SL-SDP: A SCALABLE LAYER-BY-LAYER SDP APPROACH FOR ROBUSTNESS ANALYSIS OF DEEP NEURAL NETWORKS

Anonymous authors
Paper under double-blind review

ABSTRACT

Ensuring the robustness of neural networks against small input perturbations is essential for their safe deployment in safety-sensitive applications. While semidefinite programming (SDP) provides a principled route to certified robustness, monolithic end-to-end formulations scale poorly due to the global coupling of decision variables across all layers. To overcome this computational bottleneck, this paper develops a scalable layer-by-layer SDP (SL-SDP) scheme that decomposes the monolithic verification problem into a sequence of smaller, tractable SDPs linked through propagated quadratic constraints. Crucially, our recursive approach leverages the Schur complement to derive tight, adaptive intermediate bounds with an explicit ellipsoidal interpretation, which in turn yield inexpensive neuron-wise pre-activation intervals and enable more precise abstraction of activation functions. We show how such bounds can be effectively used to tighten the relaxations of ReLU layers. Empirical results demonstrate that SL-SDP significantly improves scalability over end-to-end SDP approaches while preserving accuracy.

1 INTRODUCTION

Deep neural networks remain highly vulnerable to adversarial perturbations despite their remarkable empirical success (Szegedy et al., 2014; Goodfellow et al., 2014; Carlini & Wagner, 2017), as even small, imperceptible changes to the input can lead to drastically different and incorrect outputs. Consequently, providing formal mathematical guarantees on neural network robustness has become a vital area of research. The task of robustness analysis typically involves considering a bounded input set (*e.g.,* an ℓ_∞ ball around a nominal data point) and guaranteeing a safety property over the exact reachable output set. A common formulation involves over-approximating the logits of a classifier network with a half-space to guarantee robustness of the classifier output.

While the exact robustness problem is NP-complete (Katz et al., 2017), SDP has emerged as a powerful alternative by encapsulating nonlinear activations within quadratic constraints (QCs) (Raghunathan et al., 2018; Batten et al., 2021; Fazlyab et al., 2022; Brown et al., 2022), thereby offering significantly tighter bounds than fast but loose linear relaxations. While promising, standard end-to-end SDP formulations couple decision variables across all layers into a single linear matrix inequality (LMI) (Raghunathan et al., 2018; Fazlyab et al., 2022; Brown et al., 2022), or equivalently into a group of smaller LMIs coupled with equality constraints that should be satisfied simultaneously (Batten et al., 2021; Newton & Papachristodoulou, 2021; Xue et al., 2024). In both cases, this global coupling causes the computational complexity to scale poorly with the total number of neurons, severely bottlenecking monolithic SDPs and rendering them prohibitive for deeper architectures.

In this paper, we propose a scalable, layer-by-layer SDP framework that resolves the computational bottlenecks of end-to-end formulations while actively mitigating the accumulation of over-approximation errors (the "wrapping effect"). Inspired by recent advances in recursive bound estimation (Xu & Sivaranjani, 2024), our method solves a sequence of tractable, layer-wise SDPs and leverages the Schur complement to propagate tight, adaptive quadratic constraints.

Contributions. Our main contributions are threefold. First, we introduce a recursive QC propagation scheme that decomposes the monolithic verification problem into sequential layers. By using the

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

Schur complement, we compute adaptive ellipsoidal bounds that capture cross-neuron dependencies and preserve structural information into subsequent layers. Second, our layer-by-layer approach intrinsically provides inexpensive and accurate pre-activation bounds. While most existing methods rely on loose intermediary bounds derived from linear programming (LP)-based techniques to tighten activation relaxations (Raghunathan et al., 2018; Batten et al., 2021), our framework naturally enables tighter relaxations without the need to solve additional problems. Finally, we develop an overlapping two-layer extension of our recursive framework that systematically passes structural cross-layer dependencies forward, actively mitigating information loss and the wrapping effect in deeper networks. Proofs of all theoretical results are provided in the Appendix.

Related work. Due to the existence of nonlinear activation functions, the complete and sound formulation of the robustness problem is NP-complete (Katz et al., 2017). Therefore, exact verifiers formulated as mixed-integer programming (Tjeng et al., 2017; Cheng et al., 2017; Bunel et al., 2018), completely positive programming (Brown et al., 2022), or satisfiability modulo theories (SMT) (Katz et al., 2017; 2019) remain computationally intractable except for very small networks. To achieve massive scalability, some incomplete verification methods relax nonlinear activations with linear constraints, thereby enabling bound propagation frameworks that reduce the problem to LP (Wong & Kolter, 2018; Zhang et al., 2018; Wang et al., 2018). While computationally efficient, these linear relaxations accumulate significant over-approximation errors toward the subsequent layers of the network. To tighten these bounds, alternative methods apply multi-neuron relaxations to propagate geometric domains like zonotopes (Singh et al., 2018) or polyhedra (Singh et al., 2019), while advanced frameworks integrate these relaxations into branch-and-bound (BaB) schemes with per-neuron split constraints (Xu et al., 2021; Wang et al., 2021).

SDP-based methods naturally bridge the gap between tightness and scalability by relaxing nonlinear activations into QCs (Fazlyab et al., 2022; Raghunathan et al., 2018; Brown et al., 2022). Beyond local robustness, SDP formulations have also proven highly effective for computing global network properties, such as the network's Lipschitz constant (Fazlyab et al., 2019; Latorre et al., 2020; Pauli et al., 2021; Xu & Sivaranjani, 2024). To address the global coupling bottleneck of standard end-to-end SDPs, recent advances have explored non-convex low-rank relaxations (Chiu & Zhang, 2023), problem decomposition using chordal graph techniques (Batten et al., 2021; Xue et al., 2024; Newton & Papachristodoulou, 2021), and hybrid LP-SDP frameworks (Chiu et al., 2025). While these methods improve scalability, balancing computational complexity against the rapid degradation of intermediate bounds in sequential approximations remains a key challenge, which our recursive framework directly addresses.

Notation. Let $\mathbb{R}^n$ and $\mathbb{R}^n_+$ denote the n-dimensional Euclidean space and its non-negative orthant, respectively. The set $\mathbb{S}^n$ denotes the space of $n \times n$ *symmetric* matrices. For $M \in \mathbb{S}^n$, $M \succeq 0$ and $M \succ 0$ signify that M is positive semidefinite and positive definite, respectively. For a vector $v \in \mathbb{R}^n$, $\mathrm{diag}(v)$ denotes the diagonal matrix with v on its diagonal and for a matrix $A \in \mathbb{R}^{n\times n}$, $\mathrm{diag}(A)$ and $\mathrm{tr}(A)$ represent the vector of the diagonal elements of A and its trace, respectively. For vectors $v_1 \in \mathbb{R}^{n_1}, \ldots, v_k \in \mathbb{R}^{n_k}$, $[v_1; \ldots; v_k]$ denotes vertical concatenation. For two vectors u, v of the same length, $u \geq v$, $\min(u, v)$, and $\max(u, v)$ denote elementwise operations. The notation $\mathbf{I}_n$ represents the $n \times n$ identity matrix, $\mathbf{0}_{n\times m}$ the $n \times m$ zero matrix, and $\mathbf{0}$ a zero matrix of appropriate size. In symmetric matrices, $\star$ represents the corresponding symmetric element. For a matrix $B \in \mathbb{R}^{n\times m}$, $B^\dagger$ denotes its Moore–Penrose pseudo-inverse. An ellipsoid $\mathcal{E}(c, E)$ with center $c \in \mathbb{R}^n$ and shape matrix $E \succeq 0$ is defined as $\mathcal{E}(c, E) := \{x \in \mathbb{R}^n \mid (x - c)^\top E(x - c) \leq 1\}$. This set defines a strictly bounded, non-degenerate ellipsoid if and only if $E \succ 0$.

2 Problem formulation and background

Consider a feedforward neural network with L hidden layers as $f : \mathbb{R}^{n_0} \to \mathbb{R}^{n_y}$. The recursive transformation in the network is given by

$$\begin{aligned} z_i &= W_i x_{i-1} + b_i, \quad x_i = \phi_i(z_i), \quad i = 1, \ldots, L, \\ f(x_0) &= y = W_{L+1} x_L + b_{L+1}, \end{aligned} \tag{1}$$

where $x_0 \in \mathbb{R}^{n_0}$ is the network input, the matrix $W_i \in \mathbb{R}^{n_i \times n_{i-1}}$ and the vector $b_i \in \mathbb{R}^{n_i}$ are the weight and bias of layer i, z_i is the pre-activation for layer i, $x_i \in \mathbb{R}^{n_i}$ is the output of layer i, and $\phi_i : \mathbb{R}^{n_i} \to \mathbb{R}^{n_i}$ is the (typically element-wise) activation function.

Given a set of possible inputs $\mathcal{X}_{in}$, the exact reachable output set (or image) of the network is defined as $\mathcal{Y} = \{f(x_0) \mid x_0 \in \mathcal{X}_{in}\}$. The central problem in robustness analysis is to compute a tight, yet tractable, overapproximation $\hat{\mathcal{Y}}$ such that the inclusion $\mathcal{Y} \subseteq \hat{\mathcal{Y}}$ is guaranteed.

2.1 Representations via QCs

To leverage semidefinite programming, the non-convex verification problem should be relaxed into a tractable convex form. This is achieved by expressing the network's input set, activation functions, and objective function as a unified set of QCs, enabling their direct integration into LMIs.

We begin by defining the permissible input perturbations. To ensure compatibility with our later SDP formulation via the S-procedure (Yakubovich, 1977), this initial domain should be expressed as a quadratic inequality.

Input set. The input set is assumed to be defined by m_0 quadratic inequalities as

$$\mathcal{X}_{in} = \left\{ x_0 \in \mathbb{R}^{n_0} \;\middle|\; [1; x_0]^\top A_j [1; x_0] \geq 0, \quad j \in \{1, \dots, m_0\} \right\},$$

where $A_j \in \mathbb{S}^{n_0+1}$ are symmetric matrices. By taking a conic combination of these matrices with non-negative multipliers ω_{0j}, we construct a parameterized family of valid QCs for any $x_0 \in \mathcal{X}_{in}$ as

$$\mathbf{P}_{in}(x_0; \omega_0) \coloneqq [1; x_0]^\top P_{in}(\omega_0) [1; x_0] \geq 0, \quad P_{in}(\omega_0) = \sum_{j=1}^{m_0} \omega_{0j} A_j, \tag{2}$$

where the multiplier vector satisfies $\omega_0 \in \Omega_0 \coloneqq \mathbb{R}_+^{m_0}$. Explicit derivations of P_{in} for various geometric shapes are provided in Appendix B.1.

