness verification has seen significant success in image classification, scaling these guarantees to object detection remains notoriously difficult due to complex non-linear coordinate transformations and Intersection-over-Union (IoU) metrics. As a fundamental step towards verifying complete detection pipelines, we introduce IoUCert, a novel formal verification framework designed specifically to overcome these core mathematical bottlenecks. By isolating the object localisation task in single-object settings, we propose a coordinate transformation that circumvents precision-degrading relaxations of non-linear box prediction functions. This approach allows us to optimise bounds directly with respect to anchor box offsets, enabling a novel Interval Bound Propagation method that derives optimal IoU bounds. We demonstrate that IoUCert enables, for the first time, the robustness verification of foundational, anchor-based architectures including tractable variants of SSD, YOLOv2, and YOLOv3 against various input perturbations, providing a rigorous theoretical basis for future end-to-end detector verification.

Keywords: Neural Network Verification · Object Detection · Adversarial Robustness

1 Introduction

Neural networks are increasingly being deployed in safety-critical domains such as autonomous vehicles [13] and medical diagnostics [43]. Even when these models produce correct predictions for a given input, they can make incorrect predictions on perceptually equivalent variations of that input [2,27,61,69,71]. Formal verification approaches address this by assessing the robustness of models with respect to various input perturbations such as white noise [34,39,58,64], photometric [31,38], geometric [5,6] and convolutional [12,49] perturbations.

While significant progress in the area has been made, most methods target image classifiers [1, 4, 17, 20, 24, 41, 47, 65, 70]. As a result, the methods cannot analyse complex architectures widely used in computer vision applications

*Equal contribution.
†Work done during an internship at Safe Intelligence.

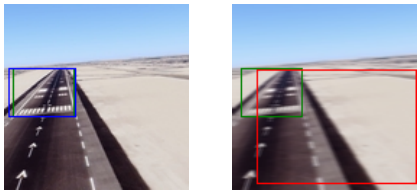

Fig. 1: A counterexample from robustness verification of a YOLOv3 model on a runway detection task. Left: original image; right: perturbed image. Green boxes: ground truth, blue box: original model prediction (IoU = 0.89), red box: prediction on the perturbed image (IoU = 0.11).

that require robustness validation. A notable example of this is the class of object detection (OD) models [44, 54, 74] which may exhibit vulnerabilities such as that illustrated in Figure 1. The technical challenge is that OD models include non-linear components such as non-maximum suppression, additional logic components such as Intersection-over-Union (IoU) calculations, and non-linear transformations to convert the raw model outputs into bounding box predictions. Existing robustness verifiers either do not support these or provide loose approximations for them [15,19,50,53]. Most work on OD verification employs regression models predicting four box coordinates which lack the typical backbone, neck and head structure of object detectors, do not consider heads operating at multiple grid scales, and use shallow backbones [15, 19, 50, 53]. They therefore lack the capabilities to analyse modern OD models.

In this paper, we overcome these limitations by introducing IoUCert, a method for the verification of OD models that is provably tighter than previous approaches, supports foundational anchor-based architectures such as SSD and YOLO, and scales beyond simplified toy models. Specifically, we make the following contributions:

- We improve existing Interval Bound Propagation (IBP) methods by deriving optimal IoU bounds for anchor-based object detection models.
- We introduce an architecture-aware verification framework for anchor-based object detectors. By using a novel coordinate transformation, we bypass the relaxation of complex non-linearities (such as class logit sigmoids), effectively avoiding the computational bottlenecks of standard verifiers. This allows us to scale efficiently and enables the first formal verification of models like SSD and YOLOv3 in single-object settings.
- We derive optimal linear relaxations for LeakyReLU activations in YOLOv3 models to minimise relaxation errors.
- We integrate IoUCert into the Venus verifier to analyse the robustness of SSD, YOLOv2 and YOLOv3 models on datasets of varying complexity, including Pascal VOC [22], COCO [42] and a runway detection task on LARD [18].

Bridging formal verification and object detection requires managing immense computational complexity: complete verification provides guarantees over infinite

continuous perturbation spaces, far more demanding than empirical testing. To remain tractable for realistic architectures, we focus on the single-object setting, verifying the core classification-regression mechanisms while leaving the combinatorial complexity of multi-object competition and Non-Maximum Suppression (NMS) for future work.

The rest of the paper is organised as follows: After discussing related work below, we discuss background material on OD models and neural network verification in Section 3 and present IoUCert in Section 4, which we evaluate in Section 5. We conclude in Section 7.

2 Related Work

Neural network verification encompasses complete methods relying on Satisfiability Modulo Theories [21, 33, 34, 51] or Mixed-Integer Linear Programming solvers [3, 8, 39, 46, 59, 60], and incomplete methods that use relaxations like Semidefinite Programming [14, 23, 40, 52] or efficient bound propagation [28, 37, 57, 63, 64, 66, 67]. State-of-the-art approaches often combine GPU-accelerated incomplete bound propagation with Branch-and-Bound heuristics to achieve completeness and scalability [10, 16, 32, 62, 72, 73].

While verification has seen significant progress, most verifiers primarily target image classifiers [11]. Recent efforts in object detection (OD) verification struggle to scale beyond highly simplified setups. For instance, [19] bounds the Intersection-over-Union (IoU) metric via Interval Bound Propagation (IBP) on predicted corner coordinates, while [53] encodes IoU as a network layer but struggles with loose bounds on operations like *max*, *min*, and division. Both methods demonstrate formal verification on simple regression toy models outputting four corner coordinates for a single object, and these models achieve low accuracies. The probabilistic verifier by [45] scales to larger architectures, but suffers from unsound robustness certificates, even for small models. Other works based on ImageStars [15] or branch-free IBP [50] are similarly evaluated on toy models and lack support for the multi-scale heads, non-linear coordinate transformations, and anchor-based structures of real-world detectors like SSD [44] and YOLO [54].

In contrast, IoUCert is a sound verifier that provides an object detection-aware framework which explicitly exploits the specific structure of anchor-based detectors to efficiently scale verification. Unlike previous approaches that naively apply image verifiers to OD models which often incur unnecessary relaxations that lead to timeouts, we introduce a coordinate transformation to directly operate on inferred offset bounds. This enables optimal bounding for realistic anchor-based detectors that achieve high accuracies. Combined with tight symbolic bounding and optimal IoU relaxations, our method scales to architectures of practical relevance. For the first time, we demonstrate robustness verification on SSD [44], YOLOv2 [55], and YOLOv3 [56] models. While newer YOLO variants exist, they employ components that remain highly challenging for current verifiers, such as attention layers [7].

3 Background

We formally define the single-object detection task, describe typical anchor-based architectures, and outline the neural network verification problem. Extending this framework to multiple objects is left for future work due to the added complexity.

3.1 Object detection

Let $I \in \mathbb{R}^{C\times H\times W}$ be an image that is annotated with a ground truth box $g = (z_0, z_1, z_2, z_3, g_c)$, where (z_0, z_1) and (z_2, z_3) are its top-left and bottom-right corners, and g_c is its class. For a single-object task the expected output is exactly one bounding box matching this ground truth, but regressing a single set of coordinates performs poorly. Instead, a modern object detector $\mathbf{O}$ predicts a large set of candidate detections $\mathbf{O}(I) = \{b^1, \ldots, b^k\}$ and relies on post-processing to isolate the final prediction, which makes the naive application of robustness verifiers prohibitively difficult.

