Verifying Neural Network Robustness with Dual Perturbations

Abstract

Safety-critical deep learning systems must be robust against real-world corruptions combining spatially correlated distortions and independent noise. Current deep neural network verification methods handle these perturbations separately, either checking independent pixel-wise perturbations or restricted convolutional transformations using predefined patterns. This gap prevents assessing robustness under realistic conditions where both perturbation types occur simultaneously. To address these limitations, we propose `VeriDou`, a framework that introduces: (i) universal convolutional perturbations that enable verification across continuous spatial distortion spaces, and (ii) dual perturbations that capture both convolutional distortions and independent pixel-level variations. Our evaluation on a set of diverse benchmarks with 14340 instances shows `VeriDou`'s dual perturbations approach found substantially more adversarial examples on networks that existing methods claimed to be highly robust. This shows that `VeriDou` is able to explore a broader range of unsafe regions and thus enhances formal assessment of robustness.

1. Introduction

Robustness [3, 8, 25, 30] is a fundamental property of deep neural networks (DNNs) that ensures consistent behavior when inputs are perturbed. The rapid deployment of DNNs in safety-critical computer vision (CV) applications, *e.g.*, autonomous driving [34] and medical diagnosis [31] has made the formal reasoning about robustness increasingly crucial for ensuring reliability. Recent years have witnessed significant advancements in DNN *verification* techniques [7, 11, 12, 40, 43] to prove robustness and safety properties of various systems, *e.g.*, image classification [9] and aircraft collision avoidance [21]. In particular, DNN verification has been developed for CV systems, *e.g.*, abstraction-based training [22, 42], tight linear approximation for MaxPool [36, 37], and verified perturbation analysis [13, 15] to certify robustness properties, explainability guarantees, and complex geometric transformations.

Tab. 1. Comparison of verification approaches (**Direct**: handling directly and not reformulating to overapproximated property; **Restricted**: limited to interpolations of predefined kernels; **Universal**: arbitrary kernel coefficient combinations within bounds).

Method	Independent Perturbation	Convolutional Perturbation		
		Direct	Restricted	Universal
$\alpha\beta$-CROWN [43]	✓	✗	✗	✗
NeuralSAT [12]	✓	✗	✗	✗
Venus [4]	✓	✗	✗	✗
Mziou-Sallami *et al.* [24]	✗	✗	✓	✗
Ruoss *et al.* [27]	✗	✗	✓	✗
Brückner *et al.* [7]	✗	✓	✓	✗
`VeriDou` (ours)	✓	✓	✓	✓

However, verifying robustness in CV systems is still in its infancy and is far from practical deployment. For example, existing DNN verification for CV systems [7, 22, 24, 27, 36, 42] cannot deal with multiple types of realistic perturbations that often occur simultaneously in real-world scenarios, *e.g.*, autonomous vehicle cameras experience vibration-induced blur and sensor noise [28]. These distortions are often categorized into two types: convolutional and independent perturbations, which exhibit distinct characteristics posing unique challenges to verification. *Convolutional perturbations* manifest images through weighted combinations of neighboring pixels, *e.g.*, motion blur [17]. In contrast, *independent perturbations* introduce separate noise to each pixel, *e.g.*, brightness fluctuations, modeled as ℓ_∞ constraints. This dual nature creates opportunities for designing verification approaches that can jointly model spatial dependencies and independent variations, which essential for robust analysis in CV systems.

Tab. 1 summarizes the capabilities of existing verification methods, highlighting that current approaches either support independent perturbations or restricted forms of convolutional perturbations, but none provide coverage of both perturbation types with universal convolution. State-of-the-art DNN verifiers primarily focus on independent perturbations using interval-based methods [12, 18, 20, 25, 40], which scale efficiently but cannot model spatial dependencies. Recent work [7, 24, 27] has attempted to address convolutional perturbations, but with restrictive constraints or limited kernels. Critically, no existing approaches support both independent and convolutional perturbations si-

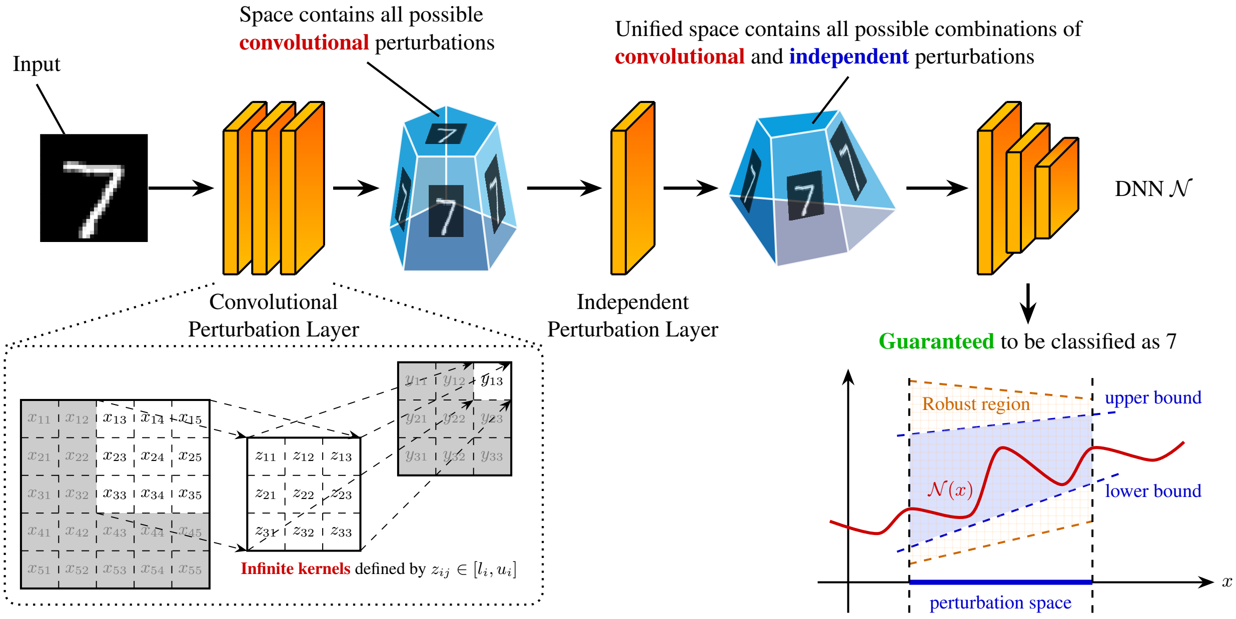
Input
Space contains all possible convolutional perturbations
Unified space contains all possible combinations of convolutional and independent perturbations
DNN $\mathcal{N}$
Convolutional Perturbation Layer
Independent Perturbation Layer
Guaranteed to be classified as 7
Robust region
upper bound
$\mathcal{N}(x)$
lower bound
perturbation space
x
x_{11} x_{12} x_{13} x_{14} x_{15}
x_{21} x_{22} x_{23} x_{24} x_{25}
x_{31} x_{32} x_{33} x_{34} x_{35}
x_{41} x_{42} x_{43} x_{44} x_{45}
x_{51} x_{52} x_{53} x_{54} x_{55}
z_{11} z_{12} z_{13}
z_{21} z_{22} z_{23}
z_{31} z_{32} z_{33}
y_{11} y_{12} y_{13}
y_{21} y_{22} y_{23}
y_{31} y_{32} y_{33}
Infinite kernels defined by $z_{ij} \in [l_i, u_i]$