Nonlinear activation functions introduce the non-convexity that makes exact verification intractable. To circumvent this, we replace the exact nonlinear mapping with a geometric overapproximation, rigorously enclosing the activation's input-output behavior within a set of QCs.

Activation functions. A key component of this framework is to abstract the nonlinear activation functions using QCs. We assume that the input-output behavior of each activation $\phi_i : \mathbb{R}^{n_i} \to \mathbb{R}^{n_i}$ (for $i = 1, \dots, L$) is characterized by m_i quadratic inequalities and m_i^e quadratic equalities, represented by symmetric matrices M_{ij} and M_{ij}^e, respectively. By taking a conic combination of the inequality matrices and a linear combination of the equality matrices, we construct a parameterized family of valid QCs as

$$[1; z_i; \phi_i(z_i)]^\top \Psi_i(\omega_i) [1; z_i; \phi_i(z_i)] \geq 0, \quad \Psi_i(\omega_i) = \sum_{j=1}^{m_i} \alpha_{ij} M_{ij} + \sum_{j=1}^{m_i^e} \beta_{ij} M_{ij}^e, \tag{3}$$

where the combined multiplier vector $\omega_i = [\alpha_i; \beta_i]$ belongs to the set $\Omega_i \coloneqq \mathbb{R}_+^{m_i} \times \mathbb{R}^{m_i^e}$. For many common activation functions, these base matrices M_{ij} and M_{ij}^e can be derived from the known properties of the activation functions. Specifically, in Appendix B.4, the properties of ReLU activation functions are leveraged to derive their specific QC family.

Subsequently, to express the activation function QC (3) in terms of the primary layer variables (x_{i-1} and x_i), we use the affine network dynamics (1), which yield $[1; z_i; \phi_i(z_i)] = T_i [1; x_{i-1}; x_i]$. By substituting this into (3), the constraint can be expressed as a new QC relating the outputs of two consecutive layers, x_{i-1} and x_i, as

$$\mathbf{P}_i(x_{i-1}, x_i; \omega_i) \coloneqq [1; x_{i-1}; x_i]^\top P_i(\omega_i) [1; x_{i-1}; x_i] \geq 0, \quad \forall \omega_i \in \Omega_i, \tag{4}$$

where $P_i(\omega_i) = T_i^\top \Psi_i(\omega_i) T_i$. To evaluate the robustness guarantee within the same unified optimization framework, the target safety property should also be structured as a quadratic constraint.

Objective function. To certify robustness, we seek to bound a quadratic objective over the reachable output set, parameterized by a matrix Ψ_y. We formulate this as finding the minimum scalar $\rho \in \mathbb{R}$ that guarantees

$$\rho - [1; y]^\top \Psi_y [1; y] \geq 0, \quad \forall x_0 \in \mathcal{X}_{in}.$$

Any valid scalar ρ provides a guaranteed upper bound on the objective, effectively defining an over-approximation of the reachable output set. Minimizing ρ tightens this bound, yielding a more precise over-approximation. This quadratic form accommodates common safety specifications, including linear bounds and ℓ_2-norm constraints. To express this condition as a quadratic constraint strictly in terms of the final hidden layer x_L, we rewrite the affine mapping of the final layer as $[1; y] = T_L[1; x_L]$, which yields the following output constraint:

$$\mathbf{H}_L(x_L, \rho) \coloneqq [1; x_L]^\top H_L(\rho) [1; x_L] \geq 0, \quad \text{where} \quad H_L(\rho) \coloneqq \begin{bmatrix} \rho & \mathbf{0} \\ \star & \mathbf{0} \end{bmatrix} - T_L^\top \Psi_y T_L. \tag{5}$$

With the quadratic constraints established for the input, activations, and output, these components should now be integrated to certify the network's overall behavior. The following subsection details how this end-to-end formulation is constructed in the literature.

2.2 End-to-end formulation via DeepSDP (Fazlyab et al., 2022)

The goal is to find the smallest ρ such that $\mathbf{H}_L(x_L; \rho) \geq 0$ is satisfied for all reachable x_L. Considering the extended neuron output vector $\mathbf{x} = [x_0; \dots; x_L] \in \mathbb{R}^N$, with $N = \sum_{i=0}^{L} n_i$, the QCs for the input set (2), activation functions (3), and the output set (5) are all quadratic functions with respect to the augmented vector $[1; \mathbf{x}]$. Hence, by the S-procedure (Yakubovich, 1977), it is sufficient to find ρ and ω_i such that

$$\mathbf{H}_L(x_L, \rho) - \sum_{i=1}^{L} \mathbf{P}_i(x_{i-1}, x_i, \omega_i) - \mathbf{P}_{in}(x_0, \omega_0) \geq 0, \quad \forall \mathbf{x} \in \mathbb{R}^N.$$

Note that since $\mathbf{P}_{in}$ and $\mathbf{P}_i$ are linearly parameterized, the nonnegative multipliers for these terms in the S-procedure can be absorbed into the ω variables. To cast this problem as an SDP, it is only necessary to expand the matrices with zero rows and columns to match the dimension of $[1; \mathbf{x}]$. Denoting this expansion with a bar notation (*e.g.,* $\bar{P}$; see Appendix B.2 for the explicit selection matrix transformations), the problem can be represented as

$$\begin{aligned} &\min_{\rho, \omega} \rho \\ &\text{s.t.} \quad \mathbf{M}(\rho, \omega) = \bar{H}_L(\rho) - \sum_{i=1}^{L} \bar{P}_i(\omega_i) - \bar{P}_{in}(\omega_0) \succeq 0, \end{aligned} \tag{6}$$

where $\rho \in \mathbb{R}$ and $\omega \in \Omega$ is the concatenation of all decision variables ω_i. The LMI constraint $\mathbf{M}(\rho, \omega) \succeq 0$ is sufficient for robustness: if a feasible pair (ρ, ω) is found, it ensures that any $\mathbf{x}$ satisfying the input and activation QCs also satisfies the output QC. While promising, the primary drawback of this monolithic approach lies in its computational scaling, as the matrix $\mathbf{M}$ has size $(1+N) \times (1+N)$ with $N = \sum_{i=0}^{L} n_i$, rendering the end-to-end SDP (6) computationally prohibitive for deeper architectures.

3 Proposed method: Scalable layer-by-layer QC propagation

The monolithic SDP formulation in (6) yields a tight over-approximation of the output set but couples the decision variables across all layers into a single LMI of size $(1 + N) \times (1 + N)$. While recent approaches seek to exploit the sparsity patterns in this LMI to solve the problem more efficiently (Batten et al., 2021; Xue et al., 2024), this global coupling remains the primary computational bottleneck. Conversely, a *naive* layer-by-layer approach, which over-approximates the output set of layer i and treats it directly as the input to layer $i + 1$, introduces significant conservatism due to the wrapping effect.

To mitigate this, we propose a recursive layer-by-layer QC propagation scheme. While any sequential decomposition inevitably incurs some wrapping error, our approach dampens its severity by aggressively tightening the derived bounds. To achieve this, the method alternates between a *containment* phase, where we certify an over-approximation for x_i, and a *tightening* phase, where we compute an adaptive QC G_i via the Schur complement. Consequently, G_i tightly bounds the layer output x_i and is passed forward, preserving structural cross-neuron dependencies that would otherwise be lost.

Beyond alleviating computational complexity, this scheme also provides tight lower and upper bounds for the pre-activations of intermediate layers, which in turn enable a more accurate abstraction of the nonlinear activation functions, as detailed for ReLU layers in Appendix B.4.

3.1 Fast layer-by-layer QC propagation

Let us consider a layer-wise decomposition of the problem. Assuming the input to the current layer i is constrained by a family of matrices $Q_{i-1}(\theta_{i-1})$, parametrized over a set of vectors Θ_{i-1}, such that

$$[1; x_{i-1}]^\top Q_{i-1}(\theta_{i-1}) [1; x_{i-1}] \geq 0, \quad \forall \theta_{i-1} \in \Theta_{i-1},$$

and incorporating the activation QCs from (4), the goal is to find the smallest ρ_i satisfying

$$[1; x_i]^\top H_i(\rho_i) [1; x_i] \geq 0.$$

Applying the S-procedure to this *single layer*, similar to the end-to-end setup presented in Subsection 2.2, yields the following optimization problem

$$\begin{aligned} &\min_{\rho_i, \omega_i, \theta_{i-1}} \rho_i \\ &\text{s.t.} \quad \mathbf{M}_i(\rho_i, \omega_i, \theta_{i-1}) = \tilde{H}_i(\rho_i) - P_i(\omega_i) - \hat{Q}_{i-1}(\theta_{i-1}) \succeq 0, \end{aligned} \tag{7}$$

with decision variables $\rho_i \in \mathbb{R}$, $\omega_i \in \Omega_i$, and $\theta_{i-1} \in \Theta_{i-1}$. Furthermore, the "tilde" and "hat" accents indicate that the matrices are zero-padded to match the dimension of the joint quadratic basis $[1; x_{i-1}; x_i]$ (see Appendix B.3 for exact transformation details). For the first layer ($i = 1$), the input constraint family is directly given as $Q_0(\theta_0) = P_{in}(\omega_0)$ with the parameter mapping $\theta_0 = \omega_0$.

The critical design choice at this stage is how the subsequent constraint Q_i for the next layer is generated. A naive approach simply leverages the optimal value $\rho_i^\star$ and fixes $Q_i(\theta_i) = \theta_i H_i(\rho_i^\star)$ with $\theta_i \in \mathbb{R}_+$. While this yields a valid constraint, the fixed structural form of H_i leads to a severe wrapping effect. In the following theorem, we present a method that alleviates this accumulation of error by adaptively containing the feasible set of x_i.