Each predicted box $b = (z_0, z_1, z_2, z_3, \mathbf{c}, s)$ contains corner coordinates, class probabilities $\mathbf{c} \in \mathbb{R}^{n_c}$, and a confidence score $s \in [0, 1]$. Let the area of box b be $\mathbf{a}(b) = (b_3 - b_1)(b_2 - b_0)$ and the intersection of two boxes b, b' be $\mathbf{i}(b, b') = (\max(b_0, b'_0), \max(b_1, b'_1), \min(b_2, b'_2), \min(b_3, b'_3))$. The Intersection-over-Union is $\mathrm{IoU}(b, b') = \mathbf{a}(\mathbf{i}(b, b'))/(\mathbf{a}(b) + \mathbf{a}(b') - \mathbf{a}(\mathbf{i}(b, b')))$.

An OD model is *correct* with respect to ground truth g and thresholds $\tau_{\text{iou}}, \tau_{\text{class}}$ if: (1) it outputs exactly one box b after post-processing, (2) the predicted class $\arg\max_i \mathbf{c}_i$ matches g_c and its class score exceeds τ_{class}, and (3) $\mathrm{IoU}(b, g) \geq \tau_{\text{iou}}$.

Modern anchor-based OD models (*e.g.*, SSD, YOLO) predict offsets for a fixed set of prior *anchor boxes* $\{p^1, \ldots, p^n\}$ via a three-step pipeline:

Step 1: Offset Prediction. The model predicts $\mathcal{A} = \{a^1, \ldots, a^n\}$, where $a^i = (o^i, l^i)$ contains coordinate offsets o^i and logits l^i for class and confidence scores.

Step 2: Box Construction. Offsets o^i and anchors p^i are combined via a model-specific function $\phi(o^i, p^i) = (c_x, c_y, w, h)$ into centre-format coordinates (see Appendix B). This is mapped to the corner-format (z_0, z_1, z_2, z_3) using $\psi(\phi(o^i, p^i)) = (c_x - \frac{w}{2}, c_y - \frac{h}{2}, c_x + \frac{w}{2}, c_y + \frac{h}{2})$. The inverse, used later in our verification procedure, is $\psi^{-1}(z_0, z_1, z_2, z_3) = (\frac{z_0+z_2}{2}, \frac{z_1+z_3}{2}, z_2 - z_0, z_3 - z_1)$. Each box is assigned a class and confidence score from the corresponding logits l^i.

Step 3: Post-processing. Non-maximum suppression (NMS) filters overlapping or low-confidence boxes. For single object detection, we assume this selects the highest-confidence bounding box above a threshold (this is without loss of generality, see Appendix A). We keep this threshold even though we deal with single-object detection, since it allows a detector to abstain when all candidates score below it. Verifying correctness therefore requires certifying both the selection of the top box and the satisfaction of the threshold.

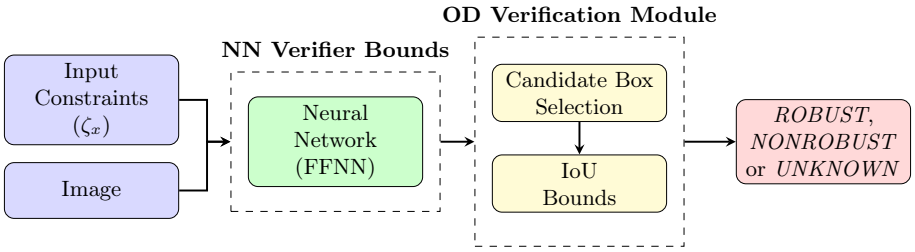

Fig. 2: Pipeline of our object detection (OD) verification framework, combining verifier bounds, candidate selection, and optimal IoU bound derivation to reach *ROBUST*, *NONROBUST* or *UNKNOWN* conclusions.

3.2 Neural Network Verification

Verification aims to certify a network's robustness. A network N is *robust* to input constraints ζ_x (*e.g.*, ℓ_∞ bounds, brightness, or blurring [12, 38]) if $N(x)$ satisfies output constraints ζ_y for all $x \in \zeta_x$. Here, ζ_y encodes OD correctness as defined above.

Interval Bound Propagation (IBP) [28] is a fast, incomplete method that propagates concrete intervals, but it often yields loose bounds as it cannot capture variable dependencies [63]. Symbolic Interval Propagation (SIP) mitigates this by propagating symbolic bounding equations. The CROWN/DeepPoly/back-substitution method [32,58,67] propagates symbolically from the layer of interest back to the input to fully capture dependencies and obtain tighter bounds. When bounds remain too loose, branch-and-bound partitions the problem space to reduce relaxation errors. This is done via *input splitting* for low input dimensions [8, 63] or *neuron splitting* for high-dimensional problems [8, 25].

Verifying OD models with standard SIP is challenging because verifiers typically target ReLU networks (YOLOv3 uses LeakyReLU) and require linear relaxations for all non-linear components (*e.g.*, box construction). In Section 4, we overcome these limitations by proposing a coordinate transformation to bypass certain non-linearities, derive optimal IoU bounds, and define optimal LeakyReLU relaxations.

4 Method

Verifying object detectors is challenging: modern OD models combine large architectures with non-linear functions mapping offsets to box coordinates, and existing verifiers fail to scale to SSD or YOLO due to loose bounds and missing support for anchor boxes, multiple prediction heads, and confidence scores. To the best of our knowledge, IoUCert is the first to support such coordinate transformations, enabling IoU bounding for detectors with non-linear box predictions, for which we further derive the provably tightest IoU bounds. Figure 2 presents an overview of our verification pipeline.

4.1 Coordinate Transformation

As outlined in Section 3.1, object detectors predict offsets $\mathbf{o}$ that are converted to corner coordinates (z_0, z_1, z_2, z_3) via the mapping $\psi \circ \phi$. We bound the IoU function *directly* with respect to the predicted offsets, rather than propagating the offset bounds through $\psi \circ \phi$ before bounding the IoU function [19], thereby circumventing the overapproximation that bound propagation over $\psi \circ \phi$ would induce.

To bound the IoU between the predicted offsets $\mathbf{o}$ of a bounding box and a fixed ground truth box g, we need to bound the expression $\mathrm{IoU}(\psi \circ \phi(\mathbf{o}), g)$. Here, we focus on deriving the optimal upper bound using the given offset bounds $[\underline{\mathbf{o}}, \overline{\mathbf{o}}]$ for $\mathbf{o} = (o_0, o_1, o_2, o_3)$; the derivation for the lower bound is analogous. We want to solve the constrained maximisation problem:

$$\begin{aligned} &\max_{\mathbf{o}=(o_0,o_1,o_2,o_3)} \mathrm{IoU}(\psi \circ \phi(\mathbf{o}), g) \\ &\text{s.t.} \quad \underline{\mathbf{o}} \leq \mathbf{o} \leq \overline{\mathbf{o}}, \end{aligned} \tag{1}$$

which is difficult to do using linear-relaxation-based bound propagation methods due to the non-linearity of $\psi \circ \phi$. We define $\gamma(\mathbf{o}) = \psi \circ \phi(\mathbf{o})$ which maps from offset space to corner space. We show the injectivity of the mappings ψ, ϕ in Appendix B. Since the composition of injective functions is injective [29, Theorem 12.2], it follows that the composition $\psi \circ \phi$ is also injective. This allows us to define $(\psi \circ \phi)^{-1}(\mathbf{z}) = \phi^{-1} \circ \psi^{-1}(\mathbf{z})$ as the mapping from corner space to offset space. We substitute $\mathbf{o} = \phi^{-1} \circ \psi^{-1}(\mathbf{z})$ in Problem 1. The objective function becomes $\mathrm{IoU}(\psi \circ \phi(\phi^{-1} \circ \psi^{-1}(\mathbf{z})), g) = \mathrm{IoU}(\mathbf{z}, g)$ while the constraints become $\underline{\mathbf{o}} \leq \phi^{-1} \circ \psi^{-1}(\mathbf{z}) \leq \overline{\mathbf{o}}$. In summary, we obtain the following equivalent optimisation problem which directly optimises over the corner coordinates $\mathbf{z} = (z_0, z_1, z_2, z_3)$ [9, pp. 130–131]:

$$\begin{aligned} &\max_{\mathbf{z}=(z_0,z_1,z_2,z_3)} \mathrm{IoU}(\mathbf{z}, g) \\ &\text{s.t.} \quad \underline{\mathbf{o}} \leq \phi^{-1} \circ \psi^{-1}(\mathbf{z}) \leq \overline{\mathbf{o}}, \end{aligned}$$