Fig. 1. Systematic overview of `VeriDou` framework. `VeriDou` encapsulates both convolutional (infinite number of possible kernels in the predefined space) and independent perturbations (infinite number of possible pixel-level variations), then formally analyzes the perturbation space to provide a guarantee for robustness (outputs are guaranteed to be classified as original class).

multaneously, limiting their applicability to realistic CV scenarios where multiple perturbation types co-occur.

To bridge this gap, we introduce `VeriDou`, a verification framework that supports independent and arbitrary convolutional perturbations, called *dual perturbations*, through a single formulation. Fig. 1 shows an overview of `VeriDou` framework. `VeriDou` first captures *convolutional perturbation*, where kernels are controlled by independent variables within specified bounds, handling *universal* convolutional perturbations across arbitrary kernels. Second, combined with *independent perturbation* for pixel-level variations, dual perturbations are encapsulated by a unified perturbation space. Both types of perturbations are encoded in perturbation DNN layers, then prepended to the original network. Third, `VeriDou` computes network overapproximation under the perturbation space via underlying verifiers; if the overapproximation is subsumed by the property (robust region), the network is guaranteed robust. This enables verification of richer properties, where a single instance encompasses infinite perturbation combinations across continuous kernel spaces, revealing counterexamples that existing approaches cannot achieve.

We demonstrate `VeriDou`'s feasibility and expressivity through evaluations on diverse benchmarks from recent Verification of Neural Networks Competitions (VNN-COMPs) [2, 6]. Our evaluation shows that while existing approaches claim high robustness rates (up to 100%), dual perturbations yields substantially more adversarial examples (up to 99%) on the same networks. This demonstrates that networks exhibiting robustness against specific perturbations lack generalization capabilities for broader distortions, revealing a critical gap in the current robustness assessment that `VeriDou` addresses.

Our contributions include: (1) A dual perturbation that models real-world distortions handling both convolutional and independent variations; (2) An algorithmic approach and implementation tool, `VeriDou`, that transforms complex specifications into standard verification problems; (3) An evaluation demonstrating that dual perturbations reveal fundamental gaps in DNN robustness assessment.

2. Background

Convolutional Perturbation Convolutional perturbation extends local robustness to handle spatial distortions. Unlike ℓ_∞-norm perturbations, convolutional perturbations apply a kernel K to the input X, resulting in the perturbed input $X * K$. These perturbations introduce spatial dependencies through convolutions, where neighboring pixels influence each other to create correlated variations rather than independent changes.

The verification problem then asks whether a DNN N complies to a property ϕ across all possible variations K within a kernel space $\mathcal{K}$.

$$\forall K \in \mathcal{K} : \phi(N(X * K)) \tag{1}$$

The key to verification is constructing an appropriate kernel space $\mathcal{K}$ (*e.g.*, how to define a convolution kernel K in $\mathcal{K}$) that precisely captures the desired perturbations.

Restricted Convolutional Perturbation Restricted convolutional perturbation $K(Z)$ uses n variables $Z = [z_1, \ldots, z_n]$, each variable z_i controls one target kernel K_i:

$$K(Z) = \sum_{i=1}^{n} z_i \cdot K_i + \left(1 - \sum_{i=1}^{n} z_i\right) \cdot Id \tag{2}$$

where $z_i \in [0, 1]$, $\sum_{i=1}^{n} z_i \leq 1$, K_i is a target kernel and Id is the identity kernel with the same size as K_i. Intuitively, $K(Z)$ is a linear combination of target kernels K_i with scale factors z_i. The term $\left(1 - \sum_{i=1}^{n} z_i\right) \cdot Id$ ensures zero perturbation when all $z_i = 0$ ($X * K(Z) = X$).

DNN Verification Given a network N and a property $\phi \equiv \phi_{in} \implies \phi_{out}$, verification problem [10, 19] reduces to checking satisfiability:

$$\alpha \wedge \phi_{in} \wedge \neg\phi_{out} \tag{3}$$

If Eq. 3 is unsatisfiable (UNSAT) and the considered property holds (*e.g.*, ϕ is a valid property of N). Otherwise, it is satisfiable (SAT) and a counterexample (*e.g.*, adversarial example) exists that disproves the property ϕ. To improve scalability, modern DNN verifiers [1, 12, 14, 26, 35, 43] adopt abstract interpretation [29, 38] to soundly overapproximate network behaviors.

3. Universal Convolutional Perturbation

There are several limitations of the restricted convolutional perturbation. First, it is unable to express slight modification of target kernel. Consider motion blur 3 × 3 at 0°, we have the following restricted kernel $K(Z)$ following Eq. 2:

$$(1 - z_1) \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} + z_1 \begin{bmatrix} 0 & 0 & 0 \\ \frac{1}{3} & \frac{1}{3} & \frac{1}{3} \\ 0 & 0 & 0 \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 \\ \frac{z_1}{3} & 1 - \frac{2z_1}{3} & \frac{z_1}{3} \\ 0 & 0 & 0 \end{bmatrix} \tag{4}$$

When z_1 varies, all entries of $K(Z)$ must change together in the same direction and proportion. It fails to capture many patterns, *e.g.*, asymmetric perturbations that require slight changes of different entries such as a slight change of an entry $\frac{z_1}{3}$ to $\frac{z_1+0.01}{3}$. Second, this approach only captures perturbations at extreme points defined by the target kernels, *e.g.*, motion blur 0° or 45°. It fundamentally cannot capture continuous intermediate orientations (*e.g.*, 30°) that lie between them.

To address the limitations of restricted perturbations in capturing continuous transformation spaces and arbitrary kernel patterns, we introduce *universal convolutional perturbations* as our core contribution that enables verification over complete kernel neighborhoods.