Theorem 1 (Recursive robustness analysis)**.** *Consider the layer-wise optimization problem in* (7) *with the optimal values $\rho_i^\star, \omega_i^\star, \theta_{i-1}^\star$. Let the aggregated source constraints for layer i be rewritten by permuting variables as*

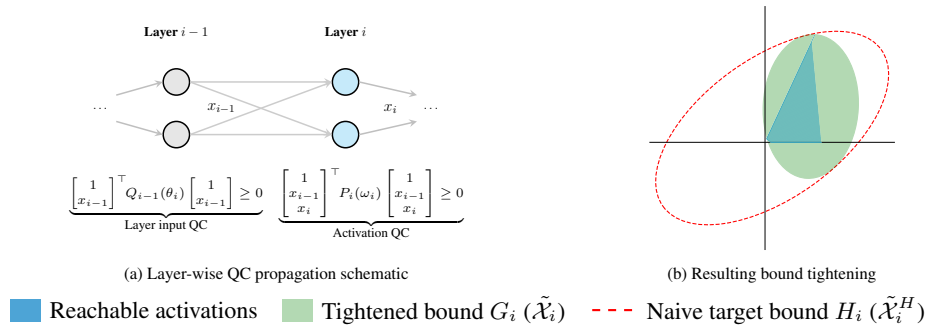
$$\begin{bmatrix} 1 \\ x_{i-1} \\ x_i \end{bmatrix}^\top P_i(\omega_i^\star) \begin{bmatrix} 1 \\ x_{i-1} \\ x_i \end{bmatrix} + \begin{bmatrix} 1 \\ x_{i-1} \end{bmatrix}^\top Q_{i-1}(\theta_{i-1}^\star) \begin{bmatrix} 1 \\ x_{i-1} \end{bmatrix} = \left[\begin{array}{c} 1 \\ \hline x_i \\ x_{i-1} \end{array}\right]^\top \left[\begin{array}{c|c} \mathbf{R}_i & \mathbf{U}_i^\top \\ \hline \mathbf{U}_i & \mathbf{V}_i \end{array}\right] \left[\begin{array}{c} 1 \\ \hline x_i \\ x_{i-1} \end{array}\right]. \tag{8}$$

Then, the output of layer i satisfies

$$[1; x_i]^\top G_i [1; x_i] \geq 0, \quad \textit{where} \quad G_i = \mathbf{R}_i - \mathbf{U}_i^\top \mathbf{V}_i^\dagger \mathbf{U}_i. \tag{9}$$

Proof sketch. The LMI $\mathbf{M}_i \succeq 0$ ensures that its lower-right block, corresponding to the preceding-layer variable x_{i-1}, is negative semidefinite. By applying the generalized Schur complement, we explicitly eliminate the variables associated with layer $i - 1$. The S-procedure then guarantees that the resulting quadratic form G_i, which acts solely on x_i, remains positive semidefinite over the feasible set, thereby propagating the constraint to the next layer. The complete proof is provided in Appendix A. □

Having established the validity of the propagated base matrix G_i, one can construct the parameterized input constraint for the subsequent layer by defining $Q_i(\theta_i) = \theta_i G_i$, with scalar multiplier $\theta_i \in \mathbb{R}_+$. We now mathematically formalize the advantage of deriving this adaptive base matrix over the naive sequential approach. To effectively mitigate the wrapping effect, the adaptive QC with G_i should be more restrictive than simply passing forward the constraint specified by $H_i(\rho_i^\star)$. The following proposition establishes this tightening effect by showing that the feasible region defined by G_i is always a subset of the region defined by $H_i(\rho_i^\star)$.



(a) Layer-wise QC propagation schematic

(b) Resulting bound tightening

Figure 1: Visualization of recursive QC propagation. **(a)** Schematic of the i-th layer, illustrating how x_{i-1} propagates to x_i, subject to the constraints defined by Q_{i-1} and P_i. **(b)** The resulting tightening effect (Proposition 1) in the output space of Layer i. While the layer-wise optimization targets a loose bounding region defined by H_i, deriving G_i via the Schur complement yields a tighter region.

Proposition 1 (Tightening effect)**.** *If the condition in* (7) *holds and G_i is defined as* (9)*, then the sets $\tilde{\mathcal{X}}_i$ and $\tilde{\mathcal{X}}_i^H$, defined as*

$$\tilde{\mathcal{X}}_i = \left\{ x_i \in \mathbb{R}^{n_i} \mid [1; x_i]^\top G_i [1; x_i] \geq 0 \right\}, \quad \tilde{\mathcal{X}}_i^H = \left\{ x_i \in \mathbb{R}^{n_i} \mid [1; x_i]^\top H_i(\rho_i^\star) [1; x_i] \geq 0 \right\} \tag{10}$$

satisfy $\tilde{\mathcal{X}}_i \subseteq \tilde{\mathcal{X}}_i^H$.

Proof sketch. Taking the Schur complement of $\mathbf{M}_i \succeq 0$ yields $H_i(\rho_i^\star) - G_i \succeq 0$. By the S-procedure, this guarantees the region defined by G_i is strictly enclosed within the target bounding surface of H_i. The complete proof is provided in Appendix A. □

Proposition 1 formally establishes that the adaptive QC defined by G_i is more restrictive than the fixed-form target QC (see Figure 1). Therefore, passing the propagated constraint G_i forward mitigates the wrapping effect by providing a tighter bounded region for the subsequent layer optimization.

While the matrix G_i provides a rigorous QC, its raw structure is not immediately intuitive for geometric analysis. To facilitate the derivation of neuron-wise intervals, we first show that G_i defines a bounded, convex region with a clear ellipsoidal interpretation; the following proposition establishes the explicit mapping between the matrix entries and the geometric parameters of the ellipsoid.

Proposition 2 (Geometric interpretation of propagated constraints)**.** *Suppose the set $\tilde{\mathcal{X}}_i^H$ in* (10) *is a non-degenerate ellipsoid and $\tilde{\mathcal{X}}_i$ has a non-empty interior. Then $\tilde{\mathcal{X}}_i$ defines a non-degenerate ellipsoid $\mathcal{E}(\bar{x}_i, E_i)$ with center $\bar{x}_i$ and shape matrix E_i, given by*

$$\bar{x}_i = -(G_i^{xx})^{-1}(G_i^{1x})^\top, \quad E_i = \frac{1}{\kappa_i}(-G_i^{xx}), \quad \textit{where} \quad G_i = \begin{bmatrix} G_i^{11} & G_i^{1x} \\ \star & G_i^{xx} \end{bmatrix},$$

with the scaling factor $\kappa_i = G_i^{11} - G_i^{1x}(G_i^{xx})^{-1}(G_i^{1x})^\top$.

Proof sketch. Since $G_i \preceq H_i$ and the target bound is a non-degenerate ellipsoid ($H_i^{xx} \prec 0$), G_i^{xx} inherits strict negative definiteness. Completing the square around G_i^{xx} geometrically transforms the constraint into a bounded ellipsoid, explicitly revealing its center and shape matrix. The complete proof is provided in Appendix A. □

While the ellipsoidal interpretation captures valuable cross-neuron dependencies, the activation function relaxations, developed in the following sections, require independent lower and upper bounds for each individual pre-activation. By projecting the ellipsoidal set onto the coordinate axes, these bounds can be derived analytically.

Proposition 3 (Explicit bounds from propagated QCs)**.** *Suppose x_i is bounded by the ellipsoid $\mathcal{E}(\bar{x}_i, E_i)$. Then, the pre-activations of the next layer, $z_{i+1} = W_{i+1}x_i + b_{i+1}$, are bounded element-wise by the interval*

$$\hat{v} - \hat{a} \leq z_{i+1} \leq \hat{v} + \hat{a}, \quad \textit{where} \quad \hat{v} = W_{i+1}\bar{x}_i + b_{i+1} \quad \textit{and} \quad \hat{a} = \sqrt{\operatorname{diag}\left(W_{i+1}E_i^{-1}W_{i+1}^\top\right)},$$

with the square root applied element-wise.

Proof sketch. Mapping the parameterized ellipsoid through the affine layer dynamics and applying the Cauchy-Schwarz inequality to the weight and shape matrices analytically yields the maximum possible deviation for each pre-activation. The complete proof is provided in Appendix A. □

With the general tightening method established, a remaining crucial step in this recursive framework is to select a suitable form for the target QC H_i during layer-wise optimization. Given the linear relationship between x_i and the subsequent pre-activations z_{i+1}, we can constrain z_{i+1} via a QC on x_i. A reasonable choice is to confine z_{i+1} within the smallest ℓ_2 norm ball centered at the propagated nominal point. Specifically, let v_{in} be the center of the input set $\mathcal{X}_{in}$. We propagate this center through the network as

$$v_0^x = v_{in}, \quad v_i^z \coloneqq W_i v_{i-1}^x + b_i, \quad v_i^x \coloneqq \phi_i(v_i^z), \quad i = 1, \ldots, L.$$

For each layer $i \in \{1, \ldots, L-1\}$, a hypersphere centered at v_{i+1}^z with squared radius $\rho_i > 0$ is considered. This objective is equivalent to enforcing a QC on x_i, given by

$$[1; x_i]^\top H_i(\rho_i) [1; x_i] \geq 0, \quad \text{where} \quad H_i(\rho_i) = \begin{bmatrix} \rho_i - (v_i^x)^\top \mathcal{W}_{i+1} v_i^x & (v_i^x)^\top \mathcal{W}_{i+1} \\ \star & -\mathcal{W}_{i+1} \end{bmatrix}, \tag{11}$$

with $\mathcal{W}_{i+1} = W_{i+1}^\top W_{i+1}$. To guarantee that the target bounding region forms a non-degenerate ellipsoid (satisfying the condition in Proposition 2), we apply a standard regularization by setting $\mathcal{W}_{i+1} = W_{i+1}^\top W_{i+1} + \epsilon \mathbf{I}_{n_i}$, where $\epsilon > 0$ is an arbitrarily small constant. Having established the target constraint H_i, we now detail the construction of the activation QCs for ReLU networks. To abstract the ReLU activation functions into QCs, we leverage their elementwise bounds and piecewise linearity, following standard practices (Batten et al., 2021; Fazlyab et al., 2022). The detailed derivation of these QCs is provided in Appendix B.4.

3.2 Overall verification algorithm

Algorithm 1 (Appendix B.5) summarizes our verification procedure, which initializes with inexpensive baseline bounds from Gowal et al. (2018); Zhang et al. (2018) and constructs refined ReLU QCs via Lemma 1. To define the layer input constraint $Q_{i-1}(\theta_{i-1})$, the first layer exactly maps the initial perturbation set as $Q_0(\theta_0) = P_{in}(\omega_0)$. For subsequent layers ($i \geq 2$), we define $Q_{i-1}(\theta_{i-1}) = c_{i-1}G_{i-1} + B_{i-1}(\nu_{i-1})$, where the parameter vector $\theta_{i-1} = [c_{i-1}; \nu_{i-1}]$ concatenates a scalar multiplier $c_{i-1} \in \mathbb{R}_+$ and non-negative multipliers $\nu_{i-1} \in \mathbb{R}_+^{n_{i-1}}$. Here, B_{i-1} confines the input to a hyper-rectangle derived from intermediate bounds, further tightening the bounding region established by G_{i-1}. Minimizing the objective ρ_i under $\mathbf{M}_i \succeq 0$ yields G_i, analytically providing tighter bounds for z_{i+1} (Proposition 3). Intersecting these with baseline intervals explicitly mitigates the wrapping effect, repeating this sequence until the final layer maps to the verification objective.