The inverse mappings ensure that the feasible region, originally defined in the offset space, is correctly expressed over $\mathbf{z}$. Rewriting the constraints explicitly in terms of the corner coordinates, we obtain:

$$\begin{aligned} &\max_{z_0,z_1,z_2,z_3} \mathrm{IoU}((z_0, z_1, z_2, z_3), g) \\ \text{s.t.} \quad &2c_x(\underline{\mathbf{o}}_0) \leq z_0 + z_2 \leq 2c_x(\overline{\mathbf{o}}_0), \\ &2c_y(\underline{\mathbf{o}}_1) \leq z_1 + z_3 \leq 2c_y(\overline{\mathbf{o}}_1), \\ &w(\underline{\mathbf{o}}_2) \leq z_2 - z_0 \leq w(\overline{\mathbf{o}}_2), \\ &h(\underline{\mathbf{o}}_3) \leq z_3 - z_1 \leq h(\overline{\mathbf{o}}_3), \end{aligned} \tag{2}$$

where c_x and c_y denote the centre coordinate transformations, and w and h represent the width and height, each expressed in terms of their respective offsets.

More generally, the transformation applies whenever the box-construction map $\psi \circ \phi$ is injective with a tractable inverse, *i.e.* when the decoding function is strictly monotonic in each offset (Appendix B). The applicability of the method to other detector families is discussed in Appendix H.

4.2 Optimal IoU IBP Bounds

Extreme points for the IoU in Problem 2 exist either (i) where the gradient of the function is zero; (ii) along the borders of the constrained region; or (iii) where the function is not differentiable. Cohen et al. [19] show that the partial derivatives of $\text{IoU}((z_0, z_1, z_2, z_3), g)$ are never zero within the feasible region, so we only consider cases (ii) and (iii). Since each constraint only includes either the variables (z_0, z_2) or (z_1, z_3), we consider the 2D plane (z_0, z_2) corresponding to width-related variables, and the plane (z_1, z_3) corresponding to height-related variables separately. We identify the coordinates within each 2D plane, and then combine the coordinates from both planes to form the complete candidate extreme points. We use L_i and U_i to denote the lower and upper bounds for the i-th constraint in Problem 2.

1. **Corner points.** The corners of each region where the constraints intersect form our first set of critical points over the boundaries. These are given by

$$P_x^c = \left\{ \left(\frac{U_0 - U_2}{2}, \frac{U_0 + U_2}{2} \right), \left(\frac{U_0 - L_2}{2}, \frac{U_0 + L_2}{2} \right), \left(\frac{L_0 - U_2}{2}, \frac{L_0 + U_2}{2} \right), \left(\frac{L_0 - L_2}{2}, \frac{L_0 + L_2}{2} \right) \right\}$$

   over the (z_0, z_2) plane, and

$$P_y^c = \left\{ \left(\frac{U_1 - U_3}{2}, \frac{U_1 + U_3}{2} \right), \left(\frac{U_1 - L_3}{2}, \frac{U_1 + L_3}{2} \right), \left(\frac{L_1 - U_3}{2}, \frac{L_1 + U_3}{2} \right), \left(\frac{L_1 - L_3}{2}, \frac{L_1 + L_3}{2} \right) \right\}$$

   over the (z_1, z_3) plane.
2. **Stationary points on boundaries.** The second set of critical points are those where the gradient along the boundaries is zero. Each boundary of the feasible region is defined by one of the following equations (with $k = 0$ or $k = 1$, depending on the plane):
   (a) $z_k + z_{k+2} = U_k$ or $z_k + z_{k+2} = L_k$
   (b) $z_{k+2} - z_k = U_{k+2}$ or $z_{k+2} - z_k = L_{k+2}$.
   As we show in Appendix E.2, the gradient along these boundaries, where it exists, is either zero at every point or non-zero at every point. When the gradient is always zero, the value on the boundary matches that at the corners of the region. If it is always non-zero, the extremum is attained at one of the ends of the corner segment. Consequently, these stationary points on the boundaries are already included in P_x^c, P_y^c.

3. **Non-differentiable points.** Since the IoU function is non-differentiable whenever $z_i = g_i$, the third set of critical points includes
   - the points at the intersection of the ground truth box coordinates with each border (when they intersect): $P_x^{\text{int}} = \{(g_0, U_0 - g_0), (g_0, L_0 - g_0), (g_0, U_2 + g_0), (g_0, L_2 + g_0), (U_0 - g_2, g_2), (L_0 - g_2, g_2), (g_2 - U_2, g_2), (g_2 - L_2, g_2)\}$ along the (z_0, z_2) plane, and $P_y^{\text{int}} = \{(g_1, U_1 - g_1), (g_1, L_1 - g_1), (g_1, U_3 + g_1), (g_1, L_3 + g_1), (U_1 - g_3, g_3), (L_1 - g_3, g_3), (g_3 - U_3, g_3), (g_3 - L_3, g_3)\}$ along the (z_1, z_3) plane.
   - the corners of the ground truth boxes when they fall within the constraint region: $P_x^{gt} = (g_0, g_2)$ and $P_y^{gt} = (g_1, g_3)$.

For $i \in \{0, 2\}$, let $P_{z_i} = \{\alpha \mid (\alpha, \beta) \in P_x^c \cup P_x^{\text{int}} \cup P_x^{gt} \text{ for some } \beta\}$, and similarly, for $i \in \{1, 3\}$, let $P_{z_i} = \{\alpha \mid (\alpha, \beta) \in P_y^c \cup P_y^{\text{int}} \cup P_y^{gt} \text{ for some } \beta\}$. These sets of points can be used to optimise the IoU function.

Theorem 1. *The IoU function is maximum for a point obtained from within the set* $C_s = \bigtimes_i P_{z_i}$.

Proof. See Appendix E.

Following Theorem 1, the maximum and minimum IoU values can be found by iterating through the critical points, checking whether they satisfy the constraints, evaluating their *validity* (whether they define valid boxes satisfying $z_0 < z_2$ and $z_1 < z_3$), and updating the maximum and minimum IoU values. The same construction applies to each 2D plane: the candidate set comprises 13 points, namely the 4 corners of the feasible region, the 8 intersections (2×4) of the two ground truth coordinate lines with the four lines bounding the region, and the single ground truth corner; these are continuous box coordinates rather than discrete pixel locations (see Figure 3 for an illustration of the (z_0, z_2) plane, including the candidates that are pruned as infeasible). Hence $|P_x^c \cup P_x^{\text{int}} \cup P_x^{gt}| = |P_y^c \cup P_y^{\text{int}} \cup P_y^{gt}| = 13$, giving $|C_s| = 13^2 = 169$ candidate points overall. Since the maximum and minimum IoU are attained at such points and only a finite number (169) exist, the algorithm is correct and will terminate in constant time. For details see Appendix F.