Def. 1 (Universal Convolutional Perturbation)**.** *A universal convolutional perturbation uses $k_1 \times k_2$ variables $Z = [z_{11}, \ldots, z_{k_1 k_2}]$, each z_{ij} controls a kernel entry:*

$$K(Z) = \sum_{i=1}^{k_1} \sum_{j=1}^{k_2} U_{ij} \cdot z_{ij} + Id \tag{5}$$

where $z_{ij} \in [l_{ij}, u_{ij}]$, each $U_{ij} \in \mathbb{R}^{k_1 \times k_2}$ is a unit kernel with a single one at position (i, j) and zeros elsewhere.

Compared to the restricted perturbation, $K(Z)$ is constructed directly from the variables Z without involving predefined target kernels. The term Id also ensures no perturbation when all $z_{ij} = 0$. Consider verifying motion blur 5 × 5 across orientations from 0° to 45°, we construct a *range-covering* kernel space for continuous angles. Specifically, we mark every kernel position that a blur line would pass through while rotating from 0° to 45°. Such positions are controlled by different variables, while others are set to zero. Using Def. 1, the kernel $K(Z)$ can be defined as:

$$K(Z) = \begin{bmatrix} z_{11} & 0 & 0 & 0 & 0 \\ z_{21} & z_{22} & 0 & 0 & 0 \\ z_{31} & z_{32} & z_{33} & z_{34} & z_{35} \\ 0 & 0 & 0 & z_{44} & z_{45} \\ 0 & 0 & 0 & 0 & z_{55} \end{bmatrix} \tag{6}$$

where non-zero entries $z_{ij} \in [0, \frac{1}{5}]$. First, this formulation addresses limitations of the restricted convolutional perturbation, *e.g.*, allowing for arbitrary combinations of kernel entries rather than requiring all entries to change together in the same direction and proportion. Second, it is able to capture any intermediate orientations of interest (*e.g.*, 30°).

3.1. Improving Expressivity

To formally characterize the expressiveness of universal perturbations, we prove that universal perturbations subsume all image transformations expressible by restricted perturbations, establishing the theoretical foundation for our approach.

Lem. 1 (Kernel Space Containment)**.** *Let $\mathcal{K}_{res}$ and $\mathcal{K}_{uni}$ be sets of all kernels expressible by restricted and universal perturbations, respectively. Then $\mathcal{K}_{res} \subseteq \mathcal{K}_{uni}$.*

Proof. Let $K_{\text{res}}(z_1, \ldots, z_n) = \sum_{i=1}^{n} z_i \cdot K_i + (1 - \sum_{i=1}^{n} z_i) \cdot Id = \sum_{i=1}^{n} z_i \cdot (K_i - Id) + Id$. We construct kernel parameters $Z^* = [z^*_{11}, z^*_{12}, \ldots, z^*_{k_1 k_2}]$ where $z^*_{pq} = \sum_{i=1}^{n} z_i \cdot (K_i - Id)_{pq}$. Then, $K_{\text{uni}}(Z^*) = \sum_{p,q} U_{pq} \cdot z^*_{pq} + Id = \sum_{i=1}^{n} z_i \cdot (K_i - Id) + Id = K_{\text{res}}$, showing any restricted kernel can be expressed as a universal kernel. □

Thm. 1 (Perturbation Space Containment)**.** *Let $\mathcal{P}_{res} = \{X * K : K \in \mathcal{K}_{res}\}$ be the set of all perturbed images generated by restricted convolutional perturbation, and $\mathcal{P}_{uni} = \{X * K : K \in \mathcal{K}_{uni}\}$ be the set of all perturbed* *images generated by universal convolutional perturbation. Then $\mathcal{P}_{res} \subseteq \mathcal{P}_{uni}$.*

Proof. Since $\mathcal{K}_{\text{res}} \subseteq \mathcal{K}_{\text{uni}}$ (from Lem. 1), for any perturbed image $X * K$ where $K \in \mathcal{K}_{\text{res}}$, we have $K \in \mathcal{K}_{\text{uni}}$, which implies $X * K \in \mathcal{P}_{\text{uni}}$. Therefore, $\mathcal{P}_{\text{res}} \subseteq \mathcal{P}_{\text{uni}}$. □

As established by Thm. 1, the universal formulation provides a more expressive perturbation space than restricted approaches. Importantly, by definition, universal convolutional perturbation enables arbitrary kernel coefficients within specified bounds, not limited to predefined kernel combinations. In particular, given a kernel $K_{\text{arbitrary}}$, we define the kernel space $\mathcal{K}$ as $z_{ij} \in [c_{ij} - \epsilon_{ij}, c_{ij} + \epsilon_{ij}]$ where centers c_{ij} are derived from specific domain requirements (*e.g.*, driving scenarios [23], or natural image statistics [39]), and ϵ_{ij} are perturbation radii.

Next, we introduce VeriDou in §4, a verification framework leveraging universal convolutional perturbation to represent multiple verification scenarios within a single formulation, where arbitrary kernels (*e.g.*, motion blur, defocus, camera shake) can be continuously varied and verified. In particular, one of VeriDou's properties encapsulates a spectrum of perturbations such that verifying a single instance covers infinite discrete kernel patterns.

4. The VeriDou Approach

Fig. 1 illustrates the VeriDou framework for transforming both perturbation types into a unified verification problem. The VeriDou algorithm, shown in Alg. 1, takes in an input (*e.g.*, an image X), and a network N, and constructs perturbation transformations by computing the convolutional component W_C through pre-convolving the input with unit kernels U_{ij} (line 2) and the independent component R' by masking perturbation ranges with coverage C (line 3). The convolutional layer (§4.1) generates all possible kernel variations within bounds $\mathcal{K}$, while the independent layer (§4.2) produces pixel-level variations within ℓ_∞ constraints. VeriDou then unifies these by constructing transformation matrix W (line 6) and perturbation layer L (line 7), yielding unified network $M = N \circ L$ (line 8) operating over the dual perturbation space. Finally, VeriDou formulates the verification problem by defining input space S as the Cartesian product of kernel and pixel bounds (line 10) and specifying robustness property ϕ requiring consistent classification (line 11). This enables the verifier V to provide formal guarantees across the entire perturbation space for both perturbations.