3.3 Balancing accuracy and scalability

While the single-layer propagation scheme detailed above is highly efficient, compressing the feasible set into a marginal QC with G_i at each step inevitably discards some cross-layer dependencies. To mitigate this information loss and improve the tightness of the final bounds, the recursive framework can be naturally extended to optimize over an overlapping window of multiple layers. In the *two-layer* variant, the layer-wise LMI is expanded to simultaneously encompass layers $i-1$ and i, conditioned on a previously propagated joint constraint bounding the augmented state $[x_{i-2}; x_{i-1}]$. By aggregating the source constraints and eliminating the oldest layer state (x_{i-2}) via the Schur complement, we generate a new joint constraint $G_{i,i-1}$ that tightly bounds $[x_{i-1}; x_i]$. This rolling overlap systematically passes structural relational information forward, enabling significantly tighter sequential pre-activation bounds at the cost of a moderately larger LMI at each step. The rigorous mathematical formulation and the corresponding algorithmic implementation (Algorithm 2) are provided in Appendix B.6.

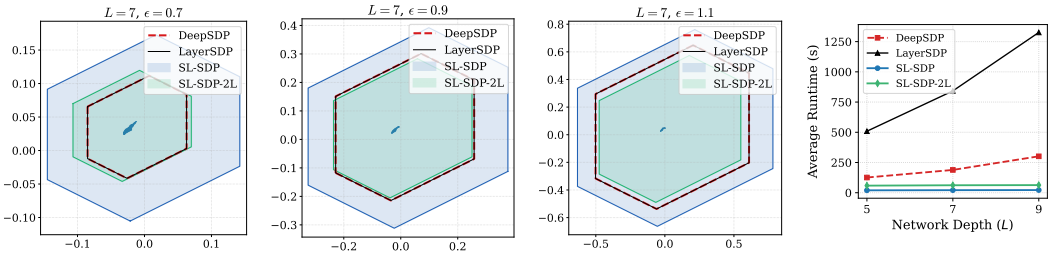


Figure 2: 2D output reachable set bounds for a network of depth $L = 7$ under varying perturbation radii $r \in \{0.7, 0.9, 1.1\}$. The blue point cloud represents the exact reachable set estimated via Monte Carlo sampling. The rightmost plot compares average runtimes of the evaluated methods across varying network depths.

4 Experiments

We evaluate our proposed scalable layer-by-layer SDP framework against state-of-the-art verification methods. Our evaluation focuses on two key aspects: (i) a qualitative assessment of the tightness of polytopic over-approximations for networks of varying depths and input perturbations, and (ii) a quantitative evaluation of certified robustness and computational scalability across different network architectures. Ultimately, we demonstrate that our layer-by-layer approach effectively bridges the gap between fast linear relaxations and monolithic SDPs, providing competitive tightness while scaling to deeper architectures where end-to-end methods fail.

Experimental setup. All experiments were executed on a machine equipped with an 8-core Intel Xeon CPU and 64 GB of RAM. Additionally, CROWN-family methods were accelerated using an NVIDIA L4 GPU with 24 GB of memory. We denote our single-layer approach as SL-SDP, and the overlapping *two-layer* approach as SL-SDP-2L. We benchmark our methods against DeepSDP (Fazlyab et al., 2022), LayerSDP (Batten et al., 2021), α-CROWN (Xu et al., 2021), and β-CROWN (Wang et al., 2021). All methods are implemented in Python, and the SDP-based methods utilize CVXPY (Diamond & Boyd, 2016) with the MOSEK solver (ApS, 2025). To establish baseline intermediate bounds, we utilized α-CROWN via the `auto_LiRPA` (Xu et al., 2020) library. All evaluated SDP-based methods leverage these intermediate bounds to tighten the relaxations for the ReLU activation functions.

Reachable set. To qualitatively evaluate bound tightness, we visualize 2D output reachable sets for networks with a 2D input space, a constant width of 64 neurons, and varying depths $L \in \{5, 7, 9\}$. Weights and biases are drawn from $\mathcal{N}(0, 1/8)$ and $\mathcal{N}(0, 1/16)$, respectively, with the input perturbation set defined as an ℓ_∞ norm ball of radius $r \in \{0.7, 0.9, 1.1\}$, centered at the all-ones vector.

Figure 2 illustrates the output bounds for $L = 7$ across different input perturbation radii. As the perturbation radius increases, the relative tightness of our method visibly improves. This advantage largely stems from our framework computing more accurate intermediate bounds than the standard α-CROWN baseline. When the input perturbation becomes large, the baseline intermediate bounds rapidly loosen, enabling our layer-by-layer methods to construct tighter relaxations. Furthermore, the right-hand side of Figure 2 demonstrates the computational efficiency of our methods. Beyond the inherent efficiency of the layer-wise decomposition, computing the six faces of the output hexagon requires optimizing only the final layer six times, while the SDPs corresponding to all preceding layers are solved only once. In contrast, end-to-end methods must re-solve the entire monolithic LMI for each face. More importantly, our approach scales substantially more efficiently with network depth. Similar reachable-set visualizations for $L = 5$ and $L = 9$ are provided in Appendix B.7. We also observe that DeepSDP and LayerSDP produce *identical* geometric bounds. Since LayerSDP incurs substantially higher computational costs as network size increases, we exclude it from the subsequent large-scale experiments.

Robustness Verification on Trained Networks. To evaluate our proposed methods on standard image classification tasks, we conduct robustness verification on the MNIST dataset. We utilize two multi-layer perceptron (MLP) architectures, a 4-layer network (`MLP 4×100`) and a 9-layer network (`MLP 9×150`), evaluated on a subset of 25 randomly selected, correctly classified test samples. We

analyze network robustness against three distinct input perturbations, incorporating both ℓ_∞ and ℓ_2 norm bounds. We benchmark our recursive formulations, SL-SDP and SL-SDP-2L, against the linear relaxation framework α-CROWN, its branch-and-bound (BaB) extension β-CROWN (Wang et al., 2021), and the monolithic DeepSDP (Fazlyab et al., 2022). Since all SDP-based methods use α-CROWN as an initial baseline, similarly to β-CROWN, only the logit margins that cannot be verified by α-CROWN are subsequently evaluated using the SDP-based methods.

Table 1 summarizes the number of verified samples and the average computational runtime. The empirical results reveal a clear shift in performance as the perturbation size increases. For small perturbation radii, CROWN-based methods achieve strong verification performance. However, as the input perturbation radius increases, the LP-based methods fail to verify any samples, and even the BaB refinements of β-CROWN cannot produce certificates within the prescribed time limit.

While DeepSDP provides tighter convex relaxations capable of handling these larger perturbations, it incurs severe computational bottlenecks and struggles to scale to the 9-layer architecture. By systematically propagating adaptive quadratic constraints, our layer-by-layer methods effectively bridge this gap, avoiding both the unpredictable runtimes of spatial branching and the dimensional explosion of monolithic SDPs.

Table 1: Verification results on the MNIST dataset. The 'Ver.' column indicates the number of samples formally certified as safe, while 'Time' denotes the average computational runtime per sample in seconds (the average runtime exclusively for verified samples is provided in parentheses, reported only when noticeably different from the overall average). The PGD row provides an empirical upper bound on the number of verifiable samples by counting instances where no adversarial example was found via a PGD attack. The timeout limits per sample are 1 hour and 1.5 hours for the 4-layer and 9-layer networks, respectively. 'TO' indicates that DeepSDP timed out without verifying any additional samples beyond those already verified by the α-CROWN baseline.

MLP 4×100						
Method	ℓ_2 ($\epsilon = 1.0$)		ℓ_2 ($\epsilon = 0.6$)		ℓ_∞ ($\epsilon = 0.05$)	
	Ver.	Time	Ver.	Time	Ver.	Time
PGD (Upper Bound)	25	–	25	–	24	–
α-CROWN	0	1.09	5	1.18	12	1.27
β-CROWN	0	3600	20	793.13 (88.39)	23	195.60 (53.37)
DeepSDP	20	2151.63 (2341.20)	24	1334.57 (1367.30)	21	625.65 (601.64)
SL-SDP (Ours)	6	347.83 (390.95)	21	295.79 (289.61)	13	161.72 (29.30)
SL-SDP-2L (Ours)	11	384.23 (514.39)	22	395.66 (387.48)	17	201.46 (146.27)
MLP 9×150						
Method	ℓ_2 ($\epsilon = 0.5$)		ℓ_2 ($\epsilon = 0.3$)		ℓ_∞ ($\epsilon = 0.04$)	
	Ver.	Time	Ver.	Time	Ver.	Time
PGD (Upper Bound)	25	–	25	–	24	–
α-CROWN	0	4.46	20	3.18	5	4.58
β-CROWN	0	4800	21	769.40 (3.67)	8	3077.54 (16.80)
DeepSDP	–	TO	–	TO	–	TO
SL-SDP (Ours)	6	734.72 (830.38)	22	161.52 (81.27)	5	541.48 (4.17)
SL-SDP-2L (Ours)	7	1260.50 (1527.34)	22	259.21 (135.46)	8	916.91 (567.31)

5 Conclusion and future work

We introduced a scalable, layer-by-layer SDP framework for certifying deep neural network robustness, replacing computationally prohibitive monolithic LMIs with tractable sequential SDPs. At its core, our method recursively propagates adaptive quadratic constraints via the Schur complement. This produces tight ellipsoidal bounds that preserve cross-neuron dependencies and yield precise, inexpensive pre-activation intervals for better activation abstraction. Empirically, this approach significantly improves scalability for deeper architectures while maintaining competitive tightness. Future work may extend this framework to complex architectures, such as residual, recurrent, and attention networks, and integrate our precise layer-wise approximations into branch-and-bound (BaB) systems to accelerate complete verification.

AI USE STATEMENT

In this work, we used generative AI tools for language polishing and text editing. We have not used generative AI tools for generating research ideas, writing the underlying codebase, or producing empirical results, and the rest of the required disclosure tasks are not applicable to this work. We have reviewed all AI-assisted work. All AI-edited text was manually proofread and cross-referenced with our initial drafts by the authors to guarantee that the scientific intent, theoretical claims, and mathematical accuracy regarding the SL-SDP framework remained unaltered. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

To facilitate full reproducibility of our work, Algorithms 1 and 2 detail the exact pseudocode for the single-layer and overlapping two-layer verification frameworks. The source code and execution scripts necessary to replicate the empirical benchmarks in Section 4 are provided in the supplementary material and will be open-sourced upon publication. Additionally, exhaustive mathematical proofs supporting all theoretical claims, particularly the recursive QC propagation derived in Theorem 1, are available in Appendix A.

REFERENCES

MOSEK ApS. *The MOSEK Python Fusion API manual. Version 11.0.*, 2025. URL `https://docs.mosek.com/latest/pythonfusion/index.html`.

Ben Batten, Panagiotis Kouvaros, Alessio Lomuscio, and Yang Zheng. Efficient neural network verification via layer-based semidefinite relaxations and linear cuts. In *International Joint Conference on Artificial Intelligence (IJCAI)*, pp. 2184–2190, 2021.

