Safety Verification and Robustness Analysis of Neural Networks via Quadratic Constraints and Semidefinite Programming

Mahyar Fazlyab, *Student Member, IEEE*, Manfred Morari, *Fellow, IEEE*, and George J. Pappas, *Fellow, IEEE*

***Abstract*—Certifying the safety or robustness of neural networks against input uncertainties and adversarial attacks is an emerging challenge in the area of safe machine learning and control. To provide such a guarantee, one must be able to bound the output of neural networks when their input changes within a bounded set. In this article, we propose a semidefinite programming (SDP) framework to address this problem for feed-forward neural networks with general activation functions and input uncertainty sets. Our main idea is to abstract various properties of activation functions (e.g., monotonicity, bounded slope, bounded values, and repetition across layers) with the formalism of quadratic constraints. We then analyze the safety properties of the abstracted network via the *S*-procedure and SDP. Our framework spans the tradeoff between conservatism and computational efficiency and applies to problems beyond safety verification. We evaluate the performance of our approach via numerical problem instances of various sizes.**

***Index Terms*—Convex optimization, deep neural networks, robustness analysis, safety verification, semidefinite programming (SDP).**

I. Introduction

NEURAL networks have become increasingly effective at many difficult machine-learning tasks. However, the nonlinear and large-scale nature of neural networks makes them hard to analyze and, therefore, they are mostly used as black-box models without formal guarantees. In particular, neural networks are highly vulnerable to attacks, or more generally, uncertainty in their input. In the context of image classification, for example, neural networks can be easily deluded into changing their classification labels by slightly perturbing the input image [1]. Indeed, it has been shown that even imperceptible perturbations in the input of the state-of-the-art neural networks cause natural images to be misclassified with high probability [2]. Input perturbations can be either of an adversarial nature [3], or they could merely occur due to compression, resizing, and cropping [4]. These drawbacks limit the adoption of neural networks in safety–critical applications such as self-driving vehicles [5], aircraft collision avoidance procedures [6], speech recognition, and recognition of voice commands (see [7] for a survey).

Motivated by the serious consequences of the fragility of neural networks to input uncertainties or adversarial attacks, there has been an increasing effort in developing tools to measure or improve the robustness of neural networks. Many results focus on specific adversarial attacks and attempt to harden the network by, for example, crafting hard-to-classify examples [8]–[11]. Although these methods are scalable and work well in practice, they still suffer from false negatives. Safety–critical applications require provable robustness against any bounded variations in the input data. As a result, many tools have recently been used, adapted, or developed for this purpose, such as mixed-integer linear programming (LP) [12]–[15], convex relaxations and duality theory [16]–[18], satisfiability modulo theory (SMT) [19], dynamical systems [20], [21], abstract interpretation [22], [23], interval-based methods [24]–[28]. All these works aim at bounding the worst-case value of a performance measure when their input is perturbed within a specified range.

Our contribution. In this article, we develop a novel framework based on semidefinite programming (SDP) for safety verification and robustness analysis of neural networks against norm-bounded perturbations in their input. Our main idea is to abstract nonlinear activation functions of neural networks by the constraints they impose on the pre- and post-activation values. In particular, we describe various properties of activation functions using quadratic constraints (QCs), such as bounded slope, bounded values, monotonicity, and repetition across layers. Using this abstraction, any property (e.g., safety or robustness) that we can guarantee for the abstracted network will automatically be satisfied by the original network as well. The quadratic form of these constraints allows us to formulate the verification problem as an SDP feasibility problem. Our main tool for developing the SDP is the $\mathcal{S}$-procedure from robust control [29], which allows us to reason

Manuscript received June 14, 2020; revised November 22, 2020; accepted December 11, 2020. Date of publication December 21, 2020; date of current version December 29, 2021. This work was supported by DARPA Assured Autonomy and NSF CPS 1837210. Recommended by Associate Editor W. Michiels. *(Corresponding author: Mahyar Fazlyab.)*

The authors are with the Department of Electrical and Systems Engineering, University of Pennsylvania, Philadelphia 19104-6321, PA USA (e-mail: mahyarfazlyab@jhu.edu; morari@control.ee.ethz.ch; pappasg@seas.upenn.edu).

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TAC.2020.3046193.

Digital Object Identifier 10.1109/TAC.2020.3046193

about multiple QCs. Our framework has the following notable features.

1) We use various forms of QCs to abstract any type of activation function.
2) Our method can capture *the cross coupling between neurons across different layers*, thereby reducing conservatism. This feature, which hinges on the assumption that the same activation function is used throughout the entire network (repetition across layers), becomes particularly effective for deep networks.
3) We can control the tradeoff between computational complexity and conservatism by systematically including or excluding different types of QCs.

In this article, we focus on the neural network verification problem (formally stated in Section II-A) but the proposed framework (input–output characterization of neural networks via QCs) can be adapted to other problems such as sensitivity analysis of neural networks to input perturbations, output reachable set estimation, probabilistic verification, bounding the Lipschitz constant of neural networks, and closed-loop stability analysis.

A. Related Work

The performance of certification algorithms for neural networks can be measured along three axes. The first axis is the tightness of the certification bounds; the second axis is the computational complexity, and, the third axis is applicability across various models (e.g. different activation functions). These axes conflict. For instance, the conservatism of the verification algorithm is typically at odds with the computational complexity. The relative advantage of any of these algorithms is application specific. For example, reachability analysis and safety verification applications call for less conservative algorithms, whereas in robust training, computationally fast algorithms are desirable [16], [24].

On the one hand, formal verification techniques such as SMT solvers [30]–[32], or integer programming approaches [14], [15] rely on combinatorial optimization to provide tight certification bounds for piece-wise linear networks, whose complexity scales exponentially with the size of the network in the worst-case. A notable work to improve scalability is [15], where the authors do exact verification of piecewise-linear networks using mixed-integer programming with an order of magnitude reduction in computational cost via tight formulations for nonlinearities and careful preprocessing.

On the other hand, certification algorithms based on continuous optimization are more scalable but less accurate. A notable work in this category is [16], in which the authors propose a LP relaxation of piece-wise linear networks and provide upper bounds on the worst-case loss using weak duality. The main advantage of this article is that the proposed algorithm solely relies on forward- and back-propagation operations on a modified network, and thus is easily integrable into existing learning algorithms. In [33], the authors propose an SDP relaxation of one-layer sigmoid-based neural networks based on bounding the worst-case loss with a first-order Taylor expansion. The closest work to the present article is [34], in which the authors propose a semidefinite relaxation (SDR) for certifying robustness of piece-wise linear multilayer neural networks. This relaxation is based on the so-called "lifting," where the original problem is embedded in a much larger space. This SDR approach provides tighter bounds than those of [16] but is less scalable. Finally, compared to the SDR method of [34], our SDP framework yield tighter bounds, especially for deep networks, and is not limited to ReLU networks. Parts of this article, specialized to probabilistic verification, have appeared in the conference paper [35].

The rest of this article is organized as follows. In Section II, we formulate the safety verification problem and present the assumptions. In Section III, we abstract the problem with QCs. In Section IV, we state our main results. In Section V, we discuss further utilities of our framework beyond safety verification. In Section VI, we provide numerical experiments to evaluate the performance of our method and compare it with competing approaches. Finally, in Section VII concludes this article.

B. Notation and Preliminaries

We denote the set of real numbers by $\mathbb{R}$, the set of nonnegative real numbers by $\mathbb{R}_+$, the set of real n-dimensional vectors by $\mathbb{R}^n$, the set of $m \times n$-dimensional matrices by $\mathbb{R}^{m\times n}$, the m-dimensional vector of all ones by 1_m, the $m \times n$-dimensional zero matrix by $0_{m\times n}$, and the n-dimensional identity matrix by I_n. We denote by $\mathbb{S}^n$, $\mathbb{S}^n_+$, and $\mathbb{S}^n_{++}$ the sets of n-by-n symmetric, positive semidefinite, and positive definite matrices, respectively. The p-norm ($p \geq 1$) is displayed by $\|\cdot\|_p : \mathbb{R}^n \to \mathbb{R}_+$. For $A \in \mathbb{R}^{m\times n}$, the inequality $A \succeq 0$ is element-wise. For $A \in \mathbb{S}^n$, the inequality $A \succ 0$ means A is positive semidefinite. For sets $\mathcal{I}$ and $\mathcal{J}$, we denote their Cartesian product by $\mathcal{I} \times \mathcal{J}$. The indicator function of a set $\mathcal{X}$ is defined as $\mathbf{1}_{\mathcal{X}}(x) = 1$ if $x \in \mathcal{X}$, and $\mathbf{1}_{\mathcal{X}}(x) = 0$ otherwise. For two matrices A, B of the same dimension, we denote their Hadamard product by $A \circ B$. A function $g : \mathbb{R}^n \to \mathbb{R}$ is α-convex ($0 \leq \alpha < \infty$) if $g - (\alpha/2)\|\cdot\|_2^2$ is convex. Furthermore, g is β-smooth ($0 \leq \beta < \infty$) if it is differentiable and $(\beta/2)\|\cdot\|_2^2 - g$ is convex. Finally, if g is α-convex and β-smooth, then

$$\frac{\alpha\beta}{\alpha+\beta}\|y-x\|_2^2 + \frac{1}{\alpha+\beta}\|\nabla g(y) - \nabla g(x)\|_2^2 \leq (\nabla g(y) - \nabla g(x))^\top (y-x)$$

for all $x, y \in \mathbb{R}^n$ [36, Th. 2.1.12].

II. Safety Verification and Robustness Analysis of Neural Networks

A. Problem Statement

Consider the nonlinear vector-valued function $f : \mathbb{R}^{n_x} \to \mathbb{R}^{n_f}$ described by a multilayer feed-forward neural network. Given a set $\mathcal{X} \subset \mathbb{R}^{n_x}$ of possible inputs (e.g., adversarial examples), the neural network maps $\mathcal{X}$ to an output set $\mathcal{Y}$ given by

$$\mathcal{Y} = f(\mathcal{X}) := \{y \in \mathbb{R}^{n_f} \mid y = f(x),\ x \in \mathcal{X}\}. \tag{1}$$

The desirable properties that we would like to verify can often be represented by a safety specification set $\mathcal{S}_y$ in the output

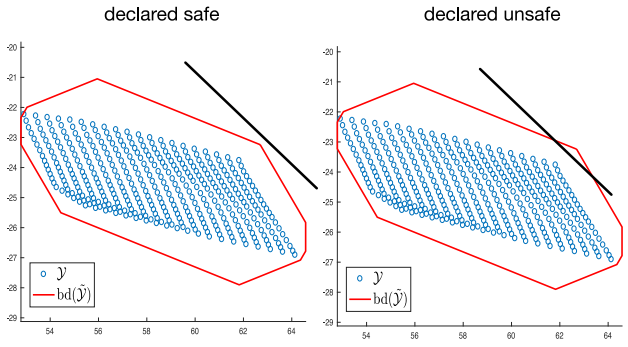

Fig. 1. Output set (in blue), the boundary of its overapproximation (in red), and the hyperplane characterizing the safe region (in black). Left: The network is deemed safe since $\tilde{\mathcal{Y}} \subseteq \mathcal{S}_y$. Right: The network is deemed unsafe since $\tilde{\mathcal{Y}} \not\subseteq \mathcal{S}_y$.

space of the neural network. In this context, the network is safe if the output set lies within the safe region, i.e., if the inclusion $f(\mathcal{X}) \subseteq \mathcal{S}_y$ holds. Alternatively, we can define $\mathcal{S}_x := f^{-1}(\mathcal{S}_y)$ as the inverse image of $\mathcal{S}_y$ under f. Then, safety corresponds to the inclusion $\mathcal{X} \subseteq \mathcal{S}_x$.