Alg. 1: VeriDou Algorithm

```
input  : Network $N$; input $X \in \mathbb{R}^d$;
         Convolutional perturbation $Z \in [z_L, z_U]^{k_1 \times k_2}$;
         Independent perturbation $R \in [\epsilon_R, \epsilon_R]^d$;
         Perturbation coverage $C \in \{0,1\}^d$;
         Oracle verifier $V$;
output : SAT/UNSAT/TIMEOUT
1  1. Perturbation construction
2  $W_C \leftarrow \sum_{i=1}^{k_1} \sum_{j=1}^{k_2} (X * U_{i,j})$          ▷ Convolutional
3  $R' \leftarrow R \times C$                                         ▷ Independent
4  $s \leftarrow \begin{bmatrix} Z & R' \end{bmatrix}$                ▷ Unified perturbation variables
5  2. Unified perturbation network
6  $W \leftarrow \begin{bmatrix} W_C & Id \end{bmatrix}^T$             ▷ Transformation matrix
7  $L \leftarrow s \cdot W + X$                                       ▷ Dual perturbation layer
8  $M \leftarrow N \circ L$                                           ▷ Perturbation network
9  3. Verification problem
10 $S \leftarrow [z_L, z_U]^{k_1 \times k_2} \times [\epsilon_R, \epsilon_R]^d$   ▷ Input Space
11 $\phi \leftarrow \{\forall s \in S \Rightarrow \arg\max(M(s)) = \arg\max(N(X))\}$
12 return $V(\phi)$                                                   ▷ Verify problem
```

4.1. Convolutional Perturbation Layer

For an input $X \in \mathbb{R}^d$ and universal kernel $K(Z) \in \mathcal{K}$ constructed in §3, the perturbed input $X_C \in \mathbb{R}^d$ is computed:

$$X_C = X * K(Z) = \sum_{i=1}^{k_1} \sum_{j=1}^{k_2} (X * U_{i,j}) \cdot z_{i,j} + X * Id \tag{7}$$

This decomposition enables independent control over each kernel coefficient, allowing arbitrary perturbation patterns within the specified bounds while maintaining computational efficiency through pre-computed terms $(X * U_{i,j})$.

Next, we construct a transformation matrix $W_C \in \mathbb{R}^{d \times k_1 \times k_2}$ that maps perturbation variables to convolution outputs. For a specific input image $X \in \mathbb{R}^d$,

$$W_C = \sum_{i=1}^{k_1} \sum_{j=1}^{k_2} (X * U_{i,j}) \tag{8}$$

where each entry of W_C represents the convolution output with the corresponding unit kernel $U_{i,j}$. The convolutional perturbation transformation is then:

$$X_C = F_C(X, Z) = X * K(Z) = W_C \cdot Z + X \tag{9}$$

4.2. Independent Perturbation Layer

For input $X \in \mathbb{R}^d$ and perturbation range $R \in [-\epsilon_R, \epsilon_R]^d$, the independent perturbation transformation is defined following the ℓ_∞ norm formulation:

$$X_I = F_I(X, R, C) = X + R \times C \tag{10}$$

where $C \in \{0,1\}^d$ is the perturbation coverage that controls the spatial distribution of perturbations, *e.g.*, if $C_i = 1$, then i-th pixel is perturbed within the range $[X_i - \epsilon_R, X_i + \epsilon_R]$, otherwise the pixel is not perturbed.

$F_I(X, R, C)$ captures all the pixel-level variations varying within the ℓ_∞ ball with center X and radius ϵ_R. It captures the independent pixel-level variations which cannot be captured by convolutional perturbation.

Note that the independent perturbation integrates seamlessly with existing DNN verification techniques, as it can be represented as a standard hyper-rectangle (*e.g.*, interval bounds on inputs) that DNN verifiers often natively support.

4.3. Dual Perturbation Verification

Given the original network N, an image X to be verified, the new network M is systematically constructed to capture the robustness of X against both universal convolutional and independent perturbations:

$$M(s) \equiv N \circ F_I \circ F_C(s) \equiv N(W_C \cdot Z + X + R \times C) \quad (11)$$

where $s \equiv (Z, R, C) \in \mathbb{R}^{k_1 \times k_2 + d}$ is the unified input to the new network M, $I \in \mathbb{R}^d$ is the original d-dimensional input, $Z \in [z_L, z_U]^{k_1 \times k_2}$ are bounded kernel perturbation variables with asymmetric bounds, $R \in [\epsilon_R, \epsilon_R]^d$ is the independent perturbation, $C \in \{0, 1\}^d$ controls the spatial distribution of independent perturbations.

Note that the order of composition results in different perturbation patterns. However, the final form of the network M is the identical. In particular, $M(s) \equiv N \circ F_C \circ F_I(s) = W_C' * Z + X + R * C$, where $W_C' = \sum_{i=1}^{k_1} \sum_{j=1}^{k_2} ((X + R \times C) * U_{i,j})$.

We use the form of Eq. 11 to construct the new network M in our evaluation. To facilitate the verification, we integrate the both convolutional and independent perturbations into a single layer $L(s)$. Particularly, we construct a linear layer L for dual perturbations $W_C \cdot Z + X + R \times C$:

$$L(s) = \underbrace{\begin{bmatrix} Z & R \times C \end{bmatrix}}_{\text{Input } s} \cdot \underbrace{\begin{bmatrix} W_C \\ Id \end{bmatrix}}_{\text{Weight } W} + \underbrace{X}_{\text{Bias } b} \quad (12)$$

where Id is the identity matrix of size $d \times d$, $W \in \mathbb{R}^{d \times (k_1 \times k_2 + d)}$ is weight, and $b \in \mathbb{R}^d$ is bias of L. The unified network M in Eq. 11 now becomes:

$$M(s) \equiv N \circ L(s) \quad (13)$$

The unified input s to the new network M then is bounded by the space $S \equiv [z_L, z_U]^{k_1 \times k_2} \times [\epsilon_R, \epsilon_R]^d$. A verification property ϕ asserts M's prediction (*e.g.*, classification task) remains unchanged despite the presence of both convolutional perturbation and independent perturbation:

$$\forall s \in S \Rightarrow \arg\max\big(M(s)\big) = \arg\max\big(N(X)\big) \quad (14)$$

5. Evaluation

We evaluate the effectiveness of `VeriDou` through five research questions: **RQ1** (§5.2): How does `VeriDou` reveal adversarial examples compared to existing methods? **RQ2** (§5.3): How does `VeriDou` perform on different types of properties? **RQ3** (§5.4): How similar are adversarial examples found by `VeriDou` to original images? **RQ4** (§5.5): How do parameters affect `VeriDou`?

5.1. Experimental Setup

Benchmarks. We use five networks from existing work [2, 5, 6] including: `MNIST-FC`, `Oval21`, `Sri-ResNet-A`, `CIFAR100`, `TinyImageNet`. These networks cover a wide variety of architectures, including FC, CNN, and ResNet, spanning on different input (*e.g.*, 784–9408) and output dimensions (*e.g.*, 10–200) , and complexities (*e.g.*, 270K–3.8M parameters).

We evaluate `VeriDou` on two types of convolutional kernels: (1) *Specific Kernels* use motion blur at specific angles (0°, 15°, 30°, 45°, 60°, 75°, 90°) and continuous angles spanning (0°–30°, 30°–60°, 60°–90°). (2) *Arbitrary Kernels* use augmented kernels [39] initialized via Kaiming normal [16] with number of z_i ranging from 20% to 100% of the total number of kernel entries. Kernel sizes are $\{5, 7, 9\}$. Four types of perturbations are evaluated: (1) *Independent* with $R \in [-0.04, 0.04]^d$ and $C \in \{50\%, 100\%\}$; (2) *Restricted*; (3) *Universal*; and (4) *Dual*. In total, we evaluate 14340 verification problem instances, where each instance is a (network, property) pair.