Robin A Brown, Edward Schmerling, Navid Azizan, and Marco Pavone. A unified view of sdp-based neural network verification through completely positive programming. In *International conference on artificial intelligence and statistics*, pp. 9334–9355. PMLR, 2022.

Rudy Bunel, Ilker Turkaslan, Philip Torr, Pushmeet Kohli, and Pawan K Mudigonda. A unified view of piecewise linear neural network verification. In *Advances in Neural Information Processing Systems*, volume 31, 2018.

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

Chih-Hong Cheng, Georg Nührenberg, and Harald Ruess. Maximum resilience of artificial neural networks. In *Automated Technology for Verification and Analysis*, pp. 251–268. Springer, 2017.

Hong-Ming Chiu and Richard Y Zhang. Tight certification of adversarially trained neural networks via nonconvex low-rank semidefinite relaxations. In *International Conference on Machine Learning*, pp. 5921–5944. PMLR, 2023.

Hong-Ming Chiu, Hao Chen, Huan Zhang, and Richard Y Zhang. Sdp-crown: Efficient bound propagation for neural network verification with tightness of semidefinite programming. *arXiv preprint arXiv:2506.06665*, 2025.

Steven Diamond and Stephen Boyd. CVXPY: A Python-embedded modeling language for convex optimization. *Journal of Machine Learning Research*, 17(83):1–5, 2016.

Mahyar Fazlyab, Alexander Robey, Hamed Hassani, Manfred Morari, and George Pappas. Efficient and accurate estimation of lipschitz constants for deep neural networks. *Advances in neural information processing systems*, 32, 2019.

Mahyar Fazlyab, Manfred Morari, and George J Pappas. Safety verification and robustness analysis of neural networks via quadratic constraints and semidefinite programming. *IEEE Transactions on Automatic Control*, 67(1):1–15, 2022.

Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014.

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

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*, pp. 97–117. Springer, 2017.

Guy Katz, Derek A Huang, Duligur Ibeling, Kyle Julian, Christopher Lazarus, Rachel Lim, Parth Shah, Shantanu Thakoor, Haoze Wu, Aleksandar Zeljić, et al. The marabou framework for verification and analysis of deep neural networks. In *International Conference on Computer Aided Verification*, pp. 443–452. Springer, 2019.

Fabian Latorre, Paul Rolland, and Volkan Cevher. Lipschitz constant estimation of neural networks via sparse polynomial optimization. *arXiv preprint arXiv:2004.08688*, 2020.

Matthew Newton and Antonis Papachristodoulou. Exploiting sparsity for neural network verification. In *Learning for dynamics and control*, pp. 715–727. PMLR, 2021.

Patricia Pauli, Anne Koch, Julian Berberich, Paul Kohler, and Frank Allgöwer. Training robust neural networks using lipschitz bounds. In *IEEE Control Systems Letters*, volume 6, pp. 121–126. IEEE, 2021.

Aditi Raghunathan, Jacob Steinhardt, and Percy S Liang. Semidefinite relaxations for certifying robustness to adversarial examples. *Advances in neural information processing systems*, 31, 2018.

Gagandeep Singh, Timon Gehr, Markus Püschel, and Martin Vechev. Fast and effective robustness certification. In *Advances in Neural Information Processing Systems*, volume 31, 2018.

Gagandeep Singh, Timon Gehr, Markus Püschel, and Martin Vechev. An abstract domain for certifying neural networks. In *Proceedings of the ACM on Programming Languages*, volume 3, pp. 1–30. ACM New York, NY, USA, 2019.

Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. In *International Conference on Learning Representations*, 2014.

Vincent Tjeng, Kai Xiao, and Russ Tedrake. Evaluating robustness of neural networks with mixed integer programming. *arXiv preprint arXiv:1711.07356*, 2017.

Shiqi Wang, Kexin Pei, Justin Whitehouse, Junfeng Yang, and Suman Jana. Efficient formal safety analysis of neural networks. In *Advances in Neural Information Processing Systems*, volume 31, 2018.

Shiqi Wang, Huan Zhang, Kaidi Xu, Xue Lin, Suchi Saria, Giulio Corsi, and Luca Daniel. Beta-crown: Efficient bound propagation with per-neuron split constraints for neural network robustness verification. In *Advances in Neural Information Processing Systems*, volume 34, pp. 29909–29921, 2021.

Eric Wong and Zico Kolter. Provable defenses against adversarial examples via the convex outer adversarial polytope. In *International Conference on Machine Learning*, pp. 5286–5295. PMLR, 2018.

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.

Kaidi Xu, Huan Zhang, Shiqi Wang, Yihan Wang, Suman Jana, Xue Lin, and Cho-Jui Hsieh. Fast and complete: Enabling complete neural network verification with rapid and massively parallel incomplete verifiers. In *International Conference on Learning Representations*, 2021.

Yuezhu Xu and S Sivaranjani. Eclipse: Efficient compositional lipschitz constant estimation for deep neural networks. *Advances in Neural Information Processing Systems*, 37:10414–10441, 2024.

Anton Xue, Lars Lindemann, and Rajeev Alur. Chordal sparsity for sdp-based neural network verification. *Automatica*, 161, 2024.

Vladimir Andreevich Yakubovich. The s-procedure in non-linear control theory. *Vestnik Leningrad Univ. Mathe.*, 4:73–93, 1977.

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

A Appendix: Proof of statements

Proof of Theorem 1. Since applying a permutation to reorder the joint basis from $[1; x_{i-1}; x_i]$ to $[1; x_i; x_{i-1}]$ preserves positive semidefiniteness, one can substitute the LMI $\mathbf{M}_i \succeq 0$ in (7) with

$$\left[\begin{array}{c|c} H_i(\rho_i) - \mathbf{R}_i & -\mathbf{U}_i^\top \\ \hline -\mathbf{U}_i & -\mathbf{V}_i \end{array}\right] \succeq 0.$$

By the properties of the generalized Schur complement, this block semi-definiteness implies the necessary conditions

$$-\mathbf{V}_i \succeq 0, \quad (\mathbf{I}_{n_{i-1}} - \mathbf{V}_i\mathbf{V}_i^\dagger)(-\mathbf{U}_i) = \mathbf{0}. \tag{12}$$

Next, we show that satisfying $\mathbf{M}_i \succeq 0$ ensures that the output of layer i satisfies the new quadratic constraint defined by G_i. Consider the quadratic form of the difference

$$\begin{aligned} &\begin{bmatrix} 1 \\ x_i \end{bmatrix}^\top G_i \begin{bmatrix} 1 \\ x_i \end{bmatrix} - \left(\begin{bmatrix} 1 \\ x_{i-1} \\ x_i \end{bmatrix}^\top P_i(\omega_i^\star) \begin{bmatrix} 1 \\ x_{i-1} \\ x_i \end{bmatrix} + \begin{bmatrix} 1 \\ x_{i-1} \end{bmatrix}^\top Q_{i-1}(\theta_{i-1}^\star) \begin{bmatrix} 1 \\ x_{i-1} \end{bmatrix} \right) \\ &= \left[\begin{array}{c} 1 \\ x_i \\ \hline x_{i-1} \end{array}\right]^\top \underbrace{\left[\begin{array}{c|c} G_i - \mathbf{R}_i & -\mathbf{U}_i^\top \\ \hline -\mathbf{U}_i & -\mathbf{V}_i \end{array}\right]}_{\mathcal{D}_i} \left[\begin{array}{c} 1 \\ x_i \\ \hline x_{i-1} \end{array}\right]. \end{aligned}$$

By the S-procedure, it is sufficient to show that $\mathcal{D}_i \succeq 0$. Since the conditions in (12) are already satisfied, the properties of the generalized Schur complement certify the positive semidefiniteness of $\mathcal{D}_i$ since

$$(G_i - \mathbf{R}_i) - (-\mathbf{U}_i^\top)(-\mathbf{V}_i)^\dagger(-\mathbf{U}_i) = (G_i - \mathbf{R}_i) + \mathbf{U}_i^\top\mathbf{V}_i^\dagger\mathbf{U}_i = \mathbf{0} \succeq 0,$$

which completes the proof. □

Proof of Proposition 1. By the S-procedure, it is sufficient to show that

$$H_i(\rho_i^\star) - G_i \succeq 0. \tag{13}$$

Under the same basis reordering used in the proof of Theorem 1, the constraint $\mathbf{M}_i \succeq 0$ implies the block LMI

$$\begin{bmatrix} H_i(\rho_i^\star) - \mathbf{R}_i & -\mathbf{U}_i^\top \\ -\mathbf{U}_i & -\mathbf{V}_i \end{bmatrix} \succeq 0,$$

which necessitates

$$H_i(\rho_i^\star) - \mathbf{R}_i + \mathbf{U}_i^\top\mathbf{V}_i^\dagger\mathbf{U}_i \succeq 0.$$

Substituting the definition of G_i from (9) into this inequality results in $H_i(\rho_i^\star) - G_i \succeq 0$. This strictly satisfies the required positive-semidefinite condition in (13), thereby concluding the proof. □

Proof of Proposition 2. By Proposition 1, one has $H_i(\rho_i^\star) - G_i \succeq 0$. Since $\tilde{\mathcal{X}}_i^H$ is a non-degenerate ellipsoid, the xx-block of its constraint matrix should satisfy $H_i^{xx}(\rho_i^\star) \prec 0$. This implies that $G_i^{xx} \preceq H_i^{xx}(\rho_i^\star) \prec 0$, ensuring G_i^{xx} is strictly negative definite and invertible. Expanding the quadratic form $[1; x_i]^\top G_i [1; x_i] \geq 0$ yields

$$x_i^\top G_i^{xx} x_i + 2x_i^\top (G_i^{1x})^\top + G_i^{11} \geq 0,$$

which can be rewritten by completing the square as

$$(x_i - \bar{x}_i)^\top G_i^{xx} (x_i - \bar{x}_i) + \kappa_i \geq 0.$$

Since $G_i^{xx} \prec 0$, the set has a non-empty interior *if and only if* $\kappa_i > 0$. Rearranging the terms gives $(x_i - \bar{x}_i)^\top (-G_i^{xx})(x_i - \bar{x}_i) \leq \kappa_i$, and normalizing by κ_i results in the standard ellipsoidal form $(x_i - \bar{x}_i)^\top E_i (x_i - \bar{x}_i) \leq 1$, which completes the proof. □

Proof of Proposition 3. By Proposition 2, one has $x_i \in \mathcal{E}(\bar{x}_i, E_i)$. This ellipsoid can be parameterized as $x_i = \bar{x}_i + E_i^{-1/2} u$, where $\|u\|_2 \leq 1$. Substituting this into the affine transformation for the pre-activations yields

$$\begin{aligned} z_{i+1} &= W_{i+1}(\bar{x}_i + E_i^{-1/2} u) + b_{i+1} \\ &= \hat{v} + W_{i+1} E_i^{-1/2} u. \end{aligned}$$

For the k-th neuron $z_{i+1,k}$ with center $\hat{v}_k$, let $W_{i+1,k}^\top$ denote the k-th row of W_{i+1}. One has

$$z_{i+1,k} = \hat{v}_k + W_{i+1,k}^\top E_i^{-1/2} u.$$

By the Cauchy-Schwarz inequality, the deviation from $\hat{v}_k$ is bounded by

$$|W_{i+1,k}^\top E_i^{-1/2} u| \leq \|(E_i^{-1/2})^\top W_{i+1,k}\|_2 \|u\|_2.$$

Since $\|u\|_2 \leq 1$ and $E_i^{-1/2}$ is symmetric, the maximum absolute deviation is exactly $\sqrt{W_{i+1,k}^\top E_i^{-1} W_{i+1,k}}$. This corresponds to the square root of the k-th diagonal element of $W_{i+1} E_i^{-1} W_{i+1}^\top$, thereby concluding the proof. □

B Appendix: Supplementary Materials

B.1 Examples for the derivation of input set QC

Several common geometric bounds for the input space $\mathcal{X}_{in} \subset \mathbb{R}^{n_0}$ can be rigorously represented via parameterized quadratic constraints. Below, inspired by Fazlyab et al. (2022), we detail the derivation of the constraint matrix $P_{in}(\omega_0)$ for these standard shapes.