4.3 Robustness Verification Algorithm

Algorithm 1 introduces the IoUCert method for establishing the robustness of an OD model for any input satisfying the specified perturbation constraints. IoUCert employs existing bound propagation frameworks (such as IBP or SIP) to obtain bounds on the output of the neural network component for a given perturbation (Line 4). Given these bounds, it identifies all bounding boxes that could potentially have the highest confidence score (Line 5). Even though an OD model only outputs the bounding box with the highest confidence score when there is only a single object, the approximate nature of the bounds may cause the identification of multiple candidate boxes. For example, if box 1 has

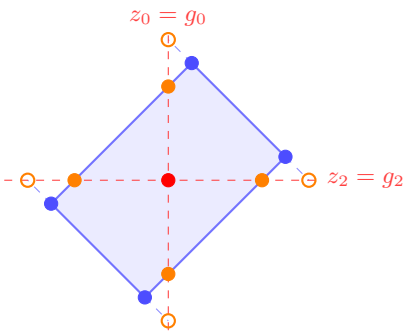

Fig. 3: Geometric intuition for the IoU candidate set in the (z_0, z_2) plane (the (z_1, z_3) plane is analogous). The four constraints define a feasible region (blue) with four corners (●, P_x^c), and the dashed ground truth coordinate lines $z_0 = g_0$, $z_2 = g_2$ mark the non-differentiability of the IoU, crossing at the ground truth corner (●, P_x^{gt}). Each ground truth line meets each of the four lines bounding the region, giving $2 \times 4 = 8$ intersections (P_x^{int}): those on the feasible boundary (●) are retained, while those outside it (○, where the ground truth lines meet the dashed boundary-line extensions) are pruned by the feasibility check. This gives $4 + 8 + 1 = 13$ enumerated candidates per plane (as used in Theorem 1). Here 4 of the 8 intersections are infeasible, leaving the 9 filled points.

score bounds ($[0.5, 0.9]$) and box 2 has ($[0.7, 0.8]$), either of them could be the top-scoring box.

To identify candidate boxes, IoUCert selects all boxes whose upper bound on the confidence score exceeds the highest lower bound among all boxes. This ensures that only boxes that could potentially be the top-scoring one are considered. For each candidate, we compute bounds on its class scores and its IoU with the ground truth box. The detailed procedure is described in Appendix F.

If all candidate boxes meet the IoU threshold, the minimum confidence score meets the class threshold, and all candidates agree on the predicted class, the verification query is *ROBUST*. If none of the candidates meets the IoU threshold, or the maximum confidence score is below the threshold, or all candidate boxes predict a class different from the ground truth, the query is *NONROBUST*. If the bounds are too loose to determine the outcome, or if the candidate boxes do not agree on a single class, IoUCert outputs *UNKNOWN*.

Theorem 2. IoUCert *is correct. It is complete when integrated with a branching framework.*

Proof. See Appendix F.

To determine the robustness of unknown cases, IoUCert can be combined with any branching framework in neural network verification, such as Venus [39].

4.4 Optimal Relaxations for LeakyReLU Activations

While most neural network verification approaches focus on ReLU activation functions, the YOLOv3 architecture employs LeakyReLU activations. We define

Algorithm 1 Robustness Verification for Object Detectors

Input: Neural network N, input constraints ζ_x, robustness constraints ζ_y
Output: *ROBUST* or *NONROBUST* or *UNKNOWN*
function VERIFY(N, ζ_x, ζ_y)
    bounds ← BOUND_PROPAGATION(ζ_x, N)
    IoUBounds, scoreBounds, predClass ←
        GETHIGHESTBOX(bounds)
    if IoUBounds.min $\geq \tau_{\text{iou}}$ **and**
        scoreBounds.lower $\geq \tau_{\text{class}}$ **and**
        predClass = *class* **then**
        return *ROBUST*
    else if IoUBounds.max $< \tau_{\text{iou}}$ **or**
        scoreBounds.upper $< \tau_{\text{class}}$ **or**
        DISAGREE(predClass) **then**
        return *NONROBUST*
    else
        return *UNKNOWN*
    end if
end function

LeakyReLU$(x) = \max\{\alpha x, x\}$ with $\alpha \in [0, 1]$ and concrete input bounds $x \in [l, u]$. If $u < 0$ or $l > 0$ the activation function is said to be stable and can be represented exactly in a linear bound propagation framework. For $l < 0 < u$, its behaviour is piece-wise linear, and linear lower and upper bounding functions $f_{\text{lower}}(x), f_{\text{upper}}(x)$ for it are given by

$$f_{\text{lower}}(x) = \tilde{\alpha} x, \quad \tilde{\alpha} \in [\alpha, 1] \tag{3}$$

$$f_{\text{upper}}(x) = \frac{u - \alpha l}{u - l} x + \frac{(\alpha - 1) l u}{u - l}. \tag{4}$$

Existing works simply set $\tilde{\alpha} = \alpha$ [48]. We observe that by selecting $\tilde{\alpha}$ depending on l, u we can reduce the local relaxation error which has been shown to improve verification performance [30, 58, 67]. We minimise the relaxation error using the following result:

Theorem 3. *For LeakyReLU$(x) = \max\{\alpha x, x\}$ with $x \in [l, u]$, the local relaxation error is minimised by setting*

$$f_{lower}(x) = \begin{cases} \alpha x & \text{if } u < |l|, \\ x & \text{else} \end{cases} \tag{5}$$

Proof. See Appendix G.

5 Evaluation

We implemented IoUCert on top of Venus, a state-of-the-art verifier [39], encoding Algorithm 1 as a custom layer appended to the target model. It takes concrete

bounds on the output logits (from back-substitution) and computes the IoU and confidence-score bounds, integrating with Venus's branch-and-bound (BaB) procedure.

5.1 Benchmarks

While some OD verification benchmarks are available within the Verification of Neural Networks Competition (VNN-COMP) [10], their verification queries only target specific anchor boxes and class predictions rather than assessing the robustness of the entire OD pipeline. Prior work on OD verification that did consider object localisation focused on toy models rather than anchor-based architectures [19, 53]. To address these gaps, we train various object detection models and modify an existing benchmark to evaluate our framework. Following standard practices in the formal verification community [10, 35], we evaluate our method on random subsets of 50 correctly classified images per dataset, balancing mathematical guarantees with computational feasibility.

- *SSD.* We trained an SSD model [44] on the safety-critical LARD runway detection task [18] (*Google Earth* images, each depicting a single runway). Images were resized to 128×128, a relatively high resolution for complete verification; for tractability we replaced piece-wise linear *MaxPool* with linear *AvgPool* layers, and trained with stochastic gradient descent (SGD) and the *MultiBox* loss [44]. NMS used a threshold of 0.5 and a confidence threshold of 0.15, with the highest-scoring box selected at inference. Overall, the model contains $11,278,046$ learnable parameters.
- *YOLOv2.* We used the YOLOv2-tiny (TinyYOLO) benchmark from VNN-COMP 2023[3] which consists of a simplified YOLOv2-tiny model trained on a subset of images from the Pascal VOC dataset [22]. The authors replaced the large backbone with a smaller variant, but the anchor-based prediction head remains representative of foundational detection architectures.
- *YOLOv3.* We trained YOLOv3-tiny models on LARD at 64×64 and 128×128 and on COCO at 128×128, again using AvgPool layers for tractability. As our framework focuses on the single-object scenario, we preprocessed COCO into single-object crops. The resulting models have between 8.7 and 8.9 million parameters; see Appendix C.1 for preprocessing and training details.