Setup. We create three variants of `VeriDou` using different verifiers: `VeriDou`$_{\alpha\beta}$ uses $\alpha\beta$-CROWN [43], `VeriDou`$_{NS}$ uses NeuralSAT [12], top verifiers in recent VNN-COMPs [5, 6] which leverage GPU-based BaB, and `VeriDou`$_{VS}$ uses Venus [4], a CPU-based verifier.

Our test machine runs Linux with AMD CPU 32-Core, 128 GB RAM, and an NVIDIA GeForce RTX 4090 GPU with 24 GB VRAM. The timeout for each problem is 30s, which is sufficient for state-of-the-art verifiers, *e.g.*, often used in VNN-COMPs [2, 5, 6].

5.2. RQ1: Comparison with Existing Methods

Fig. 3 compares the effectiveness of `VeriDou`'s dual perturbations to existing methods (*e.g.*, independent and restricted perturbations). We only consider adversarial examples with low LPIPS [41] (*e.g.*, $\leq$ 0.4) to ensure fair comparison across all perturbations by restricting them to the same perceptual similarity space relative to original images. Independent perturbations find minimal violations, *e.g.*, only 0–20%, with `CIFAR100` showing 0% and `MNIST-FC`/`Oval21` achieving only 5%. Restricted perturbations reveal more vulnerabilities compared to independent ones, *e.g.*, increasing to 53% in `TinyImageNet`.

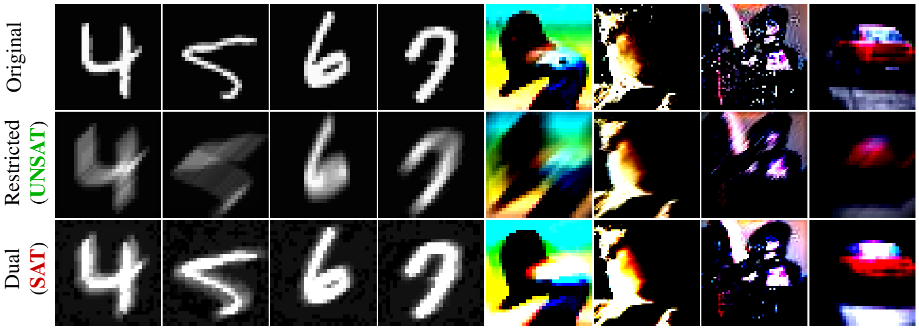

Fig. 2. Verification comparison: (top) original input images; (middle) most perturbed images of properties that restricted perturbations verified as safe (UNSAT); and (bottom) dual perturbations revealing adversarial examples (SAT). Restricted properties contain highly perturbed images but overlook adversarial examples close to originals, which `VeriDou` successfully captures.

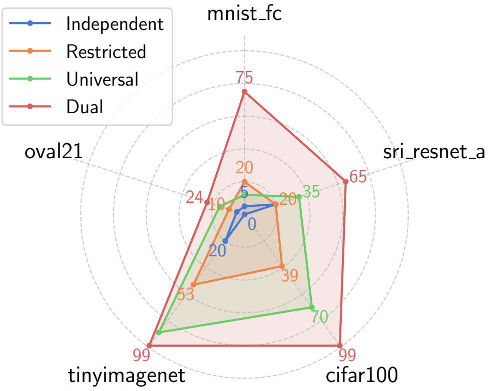

Fig. 3. Percentage of SAT instances (adversarial examples) found by different perturbation approaches across networks.

As universal properties are more expressive than restricted ones, they expose significantly more SAT instances, *e.g.*, `TinyImageNet` to 89%. Dual perturbations show a dramatic surge, reaching 65–99% SAT across networks. As shown in Fig. 2, dual perturbations successfully identify valid adversarial examples while others cannot. `MNIST-FC` increasing from 12% to 75%. This amplification demonstrates the complementary nature of the two perturbation types. While universal perturbations effectively capture structured transformations within the convolutional space, independent perturbations explore orthogonal dimensions that convolutional methods cannot reach, and vice versa. Neither perturbation type alone achieves the comprehensive coverage observed in their combination, highlighting that robustness assessment requires exploring structured and unstructured perturbations simultaneously.

Note that networks from `Oval21` include 3 ReLU-based CNNs that are robustly trained [2], and thus are harder to attack (*e.g.*, fewer SAT instances). However, `VeriDou` still manages to find 24% SAT instances for these networks, demonstrating the effectiveness of the dual perturbations. In contrast, networks from `CIFAR100` and `TinyImageNet` have high-dimensional inputs and outputs, and thus are easier (*e.g.*, 99% SAT instances).

5.3. RQ2: Analysis on `VeriDou` Performances

Discrete Kernels. Tab. 2 shows verification performance across different motion blur angles. Due to the limited expressiveness, restricted perturbations fail to capture possbile adversarial examples, thus, achieve significantly higher number of safe (UNSAT) instances (*e.g.*, 156–185 UNSAT instances) compared to dual ones (*e.g.*, 53–126 instances) using $\texttt{VeriDou}_{NS}$. The stability of verification results across different angles suggests that network robustness is not tied to specific geometric properties but rather reflect systematic properties of perturbations.

In contrast, dual perturbations consistently achieve more SAT instances than others (3×) with $\texttt{VeriDou}_{VS}$ showing the largest improvements (8×). Notably, the results expose a fundamental asymmetry that networks appearing robust to structured transformations (*e.g.*, $\texttt{VeriDou}_{\alpha\beta}$ achieves 169/73 UNSAT/SAT instances under restricted perturbations at 0°) become vulnerable against dual perturbations (*e.g.*, 50/206 UNSAT/SAT instances).

Continuous Kernels. Tab. 3 shows results for continuous angle ranges (0→30°, 30→60°, 60→90°) By definition, universal and dual perturbations naturally represent the entire spectrum, while restricted ones aggregate discrete points (*e.g.*, 0°, 15°, 30° for the 0→30° range). Restricted perturbations claim that networks are highly robust (*e.g.*, 186–191 UNSAT instances under $\texttt{VeriDou}_{NS}$) due to their limited space at specific angles, thus, missing potential adversarial examples at intermediate orientations. However, `VeriDou` reveals substantially more violations (*e.g.*, 135–161 SAT instances), illustrating that counterexamples indeed exist at intermediate angles and surrounding regions

Tab. 2. Discrete kernel results (UNSAT/SAT/TIMEOUT).