Ellipsoid and hypersphere. An ellipsoid centered at $c_0 \in \mathbb{R}^{n_0}$ with a positive-definite shape matrix $E_0 \succ 0$ is defined by $(x_0 - c_0)^\top E_0 (x_0 - c_0) \leq 1$. Expanding this inequality and arranging it into the matrix form yields the single QC

$$\begin{bmatrix} 1 \\ x_0 \end{bmatrix}^\top \begin{bmatrix} 1 - c_0^\top E_0 c_0 & c_0^\top E_0 \\ \star & -E_0 \end{bmatrix} \begin{bmatrix} 1 \\ x_0 \end{bmatrix} \geq 0.$$

Since this inequality remains valid when scaled by any non-negative multiplier, the parameterized constraint matrix becomes

$$P_{in}(\omega_0) = \omega_0 \begin{bmatrix} 1 - c_0^\top E_0 c_0 & c_0^\top E_0 \\ \star & -E_0 \end{bmatrix}, \quad \text{with} \quad \omega_0 \in \Omega_0 \coloneqq \mathbb{R}_+.$$

A hypersphere (or ℓ_2 norm ball) with radius r_0 is simply a special case of an ellipsoid where $E_0 = \frac{1}{r_0^2} \mathbf{I}_{n_0}$. Substituting this into the matrix and scaling the entire block by r_0^2 provides the representation

$$P_{in}(\omega_0) = \omega_0 \begin{bmatrix} r_0^2 - c_0^\top c_0 & c_0^\top \\ \star & -\mathbf{I}_{n_0} \end{bmatrix}, \quad \text{with} \quad \omega_0 \in \Omega_0 \coloneqq \mathbb{R}_+.$$

Hyperrectangle and cube. A hyperrectangle centered at $c_0 \in \mathbb{R}^{n_0}$ with half-widths $a_0 \in \mathbb{R}^{n_0}_+$ is bounded element-wise as $c_0 - a_0 \le x_0 \le c_0 + a_0$. For each dimension $j \in \{1, \dots, n_0\}$, this implies $(x_{0,j} - c_{0,j})^2 \le a_{0,j}^2$. Using the standard basis column vector $e_j \in \mathbb{R}^{n_0}$ (where the j-th element is 1 and all others are 0), this dimensional bound translates to the QC

$$\begin{bmatrix} 1 \\ x_0 \end{bmatrix}^\top \begin{bmatrix} a_0^\top e_j e_j^\top a_0 - c_0^\top e_j e_j^\top c_0 & c_0^\top e_j e_j^\top \\ \star & -e_j e_j^\top \end{bmatrix} \begin{bmatrix} 1 \\ x_0 \end{bmatrix} \ge 0.$$

Taking a combination of these n_0 inequalities with non-negative multipliers $\omega_0 \in \mathbb{R}^{n_0}_+$, and defining the diagonal weight matrix $\Gamma_0 \coloneqq \sum_{j=1}^{n_0} \omega_{0,j} e_j e_j^\top = \mathrm{diag}(\omega_0)$, we aggregate the constraints into

$$P_{in}(\omega_0) = \begin{bmatrix} a_0^\top \Gamma_0 a_0 - c_0^\top \Gamma_0 c_0 & c_0^\top \Gamma_0 \\ \star & -\Gamma_0 \end{bmatrix}, \quad \text{with} \quad \Omega_0 \coloneqq \mathbb{R}^{n_0}_+. \tag{14}$$

Finally, an ℓ_∞ norm ball (or cube) with radius r_0 is a hyperrectangle where all elements of a_0 are equal to r_0, and the matrix is simplified to

$$P_{in}(\omega_0) = \begin{bmatrix} r_0^2 \mathrm{tr}(\Gamma_0) - c_0^\top \Gamma_0 c_0 & c_0^\top \Gamma_0 \\ \star & -\Gamma_0 \end{bmatrix}, \quad \text{with} \quad \Omega_0 \coloneqq \mathbb{R}^{n_0}_+.$$

B.2 Matrix expansions for end-to-end SDP

In Subsection 2.2, we reformulate the layer-wise quadratic constraints into a single monolithic LMI based on the augmented network state vector $\begin{bmatrix} 1 \\ \mathbf{x} \end{bmatrix} \in \mathbb{R}^{1+N}$, where $\mathbf{x} = [x_0; x_1; \dots; x_L]$ and $N = \sum_{j=0}^{L} n_j$.

To do this, we project the local constraint matrices (P_{in}, P_i, H_L) into the higher-dimensional space $\mathbb{R}^{(1+N)\times(1+N)}$ using projection/selection matrices. Let $N_{<i} = \sum_{j=0}^{i-1} n_j$ and $N_{>i} = \sum_{j=i+1}^{L} n_j$ represent the cumulative dimensions of the neurons before and after layer i, respectively.

The selection matrix $\mathbf{E}_0 \in \mathbb{R}^{(1+n_0)\times(1+N)}$ extracts the scalar 1 and the input vector x_0 as

$$\mathbf{E}_0 = \begin{bmatrix} 1 & \mathbf{0}_{1\times n_0} & \mathbf{0}_{1\times N_{>0}} \\ \mathbf{0}_{n_0\times 1} & \mathbf{I}_{n_0} & \mathbf{0}_{n_0\times N_{>0}} \end{bmatrix}.$$

Similarly, $\mathbf{E}_L \in \mathbb{R}^{(1+n_L)\times(1+N)}$ extracts the scalar 1 and the final hidden layer x_L:

$$\mathbf{E}_L = \begin{bmatrix} 1 & \mathbf{0}_{1\times N_{<L}} & \mathbf{0}_{1\times n_L} \\ \mathbf{0}_{n_L\times 1} & \mathbf{0}_{n_L\times N_{<L}} & \mathbf{I}_{n_L} \end{bmatrix}.$$

For the activation constraints between consecutive layers $i-1$ and i, the matrix $\mathbf{E}_{i-1,i} \in \mathbb{R}^{(1+n_{i-1}+n_i)\times(1+N)}$ extracts 1, x_{i-1}, and x_i:

$$\mathbf{E}_{i-1,i} = \begin{bmatrix} 1 & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0}_{n_{i-1}\times N_{<i-1}} & \mathbf{I}_{n_{i-1}} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{I}_{n_i} & \mathbf{0}_{n_i\times N_{>i}} \end{bmatrix}.$$

Using these projection matrices, we formally define the expanded, zero-padded matrices used in the global SDP constraint (6) as

$$\begin{aligned} \bar{P}_{in}(\omega_0) &\coloneqq \mathbf{E}_0^\top P_{in}(\omega_0) \mathbf{E}_0, \\ \bar{P}_i(\omega_i) &\coloneqq \mathbf{E}_{i-1,i}^\top P_i(\omega_i) \mathbf{E}_{i-1,i}, \quad \text{for } i = 1, \dots, L, \\ \bar{H}_L(\rho) &\coloneqq \mathbf{E}_L^\top H_L(\rho) \mathbf{E}_L. \end{aligned}$$

Since the selection matrices merely distribute the elements of the local P matrices into the appropriate blocks of a larger $(1+N)\times(1+N)$ zero matrix, the symmetry and parameterized linearity of the constraints are preserved.

Algorithm 1 Scalable layer-by-layer SDP for robustness analysis of ReLU networks

Require: Network weights and biases $\{W_i, b_i\}_{i=1}^{L+1}$, input constraint matrix P_{in}, output objective matrix Ψ_y, and baseline pre-activation bounds $\{l_i^{base}, u_i^{base}\}_{i=1}^{L}$

Initialize: Set initial bounds $l_1 \leftarrow l_1^{base}$, $u_1 \leftarrow u_1^{base}$
for $i = 1, \ldots, L$ **do**
  if $i < L$ **then**
    Set $H_i(\rho_i)$ as in (11)
  else
    Set $H_L(\rho_L)$ based on the final objective Ψ_y in (5)
  end if
  Construct activation QC $P_i(\omega_i)$ based on current bounds $[l_i, u_i]$ using Lemma 1
  if $i == 1$ **then**
    Set layer input constraint $Q_0(\theta_0) \leftarrow P_{in}(\omega_0)$ with $\theta_0 = \omega_0 \in \Omega_0$
  else
    Construct hyper-rectangle QC $B_{i-1}(\nu_{i-1})$ using bounds l_{i-1}, u_{i-1} via (16)
    Set layer input constraint:

$$Q_{i-1}(\theta_{i-1}) \leftarrow c_{i-1}G_{i-1} + B_{i-1}(\nu_{i-1}) \quad \text{with} \quad \theta_{i-1} = [c_{i-1}; \nu_{i-1}] \in \mathbb{R}_+^{n_{i-1}+1}$$

  end if
  Solve the layer-wise SDP:

$$\rho_i^\star, \omega_i^\star, \theta_{i-1}^\star = \arg\min_{\rho_i, \omega_i, \theta_{i-1}} \rho_i \quad \text{s.t.} \quad \mathbf{M}_i(\rho_i, \omega_i, \theta_{i-1}) \succeq 0$$

  if $i < L$ **then**
    Compute adaptive constraint $G_i = \mathbf{R}_i - \mathbf{U}_i^\top \mathbf{V}_i^\dagger \mathbf{U}_i$ via the Schur complement (9)
    Compute center $\hat{v}$ and semi-width $\hat{a}$ for z_{i+1} using G_i via Proposition 3
    Tighten bounds for the next layer's pre-activations:

$$l_{i+1} \leftarrow \max(\hat{v} - \hat{a},\ l_{i+1}^{base}), \quad u_{i+1} \leftarrow \min(\hat{v} + \hat{a},\ u_{i+1}^{base})$$

  end if
end for
Return: Certified optimal objective bound $\rho^\star = \rho_L^\star$