Checking the condition $\mathcal{Y} \subseteq \mathcal{S}_y$, however, requires an exact computation of the nonconvex set $\mathcal{Y}$, which is very difficult. Instead, our interest is in finding a nonconservative overapproximation $\tilde{\mathcal{Y}}$ of $\mathcal{Y}$ and verifying the safety properties by checking the condition $\tilde{\mathcal{Y}} \subseteq \mathcal{S}_y$. This approach *detects all false negatives* but also produces false positives, whose rate depends on the tightness of the overapproximation (see Fig. 1). The goal of this article is to solve this problem for a broad class of input uncertainties and safety specification sets using SDP.

1) Classification Example: Consider a data (e.g., image) classification problem with n_f classes, where a feed-forward neural network $f : \mathbb{R}^{n_x} \to \mathbb{R}^{n_f}$ takes as input a data point x and returns an n_f-dimensional vector of scores (or logits)—one for each class. The classification rule is based on assigning x to the class with the highest score. That is, the class of x is given by $C(x) = \operatorname{argmax}_{1\le i\le n_f} f_i(x)$. To evaluate the local robustness of the neural network around a correctly classified point $x^\star$, we consider a set $\mathcal{X}$, containing $x^\star$, that represents the set of all possible perturbations of $x^\star$. In image classification, a popular choice are perturbations in the ℓ_∞ norm, i.e., $\mathcal{X} = \{x : \|x - x^\star\|_\infty \le \epsilon\}$, where ϵ is the maximum perturbation applied to each pixel. Then, the classifier is locally robust at $x^\star$ if it assigns all the perturbed inputs to the same class as $x^\star$, i.e., if $C(x) = C(x^\star)$ for all $x \in \mathcal{X}$. For this problem, the safe set is the polytope $\mathcal{S}_y = \{y \in \mathbb{R}^{n_f} \mid y_{i^\star} \ge y_i \text{ for all } i \ne i^\star\}$, where $i^\star = \operatorname{argmax}_{1\le i\le n_f} f_i(x^\star)$ is the class of $x^\star$.

B. Neural Network Model

For the model of the neural network, we consider an ℓ-layer feed-forward fully connected neural network $f : \mathbb{R}^{n_x} \to \mathbb{R}^{n_f}$ described by the following recursive equations:

$$\begin{aligned} x^0 &= x \\ x^{k+1} &= \phi(W^k x^k + b^k) \quad k = 0, \dots, \ell - 1 \\ f(x) &= W^\ell x^\ell + b^\ell \end{aligned} \tag{2}$$

where $x^0 = x \in \mathbb{R}^{n_0} (n_0 = n_x)$ is the input to the network and $W^k \in \mathbb{R}^{n_{k+1}\times n_k}$, $b^k \in \mathbb{R}^{n_{k+1}}$ are the weight matrix and bias vector of the $(k+1)$th layer. We denote by $n = \sum_{k=1}^{\ell} n_k$ the total number of neurons. The nonlinear activation function ϕ (ReLU,[1] sigmoid, tanh, etc.) is applied coordinate-wise to the preactivation vectors, i.e., it is of the form

$$\phi(x) := [\varphi(x_1) \ \cdots \ \varphi(x_{n_k})]^\top, \ x \in \mathbb{R}^{n_k} \tag{3}$$

where φ is the activation function of each neuron. The output $f(x)$ depends on the specific application we are considering. For example, in image classification with cross-entropy loss, $f(x)$ represents the logit input to the softmax function; or, in feedback control, x is the input to the neural network controller (e.g., tracking error) and $f(x)$ is the control input to the plant.

III. Problem Abstraction via QCs

In this section, our goal is to provide an abstraction of the verification problem described in Section II-A that can be converted into a semidefinite program. Our main tool is QCs, which were first developed in the context of robust control [37] for describing nonlinear, time-varying, or uncertain components of a system. We start with the abstraction of sets using QCs.

A. Input Set

We now provide a particular way of representing the input set $\mathcal{X}$ that will prove useful for developing the SDP.

Definition 1: Let $\mathcal{X} \subset \mathbb{R}^{n_x}$ be a nonempty set. Suppose $\mathcal{P}_\mathcal{X}$ is the set of all symmetric indefinite matrices P such that

$$\begin{bmatrix} x \\ 1 \end{bmatrix}^\top P \begin{bmatrix} x \\ 1 \end{bmatrix} \ge 0 \quad \text{for all } x \in \mathcal{X}. \tag{4}$$

We then say that $\mathcal{X}$ satisfies the QC defined by $\mathcal{P}_\mathcal{X}$.

Note that by definition, $\mathcal{P}_\mathcal{X}$ is a convex cone, i.e., if $P_1, P_2 \in \mathcal{P}_\mathcal{X}$ then $\theta_1 P_1 + \theta_2 P_2 \in \mathcal{P}_\mathcal{X}$ for all nonnegative scalars θ_1, θ_2. Furthermore, we can write

$$\mathcal{X} \subseteq \bigcap_{P\in\mathcal{P}_\mathcal{X}} \left\{ x \in \mathbb{R}^{n_x} : \begin{bmatrix} x \\ 1 \end{bmatrix}^\top P \begin{bmatrix} x \\ 1 \end{bmatrix} \ge 0 \right\}. \tag{5}$$

In other words, we can overapproximate $\mathcal{X}$ by the intersection of a possibly infinite number of sets defined by quadratic inequalities. We will see in Section IV that the matrix $P \in \mathcal{P}_\mathcal{X}$ appears as a decision variable in the SDP. In this way, we can optimize the overapproximation of $\mathcal{X}$ to minimize the conservatism of the specific verification problem we want to solve.

Proposition 1. (QC for hyper-rectangle): The hyper-rectangle $\mathcal{X} = \{x \in \mathbb{R}^{n_x} \mid \underline{x} \le x \le \bar{x}\}$ satisfies the QC defined by

$$\mathcal{P}_\mathcal{X} = \left\{ P \mid P = \begin{bmatrix} -2\Gamma & \Gamma(\underline{x} + \bar{x}) \\ (\underline{x} + \bar{x})^\top \Gamma & -2\underline{x}^\top \Gamma \bar{x} \end{bmatrix} \right\} \tag{6}$$

[1]Rectified Linear Unit.

where $\Gamma \in \mathbb{R}^{n_x \times n_x}$ is diagonal and nonnegative. For this set, (5) holds with equality.

Proof: See Appendix A.

Our particular focus in this article is on perturbations in the ℓ_∞ norm, $\mathcal{X} = \{x \mid \|x - x^\star\|_\infty \le \epsilon\}$, which are a particular class of hyper-rectangles with $\underline{x} = x^\star - \epsilon 1$ and $\bar{x} = x^\star + \epsilon 1$. We can adapt the result of Proposition 1 to other sets such as polytopes, zonotopes, and ellipsoids, as outlined below. The derivation of the corresponding QCs can be found in Appendix B.

1) Polytopes: Let $\mathcal{X} = \{x \in \mathbb{R}^{n_x} \mid Hx \le h\}$ be a polytope, where $H \in \mathbb{R}^{m \times n_x}, h \in \mathbb{R}^m$. Then, $\mathcal{X}$ satisfies the QC defined by

$$\mathcal{P}_\mathcal{X} = \left\{ P \mid P = \begin{bmatrix} H^\top \Gamma H & -H^\top \Gamma h \\ -h^\top \Gamma H & h^\top \Gamma h \end{bmatrix} \right\} \tag{7}$$

where $\Gamma \in \mathbb{S}^m, \Gamma \ge 0, \Gamma_{ii} = 0$. Furthermore, if the set $\{x \in \mathbb{R}^{n_x} \mid Hx \ge h\}$ is empty, then (5) holds with equality.

2) Zonotopes: A zonotope is an affine transformation of the unit cube, $\mathcal{X} = \{x \in \mathbb{R}^{n_x} \mid x = x_c + A\lambda, \quad \lambda \in [0,1]^m\}$, where $A \in \mathbb{R}^{n_x \times m}$ and $x_c \in \mathbb{R}^{n_x}$. Then, any $P \in \mathcal{P}_\mathcal{X}$ satisfies

$$\begin{bmatrix} A & x_c \\ 0 & 1 \end{bmatrix}^\top P \begin{bmatrix} A & x_c \\ 0 & 1 \end{bmatrix} + \begin{bmatrix} 2\Gamma & -\Gamma 1_m \\ -1_m^\top \Gamma & 0 \end{bmatrix} \succeq 0 \tag{8}$$

for some diagonal and nonnegative $\Gamma \in \mathbb{R}^{m \times m}$.

3) Ellipsoids: Suppose the input set $\mathcal{X}$ is an ellipsoid defined by $\mathcal{X} = \{x \in \mathbb{R}^{n_x} \mid \|Ax + b\|_2 \le 1\}$, where $A \in \mathbb{S}^{n_x}$ and $b \in \mathbb{R}^{n_x}$. Then, $\mathcal{X}$ satisfies the QC defined by

$$\mathcal{P}_\mathcal{X} = \left\{ P \mid P = \mu \begin{bmatrix} -A^\top A & -A^\top b \\ -b^\top A & 1 - b^\top b \end{bmatrix}, \ \mu \ge 0 \right\}. \tag{9}$$

B. Safety Specification Set

As mentioned in the introduction, the safe set can be characterized either in the output space ($\mathcal{S}_y$) or in the input space ($\mathcal{S}_x$). In this article, we consider the latter. Specifically, we assume $\mathcal{S}_x$ can be represented (or inner approximated) by the intersection of finitely many quadratic inequalities

$$\mathcal{S}_x = \bigcap_{i=1}^m \left\{ x \in \mathbb{R}^{n_x} \mid \begin{bmatrix} x \\ f(x) \\ 1 \end{bmatrix}^\top S_i \begin{bmatrix} x \\ f(x) \\ 1 \end{bmatrix} \le 0 \right\} \tag{10}$$

where the $S_i \in \mathbb{S}^{n_x + n_f + 1}$ are given. In particular, this characterization includes ellipsoids and polytopes in the output space. For instance, for an output safety specification set described by the polytope $\mathcal{S}_y = \cap_{i=1}^m \{y \in \mathbb{R}^{n_f} \mid c_i^\top y - d_i \le 0\}$, the S_i's are given by

$$S_i = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & c_i \\ 0 & c_i^\top & -2d_i \end{bmatrix} \quad i = 1, \ldots, m.$$

C. Abstraction of Nonlinearities by QCs

One of the main difficulties in the analysis of neural networks is the composition of nonlinear activation functions. To simplify the analysis, instead of analyzing the network directly, our main idea is to remove the nonlinear activation functions from the network but retain the constraints they impose on the pre- and postactivation signals. Using this abstraction, any property (e.g., safety or robustness) that we can guarantee for the "constrained" network will automatically be satisfied by the original network as well. In the following, we show how we can encode various properties of activation functions (e.g., monotonicity, bounded slope, and bounded values) using QCs. We first provide a formal definition as follows.