	Perturbation	**Motion 0°**	**Motion 15°**	**Motion 30°**	**Motion 45°**	**Motion 60°**	**Motion 75°**	**Motion 90°**
`VeriDou`$_{\alpha\beta}$	Restricted	169/73/58	164/85/51	148/93/59	138/105/57	154/88/58	171/73/56	169/81/50
	Universal	135/120/45	134/117/49	127/126/47	122/128/50	134/125/41	135/117/48	125/132/43
	Dual ($C = 0.5$)	75/163/62	79/170/51	75/170/55	65/178/57	73/173/54	81/172/47	86/172/42
	Dual ($C = 1.0$)	50/206/44	51/203/46	51/211/38	48/219/33	51/208/41	49/201/50	49/198/53
`VeriDou`$_{NS}$	Restricted	183/69/48	176/82/42	170/91/39	156/100/44	172/83/45	193/73/34	185/80/35
	Universal	126/120/54	123/117/60	116/125/59	113/129/58	121/125/54	121/117/62	114/132/54
	Dual ($C = 0.5$)	81/162/57	83/169/48	79/170/51	71/179/50	89/170/41	91/169/40	88/171/41
	Dual ($C = 1.0$)	56/203/41	58/199/43	57/206/37	53/215/32	59/205/36	59/199/42	56/195/49
`VeriDou`$_{VS}$	Restricted	142/34/124	136/40/124	129/40/131	121/46/133	138/35/127	151/23/126	142/35/123
	Universal	113/116/71	112/115/73	108/122/70	102/125/73	116/124/60	115/117/68	111/131/58
	Dual ($C = 0.5$)	60/150/90	60/160/80	60/161/79	51/167/82	59/165/76	68/162/70	66/164/70
	Dual ($C = 1.0$)	28/192/80	26/194/80	24/198/78	24/203/73	28/195/77	29/190/81	33/186/81

Tab. 3. Continuous kernel results (UNSAT/SAT/TIMEOUT).

	Perturbation	**0 → 30°**	**30 → 60°**	**60 → 90°**
`VeriDou`$_{\alpha\beta}$	Restricted	189/48/33	172/62/36	187/53/30
	Universal	98/136/36	88/115/67	103/129/38
	Dual ($C = 0.5$)	71/153/46	63/132/75	84/142/44
	Dual ($C = 1.0$)	51/168/51	47/138/85	54/171/45
`VeriDou`$_{NS}$	Restricted	190/47/33	186/58/26	191/52/27
	Universal	91/127/52	89/111/70	101/119/50
	Dual ($C = 0.5$)	74/139/57	66/127/77	85/136/49
	Dual ($C = 1.0$)	56/155/59	53/135/82	59/161/50
`VeriDou`$_{VS}$	Restricted	156/20/94	154/23/93	161/19/90
	Universal	87/121/62	76/101/93	97/112/61
	Dual ($C = 0.5$)	54/128/88	44/107/119	67/122/81
	Dual ($C = 1.0$)	33/134/103	28/113/129	37/136/97

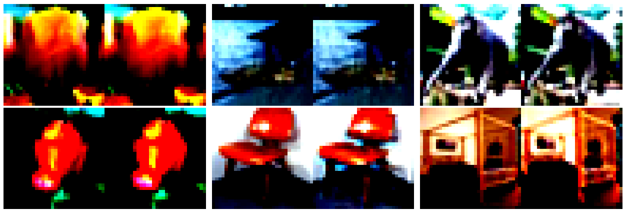

Fig. 4. Adversarial examples in arbitrary kernel properties. Each pair shows original image (left) and adversarial example (right).

Tab. 4. Arbitrary kernel results (UNSAT/SAT/TIMEOUT).

	Num. z_i **(%)**	**Domain 1**	**Domain 2**	**Domain 3**
`VeriDou`$_{\alpha\beta}$	20	132/81/87	121/92/87	114/99/87
	50	99/73/128	91/73/136	99/67/134
	100	74/58/168	81/50/169	73/65/162
`VeriDou`$_{NS}$	20	143/81/76	135/91/74	126/99/75
	50	99/73/128	94/73/133	102/67/131
	100	72/58/170	78/51/171	70/65/165
`VeriDou`$_{VS}$	20	117/81/102	105/92/103	101/99/100
	50	93/73/134	81/72/147	94/68/138
	100	68/57/175	78/51/171	71/65/164

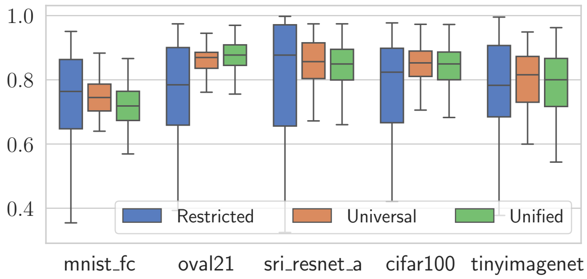

Fig. 5. SSIM scores of adversarial examples. Higher is better.

that restricted ones overlook entirely. Independent part is particularly effective at further discovering hidden vulnerabilities (*e.g.*, 142→171 under `VeriDou`$_{\alpha\beta}$) that neither restricted nor universal perturbations could.

Arbitrary Kernels. Tab. 4 presents results using diverse convolutional kernels across three different domains with number of z_i ranging from 20% to 100%. The results show a balanced distribution of UNSAT and SAT instances (*e.g.*, at 20% coverage, UNSAT ranges 101–143 while SAT ranges 81–99). The verification becomes more challenging as kernel parameter coverage increases, thus, timeout instances become increasingly prevalent, *e.g.*, under `VeriDou`$_{NS}$, timeouts of Domain 3 increase from 75 instances at 20% to 165 instances at 100% coverage. Note that the counterexamples discovered remain visually similar to original images (Fig. 4), indicating that `VeriDou` finds meaningful adversarial examples rather than imperceptible artifacts within diverse kernel transformation spaces.

5.4. RQ3: Adversarial Example Similarity

The similarity analyses in Fig. 5 and Fig. 6 quantify both structural (SSIM [33]) and perceptual (LPIPS [41]) fidelity

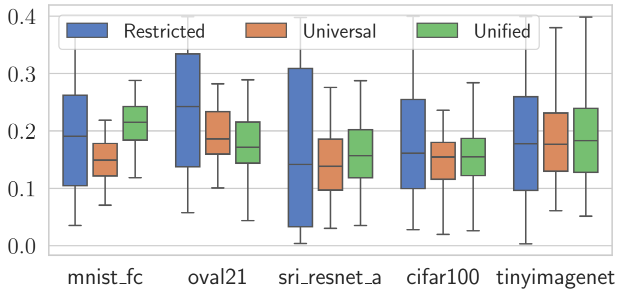

Fig. 6. LPIPS scores of adversarial examples. Lower is better.

