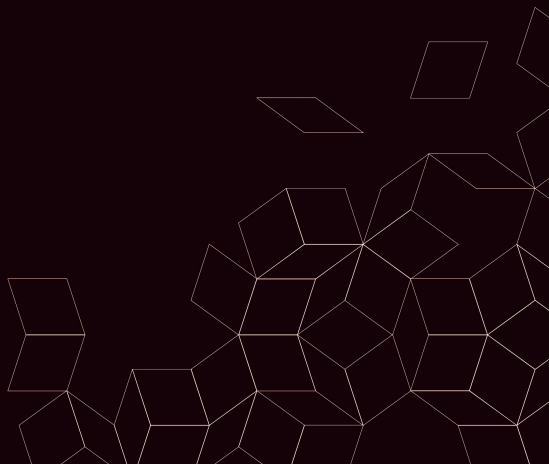
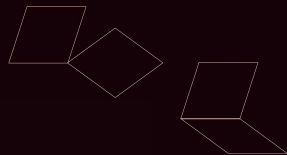


Certifying Differential Invariants of Neural Networks using Abstract Duals

CHANDRA KANTH NAGESH

*Department of Computer Science
University of Colorado Boulder*

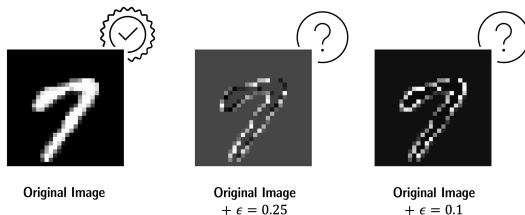
Final Project; CSCI – 7135, Fall 2025



Safety Property

Given a neural network classifier $f : \mathbb{R}^n \rightarrow \mathbb{R}^k$, an input x_0 , and a perturbation radius ϵ . Verify that classification of x_0 is invariant within an ϵ -ball i.e.,

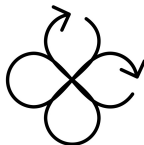
$$\forall x \in \mathbb{B}_\infty(x_0, \epsilon) : \operatorname{argmax}(f(x)) = \operatorname{argmax}(f(x_0))$$



Neural Networks are highly non-linear and vulnerable to *adversarial attacks*.



Original Image + tiny perturbations



Continuous input spaces



Safety-Critical Domains

Possible solution

Try computing an over-approximation of the reachable output set $\mathcal{R}_f(X_0)$ to prove safety

The power of global bounds with simplicity of gradient analysis

1. **Simplicity beats complexity**

- DeepPoly¹ is strictly better than our method
- However, *suprising results found!*

2. **Gradient Instability**

- Found 100% correlation between instability and failure at high ϵ
- Our method detects when the *local linearity assumption collapses*

¹Singh, G. et. al. (2019). An abstract domain for certifying neural networks. 3(POPL), pp.1–30.

Mathematical Bridge: Mean Value Theorem

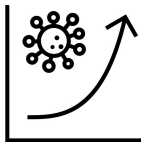
$$|f(x) - f(x_0)| \leq \underbrace{\sup_{z \in X} \|\nabla f(z)\|}_{\text{Abstract Duals}} \cdot \|x - x_0\|$$

Key Difference

- **DeepPoly:** Tracks affine constraints layer-by-layer ($x_j \geq \sum w_i x_i$). Precise but accumulates error ($O(L)$)
- **Ours:** Approximates the derivative $\nabla f(x)$. Coarser, but “jumps” by bypassing intermediate wrapping errors



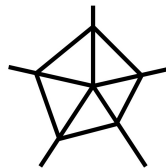
Formalization:
Affine Arithmetic



Gradient Anomaly:
 $\text{Depth} < 3$; Sigmoid



Instability Metric:
Identifies failures



Evaluation:
Tiny/Small/Std. ϵ

Statement

“Global gradient bounds provide a simpler verification domain for shallow, smooth networks, while relational domains are essential for deep, non-smooth architectures”

Lift standard dual numbers $\mathbb{D} = \{a + b\eta\}$ to Affine Forms to compute global gradient bounds.

→ **Value Component:** $\hat{x} = \alpha_0 + \sum_{i=1}^n \alpha_i \epsilon_i$ (Abstract Input Region)

→ Propagate the interval through the network:

$$\hat{y} = f(\hat{x}) = \underbrace{\left(y_0 + \sum y_i \epsilon_i\right)}_{\text{Value Interval}} + \underbrace{\left(d_0 + \sum d_i \epsilon_i\right)}_{\text{Gradient } \nabla f} \eta$$

→ *Global Lipschitz bound is the supremum:*

$$K = \sup_{x \in X_0} \|\nabla f(x)\|_1 = \sum_j \left(|\beta_{0,j}| + \sum_i |\beta_{i,j}| \right)$$

A neuron n is **gradient-unstable** if its gradient interval $[\underline{g}, \overline{g}]$ contains 0

$$\exists x_1, x_2 \in X_0 \text{ s.t. } \text{sign} \left(\frac{\partial f}{\partial n}(x_1) \right) \neq \text{sign} \left(\frac{\partial f}{\partial n}(x_2) \right)$$

Implication

- **Geometric Interpretation:** The function is non-monotonic with respect to neuron n within the input region.
- Linear approximations (DeepPoly) become loose when curvature is high

Counter-Intuitive Finding

In **16.1%** of Tiny Net cases ($\epsilon = 0.01$), the simpler Gradient method certified robustness where DeepPoly failed.

Network	ϵ	Count	%
Tiny Net	0.01	5	16.1%
Tiny Net	0.02	3	9.7%
Tiny Net	0.03	2	6.5%
Small/Std	All	0	0.0%

Table: Gradient method outperforms DeepPoly in shallow, smooth networks; {Tiny: 2×10 , Small: 5×20 , Standard: 10×20 }

1. Gradient Instability Analysis

- **Hypothesis:** Does instability (zero-crossing gradient) predict failure?
- **Result:** 100% Correlation at high perturbation ($\epsilon = 0.12$)

2. ReLU vs. Sigmoid

- **Hypothesis:** Does activation smoothness affect performance?
- **Result:** DeepPoly dominates (100% win rate) on ReLU networks, whereas, smooth derivatives of Sigmoid allow tight global bounds.

Leveraged `ocaml-nn2` library to run on MNIST classifiers with perturbation $\epsilon \in \{0.01 \dots 0.12\}$.

²<https://github.com/ck090/ocaml-nn/tree/main>

- **Shallow + Smooth** → Use **Gradient (Abstract Duals)**
- **Deep + Non-smooth** → Use **Relational (DeepPoly)**

Final Takeaway

“Domain complexity does not dictate robustness, however, alignment with function smoothness does. Expanding to Physics-Informed Neural Networks is an exciting future direction”