Impact of Model Adaptations. The adaptations required for tractable verification only mildly affect standard performance: for YOLOv3-tiny on LARD (64×64), replacing MaxPool with AvgPool moves $\text{mAP}_{0.5}$ from 86.88% to 86.59% ($\text{mAP}_{0.5:0.95}$: $41.67\% \rightarrow 40.90\%$) while reducing verification time by over an order of magnitude. The backbones, resolution, and single-object COCO crops all preserve the anchor-based structure our method targets. Full before/after accuracies and the pooling ablation are provided in Appendices C.1 and C.3.

[3] https://github.com/xiangruzh/Yolo-Benchmark

5.2 Experimental Results

We evaluate the effectiveness of IoUCert using a number of different models and perturbations. To compare the tightness of our bounds as well as the complete verification performance against the state-of-the-art, we reimplement the bounding method proposed by Cohen et al. [19] in our verification framework. All experiments were run on a machine equipped with an AMD Ryzen 9 9950X3D 16-core CPU, 192 GB of RAM, and an NVIDIA RTX 5090 GPU with 32 GB of VRAM, running Ubuntu with kernel 6.8.

Table 1: IoUCert verification results on SSD (trained on LARD) and YOLOv2 (trained on Pascal VOC) for different perturbation sizes ϵ and perturbations showing robust (R), non-robust (NR), and timeout (T) counts and mean time (all cases) in seconds.

Model	ϵ	Brightness				Contrast			
		R	**NR**	**T**	**Time**	**R**	**NR**	**T**	**Time**
SSD	0.01	48	2	0	29.06	49	1	0	24.21
	0.05	45	5	0	404.54	47	3	0	171.49
	0.10	40	10	0	731.41	42	8	0	359.41
	0.30	9	41	0	458.21	30	20	0	885.51
	0.50	0	47	3	221.71	14	36	0	743.04
	0.80	0	50	0	10.62	0	50	0	3.78
	1.00	0	50	0	5.79	0	50	0	3.90
YOLOv2	0.01	50	0	0	3.69	50	0	0	3.23
	0.05	50	0	0	12.97	50	0	0	4.86
	0.10	47	3	0	23.81	50	0	0	9.23
	0.30	28	22	0	39.40	48	2	0	33.36
	0.50	4	46	0	9.99	36	14	0	35.86
	0.80	0	50	0	1.21	0	50	0	2.52
	1.00	0	50	0	0.81	0	50	0	2.36

SSD and YOLOv2 Results. We evaluated the ReLU-based SSD and YOLOv2 models under both brightness and contrast perturbations. Table 1 reports the number of *ROBUST*, *TIMEOUT*, and *NONROBUST* cases as well as the average verification time. IoUCert is fast for the small YOLOv2 model. It verifies all properties for small perturbation budgets and most properties for medium-sized budgets. For larger budgets, IoUCert effectively identifies counterexamples showcasing the vulnerabilities of the model. Although the SSD model is significantly larger, we are able to identify robust cases for ϵ values of up to 0.3 for brightness and 0.5 for contrast perturbations. As expected, verification times are higher than for the small YOLOv2 model, because bound-propagation passes through the larger model are more expensive and more branching is needed for tight bounds. Comparing our bounds (Section 4.2) with a reimplementation of the looser method of Cohen et al. [19], performance is similar: tighter bounds avoid branching but cost more to compute (detailed analysis in Appendix D).

Table 2: Summary of IoU bound ranges, number of sampled bounds, average tightness improvement, and avoided subproblem exploration percentages.

Range	#Bounds	Improv. (%)	Avoided (%)
0.01 - 0.10	14642	50.67	0.59
0.10 - 0.20	8479	65.09	0.12
0.20 - 0.30	8084	58.54	0.11
0.30 - 0.40	6383	56.09	0.05
0.40 - 0.50	4440	55.32	0.14
0.50 - 0.60	3569	54.75	99.66
0.60 - 0.70	2707	53.74	98.93
0.70 - 0.80	2336	53.14	97.60
0.80 - 0.90	1802	52.20	96.50
0.90 - 0.99	1374	52.30	95.92

Bound Tightness. To assess the bound tightness, we recorded bounds for all boxes (not just the top-scoring one) during verification runs on the SSD model under a brightness perturbation with $\epsilon = 0.02$ and measure the difference between the upper and lower bounds. Table 2 summarises the results. The first column shows the range of the recorded IoU bounds; the second, the number of sampled bounds; the third, the percentage tightness improvement over [19]; and the fourth, the percentage of branches whose exploration was avoided due to tighter bounds. Our method consistently improved bound tightness by over 50% across all depths. At shallower depths, where bounds are generally looser, this translated to over 95% of branches being pruned from the verification process.

YOLOv3 Results. Table 3 reports results on the LeakyReLU-based YOLOv3 architecture under brightness, contrast, and motion blur (kernel size 5) perturbations. Thanks to its coordinate transformation, IoUCert retains enough tightness to verify YOLOv3 across a wide range of perturbations. All models are highly robust to 0° motion blur, though our procedure still finds edge cases yielding incorrect predictions at high budgets. The same trend holds for other blur angles (Appendix C.2, Table S3).

The model trained on the more complex COCO dataset is generally more vulnerable to brightness and motion blur than LARD at the same 128×128 resolution (*e.g.* 29 vs. 40 robust cases at $\epsilon = 0.5$ under brightness), though slightly more resilient to mid-range contrast. On LARD, the 128×128 model attains higher clean accuracy than the 64×64 one but is more vulnerable to perturbations: higher input dimensionality leads to looser bounds, so higher clean accuracy does not necessarily lead to higher certified robustness. LeakyReLU relaxation tightness is discussed in Appendix G.

Pooling Choice. Our YOLOv3 models replace the MaxPool downsampling layers of the original architecture with AvgPool layers, which are linear and can be represented exactly in the bound-propagation framework while retaining comparable clean accuracy ($mAP_{0.5}$ of 86.59% vs. 86.88% on LARD at 64×64). This choice is decisive for verifiability: on a 50-image subset, the AvgPool model

Table 3: IoUCert results on YOLOv3 for different perturbation sizes ϵ and perturbations with robust (R), non-robust (NR), and timeout (T) counts shown alongside mean verification time in seconds.

Model	ϵ	Brightness				Contrast				Motion Blur (0°)			
		R	**NR**	**T**	**Time**	**R**	**NR**	**T**	**Time**	**R**	**NR**	**T**	**Time**
LARD 64×64	0.01	50	0	0	3.35	50	0	0	3.28	50	0	0	3.15
	0.05	50	0	0	9.72	50	0	0	8.06	50	0	0	3.30
	0.10	50	0	0	20.45	50	0	0	15.82	50	0	0	3.83
	0.30	50	0	0	56.26	50	0	0	39.25	50	0	0	16.89
	0.50	47	3	0	89.58	49	1	0	58.63	50	0	0	31.91
	0.80	36	14	0	105.72	49	1	0	114.87	49	1	0	68.61
	1.00	28	22	0	103.87	0	50	0	43.65	45	5	0	91.45
LARD 128×128	0.01	50	0	0	10.32	50	0	0	9.53	50	0	0	6.96
	0.05	50	0	0	107.77	50	0	0	100.64	50	0	0	8.44
	0.10	50	0	0	190.99	50	0	0	168.27	50	0	0	10.76
	0.30	42	8	0	381.03	43	7	0	349.66	50	0	0	74.52
	0.50	40	10	0	592.56	40	10	0	428.93	50	0	0	152.65
	0.80	23	27	0	534.56	35	15	0	571.47	50	0	0	320.19
	1.00	12	38	0	304.45	0	50	0	136.48	49	1	0	433.41
COCO 128×128	0.01	50	0	0	8.93	50	0	0	7.90	50	0	0	7.38
	0.05	47	3	0	56.09	50	0	0	22.08	50	0	0	9.19
	0.10	46	4	0	120.60	50	0	0	61.46	50	0	0	14.55
	0.30	36	14	0	272.45	45	5	0	194.66	49	1	0	85.77
	0.50	29	21	0	376.78	43	7	0	314.13	48	2	0	167.99
	0.80	17	33	0	355.23	31	19	0	450.79	40	10	0	289.62
	1.00	6	44	0	169.74	0	50	0	79.65	38	12	0	401.69

is verified more than an order of magnitude faster than its MaxPool counterpart and incurs far fewer timeouts (*e.g.* under brightness at $\epsilon = 0.3$, 50/50 robust cases in 56 s for AvgPool versus 15/50 with 33 timeouts and over 1600 s for MaxPool). The full ablation across all perturbations is reported in Appendix C.3, Table S4.