Definition 2. (QC for functions): Let $\phi : \mathbb{R}^n \to \mathbb{R}^n$ and suppose $\mathcal{Q}_\phi \subset \mathbb{S}^{2n+1}$ is the set of all symmetric indefinite matrices Q such that

$$\begin{bmatrix} x \\ \phi(x) \\ 1 \end{bmatrix}^\top Q \begin{bmatrix} x \\ \phi(x) \\ 1 \end{bmatrix} \ge 0 \quad \text{for all } x \in \mathcal{X} \tag{11}$$

where $\mathcal{X} \subseteq \mathbb{R}^n$ is a nonempty set. Then, we say ϕ satisfies the QC defined by $\mathcal{Q}_\phi$ on $\mathcal{X}$.

We remark that our definition of a QC slightly differs from the one used in robust control [37], by including a constant in the vector surrounding the matrix Q, which allows us to incorporate affine constraints (e.g., bounded nonlinearities). In view of Definition 1, we can interpret (11) as a QC satisfied by the graph of ϕ, $\mathcal{G}(\phi) := \{(x, y) \mid y = \phi(x), \ x \in \mathcal{X}\} \subset \mathbb{R}^{2n}$, i.e., $\mathcal{Q}_\phi = \mathcal{P}_{\mathcal{G}(\phi)}$. Therefore, we can write

$$\mathcal{G}(\phi) \subseteq \bigcap_{Q \in \mathcal{Q}_\phi} \left\{ (x, y) \in \mathbb{R}^{2n} : \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}^\top Q \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \ge 0 \right\}.$$

In other words, we overapproximate the graph of ϕ by a quadratically constrained set.

The derivation of QCs is function specific but there are certain rules and heuristics that can be used for all of them which we describe as follows.

1) Sector-Bounded Nonlinearities: Consider the nonlinear function $\varphi : \mathbb{R} \to \mathbb{R}$ with $\varphi(0) = 0$. We say that φ is *sector-bounded* in the sector $[\alpha, \beta]$ ($\alpha \le \beta < \infty$) if the following condition holds for all $x \in \mathbb{R}$,[2]:

$$(\varphi(x) - \alpha x)(\varphi(x) - \beta x) \le 0. \tag{12}$$

Geometrically, this inequality means that the function $y = \varphi(x)$ lies in the sector formed by the lines $y = \alpha x$ and $y = \beta x$ (see Fig. 2). As an example, the ReLU function belongs to the sector [0,1] and in fact, lies on its boundary.

For the vector case, let $K_1, K_2 \in \mathbb{R}^{n \times n}$ be two matrices such that $K_2 - K_1$ is symmetric positive semidefinite. We say that $\phi : \mathbb{R}^n \to \mathbb{R}^n$ is sector-bounded in the sector $[K_1, K_2]$ if the following condition holds for all $x \in \mathbb{R}^n$ [38]:

$$(\phi(x) - K_1 x)^\top (\phi(x) - K_2 x) \le 0 \tag{13}$$

or, equivalently

$$\begin{bmatrix} x \\ \phi(x) \\ 1 \end{bmatrix}^\top \begin{bmatrix} -K_1^\top K_2 - K_2^\top K_1 & K_1^\top + K_2^\top & 0 \\ K_1 + K_2 & -2I_n & 0 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} x \\ \phi(x) \\ 1 \end{bmatrix} \ge 0.$$

[2]For the case, where $\alpha = -\infty$ or $\beta = +\infty$, we define the sector bound inequality as $x(\varphi(x) - \beta x) \le 0$ and $x(\alpha x - \varphi(x)) \le 0$, respectively.

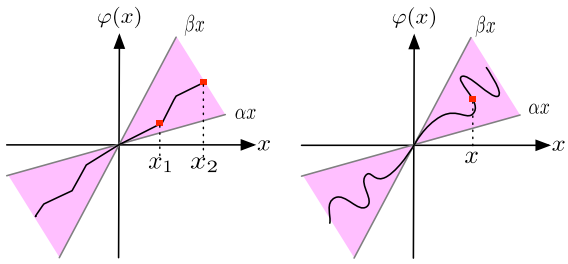

Fig. 2. Slope-restricted nonlinearity (left) and a sector-bounded nonlinearity (right).

The sector condition does not impose any restriction on the slope of the function. This motivates a more accurate description of nonlinearities with bounded slope [39].

2) Slope-Restricted Nonlinearities: A nonlinear function $\phi : \mathbb{R}^n \to \mathbb{R}^n$ is slope-restricted in the sector $[\alpha, \beta]$ ($\alpha \le \beta < \infty$), if for any $x, x^\star \in \mathbb{R}^n$

$$(\phi(x)-\phi(x^\star)-\alpha(x-x^\star))^\top(\phi(x)-\phi(x^\star)-\beta(x-x^\star)) \le 0. \tag{14}$$

For the one-dimensional case ($n = 1$), (14) states that the chord connecting any two points on the curve of ϕ has a slope that is at least α and at most β

$$\alpha \le \frac{\phi(x)-\phi(x^\star)}{x-x^\star} \le \beta \quad \forall x, x^\star \in \mathbb{R}.$$

Note that a slope-restricted nonlinearity with $\phi(0) = 0$ is also sector bounded. Furthermore, if ϕ is slope-restricted in $[\alpha, \beta]$, then the function $x \mapsto \phi(x + x^\star) - \phi(x^\star)$ belongs to the sector $[\alpha I_n, \beta I_n]$ for any $x^\star$. Finally, the gradient of an α-convex and β-smooth function is slope-restricted in $[\alpha, \beta]$.

To connect the results of the previous section to activation functions in neural networks, we recall the following result from convex analysis [36].

Lemma 1. (gradient of convex functions): Consider a function $g : \mathbb{R}^n \to \mathbb{R}$ that is α-convex and β-smooth. Then, the gradient function $\nabla g : \mathbb{R}^n \to \mathbb{R}^n$ is slope-restricted in the sector $[\alpha, \beta]$.

Notably, all commonly used activation functions for deep neural networks are gradients of convex functions. Therefore, they belong to the class of slope-restricted nonlinearities, according to Lemma 1. We have the following result.

Proposition 2: The following statements hold true.

a) The ReLU function $\varphi(x) = \max(0, x)$, $x \in \mathbb{R}$ is slope-restricted and sector-bounded in [0, 1].
b) The sigmoid function, $\varphi(x) = \frac{1}{1+e^{-x}}$, $x \in \mathbb{R}$ is slope-restricted in [0, 1].
c) The tanh function, $\varphi(x) = \tanh(x)$, $x \in \mathbb{R}$ is slope-restricted and sector-bounded in [0, 1].
d) The leaky ReLU function, $\varphi(x) = \max(ax, x)$, $x \in \mathbb{R}$ with $a > 0$ is slope-restricted and sector-bounded in $[\min(a, 1), \max(a, 1)]$.
e) The exponential linear function (ELU), $\varphi(x) = \max(x, a(e^x - 1))$, $x \in \mathbb{R}$ with $a > 0$ is slope-restricted and sector-bounded in [0,1].
f) The softmax function, $\phi(x) = [\frac{e^{x_1}}{\sum_{i=1}^d e^{x_i}}, \cdots, \frac{e^{x_n}}{\sum_{i=1}^d e^{x_i}}]^\top$, $x \in \mathbb{R}^n$ is slope-restricted in [0,1].

In the context of neural networks, our interest is in *repeated nonlinearities* of the form $\phi(x) = [\varphi(x_1) \ \cdots \ \varphi(x_n)]^\top$. Furthermore, the activation values might be bounded from below or above (e.g., the ReLU function which outputs a nonnegative value). The sector bound and slope restricted inequalities can become too conservative as they do not capture these properties. In the following, we discuss QCs for these properties.

3) Repeated Nonlinearities: Suppose $\varphi : \mathbb{R} \to \mathbb{R}$ is slope-restricted in $[\alpha, \beta]$ ($\alpha \le \beta$) and let $\phi(x) = [\varphi(x_1) \cdots \varphi(x_n)]^\top$ be a vector-valued function constructed by component-wise repetition of φ. It is not hard to verify that ϕ is also slope-restricted in $[\alpha, \beta]$. Indeed, by summing the slope-restriction conditions

$$(\varphi(x_i)-\varphi(x_i^\star)-\alpha(x_i-x_i^\star))(\varphi(x_i)-\varphi(x_i^\star)-\beta(x_i-x_i^\star)) \le 0.$$

over $i = 1, \ldots, n$, we obtain (14). However, this representation simply ignores the fact that all the nonlinearities that compose ϕ are the same. By taking advantage of this structure, we can refine the QC that describes ϕ. To be specific, for an input–output pair $(x, \phi(x))$, $x \in \mathbb{R}^n$, we can write the inequality

$$(\varphi(x_i)-\varphi(x_j)-\alpha(x_i-x_j))(\varphi(x_i)-\varphi(x_j)-\beta(x_i-x_j)) \le 0 \tag{15}$$

for all distinct $i, j = 1, \ldots, n$, $i \ne j$. This particular QC can considerably reduce conservatism, especially for deep networks, as it reasons about *the coupling between the neurons throughout the entire network*. By making an analogy to dynamical systems, we can interpret the neural network as a time-varying discrete-time dynamical system where the same nonlinearity is repeated for all "time" indexes k (the layer number). Then, the QC in (15) couples all the possible neurons. In the following lemma, we characterize QCs for repeated nonlinearities.

Lemma 2. (QC for repeated nonlinearities): Suppose $\varphi : \mathbb{R} \to \mathbb{R}$ is slope-restricted in the sector $[\alpha, \beta]$. Then, the vector-valued function $\phi(x) = [\varphi(x_1) \ \cdots \varphi(x_n)]^\top$ satisfies the QC

$$\begin{bmatrix} x \\ \phi(x) \\ 1 \end{bmatrix}^\top \begin{bmatrix} -2\alpha\beta T & (\alpha+\beta)T & 0 \\ (\alpha+\beta)T & -2T & 0 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} x \\ \phi(x) \\ 1 \end{bmatrix} \ge 0 \tag{16}$$

for all $x \in \mathbb{R}^n$, where

$$T = \sum_{1 \le i < j \le n} \lambda_{ij}(e_i - e_j)(e_i - e_j)^\top, \ \lambda_{ij} \ge 0 \tag{17}$$

and $e_i \in \mathbb{R}^n$ is the ith unit vector.

Proof: By a conic combination of $\binom{n}{2}$ QCs of the form (15), we obtain (16). See Appendix C for a detailed proof.

There are several results in the literature about repeated nonlinearities. For instance, in [40] and [41], the authors derive QCs for repeated and odd nonlinearities (e.g., tanh function).