B.3 Matrix expansions for layer-wise SDP

In Subsection 3.1, the layer-wise optimization problem (7) is constructed using a joint quadratic basis for two consecutive layers. Let this basis vector be $\xi_i = [1; x_{i-1}; x_i] \in \mathbb{R}^{1+n_{i-1}+n_i}$. Since the local constraint matrices $H_i(\rho_i)$ and $Q_{i-1}(\theta_{i-1})$ are defined on smaller dimensional subspaces, they should be expanded with zeros to match the dimension of ξ_i. We define the selection matrices $\mathcal{S}_{x_{i-1}} \in \mathbb{R}^{(1+n_{i-1})\times(1+n_{i-1}+n_i)}$ and $\mathcal{S}_{x_i} \in \mathbb{R}^{(1+n_i)\times(1+n_{i-1}+n_i)}$ as

$$\mathcal{S}_{x_{i-1}} = \begin{bmatrix} 1 & \mathbf{0}_{1\times n_{i-1}} & \mathbf{0}_{1\times n_i} \\ \mathbf{0}_{n_{i-1}\times 1} & \mathbf{I}_{n_{i-1}} & \mathbf{0}_{n_{i-1}\times n_i} \end{bmatrix},$$
$$\mathcal{S}_{x_i} = \begin{bmatrix} 1 & \mathbf{0}_{1\times n_{i-1}} & \mathbf{0}_{1\times n_i} \\ \mathbf{0}_{n_i\times 1} & \mathbf{0}_{n_i\times n_{i-1}} & \mathbf{I}_{n_i} \end{bmatrix}.$$

Using these matrices, the zero-padded expansions used in the linear matrix inequality (7) are defined as

$$\hat{Q}_{i-1}(\theta_{i-1}) \coloneqq \mathcal{S}_{x_{i-1}}^\top Q_{i-1}(\theta_{i-1}) \mathcal{S}_{x_{i-1}},$$
$$\tilde{H}_i(\rho_i) \coloneqq \mathcal{S}_{x_i}^\top H_i(\rho_i) \mathcal{S}_{x_i}.$$

Note that, by definition in (4), the activation constraint matrix $P_i(\omega_i)$ operates on the joint vector $[1; x_{i-1}; x_i]$ and therefore requires no expansion in the layer-wise step.

B.4 ABSTRACTION FOR RELU LAYERS

The ellipsoidal over-approximations obtained at each layer allow us to derive tight interval bounds for every neuron, as detailed in Proposition 3. These bounds are particularly advantageous for ReLU-type activations: if a neuron's pre-activation interval does not cross zero, its activation behaves purely as a linear function, effectively eliminating the nonlinearity for that specific neuron. Moreover, even when a neuron remains uncertain (its pre-activation interval crosses zero), tighter interval bounds improve the accuracy of the abstraction.

Building on these bounds, we derive a refined quadratic constraint for ReLU activations in the following lemma.

Lemma 1. *Suppose the pre-activation vector $z \in \mathbb{R}^n$ is bounded element-wise as $l \leq z \leq u$. Let the corresponding element-wise bounds for the positive ReLU split $\Delta = \phi(z)$ and the negative split $\delta = \phi(z) - z$ be defined as*

$$l_\Delta = \max(l, 0), \quad u_\Delta = \max(u, 0), \quad l_\delta = \max(-u, 0), \quad u_\delta = \max(-l, 0).$$

A valid quadratic constraint on the augmented vector $v = [1; z; \phi(z)]$ is given by

$$\begin{bmatrix} 1 \\ z \\ \phi(z) \end{bmatrix}^\top \mathcal{T}^\top \mathcal{Q} \mathcal{T} \begin{bmatrix} 1 \\ z \\ \phi(z) \end{bmatrix} \geq 0, \quad \mathcal{T} = \begin{bmatrix} 1 & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{I}_n \\ \mathbf{0} & -\mathbf{I}_n & \mathbf{I}_n \end{bmatrix}, \tag{15}$$

where the constraint matrix $\mathcal{Q} \in \mathbb{S}^{1+2n}$ is constructed as

$$\mathcal{Q} = \begin{bmatrix} -l_\delta^\top \mathrm{diag}(\gamma_\delta) u_\delta - l_\Delta^\top \mathrm{diag}(\gamma_\Delta) u_\Delta & \frac{1}{2}(l_\Delta + u_\Delta)^\top \mathrm{diag}(\gamma_\Delta) & \frac{1}{2}(l_\delta + u_\delta)^\top \mathrm{diag}(\gamma_\delta) \\ \star & -\mathrm{diag}(\gamma_\Delta) & \mathrm{diag}(\lambda) \\ \star & \star & -\mathrm{diag}(\gamma_\delta) \end{bmatrix},$$

with multipliers $\gamma_\delta, \gamma_\Delta \in \mathbb{R}^n_+$, and $\lambda \in \mathbb{R}^n$.

Proof of Lemma 1. By defining the vectors $u = [1; \Delta; \delta]$ and $v = [1; z; \phi(z)]$, the linear relationship $u = \mathcal{T} v$ naturally follows from the definition of $\mathcal{T}$ in (15). We derive the quadratic constraints on u by analyzing the properties of the ReLU function for each individual neuron $k \in \{1, \ldots, n\}$.

Since $\phi(z) = \max(z, 0)$, the positive and negative splits satisfy

$$l_{\Delta,k} \leq \Delta_k \leq u_{\Delta,k}, \quad l_{\delta,k} \leq \delta_k \leq u_{\delta,k}, \quad \text{and} \quad \Delta_k \delta_k = 0.$$

These relationships yield the following valid inequalities for any $\gamma_{\Delta,k}, \gamma_{\delta,k} \in \mathbb{R}_+$ and $\lambda_k \in \mathbb{R}$:

$$\begin{aligned} -\gamma_{\Delta,k}(\Delta_k - u_{\Delta,k})(\Delta_k - l_{\Delta,k}) &\geq 0, \\ -\gamma_{\delta,k}(\delta_k - u_{\delta,k})(\delta_k - l_{\delta,k}) &\geq 0, \\ 2\lambda_k \Delta_k \delta_k &= 0. \end{aligned}$$

Summing these conditions across all neurons k and expressing the aggregation in quadratic matrix form yields $u^\top \mathcal{Q} u \geq 0$. Substituting $u = \mathcal{T} v$ results in $v^\top (\mathcal{T}^\top \mathcal{Q} \mathcal{T}) v \geq 0$, which completes the proof. □

The proposed formulation accounts for both stable and unstable neurons. Specifically, for stably active neurons ($l_k \geq 0$), the bounds naturally yield $u_{\delta,k} = l_{\delta,k} = 0$, which enforces the equality $\delta_k = 0$ within the constraint matrix itself. Similarly, for stably inactive neurons ($u_k \leq 0$), we obtain $u_{\Delta,k} = l_{\Delta,k} = 0$, enforcing $\Delta_k = 0$. Therefore, incorporating additional constraints for stable neurons is not required.

B.5 OVERALL VERIFICATION ALGORITHM

Algorithm 1 provides the complete pseudocode for the single-layer verification procedure (SL-SDP). Building upon inexpensive baseline bounds, the algorithm sequentially formulates and solves local SDPs. At each hidden layer, it extracts the adaptive constraint G_i via the Schur complement, calculates tightened pre-activation bounds using Proposition 3, and propagates this refined geometry forward.

To formulate the bounding hyper-rectangle $B_{i-1}(\nu_{i-1})$ used in the layer input constraint $Q_{i-1}(\theta_{i-1})$ for $i \geq 2$, we map the pre-activation bounds $l_{i-1} \leq z_{i-1} \leq u_{i-1}$ through the ReLU function.

The corresponding element-wise valid bounds for the layer output $x_{i-1} = \phi(z_{i-1})$ are given by $\hat{l}_{i-1} = \max(0, l_{i-1})$ and $\hat{u}_{i-1} = \max(0, u_{i-1})$. These bounds confine x_{i-1} to a hyper-rectangle with the geometrical center $(\hat{u}_{i-1} + \hat{l}_{i-1})/2$ and half-width $(\hat{u}_{i-1} - \hat{l}_{i-1})/2$. The corresponding constraint matrix $B_{i-1}(\nu_{i-1})$, parameterized by $\nu_{i-1} \in \mathbb{R}_+^{n_{i-1}}$, is explicitly constructed according to (14) as

$$B_{i-1}(\nu_{i-1}) = \begin{bmatrix} -\hat{u}_{i-1}^\top \mathrm{diag}(\nu_{i-1})\hat{l}_{i-1} & \frac{1}{2}(\hat{u}_{i-1} + \hat{l}_{i-1})^\top \mathrm{diag}(\nu_{i-1}) \\ \star & -\mathrm{diag}(\nu_{i-1}) \end{bmatrix}. \tag{16}$$

For the layer-wise optimization, the constraint term $Q_{i-1}(\theta_{i-1})$ in the LMI $\mathbf{M}_i$ is instantiated as $P_{in}(\omega_0)$ for the first layer ($i = 1$), and as $c_{i-1}G_{i-1} + B_{i-1}(\nu_{i-1})$ for all subsequent layers ($i \geq 2$). Once the layer-wise SDP is solved, the adaptive constraint G_i is extracted via the Schur complement in (9). Finally, G_i is used to analytically compute the tightened pre-activation bounds for the next layer, seamlessly propagating the geometric constraints forward.