Discussion. Overall, IoUCert effectively verifies a range of anchor-based detectors via our coordinate transformation and, for YOLOv3, the tight LeakyReLU relaxations, scaling even to complex multi-class datasets like COCO without sacrificing bound tightness. Verification is effective regardless of the bounding method: tighter bounds cost more per call but prune more branches, while looser bounds branch more but process each branch faster (Appendix D).

6 Scope and Limitations

The coordinate transformation (Section 4.1) and optimal IoU bounds (Section 4.2) at the core of IoUCert apply to any detector whose box-decoding map

$\psi \circ \phi$ is injective with a tractable inverse, *i.e.* strictly monotonic in each offset (Appendix B). This holds for the dense anchor-based heads of the SSD and YOLO families we evaluate, and is independent of training: label-assignment strategies such as ATSS [68], PAA [36] or OTA [26] alter the training target, not the inference-time decoding map. The same principle covers the region proposal network of two-stage detectors such as Faster R-CNN and anchor-free heads that regress invertible offsets, whereas transformer-based detectors such as DETR add attention and set-prediction mechanisms that remain challenging for current verifiers (Appendix H).

IoUCert performs verification offline and currently targets anchor-based detectors for the single-object case, where correctness depends only on the IoU between the prediction and the ground truth. Extending it to the full multi-object pipeline additionally requires bounding the pairwise overlaps used by non-maximum suppression (NMS), *i.e.* the bounds $\underline{J}_{ik} \leq \mathrm{IoU}(B_i, B_k) \leq \overline{J}_{ik}$ for candidate pairs i, k. This is substantially harder, requiring up to $O(n^2)$ certificates over *pairs* of variable boxes and becoming ambiguous whenever the overlap bounds straddle the NMS threshold. We therefore view NMS-aware verification over the candidate boxes already bounded tightly by IoUCert as the most promising next step, and discuss it further in Appendix H.

7 Conclusion

Verifying object detectors before deployment matters in safety-critical settings such as autonomous driving, yet their complex architectures and non-linear localisation place realistic anchor-based detectors such as YOLOv3 beyond existing robustness verification methods.

We introduced IoUCert, which combines optimal Interval Bound Propagation (IBP) bounds for the Intersection-over-Union (IoU) metric with a coordinate transformation for the box prediction function and tight LeakyReLU relaxations, enabling the analysis of complex anchor-based detectors such as YOLOv3 across diverse datasets at a scale not previously demonstrated.

IoUCert performs verification offline, prior to deployment, rather than at runtime. Its current scope, its applicability to other detector families, and the path towards full multi-object, NMS-aware verification are discussed in Section 6.

Acknowledgements

Benedikt Brückner acknowledges support from the UKRI Centre for Doctoral Training in Safe and Trusted Artificial Intelligence [EP/S023356/1]. Alejandro Mercado acknowledges support from an Imperial College London President's PhD Scholarship. Alessio Lomuscio acknowledges partial support from the Royal Academy of Engineering via a Chair of Emerging Technologies.

References

1. Althoff, M.: An introduction to cora 2015. In: Proceedings of the 1st and 2nd Workshop on Applied Verification for Continuous and Hybrid Systems. pp. 120–151. EasyChair (2015)
2. Amirkhani, A., Karimi, M.P., Banitalebi-Dehkordi, A.: A survey on adversarial attacks and defenses for object detection and their applications in autonomous vehicles. The Visual Computer **39**(11), 5293–5307 (2023)
3. Anderson, R., Huchette, J., Ma, W., Tjandraatmadja, C., Vielma, J.: Strong mixed-integer programming formulations for trained neural networks. Mathematical Programming **183**, 3–39 (2020)
4. Bak, S.: nnenum: Verification of relu neural networks with optimized abstraction refinement. In: Proceedings of the 13th International Symposium on NASA Formal Methods (NFM21). Lecture Notes in Computer Science, vol. 12673, pp. 19–36. Springer (2021)
5. Balunovic, M., Baader, M., Singh, G., Gehr, T., Vechev, M.: Certifying geometric robustness of neural networks. In: Proceedings of the 33rd Annual Conference on Neural Information Processing Systems (NeurIPS19), pp. 15313–15323. Curran Associates, Inc. (2019)
6. Batten, B., Zheng, Y., De Palma, A., Kouvaros, P., Lomuscio, A.: Verification of geometric robustness of neural networks via piecewise linear approximation and lipschitz optimisation. In: Proceedings of the 27th European Conference on Artificial Intelligence (ECAI24). pp. 2362–2369. IOS Press (2024)
7. Bochkovskiy, A., Wang, C.., Liao, H.Y.M.: Yolov4: Optimal speed and accuracy of object detection. arXiv preprint arXiv:2004.10934 (2020)
8. Botoeva, E., Kouvaros, P., Kronqvist, J., Lomuscio, A., Misener, R.: Efficient verification of neural networks via dependency analysis. In: Proceedings of the 34th AAAI Conference on Artificial Intelligence (AAAI20). pp. 3291–3299. AAAI Press (2020)
9. Boyd, S., Vandenberghe, L.: Convex Optimization. Cambridge University Press (2004)
10. Brix, C., Bak, S., Johnson, T.T., Wu, H.: The fifth international verification of neural networks competition (vnn-comp 2024): Summary and results. arXiv preprint arXiv:2412.19985 (2024)
11. Brix, C., Müller, M.N., Bak, S., Johnson, T.T., Liu, C.: First three years of the international verification of neural networks competition (vnn-comp). arXiv preprint arXiv:2301.05815 (2023)
12. Brückner, B., Lomuscio, A.: Verification of neural networks against convolutional perturbations via parameterised kernels. In: Proceedings of the 39th AAAI Conference on Artificial Intelligence (AAAI25). pp. 27215–27223. AAAI Press (2025)
13. Cao, Y., Xiao, C., Cyr, B., Zhou, Y., Park, W., Rampazzi, S., Chen, Q.A., Fu, K., Mao, Z.M.: Adversarial sensor attack on lidar-based perception in autonomous driving. In: Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security (CCS 2019). pp. 2267–2281. ACM (2019)
14. Chiu, H.M., Zhang, R.Y.: Tight certification of adversarially trained neural networks via nonconvex low-rank semidefinite relaxations. In: Proceedings of the 40th International Conference on Machine Learning (ICML23). vol. 202, pp. 5631–5660. PMLR (2023)
15. Chowdhury, S., Khandelwal, H., D'Souza, M.: Robustness verification for object detectors using set-based reachability analysis. In: Proceedings of the 34th Interna-