4) Bounded Nonlinearities: Finally, suppose the nonlinear function values are bounded, i.e., $\underline{\phi} \le \phi(x) \le \bar{\phi}$ for all $x \in \mathbb{R}^n$. Using Proposition 1, $\phi(x)$ satisfies the QC

$$\begin{bmatrix} x \\ \phi(x) \\ 1 \end{bmatrix}^\top \begin{bmatrix} 0 & 0 & 0 \\ 0 & -2D & D(\underline{\phi}+\bar{\phi}) \\ 0 & (\underline{\phi}+\bar{\phi})^\top D & -2\underline{\phi}^\top D\bar{\phi} \end{bmatrix} \begin{bmatrix} x \\ \phi(x) \\ 1 \end{bmatrix} \ge 0 \tag{18}$$

for all x, where $D \in \mathbb{R}^{n\times n}$ is diagonal and nonnegative. We can write a similar inequality when the preactivation values are known to be bounded. More generally, if the graph of ϕ is known to satisfy $\mathcal{G}(\phi) \subseteq \mathcal{G}$, then any QC for $\mathcal{G}$ is also a valid QC for ϕ.

We observe that the inequalities (15)–(18) are all quadratic in $(x, \phi(x), 1)$, and therefore can be encapsulated into QCs of the form (11). As we show in Section IV, the matrix $Q \in \mathcal{Q}_\phi$ that abstracts the nonlinearity ϕ appears as a decision variable in the SDP.

Although the above rules can be used to guide the search for valid QCs for activation functions, a less conservative description of activation functions requires a case-by-case treatment to further exploit the structure of the nonlinearity. In the next section, we elaborate on QCs for ReLU activation functions.

D. QCs for ReLU Activation Function

The ReLU function precisely lies on the boundary of the sector [0,1]. This observation can be used to refine the QC description of ReLU. Specifically, let $y = \max(\alpha x, \beta x),\ x \in \mathbb{R}^n$ be the concatenation of n ReLU activation functions.[3] Then, each individual activation function can be described by the following constraints [34]:

$$y_i = \max(\alpha x_i, \beta y_i) \iff \begin{cases} (y_i - \alpha x_i)(y_i - \beta x_i) = 0 \\ \beta x_i \le y_i \\ \alpha x_i \le y_i. \end{cases} \tag{19}$$

The first constraint is the boundary of the sector $[\alpha, \beta]$ and the other constraints simply prune these boundaries to recover the ReLU function. Furthermore, for any two distinct indices $i \neq j$, we can write the constraint (15)

$$(y_j - y_i - \alpha(x_j - x_i))(y_j - y_i - \beta(x_j - x_i)) \le 0. \tag{20}$$

By adding a weighted combination of all these constraints (nonnegative weights for inequalities), we find that the function $y = \max(\alpha x, \beta x)$ satisfies

$$\sum_{i=1}^{n} \{\lambda_i (y_i - \alpha x_i)(y_i - \beta x_i) - \nu_i (y_i - \beta x_i) - \eta_i (y_i - \alpha x_i)\} + \sum_{i\neq j} \lambda_{ij} (y_j - y_i - \alpha(x_j - x_i))(y_j - y_i - \beta(x_j - x_i)) \le 0 \tag{21}$$

for all $x \in \mathbb{R}^n$. In the following lemma, we provide a full QC characterization of the ReLU function.

Lemma 3. (Global QC for ReLU *function):* The function $\phi(x) = \max(\alpha x, \beta x)$ satisfies the QC

$$\begin{bmatrix} x \\ \phi(x) \\ 1 \end{bmatrix}^\top \begin{bmatrix} Q_{11} & Q_{12} & Q_{13} \\ Q_{12}^\top & Q_{22} & Q_{23} \\ Q_{13}^\top & Q_{23}^\top & Q_{33} \end{bmatrix} \begin{bmatrix} x \\ \phi(x) \\ 1 \end{bmatrix} \ge 0 \tag{22}$$

for all $x \in \mathbb{R}^n$, where

$$Q_{11} = -2\alpha\beta(\mathrm{diag}(\lambda) + T),\ Q_{12} = (\alpha+\beta)(\mathrm{diag}(\lambda) + T)$$
$$Q_{13} = -\beta\nu - \alpha\eta,\ Q_{22} = -2(\mathrm{diag}(\lambda) + T)$$
$$Q_{23} = \nu + \eta,\ Q_{33} = 0$$

[3]For ReLU, we have $\alpha = 0$ and $\beta = 1$.

$\nu, \eta \in \mathbb{R}^n_+$, and T is given by (17).

Proof: See Appendix D.

1) Tightening Relaxations: The QC of Lemma 3 holds globally for the whole space $\mathbb{R}^n$. When restricted to a local region $\mathcal{X}$, these QCs can be tightened. Specifically, suppose $y = \max(x, 0)$ and define $\mathcal{I}^+$, $\mathcal{I}^-$, and $\mathcal{I}^\pm$ as the set of activations that are known to be always active, always inactive, or unknown for all $x \in \mathcal{X} \subseteq \mathbb{R}^n$, i.e.,

$$\mathcal{I}^+ = \{i \mid x_i \ge 0 \text{ for all } x \in \mathcal{X}\}$$
$$\mathcal{I}^- = \{i \mid x_i < 0 \text{ for all } x \in \mathcal{X}\}$$
$$\mathcal{I}^\pm = \{1, \dots, n\} \setminus (\mathcal{I}^+ \cup \mathcal{I}^-). \tag{23}$$

Then, the function $y_i = \max(\alpha x_i, \beta x_i)$ belongs to the sector $[\alpha, \alpha]$, $[\alpha, \beta]$ and $[\beta, \beta]$ for inactive, unknown, and active neurons, respectively. Furthermore, since the constraint $y_i \ge \beta x_i$ holds with equality for active neurons, we can write $\nu_i \in \mathbb{R}$ if $i \in \mathcal{I}^+$, $\nu_i \ge 0$ otherwise. Similarly, the constraint $y_i \ge \alpha x_i$ holds with equality for inactive neurons. Therefore, we can write $\eta_i \in \mathbb{R}$ if $i \in \mathcal{I}^-$, $\eta_i \ge 0$ otherwise. Finally, the chord connecting the input–=output pairs of always-active or always-inactive neurons has slope of α or β. Equivalently, for any $(i, j) \in (\mathcal{I}^+ \times \mathcal{I}^+) \cup (\mathcal{I}^- \times \mathcal{I}^-)$, we can write

$$\left(\frac{y_j - y_i}{x_j - x_i} - \alpha\right)\left(\frac{y_j - y_i}{x_j - x_i} - \beta\right) = 0.$$

Therefore, in (21), $\lambda_{ij} \in \mathbb{R}$ for $(i, j) \in (\mathcal{I}^+ \times \mathcal{I}^+) \cup (\mathcal{I}^- \times \mathcal{I}^-)$ and $\lambda_{ij} \ge 0$ otherwise. The above additional degrees of freedom on the multipliers can tighten the relaxation incurred in (21). In the following Lemma, we summarize the above observations.

Lemma 4. (Local QC for ReLU *function):* Let $\phi(x) = \max(\alpha x, \beta x),\ x \in \mathcal{X} \subset \mathbb{R}^n$ and define $\mathcal{I}^+, \mathcal{I}^-$ as in (23). Then ϕ satisfies the QC

$$\begin{bmatrix} x \\ \phi(x) \\ 1 \end{bmatrix}^\top \begin{bmatrix} Q_{11} & Q_{12} & Q_{13} \\ Q_{12}^\top & Q_{22} & Q_{23} \\ Q_{13}^\top & Q_{23}^\top & Q_{33} \end{bmatrix} \begin{bmatrix} x \\ \phi(x) \\ 1 \end{bmatrix} \ge 0 \tag{24}$$

for all $x \in \mathcal{X}$, where

$$Q_{11} = -2\mathrm{diag}(\boldsymbol{\alpha} \circ \boldsymbol{\beta} \circ \lambda) - 2\alpha\beta T$$
$$Q_{12} = \mathrm{diag}((\boldsymbol{\alpha} + \boldsymbol{\beta}) \circ \lambda) + (\alpha + \beta)T$$
$$Q_{13} = -\beta\nu - \alpha\eta, Q_{22} = -2(\mathrm{diag}(\lambda) + T)$$
$$Q_{23} = \nu + \eta,\ Q_{33} = 0$$

with $T = \sum_{1\le i<j\le n} \lambda_{ij}(e_i - e_j)(e_i - e_j)^\top$ and

$$\boldsymbol{\alpha} = [\alpha + (\beta - \alpha)\mathbf{1}_{\mathcal{I}^+}(1), \dots, \alpha + (\beta - \alpha)\mathbf{1}_{\mathcal{I}^+}(n)]$$
$$\boldsymbol{\beta} = [\beta - (\beta - \alpha)\mathbf{1}_{\mathcal{I}^-}(1), \dots, \beta - (\beta - \alpha)\mathbf{1}_{\mathcal{I}^-}(n)]$$
$$\nu_i \in \mathbb{R}_+ \text{ for } i \notin \mathcal{I}^+$$
$$\eta_i \in \mathbb{R}_+ \text{ for } i \notin \mathcal{I}^-$$
$$\lambda_{ij} \in \mathbb{R}_+ \text{ for } \{i, j\} \notin (\mathcal{I}^+ \times \mathcal{I}^+) \cup (\mathcal{I}^- \times \mathcal{I}^-).$$

Proof: See Appendix D.

We do not know *a priori*, which neurons are always active or always inactive. However, we can partially find them

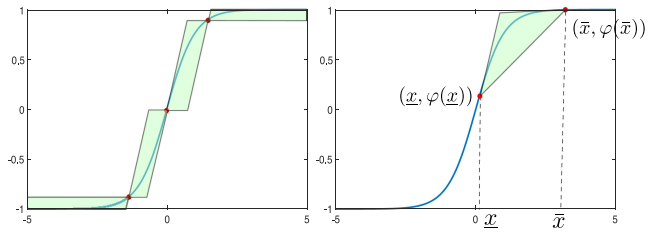

Fig. 3. (Left) The curve of the tanh function overapproximated on $\mathbb{R}$ by the intersection of three sectors. (Right) The curve of the tanh function overapproximated on $[\underline{x}, \bar{x}]$ by a polytope.

by computationally cheap presolve steps. Specifically, if x is known to satisfy $\underline{x} \le x \le \bar{x}$ (bounds on the preactivation values), then we have $\mathcal{I}^+ = \{i \mid \underline{x}_i \ge 0\}$, $\mathcal{I}^- = \{i \mid \bar{x}_i < 0\}$, and $\mathcal{I}^\pm = \{i \mid \bar{x}_i \underline{x}_i \le 0\}$. These element-wise bounds can be found by, for example, interval bound propagation [42], [43] or the LP approach of [16]. Indeed, tighter bounds result in a less conservative description of the ReLU function outlined in Lemma 4.

E. Other Activation Functions

Deriving nonconservative QCs for other activation functions (other than ReLU) is more complicated as they are not on the boundary of any sector. However, by bounding these functions at multiple points by sector bounds of the form (12), we can obtain a substantially better overapproximation. In Fig. 3, we illustrate this idea for the $\tanh$ function.