B.6 Extended formulation for overlapping layer-by-layer QC propagation

As discussed in Subsection 3.3, the overlapping two-layer strategy mitigates information loss by preserving cross-layer dependencies. Assume we have propagated a joint QC bounding the concatenation of the outputs of the previous two layers, $[x_{i-2}; x_{i-1}]$. To further restrict this feasible region, we define a parameterized constraint family $Q_{i-1,i-2}(\theta_{i-2}) = c_{i-2}G_{i-1,i-2} + B_{i-1,i-2}(\nu_{i-2})$. Here, $G_{i-1,i-2}$ is the joint adaptive constraint propagated from the previous step, and $B_{i-1,i-2}(\nu_{i-2})$ is a bounding hyper-rectangle rigorously derived by applying the formulation in (14) to the concatenated pre-activation bounds of $[x_{i-2}; x_{i-1}]$. The parameter vector $\theta_{i-2} = [c_{i-2}; \nu_{i-2}]$ encapsulates both the scalar multiplier $c_{i-2} \in \mathbb{R}_+$ and the non-negative multipliers $\nu_{i-2} \in \mathbb{R}_+^{n_{i-2}+n_{i-1}}$. Thus, the input to the overlapping window satisfies

$$\begin{bmatrix} 1 \\ x_{i-2} \\ x_{i-1} \end{bmatrix}^\top Q_{i-1,i-2}(\theta_{i-2}) \begin{bmatrix} 1 \\ x_{i-2} \\ x_{i-1} \end{bmatrix} \geq 0.$$

The layer-wise LMI from (7) can be expanded to encompass both layers $i-1$ and i, simultaneously. We formulate the overlapping optimization problem as

$$\begin{aligned} &\min_{\rho_i, \omega_i, \omega_{i-1}, \theta_{i-2}} \rho_i \\ &\text{s.t.} \quad \mathbf{M}_{i,i-1}(\rho_i, \omega_i, \omega_{i-1}, \theta_{i-2}) = \bar{H}_i(\rho_i) - \bar{P}_i(\omega_i) - \bar{P}_{i-1}(\omega_{i-1}) - \bar{Q}_{i-1,i-2}(\theta_{i-2}) \succeq 0, \end{aligned} \tag{17}$$

where the "bar" accents indicate that all local matrices are zero-padded to match the expanded joint basis $[1; x_{i-2}; x_{i-1}; x_i]$. Following the logic of Theorem 1, once a solution is obtained, we aggregate the source constraints and permute the basis to isolate x_{i-2} as

$$\begin{aligned} &\begin{bmatrix} 1 \\ x_{i-1} \\ x_i \end{bmatrix}^\top P_i(\omega_i^\star) \begin{bmatrix} 1 \\ x_{i-1} \\ x_i \end{bmatrix} + \begin{bmatrix} 1 \\ x_{i-2} \\ x_{i-1} \end{bmatrix}^\top P_{i-1}(\omega_{i-1}^\star) \begin{bmatrix} 1 \\ x_{i-2} \\ x_{i-1} \end{bmatrix} + \begin{bmatrix} 1 \\ x_{i-2} \\ x_{i-1} \end{bmatrix}^\top Q_{i-1,i-2}(\theta_{i-2}^\star) \begin{bmatrix} 1 \\ x_{i-2} \\ x_{i-1} \end{bmatrix} \\ &\quad = \left[\begin{array}{c} 1 \\ x_{i-1} \\ x_i \\ \hline x_{i-2} \end{array}\right]^\top \left[\begin{array}{c|c} \mathbf{R}_i & \mathbf{U}_i^\top \\ \hline \mathbf{U}_i & \mathbf{V}_i \end{array}\right] \left[\begin{array}{c} 1 \\ x_{i-1} \\ x_i \\ \hline x_{i-2} \end{array}\right]. \end{aligned} \tag{18}$$

Eliminating x_{i-2} via the Schur complement yields a new joint constraint $G_{i,i-1} = \mathbf{R}_i - \mathbf{U}_i^\top \mathbf{V}_i^\dagger \mathbf{U}_i$ acting on $[1; x_{i-1}; x_i]$. This tightened matrix is then passed forward to construct $Q_{i,i-1}$ for the next overlapping window.

Considering the augmented affine transformation

$$\begin{bmatrix} z_i \\ z_{i+1} \end{bmatrix} = \begin{bmatrix} W_i & \mathbf{0} \\ \mathbf{0} & W_{i+1} \end{bmatrix} \begin{bmatrix} x_{i-1} \\ x_i \end{bmatrix} + \begin{bmatrix} b_i \\ b_{i+1} \end{bmatrix},$$

Algorithm 2 Overlapping layer-by-layer SDP for robustness analysis

Require: Network weights and biases $\{W_i, b_i\}_{i=1}^{L+1}$, input constraint matrix P_{in}, output objective matrix Ψ_y, and baseline pre-activation bounds $\{l_i^{base}, u_i^{base}\}_{i=1}^{L}$

Initialize: Set initial bounds $l_1 \leftarrow l_1^{base}$, $u_1 \leftarrow u_1^{base}$
for $i = 2, \ldots, L$ **do**
  if $i < L$ **then**
    Set the target constraint $H_i(\rho_i)$ as in (11)
  else
    Set $H_L(\rho_L)$ based on the final objective Ψ_y in (5)
  end if
  Construct activation QCs $P_i(\omega_i)$ and $P_{i-1}(\omega_{i-1})$ using current bounds for z_{i-1} and z_i
  if $i == 2$ **then**
    Set layer input constraint $Q_{1,0}(\theta_0) \leftarrow$ zero-padded expansion of $P_{in}(\omega_0)$ onto $[1; x_0; x_1]$
  else
    Construct hyper-rectangle QC $B_{i-1,i-2}(\nu_{i-2})$ using concatenated bounds for $[x_{i-2}; x_{i-1}]$
    Set layer input constraint:

$$Q_{i-1,i-2}(\theta_{i-2}) \leftarrow c_{i-2}G_{i-1,i-2} + B_{i-1,i-2}(\nu_{i-2}) \quad \text{with} \quad \theta_{i-2} = [c_{i-2}; \nu_{i-2}]$$

  end if
  Solve the overlapping layer SDP (17):

$$\rho_i^\star, \omega_i^\star, \omega_{i-1}^\star, \theta_{i-2}^\star = \arg \min_{\rho_i, \omega_i, \omega_{i-1}, \theta_{i-2}} \rho_i \quad \text{s.t.} \quad \mathbf{M}_{i,i-1}(\rho_i, \omega_i, \omega_{i-1}, \theta_{i-2}) \succeq 0$$

  if $i < L$ **then**
    Compute the new joint constraint $G_{i,i-1} = \mathbf{R}_i - \mathbf{U}_i^\top \mathbf{V}_i^\dagger \mathbf{U}_i$ via the Schur complement (18)
    Compute centers $\hat{v}$ and semi-widths $\hat{a}$ for the augmented vector $[z_i; z_{i+1}]$ using $G_{i,i-1}$ via Proposition 3
    Tighten bounds for the current and next layer's pre-activations:

$$l_i \leftarrow \max(\hat{v}_{z_i} - \hat{a}_{z_i},\ l_i), \qquad u_i \leftarrow \min(\hat{v}_{z_i} + \hat{a}_{z_i},\ u_i)$$
$$l_{i+1} \leftarrow \max(\hat{v}_{z_{i+1}} - \hat{a}_{z_{i+1}},\ l_{i+1}^{base}), \qquad u_{i+1} \leftarrow \min(\hat{v}_{z_{i+1}} + \hat{a}_{z_{i+1}},\ u_{i+1}^{base})$$

  end if
end for
Return: Certified optimal objective bound $\rho^\star = \rho_L^\star$

similar to G_i in Proposition 2, the joint matrix $G_{i,i-1}$ bounds $[x_{i-1}; x_i]$ within an ellipsoid and can be leveraged as in Proposition 3 to derive tighter pre-activation intervals for both z_i and z_{i+1}. This consequently makes the QC abstraction for the activation functions more accurate.

Algorithm 2 details the verification procedure using the two-layer overlapping strategy. Similar to the single-layer approach, we assume access to baseline pre-activation interval bounds. We initialize the process for $i = 2$ by deriving $Q_{1,0}$ directly via a zero-padded expansion of the input constraint P_{in}. At each step $i \geq 2$, the algorithm solves the corresponding SDP, then eliminates x_{i-2} via the Schur complement to produce the updated joint constraint $G_{i,i-1}$. As this matrix inherently captures the coupling between x_{i-1} and x_i, applying Proposition 3 allows us to simultaneously refine the pre-activation intervals for both the current layer z_i and the subsequent layer z_{i+1}, systematically preserving relational information throughout the network.

B.7 Detailed numerical results

This section provides additional results and visualizations to supplement the experiments discussed in Section 4. Figure 3 extends the qualitative reachability analysis, illustrating the 2D output bounding regions for networks with depths $L = 5$ and $L = 9$ across varying perturbation radii ($r \in \{0.7, 0.9, 1.1\}$). Consistent with the findings for $L = 7$ in Figure 2, the visualizations confirm that our

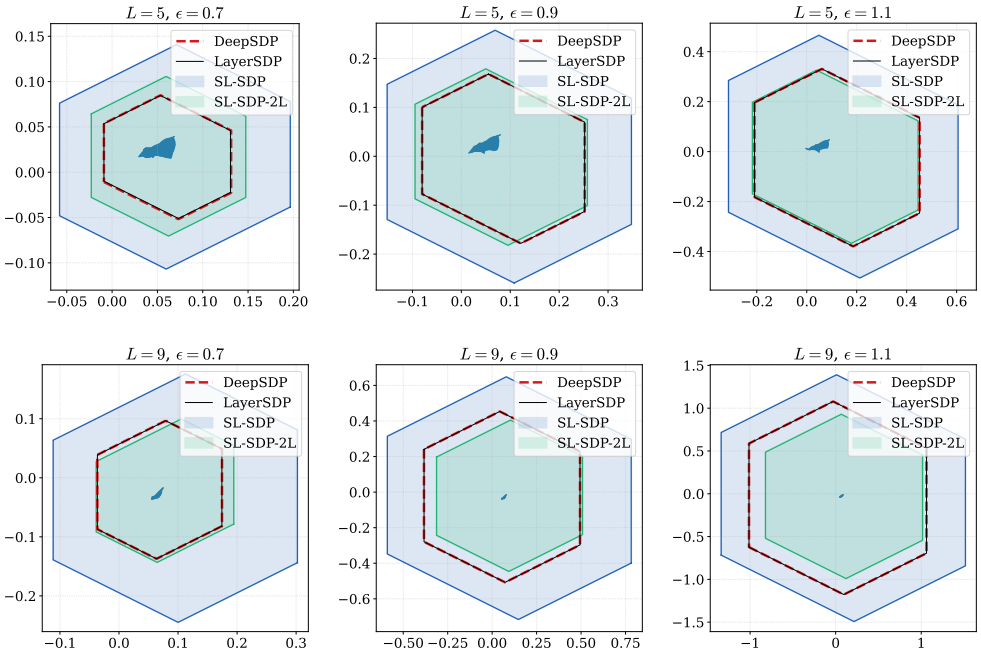


Figure 3: 2D output reachable set bounds for networks of depth $L = 5$ (top row) and $L = 9$ (bottom row) under varying perturbation radii $r \in \{0.7, 0.9, 1.1\}$. The blue point cloud represents the exact reachable set estimated via Monte Carlo sampling. The relative tightness of the SL-SDP and SL-SDP-2L bounds improves notably over baseline methods as the perturbation radius grows, validating the effectiveness of tighter intermediary bounds.

layer-by-layer constraint propagation effectively mitigates the wrapping effect, maintaining tighter over-approximations than monolithic SDPs when both network depth and perturbation magnitude increase. Table 1 in the main text reports the computational runtimes and output upper bounds obtained in the scalability and bound-tightness experiments.