tional Conference on Artificial Neural Networks (ICANN25). pp. 481–492. Lecture Notes in Computer Science, Springer (2025)
16. De Palma, A., Bunel, R., Desmaison, A., Dvijotham, K., Kohli, P., Torr, P., Kumar, M.P.: Improved branch and bound for neural network verification via lagrangian decomposition. arXiv preprint arXiv:2104.06718 (2021)
17. Demarchi, S., Guidotti, D., Pulina, L., Tacchella, A.: Never2: Learning and verification of neural networks. Soft Computing **28**(19), 11647–11665 (2024)
18. Ducoffe, M., Carrere, M., Féliers, L., Gauffriau, A., Mussot, V., Pagetti, C., Sammour, T.: Lard – landing approach runway detection – dataset for vision based landing. arXiv preprint arXiv:2304.09938 (2023)
19. Ducoffe, N.C.M., Boumazouza, R., Gabrea, C., Pagetti, C., Pucel, X., Galametz, A.: Verifiou – robustness of object detection to perturbations. In: Proceedings of the 44th Digital Avionics Systems Conference (DASC25). pp. 1–10 (2025)
20. Duong, H., Li, L., Nguyen, T., Dwyer, M.B.: A dpll(t) framework for verifying deep neural networks. arXiv preprint arXiv:2307.10266 **abs/2307.10266** (2023)
21. Ehlers, R.: Formal verification of piece-wise linear feed-forward neural networks. In: Proceedings of the 15th International Symposium on Automated Technology for Verification and Analysis (ATVA17). Lecture Notes in Computer Science, vol. 10482, pp. 269–286. Springer (2017)
22. Everingham, M., Gool, L.V., Williams, C.K.I., Winn, J.M., Zisserman, A.: The pascal visual object classes (voc) challenge. International Journal of Computer Vision **88**(2), 303–338 (2010)
23. Fazlyab, M., Morari, M., Pappas, G.J.: Safety verification and robustness analysis of neural networks via quadratic constraints and semidefinite programming. IEEE Transactions on Automatic Control (2020)
24. Ferlez, J., Khedr, H., Shoukry, Y.: Fast batllnn: Fast box analysis of two-level lattice neural networks. In: Proceedings of the 25th ACM International Conference on Hybrid Systems: Computation and Control (HSCC22). pp. 23:1–23:11. ACM (2022)
25. Ferrari, C., Mueller, M., Jovanović, N., Vechev, M.: Complete verification via multi-neuron relaxation guided branch-and-bound. In: Proceedings of the 10th International Conference on Learning Representations (ICLR22). Openreview.net (2022)
26. Ge, Z., Liu, S., Li, Z., Yoshie, O., Sun, J.: Ota: Optimal transport assignment for object detection. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR21). pp. 303–312 (2021)
27. Goodfellow, I., Shlens, J., Szegedy, C.: Explaining and harnessing adversarial examples. In: Proceedings of the 3rd International Conference on Learning Representations (ICLR15) (2015)
28. Gowal, S., Dvijotham, K., Stanforth, R., Bunel, R., Qin, C., Uesato, J., Arandjelovic, R., Mann, T., Kohli, P.: On the effectiveness of interval bound propagation for training verifiably robust models. arXiv preprint arXiv:1810.12715 (2019)
29. Hammack, R.: Book of Proof. Richard Hammack, 3rd edn. (2018)
30. Hashemi, V., Kouvaros, P., Lomuscio, A.: Osip: Tightened bound propagation for the verification of relu neural networks. In: Proceedings of the 19th International Conference on Software Engineering and Formal Methods (SEFM21). pp. 463–480. IEEE Computer Society (2021)
31. Henriksen, P., Hammernik, K., Rueckert, D., Lomuscio, A.: Bias field robustness verification of large neural image classifiers. In: Proceedings of the 32nd British Machine Vision Conference (BMVC21). BMVA Press (2021)

32. Henriksen, P., Lomuscio, A.: DEEPSPLIT: an efficient splitting method for neural network verification via indirect effect analysis. In: Proceedings of the 30th International Joint Conference on Artificial Intelligence (IJCAI21). pp. 2549–2555. ijcai.org (2021)
33. Katz, G., Barrett, C., Dill, D., Julian, K., Kochenderfer, M.: Reluplex: An efficient SMT solver for verifying deep neural networks. In: Proceedings of the 29th International Conference on Computer Aided Verification (CAV17). Lecture Notes in Computer Science, vol. 10426, pp. 97–117. Springer (2017)
34. Katz, G., Huang, D., Ibeling, D., Julian, K., Lazarus, C., Lim, R., Shah, P., Thakoor, S., Wu, H., Zeljic, A., Dill, D., Kochenderfer, M., Barrett, C.: The marabou framework for verification and analysis of deep neural networks. In: Proceedings of the 31st International Conference on Computer Aided Verification (CAV19). pp. 443–452 (2019)
35. Kaulen, K., Ladner, T., Bak, S., Brix, C., Duong, H., Flinkow, T., Johnson, T.T., Koller, L., Manino, E., Nguyen, T.H., Wu, H.: The 6th international verification of neural networks competition (vnn-comp 2025): Summary and results. arXiv preprint 2512.19007 (2025)
36. Kim, K., Lee, H.S.: Probabilistic anchor assignment with iou prediction for object detection. In: Proceedings of the 16th European Conference on Computer Vision (ECCV20). pp. 355–371. Springer (2020)
37. Kouvaros, P., Brückner, B., Henriksen, P., Lomuscio, A.: Dynamic back-substitution in bound-propagation-based neural network verification. In: Proceedings of the 39th AAAI Conference on Artificial Intelligence (AAAI25). pp. 27383–27391. AAAI Press (2025)
38. Kouvaros, P., Kyono, T., Leofante, F., Lomuscio, A., Margineantu, D., Osipychev, D., Zheng, Y.: Formal analysis of neural network-based systems in the aircraft domain. In: Proceedings of the 24th International Symposium on Formal Methods (FM21). Lecture Notes in Computer Science, vol. 13047, pp. 730–740. Springer (2021)
39. Kouvaros, P., Lomuscio, A.: Towards scalable complete verification of relu neural networks via dependency-based branching. In: Proceedings of the 30th International Joint Conference on Artificial Intelligence (IJCAI21). pp. 2643–2650. ijcai.org (2021)
40. Lan, J., Brückner, B., Lomuscio, A.: A semidefinite relaxation based branch-and-bound method for tight neural network verification. In: Proceedings of the 37th AAAI Conference on Artificial Intelligence (AAAI23). pp. 14946–14954. AAAI Press (2023)
41. Lemesle, A., Lehmann, J., Gall, T.L.: Neural network verification with pyrat. arXiv preprint arXiv:2410.23903 (2024)
42. Lin, T.Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., Zitnick, C.L.: Microsoft coco: Common objects in context. In: Proceedings of the 13th European Conference on Computer Vision (ECCV 2014). pp. 740–755. Lecture Notes in Computer Science, Springer (2014)
43. Litjens, G., Kooi, T., Bejnordi, B.E., Setio, A.A., Ciompi, F., Ghafoorian, M., Laak, J.A.V.D., Ginneken, B.V., Sánchez, C.I.: A survey on deep learning in medical image analysis. Medical Image Analysis **42**, 60–88 (2017)
44. Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S.E., Fu, C.Y., Berg, A.C.: Ssd: Single shot multibox detector. In: Proceedings of the 14th European Conference on Computer Vision (ECCV16). Lecture Notes in Computer Science, vol. 9905, pp. 21–37. Springer (2016)