A secondary approach is to use the element-wise bounds on the inputs to the activation functions to use a tighter sector bound condition in (12). For instance, suppose $x \in [\underline{x}, \bar{x}] \subseteq \mathbb{R}$. Then, the function $\varphi(x) = \tanh(x)$ satisfies the sector condition in (12), where α and β are given by

$$\alpha = \begin{cases} \tanh(\bar{x})/\bar{x} & \text{if } \underline{x}\bar{x} \ge 0 \\ \min(\tanh(\underline{x})/\underline{x}, \tanh(\bar{x})/\bar{x}) & \text{otherwise} \end{cases}$$

$$\beta = \begin{cases} \tanh(\underline{x})/\underline{x} & \text{if } \underline{x}\bar{x} \ge 0 \\ 1 & \text{otherwise.} \end{cases}$$

More generally, suppose the graph of $\varphi : [\underline{x}, \bar{x}] \to \mathbb{R}$ is known to satisfy $\mathcal{G}(\varphi) \subseteq \mathcal{G} \subset \mathbb{R}^2$. Then, any QC satisfied by $\mathcal{G}$ is also a valid QC for φ. We can use this property to build local QCs for general activation functions provided that we can overapproximate their graph locally. This idea is illustrated in Fig. 3 for the case of $\tanh$ function.

IV. Neural Network Verification via SDP

In the previous section, we developed an abstraction of sets and nonlinearities using QCs. In this section, we use this abstraction to develop an linear matrix inequality (LMI) feasibility problem that can assert whether $f(\mathcal{X}) \subseteq \mathcal{S}_y$ (or $\mathcal{X} \subseteq \mathcal{S}_x = f^{-1}(\mathcal{S}_y)$). The crux of our idea in the development of the LMI is the $\mathcal{S}$-procedure [29], a technique to reason about multiple QCs, and is frequently used in robust control and optimization [44], [45].

A. Single-Layer Neural Networks

For the sake of simplicity in the exposition, we start with the analysis of one-layer neural networks and then extend the results to the multilayer case in Section IV-B. We further assume that the safe set $\mathcal{S}_x$ in (10) is specified by a single quadratic form, i.e., $m = 1$. We state our main result in the following theorem.

Theorem 1 (SDP for one layer): Consider a one-layer neural network $f : \mathbb{R}^{n_x} \to \mathbb{R}^{n_f}$ described by the equation

$$f(x) = W^1 \phi(W^0 x + b^0) + b^1 \tag{25}$$

Suppose $x \in \mathcal{X} \subset \mathbb{R}^{n_x}$, where $\mathcal{X}$ satisfies the QC defined by $\mathcal{P}_\mathcal{X}$, i.e., for any $P \in \mathcal{P}_\mathcal{X}$

$$\begin{bmatrix} x \\ 1 \end{bmatrix}^\top P \begin{bmatrix} x \\ 1 \end{bmatrix} \ge 0 \quad \text{for all } x \in \mathcal{X}. \tag{26}$$

Let $\mathcal{Z} = \{z \mid z = W^0 x + b^0,\ x \in \mathcal{X}\}$ and suppose ϕ satisfies the QC defined by $\mathcal{Q}_\phi$ on $\mathcal{Z}$, i.e., for any $Q \in \mathcal{Q}_\phi$

$$\begin{bmatrix} z \\ \phi(z) \\ 1 \end{bmatrix}^\top Q \begin{bmatrix} z \\ \phi(z) \\ 1 \end{bmatrix} \ge 0 \quad \text{for all } z \in \mathcal{Z}. \tag{27}$$

Consider the following matrix inequality:

$$M_{\mathrm{in}}(P) + M_{\mathrm{mid}}(Q) + M_{\mathrm{out}}(S) \preceq 0 \tag{28}$$

where

$$M_{\mathrm{in}}(P) = \begin{bmatrix} I_{n_0} & 0 \\ 0 & 0 \\ 0 & 1 \end{bmatrix} P \begin{bmatrix} I_{n_0} & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{29a}$$

$$M_{\mathrm{mid}}(Q) = \begin{bmatrix} {W^0}^\top & 0 & 0 \\ 0 & I_{n_1} & 0 \\ {b^0}^\top & 0 & 1 \end{bmatrix} Q \begin{bmatrix} W^0 & 0 & b^0 \\ 0 & I_{n_1} & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{29b}$$

$$M_{\mathrm{out}}(S) = \begin{bmatrix} I_{n_0} & 0 & 0 \\ 0 & {W^1}^\top & 0 \\ 0 & {b^1}^\top & 1 \end{bmatrix} S \begin{bmatrix} I_{n_0} & 0 & 0 \\ 0 & W^1 & b^1 \\ 0 & 0 & 1 \end{bmatrix} \tag{29c}$$

and $S \in \mathbb{S}^{n_x + n_f + 1}$ is a given symmetric matrix. If (28) is feasible for some $P \in \mathcal{P}_\mathcal{X},\ Q \in \mathcal{Q}_\phi$, then

$$\begin{bmatrix} x \\ f(x) \\ 1 \end{bmatrix}^\top S \begin{bmatrix} x \\ f(x) \\ 1 \end{bmatrix} \le 0 \text{ for all } x \in \mathcal{X}.$$

Proof: See Appendix F.

Theorem 1 states that if the matrix inequality (28) is feasible for some $(P, Q) \in \mathcal{P}_\mathcal{X} \times \mathcal{Q}_\phi$, then we can certify that the network $\mathcal{X} \subseteq \mathcal{S}_x$ or $f(\mathcal{X}) \subseteq \mathcal{S}_y$. Since $\mathcal{P}_\mathcal{X}$ and $\mathcal{Q}_\phi$ are both convex, (28) is a LMI feasibility problem and, hence, can be efficiently solved via interior-point method solvers for convex optimization.

Remark 1 (End-to-end QC for neural network): It follows from the proof of Theorem 1 that, in view of Definition 2, the neural network in (25) satisfies the QC defined by $(\mathcal{X}, \mathcal{Q}_f)$, where

$$\mathcal{Q}_f = \{Q_f \mid \exists Q \in \mathcal{Q}_\phi \text{ s.t. } M_{\mathrm{mid}}(Q) \preceq M_{\mathrm{out}}(Q_f)\}. \tag{30}$$

In other words, for any $Q_f \in \mathcal{Q}_f$ we have

$$\begin{bmatrix} x \\ f(x) \\ 1 \end{bmatrix}^\top Q_f \begin{bmatrix} x \\ f(x) \\ 1 \end{bmatrix} \geq 0 \quad \text{for all } x \in \mathcal{X}.$$

B. Multilayer Neural Networks

We now turn to multilayer neural networks. Assuming that all the activation functions are the same across the layers (repetition across layers), we can concatenate all the pre- and postactivation signals together and form a more compact representation. To see this, we first introduce $\mathbf{x} = [x^{0\top} \cdots x^{\ell\top}]^\top \in \mathbb{R}^{n_0+n}$, where $\ell \geq 1$ is the number of hidden layers. We further define the entry selector matrices $\mathbf{E}^k \in \mathbb{R}^{n_k \times (n_0+n)}$ such that $x^k = \mathbf{E}^k \mathbf{x}$ for $k = 0, \ldots, \ell$. Then, we can write (2) compactly as

$$x = \mathbf{E}^0 \mathbf{x},\ \mathbf{B}\mathbf{x} = \phi(\mathbf{A}\mathbf{x} + \mathbf{b}),\ f(x) = W^\ell \mathbf{E}^\ell \mathbf{x} + b^\ell \tag{31a}$$

where

$$\mathbf{A} = \begin{bmatrix} W^0 & 0 & \cdots & 0 & 0 \\ 0 & W^1 & \cdots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & W^{\ell-1} & 0 \end{bmatrix} \quad \mathbf{b} = \begin{bmatrix} b^0 \\ b^1 \\ \vdots \\ b^{\ell-1} \end{bmatrix}$$

$$\mathbf{B} = \begin{bmatrix} 0 & I_{n_1} & \cdots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & I_{n_{\ell-1}} & 0 \\ 0 & 0 & \cdots & 0 & I_{n_\ell} \end{bmatrix}. \tag{31b}$$

In the following result, we develop the multilayer counterpart of Theorem 1 for the multilayer neural network in (31).

Theorem 2 (SDP for multiple layers): Consider the multilayer neural network described by (31). Suppose $\mathcal{X} \subset \mathbb{R}^{n_x}$ satisfies the QC defined by $\mathcal{P}_\mathcal{X}$. Define $\mathcal{Z} = \{\mathbf{A}\mathbf{x} + \mathbf{b} \mid x \in \mathcal{X}\}$ and suppose ϕ satisfies the QC defined by $\mathcal{Q}_\phi$ on $\mathcal{Z}$. Consider the following LMI.

$$M_{\text{in}}(P) + M_{\text{mid}}(Q) + M_{\text{out}}(S) \preceq 0 \tag{32}$$

where

$$M_{\text{in}}(P) = \begin{bmatrix} \mathbf{E}^0 & 0 \\ 0 & 1 \end{bmatrix}^\top P \begin{bmatrix} \mathbf{E}^0 & 0 \\ 0 & 1 \end{bmatrix} \tag{33a}$$

$$M_{\text{mid}}(Q) = \begin{bmatrix} \mathbf{A} & \mathbf{b} \\ \mathbf{B} & 0 \\ 0 & 1 \end{bmatrix}^\top Q \begin{bmatrix} \mathbf{A} & \mathbf{b} \\ \mathbf{B} & 0 \\ 0 & 1 \end{bmatrix} \tag{33b}$$

$$M_{\text{out}}(S) = \begin{bmatrix} \mathbf{E}^0 & 0 \\ W^\ell \mathbf{E}^\ell & b^\ell \\ 0 & 1 \end{bmatrix}^\top S \begin{bmatrix} \mathbf{E}^0 & 0 \\ W^\ell \mathbf{E}^\ell & b^\ell \\ 0 & 1 \end{bmatrix} \tag{33c}$$

and $S \in \mathbb{S}^{n_x+n_f+1}$ is a given symmetric matrix. If (32) is feasible for some $(P, Q) \in \mathcal{P}_\mathcal{X} \times \mathcal{Q}_\phi$, then

$$\begin{bmatrix} x \\ f(x) \\ 1 \end{bmatrix}^\top S \begin{bmatrix} x \\ f(x) \\ 1 \end{bmatrix} \leq 0 \quad \text{for all } x \in \mathcal{X}. \tag{34}$$

Proof: See Appendix G.

Remark 2: For the case that the safe set is characterized by more than one quadratic inequality, i.e., when $m > 1$ in (10), then $\mathcal{X} \subseteq \mathcal{S}_x$ if the following LMIs

$$M_{\text{in}}(P_i) + M_{\text{mid}}(Q_i) + M_{\text{out}}(S_i) \leq 0\ i = 1, \ldots, m \tag{35}$$

hold for some $P_i \in \mathcal{P}_\mathcal{X}$ and $Q_i \in \mathcal{Q}_\phi$.

V. Optimization Over the Abstracted Network

In the previous section, we developed an LMI feasibility problem as a sufficient to verify the safety of the neural network. We can incorporate this LMI as a constraint of an optimization problem to solve problems beyond safety verification. Specifically, we can define the following SDP:

$$\begin{aligned} \text{minimize} \quad & g(P, Q, S) \\ \text{subject to} \quad & M_{\text{in}}(P) + M_{\text{mid}}(Q) + M_{\text{out}}(S) \preceq 0 \\ & (P, Q, S) \in \mathcal{P}_\mathcal{X} \times \mathcal{Q}_\phi \times \mathcal{S} \end{aligned} \tag{36}$$

where $g(P, Q, S)$ is a convex function of P, Q, S, and $\mathcal{S}$ is a convex subset of $\mathbb{S}^{n_x+n_f+1}$. In the following, we allude to some utilities of the SDP (36), which we call DeepSDP.