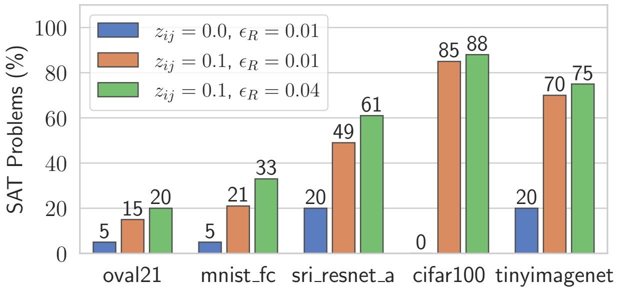

Fig. 7. Impact of Z and R on the robustness of networks.

of adversarial examples discovered by different perturbations. In general, adversarial examples maintain high structural similarity with SSIM scores consistently around 0.8, while perceptual distances remain minimal with LPIPS values below 0.3. However, the distribution characteristics reveal important differences, *e.g.*, universal and dual perturbations exhibit significantly tighter value distributions (narrower interquartile ranges) compared to restricted methods, indicating more consistent similarity preservation.

The narrower box plots for VeriDou's approaches suggest systematic discovery of adversarial examples within well-defined similarity bounds, rather than the scattered variations observed with restricted perturbations. Complementing these quantitative metrics, Fig. 2 provides visual evidence that dual perturbations identify counterexamples that appear virtually similar to original inputs, while restricted perturbations either overlook these subtle violations or contain more visually detectable distortions. This combination of quantitative consistency and visual authenticity suggests that VeriDou enables effective robustness assessment while preserving example realism.

5.5. RQ4: Ablation Study

Fig. 7 analyzes the contributions of convolutional (Z) and independent (R) parts on VeriDou's performances. The convolutional component Z shows substantial impact on revealing adversarial examples, *e.g.*, increasing from $z_{ij} = 0.0$ (independent) to $z_{ij} = 0.1$ (universal) reveals significantly more numbers of SAT instances across all networks, with CIFAR100 jumping from 0% to 85% and TinyImageNet rising from 20% to 70%.

The independent component R provides complementary benefits, with radius increases from $\epsilon_R = 0.01$ to $\epsilon_R = 0.04$ consistently boosting numbers of counterexamples found across all networks, *e.g.*, Oval21 improves from 15% to 20% and MNIST-FC advances from 21% to 33%. This orthogonal contributions between Z and R allows VeriDou to systematically probe different noise variations that reflect realistic imaging conditions.

6. Related Work

Modern DNN verifiers, *e.g.*, $\alpha\beta$-CROWN [32, 40, 43], NeuralSAT [10–12], PyRAT [26] and Marabou [35], primarily focus on independent perturbations, e.g., local robustness [2, 6]. Recent work has advanced the field by addressing vision-specific challenges, *e.g.*, training methods that improve verification for image classifiers [22, 42], tighter approximation for MaxPool-based CNNs [36, 37].

Independent perturbations are widely supported, but do not capture spatially coupled distortions. The work in [24, 27] considered a limited convolution kernel space, *e.g.*, restricting convolution kernels to values in $[0, 1]$ that sum to one [24] or pixel-level spatial smoothness constraints [27]. Brückner *et al.* [7] analyzes specific types of distortions through parameterizations, enabling continuous transitions from non-perturbed to fully-perturbed patterns.

Recent work has explored alternative verification paradigms for CV applications, *e.g.*, verifying perturbation analysis for explainability [13], verification methods for latent variable model-based specifications [15], addressing complex transformations in vision systems. Our work differs from these prior approaches by providing a unified framework that handles both convolutional and independent perturbations (dual perturbations). VeriDou addresses limitations in existing methods that struggle with complex distortions involving multiple variations.

7. Conclusion

We introduce VeriDou framework to verify robustness through convolutional and independent perturbations, addressing the gap in expressivity of robustness properties. Our evaluation reveals a critical finding that networks demonstrating strong robustness against specific perturbations become significantly more vulnerable under dual perturbations. These findings suggest that verification frameworks should explore combinations of transformation types to avoid overconfident safety conclusions. Future work should investigate training techniques that improve robustness across multiple perturbations.

References

[1] Stanley Bak. nnenum: Verification of ReLU Neural Networks with Optimized Abstraction Refinement. In *NASA Formal Methods Symposium*, pages 19–36. Springer, 2021.

[2] Stanley Bak, Changliu Liu, and Taylor Johnson. The Second International verification of Neural Networks Competition (VNN-COMP 2021): Summary and Results. *arXiv preprint arXiv:2109.00498*, 2021.

[3] Osbert Bastani, Yani Ioannou, Leonidas Lampropoulos, Dimitrios Vytiniotis, Aditya Nori, and Antonio Criminisi. Measuring neural net robustness with constraints. *Advances in neural information processing systems*, 29, 2016.

[4] Elena Botoeva, Panagiotis Kouvaros, Jan Kronqvist, Alessio Lomuscio, and Ruth Misener. Efficient verification of relu-based neural networks via dependency analysis. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 3291–3299, 2020.

[5] Christopher Brix, Mark Niklas Müller, Stanley Bak, Taylor T Johnson, and Changliu Liu. First three years of the international verification of neural networks competition (VNN-COMP). *International Journal on Software Tools for Technology Transfer*, pages 1–11, 2023.

[6] Christopher Brix, Stanley Bak, Taylor T Johnson, and Haoze Wu. The fifth international verification of neural networks competition (vnn-comp 2024): Summary and results. *arXiv preprint arXiv:2412.19985*, 2024.

[7] Benedikt Brückner and Alessio Lomuscio. Verification of neural networks against convolutional perturbations via parameterised kernels. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 27215–27223, 2025.

[8] Nicholas Carlini and David Wagner. Towards evaluating the robustness of neural networks. In *2017 ieee symposium on security and privacy (sp)*, pages 39–57. Ieee, 2017.

[9] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

[10] Hai Duong, ThanhVu Nguyen, and Matthew Dwyer. A DPLL(T) Framework for Verifying Deep Neural Networks. *arXiv preprint arXiv:2307.10266*, 2024.

[11] Hai Duong, Dong Xu, Thanhvu Nguyen, and Matthew B. Dwyer. Harnessing neuron stability to improve dnn verification. *Proc. ACM Softw. Eng.*, 1(FSE), 2024.

[12] Hai Duong, ThanhVu Nguyen, and Matthew B Dwyer. Neuralsat: A high-performance verification tool for deep neural networks. In *International Conference on Computer Aided Verification*, page to appear, 2025.