45. Liu, Z., Yu, L., Lin, T., Chi, Z., Zhang, L.: Certifying the full YOLO pipeline: A probabilistic verification approach. In: Proceedings of the 14th International Conference on Learning Representations (ICLR26). OpenReview.net (2026)
46. Lomuscio, A., Maganti, L.: An approach to reachability analysis for feed-forward relu neural networks. arXiv preprint 1706.07351 (2017)
47. Lopez, D.M., Choi, S.W., Tran, H.D., Johnson, T.T.: Nnv 2.0: The neural network verification tool. In: Proceedings of the 35th International Conference on Computer Aided Verification (CAV23). Lecture Notes in Computer Science, vol. 13965, pp. 397–412. Springer (2023)
48. Mellouki, O.E., Khedher, M.I., El-Yacoubi, M.A.: Abstract layer for leakyrelu for neural network verification based on abstract interpretation. IEEE Access **11**, 33401–33413 (2023)
49. Mziou-Sallami, M., Adjed, F.: Towards a certification of deep image classifiers against convolutional attacks. In: Proceedings of the 14th International Conference on Agents and Artificial Intelligence (ICAART22). pp. 419–428 (2022)
50. Nirala, A.K., Sarkar, S.: Towards certified object detectors: Certified runway detection using yolo. In: Proceedings of the 2025 IEEE International Conference on Image Processing (ICIP25). pp. 827–832 (2025)
51. Pulina, L., Tacchella, A.: Challenging SMT solvers to verify neural networks. AI Communications **25**(2), 117–135 (2012)
52. Raghunathan, A., Steinhardt, J., Liang, P.S.: Semidefinite relaxations for certifying robustness to adversarial examples. In: Proceedings of the 32nd Annual Conference on Neural Information Processing Systems (NeurIPS18). pp. 10900–10910 (2018)
53. Raviv, A., Elboher, Y.Y., Aluf-Medina, M., Weiss, Y.L., Cohen, O., Assa, R., Katz, G., Kugler, H.: Formal verification of object detection. arXiv preprint arXiv:2407.01295 (2024)
54. Redmon, J., Divvala, S., Girshick, R., Farhadi, A.: You only look once: Unified, real-time object detection. In: Proceedings of the 29th IEEE Conference on Computer Vision and Pattern Recognition (CVPR16). pp. 779–788 (2016)
55. Redmon, J., Farhadi, A.: Yolo9000: Better, faster, stronger. In: Proceedings of the 30th IEEE Conference on Computer Vision and Pattern Recognition (CVPR17). pp. 6517–6525 (2017)
56. Redmon, J., Farhadi, A.: Yolov3: An incremental improvement. arXiv preprint 1804.02767 (2018)
57. Singh, G., Gehr, T., Mirman, M., Püschel, M., Vechev, M.: Fast and effective robustness certification. In: Proceedings of the 32nd Annual Conference on Neural Information Processing Systems (NeurIPS18). pp. 10802–10813 (2018)
58. Singh, G., Gehr, T., Püschel, M., Vechev, M.: An abstract domain for certifying neural networks. Proceedings of the ACM on Programming Languages **3**(POPL), 41 (2019)
59. Singh, G., Gehr, T., Püschel, M., Vechev, M.: Boosting robustness certification of neural networks. In: Proceedings of the 7th International Conference on Learning Representations (ICLR19). OpenReview.net (2019)
60. Tjeng, V., Xiao, K., Tedrake, R.: Evaluating robustness of neural networks with mixed integer programming. In: Proceedings of the 7th International Conference on Learning Representations (ICLR19) (2019)
61. Wang, F., Zhang, Y., Yin, X., Cheng, G., Fu, Z., Huang, X., Ruan, W.: A black-box evaluation framework for semantic robustness in bird's eye view detection. In: Proceedings of the 39th AAAI Conference on Artificial Intelligence (AAAI-25). pp. 7637–7645. AAAI Press (2025)

62. Wang, S., Pei, K., Whitehouse, J., Yang, J., Jana, S.: Efficient formal safety analysis of neural networks. In: Proceedings of the 32nd Annual Conference on Neural Information Processing Systems (NeurIPS18). pp. 6367–6377. Curran Associates, Inc. (2018)
63. Wang, S., Pei, K., Whitehouse, J., Yang, J., Jana, S.: Formal security analysis of neural networks using symbolic intervals. In: Proceedings of the 27th USENIX Security Symposium (USENIX18) (2018)
64. Wang, S., Zhang, H., Xu, K., Lin, X., Jana, S., Hsieh, C., Kolter, J.: Beta-crown: Efficient bound propagation with per-neuron split constraints for neural network robustness verification. In: Proceedings of the 34th Annual Conference on Neural Information Processing Systems (NeurIPS21). vol. 34, pp. 29909–29921. Curran Associates, Inc. (2021)
65. Wu, H., Isac, O., Zeljić, A., Tagomori, T., Daggitt, M., Kokke, W., Refaeli, I., Amir, G., Julian, K., Bassan, S., Huang, P., Lahav, O., Wu, M., Zhang, M., Komendantskaya, E., Katz, G., Barrett, C.: Marabou 2.0: A versatile formal analyzer of neural networks. In: Computer Aided Verification. Lecture Notes in Computer Science, vol. 14682, pp. 249–264. Springer (2024)
66. Zhang, H., Wang, S., Xu, K., Li, L., Li, B., Jana, S., Hsieh, C.J., Kolter, J.Z.: General cutting planes for bound-propagation-based neural network verification. In: Proceedings of the 36th Conference on Neural Information Processing Systems (NeurIPS22). pp. 1656–1670. Curran Associates, Inc. (2022)
67. Zhang, H., Weng, T., Chen, P., Hsieh, C., Daniel, L.: Efficient neural network robustness certification with general activation functions. In: Proceedings of the 32nd Annual Conference on Neural Information Processing Systems (NeurIPS18). pp. 4944–4953 (2018)
68. Zhang, S., Chi, C., Yao, Y., Lei, Z., Li, S.Z.: Bridging the gap between anchor-based and anchor-free detection via adaptive training sample selection. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR20). pp. 9756–9765 (2020)
69. Zhang, Y., Chen, J., Peng, Z., Dang, Y., Shi, Z., Zou, Z.: Physical adversarial attacks against aerial object detection with feature-aligned expandable textures. IEEE Transactions on Geoscience and Remote Sensing **62**, 1–15 (2024)
70. Zhang, Y., Kouvaros, P., Lomuscio, A.: Scalable neural network geometric robustness validation via hölder optimisation. In: Proceedings of the 39th Annual Conference on Neural Information Processing Systems (NeurIPS25). OpenReview.net (2025)
71. Zhang, Y., Wang, F., Ruan, W.: Fooling object detectors: Adversarial attacks by half-neighbor masks. arXiv preprint arXiv:2101.00989 (2021)
72. Zhou, D., Brix, C., Hanasusanto, G.A., Zhang, H.: Scalable neural network verification with branch-and-bound inferred cutting planes. In: Proceedings of the 38th Annual Conference on Neural Information Processing Systems (NeurIPS24) (2024)
73. Zhou, D., Chavez, J., Chen, H., Hanasusanto, G.A., Zhang, H.: Clip-and-verify: Linear constraint-driven domain clipping for accelerating neural network verification. In: Proceedings of the 39th Annual Conference on Neural Information Processing Systems (NeurIPS25). OpenReview.net (2025)
74. Zou, Z., K.Chen, Shi, Z., Guo, Y., Ye, J.: Object detection in 20 years: A survey. Proceedings of the IEEE **111**(3), 257–276 (2023)