A. Reachable Set Estimation

In Theorem 1, we developed a feasibility problem to assert whether $\mathcal{X} \subseteq \mathcal{S}_x$, or equivalently, $f(\mathcal{X}) \subseteq \mathcal{S}_y$. By parameterizing $\mathcal{S}_x$, we can find the best overapproximation of $f(\mathcal{X})$ by solving (36). Suppose $\mathcal{S}_x$ is described by $\mathcal{S}_x = \{x \mid c^\top f(x) - d \leq 0\}$ with a given $c \in \mathbb{R}^{n_f}$ and $d \in \mathbb{R}$. By defining

$$S = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & c \\ 0 & c^\top & -2d \end{bmatrix} \tag{37}$$

the feasibility of (32) for some $(P, Q) \in \mathcal{P}_\mathcal{X} \times \mathcal{Q}_\phi$ implies $c^\top f(x) \leq d$ for all $x \in \mathcal{X}$. In other words, d is a certified upper bound on the optimal value of the optimization problem

$$\text{maximize } c^\top f(x) \quad \text{subject to } x \in \mathcal{X}. \tag{38}$$

Now, if we treat $d \in \mathbb{R}$ as a decision variable, we can minimize this bound by solving (36) with $g(P, Q, S) = d$. This is particularly useful for overapproximating the reachable set $f(\mathcal{X})$ by a polyhedron of the form $\mathcal{S}_y = \cap_i \{y \in \mathbb{R}^{n_f} \mid c_i^\top y - d_i \leq 0\}$, where c_i are given and the goal is to find the smallest value of d_i, for each i, such that $f(\mathcal{X}) \subseteq \mathcal{S}_y$.

By reparameterizing S in (37), we can also compute the best ellipsoidal overapproximation of $f(\mathcal{X})$. Specifically, define

$$S = \begin{bmatrix} 0 & 0 & 0 \\ 0 & A_y^2 & A_y b_y \\ 0 & b_y^\top A_y & b_y^\top b_y - 1 \end{bmatrix}.$$

Then, the inclusion $f(\mathcal{X}) \subseteq \mathcal{S}_y = f(\mathcal{S}_x)$ implies that $f(\mathcal{X})$ is enclosed by the ellipsoid $\mathcal{S}_y = \{y \in \mathbb{R}^{n_f} \mid \|A_y y + b_y\|_2 \leq 1\}$. Therefore, finding the minimum-volume ellipsoid enclosing $f(\mathcal{X})$ amounts to the optimization problem

$$\begin{aligned} \text{minimize} \quad & \log\det(A_y^{-1}) \\ \text{subject to} \quad & M_{\text{in}}(P) + M_{\text{mid}}(Q) + M_{\text{out}}(S(A_y, b_y)) \preceq 0 \\ & (P, Q, A_y, b_y) \in \mathcal{P}_\mathcal{X} \times \mathcal{Q} \times \mathbb{S}^{n_f} \times \mathbb{R}^{n_f}. \end{aligned} \tag{39}$$

Note that this problem is not convex in (A_y, b_y) due to the nonaffine dependence of S on these variables. However, by using Schur complements, we can formulate an equivalent convex program. We skip the details for the sake of space and refer the reader to [35].

B. Closed-Loop Reachability Analysis

By modifying the matrix S in (37), we can use a similar approach as presented in Section V-A to overapproximate the reachable sets of closed-loop systems involving neural networks. Specifically, consider a discrete-time linear time-invariant (LTI) system driven by a neural network controller

$$x^+ = f_{cl}(x) := Ax + Bf(x),\ x \in \mathcal{X}. \tag{40}$$

Given a set of current states $\mathcal{X}$, the one-step forward reachable set is $\mathcal{X}^+ = f_{cl}(\mathcal{X})$. Suppose $\mathcal{S}_x$ in (10) is defined by $\mathcal{S}_x = \{x \in \mathbb{R}^{n_x} \mid c^\top f_{cl}(x) \le d\}$, where

$$S = \begin{bmatrix} 0 & 0 & A^\top c \\ 0 & 0 & B^\top c \\ c^\top A & c^\top B & -2\,d \end{bmatrix}.$$

According to Theorem 2, the feasibility of the LMI (32) for some $(P, Q) \in \mathcal{P}_{\mathcal{X}} \times \mathcal{Q}_\phi$ would allow us to conclude $\mathcal{X} \subseteq \mathcal{S}_x$, or equivalently, $c^\top f_{cl}(x) \le d$ for all $x \in \mathcal{X}$. By repeating this for different pairs $(c_i, d_i) \in \mathbb{R}^{n_f} \times \mathbb{R},\ i = 1, \ldots, m$, we can overapproximate the one-step reachable set $f_{cl}(\mathcal{X})$ by the polyhedron $\mathcal{P} = \{x^+ \in \mathbb{R}^{n_x} \mid c_i^\top x^+ - d_i \le 0\, i = 1, \ldots, m\}$. Similarly, we can also overapproximate the closed-loop reachable sets by ellipsoids. In Section VI-C, we use this approach to verify a model predictive controller approximated by a neural network.

VI. Discussion and Numerical Experiments

In this section, we discuss the numerical aspects of our approach. For solving the SDP, we used MOSEK [46] with CVX [47] on a 5-core personal computer with 8 GB of RAM. For all experiments, we used ReLU activation functions and did interval bound propagation as a presolve step to determine the element-wise bounds on the activation functions.[4] We start with the computational complexity of the proposed SDP.

A. Computational Complexity

1) Input Set: The number of decision variables for the input set depends on the set representation. The quadratically constrained set that overapproximates hyper-rectangles is indexed by n_x decision variables, where n_x is the input dimension (see Proposition 1). Note that for hyper-rectangles, we can include additional QCs. Indeed, any x satisfying $\underline{x} \le x \le \bar{x}$ satisfies $2n_x^2 - n_x$ QCs of the form $(x_i - \underline{x}_i)(\bar{x}_j - x_i) \ge 0$, $(x_i - \underline{x}_i)(x_j - \underline{x}_j) \ge 0\ i \ne j$, $(x_i - \bar{x}_i)(x_j - \bar{x}_j) \ge 0\ i \ne j$. However, one can precisely characterize a hyper-rectangle with only n_x of these QCs, namely, $(x_i - \underline{x}_i)(\bar{x}_i - x_i) \ge 0$. Our numerical computations reveal that adding the remaining QCs would not tighten the relaxation.

[4]All code, data, and experiments for this article are available at https://github.com/mahyarfazlyab/DeepSDP

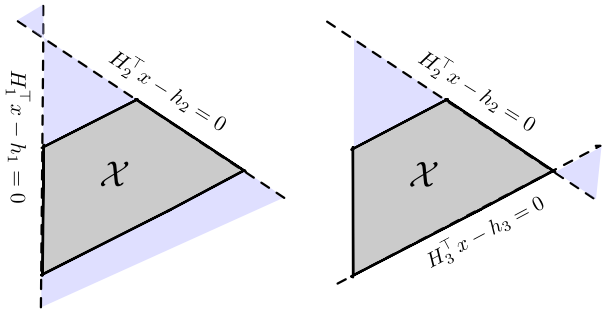

Fig. 4. Sector bound that is not tight (Left) and a sector bound that is tight (Right).

For polytopes, the maximum number of decision variables is $\binom{m}{2}$, where m is the number of half-spaces defining the polytope. However, we can use some heuristics to remove QCs that are not "tight." For instance, for the polytope $\mathcal{X} = \{x \mid Hx \le h\}$, we can write $\binom{m}{2}$ sector bounds of the form $(H_i^\top x - h_i)(H_j^\top x - h_j) \ge 0$. Now, if the intersection of these hyperplanes belongs to $\mathcal{X}$, then the sector would be tight (see Fig. 4). We can verify this by checking the feasibility of

$$H_i^\top x - h_i = H_j^\top x - h_j = 0,\ H_k^\top x - h_k \le 0,\ k \ne i, j.$$

Finally, for the case of ellipsoids, we only have one decision variable, the parameter μ in (9).

2) Activation Functions: For a network with n hidden neurons, if we use all possible QCs, the number of decision variables will be $\mathcal{O}(n + n^2)$. If we ignore repeated nonlinearities, we will arrive at $\mathcal{O}(n)$ decision variables. In our numerical experiments, we did not observe any additional conservatism after removing repeated nonlinearities across the neurons of the same layer. However, accounting for repeated nonlinearities was sometimes very effective for the case of multiple layers.

3) Safety Specification Set: The number of decision variables for the safety specification set depends on how we would like to bound the output set. For instance, for finding a single hyperplane, we have only one decision variable. For the case of ellipsoids, there will be $\mathcal{O}(n_f^2)$ decision variables.

B. Synthetic Examples

1) Number of Hidden Layers: As the first experiment, we consider finding overapproximations of the reachable set of a neural network with a varying number of layers, for a given input set. Specifically, we consider randomly generated neural networks with $n_x = 2$ inputs, $n_f = 2$ outputs, and $\ell = \{1, 2, 3, 4\}$ hidden layers, each having $n_k = 100$ neurons per layer. For the input set, we consider ℓ_∞ balls with center $x^\star = (1, 1)$ and radius $\epsilon = 0.1$. We use DeepSDP to find overapproximations of $f(\mathcal{X})$ in the form of polytopes (see Section V-A). In Fig. 5, we compare the output set $f(\mathcal{X})$ (using exhaustive search over $\mathcal{X}$) with two overapproximations: the red polytope is obtained by solving DeepSDP. The dashed black polytope is obtained by the SDR approach of [34]. We observe that the bounds obtained by DeepSDP are relatively tighter, especially for deeper networks. In Appendix H, we provide more visualizations.

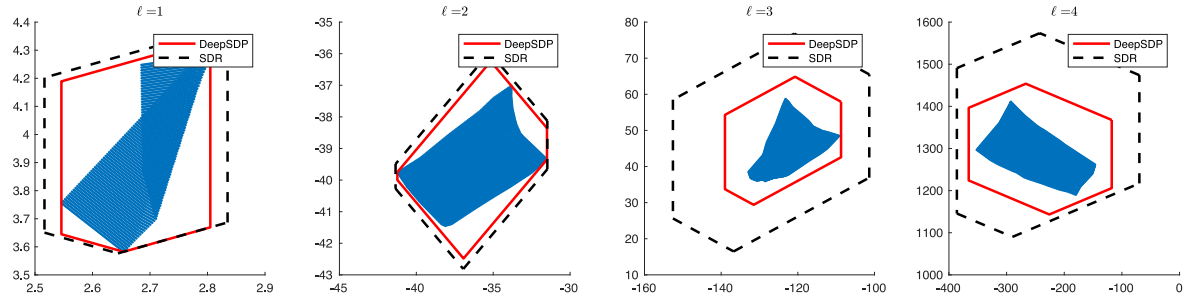

Fig. 5. Illustrations of the output set (blue), the polytope obtained from the results of this article (red), and the polytope obtained by the SDR of [34] (dashed black). The number of neurons per layer is 100, and the input set is the ℓ_∞ ball with center $x^\star = (1, 1)$ and radius $\epsilon = 0.1$. The weights of the neural networks are drawn according to the Gaussian distribution $\mathcal{N}(0, 1/\sqrt{n_x})$. From the left to right, the number of hidden layers is 1, 2, 3, and 4 (the activation function is ReLU).