[13] Thomas Fel, Mélanie Ducoffe, David Vigouroux, Rémi Cadène, Mikael Capelle, Claire Nicodème, and Thomas Serre. Don't lie to me! robust and efficient explainability with verified perturbation analysis. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 16153–16163, 2023.

[14] Claudio Ferrari, Mark Niklas Mueller, Nikola Jovanović, and Martin Vechev. Complete Verification via Multi-Neuron Relaxation Guided Branch-and-Bound. In *International Conference on Learning Representations*, 2022.

[15] Harleen Hanspal and Alessio Lomuscio. Efficient verification of neural networks against lvm-based specifications. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3894–3903, 2023.

[16] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE international conference on computer vision*, pages 1026–1034, 2015.

[17] Dan Hendrycks and Thomas Dietterich. Benchmarking neural network robustness to common corruptions and perturbations. *arXiv preprint arXiv:1903.12261*, 2019.

[18] Xiaowei Huang, Marta Kwiatkowska, Sen Wang, and Min Wu. Safety verification of deep neural networks. In *International conference on computer aided verification*, pages 3–29. Springer, 2017.

[19] Guy Katz, Clark Barrett, David L Dill, Kyle Julian, and Mykel J Kochenderfer. Reluplex: An efficient SMT solver for verifying deep neural networks. In *International Conference on Computer Aided Verification*, pages 97–117. Springer, 2017.

[20] Guy Katz, Clark Barrett, David L Dill, Kyle Julian, and Mykel J Kochenderfer. Towards proving the adversarial robustness of deep neural networks. *Proc. 1st Workshop on Formal Verification of Autonomous Vehicles (FVAV), pp. 19-26*, 2017.

[21] Mykel J Kochenderfer, Jessica E Holland, and James P Chryssanthacopoulos. Next-generation airborne collision avoidance system. Technical report, Massachusetts Institute of Technology-Lincoln Laboratory Lexington United States, 2012.

[22] Zhaoyang Lyu, Minghao Guo, Tong Wu, Guodong Xu, Kehuan Zhang, and Dahua Lin. Towards evaluating and training verifiably robust neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4308–4317, 2021.

[23] Claudio Michaelis, Benjamin Mitzkus, Robert Geirhos, Evgenia Rusak, Oliver Bringmann, Alexander S Ecker, Matthias Bethge, and Wieland Brendel. Benchmarking robustness in object detection: Autonomous driving when winter is coming. *arXiv preprint arXiv:1907.07484*, 2019.

[24] Mallek Mziou-Sallami and Faouzi Adjed. Towards a certification of deep image classifiers against convolutional attacks. In *ICAART (2)*, pages 419–428, 2022.

[25] Nicolas Papernot, Patrick McDaniel, Somesh Jha, Matt Fredrikson, Z Berkay Celik, and Ananthram Swami. The limitations of deep learning in adversarial settings. In *2016 IEEE European symposium on security and privacy (EuroS&P)*, pages 372–387. IEEE, 2016.

[26] PyRAT. A tool to analyze the robustness and safety of neural networks, 2024.

[27] Anian Ruoss, Maximilian Baader, Mislav Balunović, and Martin Vechev. Efficient certification of spatial robustness. In *Proceedings of the AAAI conference on artificial intelligence*, pages 2504–2513, 2021.

[28] Daniel Seita. Bdd100k: A large-scale diverse driving video database. *The Berkeley Artificial Intelligence Research Blog. Version*, 511(41):11, 2018.

[29] 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.

[30] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*, 2013.

[31] Joseph J Titano, Marcus Badgeley, Javin Schefflein, Margaret Pain, Andres Su, Michael Cai, Nathaniel Swinburne, John Zech, Jun Kim, Joshua Bederson, et al. Automated deep-neural-network surveillance of cranial images for acute neurologic events. *Nature medicine*, 24(9):1337–1341, 2018.

[32] Shiqi Wang, Huan Zhang, Kaidi Xu, Xue Lin, Suman Jana, Cho-Jui Hsieh, and J Zico Kolter. Beta-CROWN: Efficient Bound Propagation with Per-neuron Split Constraints for Complete and Incomplete Neural Network Robustness Verification. *Advances in Neural Information Processing Systems*, 34:29909–29921, 2021.

[33] Zhou Wang, Alan C Bovik, Hamid R Sheikh, and Eero P Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE transactions on image processing*, 13(4):600–612, 2004.

[34] Bichen Wu, Forrest Iandola, Peter H Jin, and Kurt Keutzer. Squeezedet: Unified, small, low power fully convolutional neural networks for real-time object detection for autonomous driving. In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, pages 129–137, 2017.

[35] Haoze Wu, Omri Isac, Aleksandar Zeljić, Teruhiro Tagomori, Matthew Daggitt, Wen Kokke, Idan Refaeli, Guy Amir, Kyle Julian, Shahaf Bassan, et al. Marabou 2.0: a versatile formal analyzer of neural networks. In *International Conference on Computer Aided Verification*, pages 249–264. Springer, 2024.

[36] Yuan Xiao, Shiqing Ma, Juan Zhai, Chunrong Fang, Jinyuan Jia, and Zhenyu Chen. Towards general robustness verification of maxpool-based convolutional neural networks via tightening linear approximation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 24766–24775, 2024.

[37] Yuan Xiao, Yuchen Chen, Shiqing Ma, Chunrong Fang, Tongtong Bai, Mingzheng Gu, Yuxin Cheng, Yanwei Chen, and Zhenyu Chen. Tightening robustness verification of maxpool-based neural networks via minimizing the over-approximation zone. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 20695–20705, 2025.

[38] Kaidi Xu, Zhouxing Shi, Huan Zhang, Yihan Wang, Kai-Wei Chang, Minlie Huang, Bhavya Kailkhura, Xue Lin, and Cho-Jui Hsieh. Automatic perturbation analysis for scalable certified robustness and beyond. *Advances in Neural Information Processing Systems*, 33:1129–1141, 2020.

[39] Zhenlin Xu, Deyi Liu, Junlin Yang, Colin Raffel, and Marc Niethammer. Robust and generalizable visual representation learning via random convolutions. *arXiv preprint arXiv:2007.13003*, 2020.

[40] 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. *Proceedings of the 36th International Conference on Neural Information Processing Systems*, 2022.

[41] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 586–595, 2018.

[42] Zhaodi Zhang, Zhiyi Xue, Yang Chen, Si Liu, Yueling Zhang, Jing Liu, and Min Zhang. Boosting verified training for robust image classifications via abstraction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16251–16260, 2023.

[43] Duo Zhou, Christopher Brix, Grani A Hanasusanto, and Huan Zhang. Scalable neural network verification with branch-and-bound inferred cutting planes. *arXiv preprint arXiv:2501.00200*, 2024.