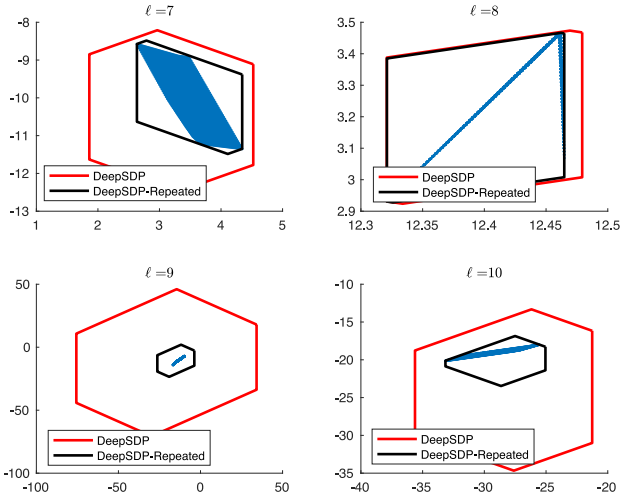

Fig. 6. Plots of the output set (blue), the polytope obtained from $\mathrm{DeepSDP}$ without repeated nonlinearities (red), and the polytope obtained by $\mathrm{DeepSDP}$ after including repeated nonlinearities (black).

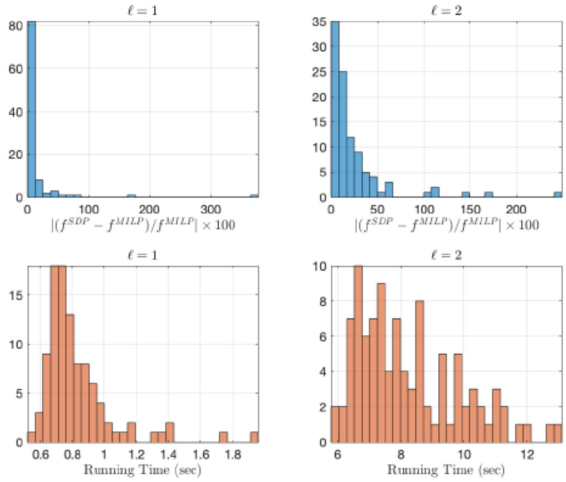

Fig. 7. Histograms of the normalized gap between the optimal values and their corresponding bounds obtained by $\mathrm{DeepSDP}$ (Top). Histograms of solve times in seconds (Bottom).

2) Repeated Nonlinearities: As the second experiment, we study the effect of including repeated nonlinearities on the tightness of the bounds. Specifically, we bound the output of a randomly generated neural network with $n_x = 2$ inputs, $n_f = 2$ output, and $n_k = 10$ neurons per layer by a polytope with 6 facets. For the input set we consider ℓ_∞ ball with center $x^\star = (1, 1)$ and radius $\epsilon = 0.1$. In Fig. 6, we plot the output set, and it overapproximation by $\mathrm{DeepSDP}$ before and after including repeated nonlinearities. We observe that by including repeated nonlinearities, the bounds become tighter, especially for deep networks.

3) Comparison With Other Methods: As the third experiment, we consider the following optimization problem:

$$f^\star = \sup_{\|x - x^\star\|_\infty \le \epsilon} c^\top f(x). \tag{41}$$

To evaluate the tightness of our bounds, we compare $\mathrm{DeepSDP}$ with the MILP formulation of [13], the SDR of [34], and the LP relaxation of [16]. For the problem data, we generated random instances of neural networks with $n_x = 10$ inputs, $n_f = 1$ output and $\ell \in \{1, \ldots, 5\}$ hidden layers; for each layer size, we generated 100 random neural networks with their weights and biases chosen independently from the normal distribution $\mathcal{N}(0, 1/\sqrt{n_x})$. For the input set, we consider $x^\star = 1_{n_x}$ and $\epsilon = 0.2$. In Table I, we report the comparisons of bounds and running times. The MILP formulation finds the global solution but the running time grows quickly as the number of neurons increases. Compared to SDR, the bounds of $\mathrm{DeepSDP}$ are relatively tighter, especially for deeper networks. Finally, the LP relaxation bounds are considerably looser but the running time is negligible. In Fig. 7, we plot the histograms of the normalized gap between the optimal value $f^\star$ (obtained by MILP) and the upper bound f^{SDP} for layer sizes $\ell = 1, 2$.

C. Verification of Approximate Model Predictive Control (MPC)

Consider an LTI system

$$x_{k+1} = Ax_k + Bu_k,\ x_k \in \mathcal{X},\ u_k \in \mathcal{U} \tag{42}$$

where $x_k \in \mathbb{R}^{n_x}$ is the state at time k, $u_k \in \mathbb{R}^{n_u}$ is the control input, and A, B are matrices of appropriate size. The state and control input are subject to the box constraints $\mathcal{X} = \{x \mid \underline{x} \le x \le \bar{x}\}$ and $\mathcal{U} = \{u \mid \underline{u} \le u \le \bar{u}\}$.

TABLE I
AVERAGE VALUES (OVER 100 RUNS) OF DIFFERENT UPPER BOUNDS FOR THE PROBLEM $\sup_{x\in\mathcal{X}} f(x)$ WITH $\mathcal{X} = \|x - X_\star\|_\infty \leq \epsilon$, $x_\star = 1_{n_x}$ AND $\epsilon = 0.2$

	Bounds				Running Time (Sec)			
ℓ	MILP	DeepSDP	SDR	LP	MILP	DeepSDP	SDR	LP
1	1.07	1.12	1.13	1.81	0.04	0.82	0.55	
2	2.04	2.52	2.74	7.62	25.96	8.26	4.71	
3	-	11.08	12.21	50.60	-	34.18	31.20	
4	-	47.74	54.15	368.65	-	78.95	94.74	
5	-	218.8	266.3	3004.9	-	164.63	207.77	

Neural network f has $n_x = 10$ inputs, $n_f = 1$ output and $\ell \in \{1, \ldots, 5\}$ hidden layers.

Suppose the control policy is parameterized by a multilayer fully connected feed-forward network f that is trained offline to approximate a MPC law $\mu^\star(x)$. The motivation is to reduce the computational burden of solving an optimization problem online to determine the MPC control action. The trained neural network, however, does not necessarily satisfy the specifications of the MPC control law such as state and control constraint satisfaction. To ensure input constraint satisfaction, we project the neural network output onto $\mathcal{U}$, resulting in the closed-loop system

$$x_{k+1} = f_{cl}(x_k) := Ax_k + B\mathrm{Proj}_{\mathcal{U}}(f(x_k)). \tag{43}$$

Note that for input box constraints, $\mathcal{U} = \{u \mid \underline{u} \leq u \leq \bar{u}\}$, we can embed the projection operator as two additional layers with a specific choice of weights and biases. Indeed, for an ℓ-layer f, we can describe $f_p(x) = \mathrm{Proj}_{\mathcal{U}}(f(x_k))$ via the $(\ell+2)$-layer ReLU network

$$\begin{aligned} x^0 &= x \\ x^{k+1} &= \max(W^k x^k + b^k, 0) \quad k = 0, \ldots, \ell - 1 \\ x^{\ell+1} &= \max(W^\ell x^\ell + b^\ell - \underline{u}, 0) \\ x^{\ell+2} &= \max(-x^{\ell+1} + \bar{u} - \underline{u}, 0) \\ f_p(x) &= -x^{\ell+2} + \bar{u}. \end{aligned} \tag{44}$$

To validate state constraint satisfaction, we must ensure that there is a set of initial states $\mathcal{E} \subseteq \mathcal{X}$, whose trajectories would always satisfy the state constraints. One such set is a positive invariant set. By definition, a set $\mathcal{E}$ is positively invariant with respect to f_{cl}, if and only if $x_0 \in \mathcal{E}$ implies $x_k \in \mathcal{E}$ for all $k \geq 1$. Equivalently, $\mathcal{E}$ is positively invariant if $f_{cl}(\mathcal{E}) \subseteq \mathcal{E}$. We now show that how we can compute a positive invariant set for (43) using SDP.

To find a positive invariant set for the closed-loop system, we consider the candidate set $\mathcal{E} = \{x \mid \|x\|_\infty \leq \epsilon\}$. We first over approximate the one-step reachable set $f_{cl}(\mathcal{E})$ by the polytope $\mathcal{P} = \{x \mid Hx \leq h\}$, $H \in \mathbb{R}^{m\times n_x}, h \in \mathbb{R}^m$ (see § V-B). To do this, we form the following m SDPs:

$$\begin{aligned} \text{minimize} \quad & h_i \\ \text{subject to} \quad & M_{\mathrm{in}}(P) + M_{\mathrm{mid}}(Q) + M_{\mathrm{out}}(S_i) \prec 0 \\ & (P, Q, h_i) \in \mathcal{P}_{\mathcal{E}} \times \mathcal{Q}_\phi \times \mathbb{R} \end{aligned} \tag{45}$$

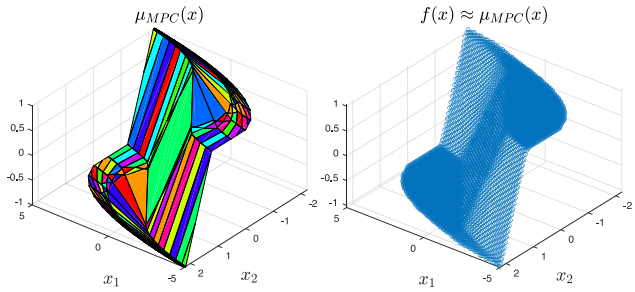

Fig. 8. Explicit MPC control law for the system described in Section VI-C (left), and its approximation by a neural network (right).

where

$$S_i = \begin{bmatrix} 0 & 0 & A^\top H^\top e_i \\ 0 & 0 & B^\top H^\top e_i \\ e_i^\top HA & e_i^\top HB & -2e_i^\top h \end{bmatrix} \quad i = 1, \ldots, m.$$

With this choice of S_i, it is not difficult to show that the feasibility of the LMIs in (45) implies $f_{cl}(\mathcal{E}) \subseteq \mathcal{P}$, and therefore, (45) finds the smallest $\mathcal{P}$ that encloses $f_{cl}(\mathcal{E})$. Then, $\mathcal{E}$ is positively invariant if $\mathcal{P} \subseteq \mathcal{E}$.

For the numerical experiment, we first consider a 2-D system

$$x_{t+1} = 1.2 \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix} x_t + \begin{bmatrix} 1 \\ 0.5 \end{bmatrix} u_t \tag{46}$$

subject to the state and input constraints $x_t \in \mathcal{X} = \{x \mid \|x\|_\infty \leq 5\}$ and $u \in \mathcal{U} = \{u \mid \|u\|_\infty \leq 1\}$. We are interested in stabilizing the system by solving the finite horizon problem

$$\begin{aligned} \text{minimize} \quad & \sum_{t=0}^{T} \|x_t\|_2^2 + u_t^2 \\ \text{s.t.} \quad & (x_t, u_t) \in \mathcal{X} \times \mathcal{U} \;\; t = 0, \ldots, T, \; x_0 = x \end{aligned} \tag{47}$$

and choosing the control law as $\mu_{\mathrm{MPC}}(x) = u_0^\star$. For generating the training data, we compute $\mu_{\mathrm{MPC}}(x)$ at 6284 uniformly chosen random points from the control invariant set. We then train a neural network with two inputs, one output, and two hidden layers with 32 and 16 neurons, respectively using the mean-squared loss. In Fig. 8, we plot the explicit MPC control law as well as its approximation by the neural network.

In Fig. 9, we plot the largest invariant set $\mathcal{E}$ that we could find, which is $\mathcal{E} = \{x \mid \|x\|_\infty \leq 0.65\}$. In this figure, we also plot the output reachable sets for the first four time steps, starting from the initial set $\mathcal{E}$, as well as their overapproximations by DeepSDP.

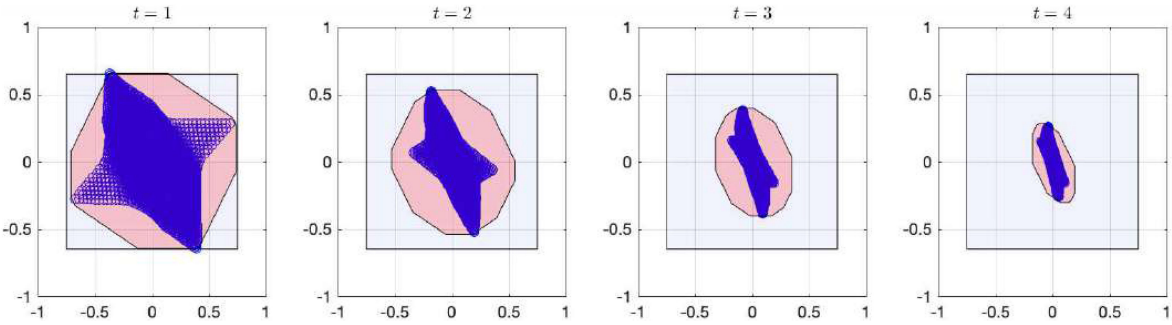

Fig. 9. Illustration of the invariant set $\mathcal{E}$ (light blue), the output reachable sets (dark blue) and their overapproximations (light red) for the system described in Section VI-C. To overapproximate the reachable set at each time step t, we use the overapproximation of the reachable set computed by DeepSDP at $t-1$ as the initial set.

VII. Conclusion

We proposed a SDP framework for robustness analysis and safety verification of feed-forward fully connected neural networks with general activation functions. Our main idea is to abstract the nonlinear activation functions by QCs that are known to be satisfied by all possible input–output instances of the activation functions. We then showed that we can analyze the abstracted network via SDP. We conclude this article with several future directions.

First, a notable advantage of the proposed SDP compared to other convex relaxations is the relative tightness of the bounds. In particular, coupling all pairs of neurons in the network (repeated nonlinearities) can considerably reduce conservatism. However, coupling all neurons is not feasible for even medium-sized networks as the number of decision variables would scale quadratically with the number of neurons. Nevertheless, our numerical experiments show that most of these pair-wise couplings of neurons are redundant and do not tighten the bounds. It would be interesting to develop a method that can decide *a priori* that coupling which pairs of neurons would tighten the relaxation. Second, one of the drawbacks of SDPs is their limited scalability in general. Exploiting the structure of the problem (e.g., sparsity patterns induced by the network strucrure) to reduce the computational complexity would be an important future direction. Third, we have only considered fully connected networks in this article. It would be interesting to extend the results to other architectures. Finally, incorporating the proposed framework in training neural networks with desired robustness properties would be another important future direction.

Appendix

A. Proof of Proposition 1

The inequality $\underline{x} \le x \le \bar{x}$ is equivalent to n_x quadratic inequalities of the form $(x_i - \underline{x}_i)(\bar{x}_i - x_i) \ge 0 \quad i = 1, \dots, n_x$. Multiplying both sides of with $\Gamma_i \ge 0$, summing over $i = 1, \dots, n_x$, and denoting $\Gamma = \operatorname{diag}(\gamma_1, \dots, \gamma_{n_x})$ yields the claimed inequality. □

B. QCs for Polytopes, Zonotopes, and Ellipsoids

1) Polytopes: For every vector x satisfying $Hx \le h$, we have $(H_i^\top x - h_i)(H_j^\top x - h_j) \ge 0$, $i \ne j$, where $H_i^\top$ is the i-th row of H. These inequalities imply

$$\sum_{1 \le i,j \le m} \Gamma_{ij}(H_i^\top x - h_i)(H_j^\top x - h_j) \ge 0$$

where $\Gamma_{ij} = \Gamma_{ji} \ge 0$, $i \ne j$, $\Gamma_{ii} = 0$. The preceding inequality is equivalent to (7). Now suppose the set $\{x \mid Hx \ge h\}$ is empty. Then

$$\mathcal{X} = \{x \mid (H_i^\top x - h_i)^\top (H_j^\top x - h_j) \ge 0,\ i \ne j\}.$$

To show this set equality define $\mathcal{X}_Q$ as the set on the right-hand side. We have $\mathcal{X} \subset \mathcal{X}_Q$. To show $\mathcal{X}_Q \subset \mathcal{X}$, suppose $x \in \mathcal{X}_Q$, implying that either $H_i^\top x - h_i \le 0$ for all i or $H_i^\top x - h_i \ge 0$ for all i. But the latter cannot happen since the set $\{x \mid Hx \ge h\}$ is empty. Therefore, we have $H_i^\top x - h_i \le 0$ for all i.

2) Zonotopes: By multiplying both sides of (8) by $[\lambda^\top\ 1]$ and $[\lambda^\top\ 1]^\top$, respectively, and noting that $x = x_c + A\lambda$ we obtain

$$\begin{bmatrix} x \\ 1 \end{bmatrix}^\top P \begin{bmatrix} x \\ 1 \end{bmatrix} \ge \begin{bmatrix} \lambda \\ 1 \end{bmatrix}^\top \begin{bmatrix} -2\Gamma & \Gamma 1_m \\ -1_m^\top \Gamma & 0 \end{bmatrix} \begin{bmatrix} \lambda \\ 1 \end{bmatrix} \ge 0$$

where the right inequality follows from the fact that $\lambda \in [0,1]^m$, hence satisfying the QC of Proposition 1.

3) Ellipsoids: Any $x \in \mathcal{X}$ satisfies $\mu(1 - (Ax+b)^\top (Ax+b)) \ge 0$ for $\mu \ge 0$. The latter inequality is equivalent to (9). □

C. Proof of Lemma 2

For any distinct pairs $(x_i, \varphi(x_i))$ and $(x_j, \varphi(x_j))$, $1 \le i < j \le n$, we can write the slope restriction inequality in (14) as

$$\begin{bmatrix} x_i - x_j \\ \varphi(x_i) - \varphi(x_j) \end{bmatrix}^\top \begin{bmatrix} -2\alpha\beta & \alpha+\beta \\ \alpha+\beta & -2 \end{bmatrix} \begin{bmatrix} x_i - x_j \\ \varphi(x_i) - \varphi(x_j) \end{bmatrix} \ge 0.$$

By multiplying both sides by $\lambda_{ij} \ge 0$, we obtain

$$\begin{bmatrix} x \\ \phi(x) \\ 1 \end{bmatrix}^\top \begin{bmatrix} -2\alpha\beta E_{ij}\lambda_{ij} & (\alpha+\beta)E_{ij}\lambda_{ij} & 0 \\ (\alpha+\beta)E_{ij}\lambda_{ij} & -2E_{ij}\lambda_{ij} & 0 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} x \\ \phi(x) \\ 1 \end{bmatrix} \ge 0$$

where $E_{ij} = (e_i - e_j)(e_i - e_j)^\top$ and $e_i \in \mathbb{R}^n$ is the ith unit vector in $\mathbb{R}^n$. Summing over all $1 \le i < j \le n$ will yield the desired result.

D. Proof of Lemma 3

Consider the equivalence in (19) for the ith coordinate of $y = \max(\alpha x, \beta x),\ x \in \mathbb{R}^n$:

$$(y_i - \alpha x_i)(y_i - \beta x_i) = 0,\ y_i \geq \beta x_i, \quad y_i \geq \alpha x_i.$$

Multiplying these constraints by $\lambda_i \in \mathbb{R}, \nu_i \in \mathbb{R}_+$, and $\eta_i \in \mathbb{R}_+$, respectively, and adding them together, we obtain

$$\begin{bmatrix} x_i \\ y_i \\ 1 \end{bmatrix}^\top \begin{bmatrix} -2\alpha\beta\lambda_i & (\alpha+\beta)\lambda_i & -\beta\nu_i - \alpha\eta_i \\ (\alpha+\beta)\lambda_i & -2\lambda_i & \nu_i + \eta_i \\ -\beta\nu_i - \alpha\eta_i & \nu_i + \eta_i & 0 \end{bmatrix} \begin{bmatrix} x_i \\ y_i \\ 1 \end{bmatrix} \geq 0.$$

Substituting $x_i = e_i^\top x$ and $y_i = e_i^\top y$, where e_i is the ith unit vector in $\mathbb{R}^n$, and rearranging terms, we get

$$\begin{bmatrix} x \\ y \\ 1 \end{bmatrix}^\top Q_i \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \geq 0,\ i = 1, \ldots, n \tag{48}$$

where

$$Q_i = \begin{bmatrix} -2\alpha\beta\lambda_i & (\alpha+\beta)\lambda_i e_i e_i^\top & (-\beta\nu_i - \alpha\eta_i)e_i \\ (\alpha+\beta)\lambda_i e_i & -2\lambda_i e_i & (\nu_i + \eta_i)e_i \\ (-\beta\nu_i - \alpha\eta_i)e_i & (\nu_i + \eta_i)e_i & 0 \end{bmatrix}.$$

Furthermore, since $y_i = \max(\alpha x_i, \beta x_i)$ is slope-restricted in $[\alpha, \beta]$, by Lemma 2, we can write

$$\begin{bmatrix} x \\ y \\ 1 \end{bmatrix}^\top \begin{bmatrix} -2\alpha\beta T & (\alpha+\beta)T & 0 \\ (\alpha+\beta)T & -2\,T & 0 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \geq 0. \tag{49}$$

Summing (48) over all $i = 1, \ldots, n$ and adding the result to (49) would yield (22). □

E. Proof of Lemma 4

Consider the relation $y = \max(\alpha x, \beta x)$. For active neurons, $i \in \mathcal{I}^+$, we can write

$$(y_i - \beta x_i)(y_i - \beta x_i) = 0,\ y_i = \beta x_i, \quad y_i \geq \alpha x_i.$$

Similarly, for inactive neurons, $i \in \mathcal{I}^-$,we can write

$$(y_i - \alpha x_i)(y_i - \alpha x_i) = 0,\ y_i \geq \beta x_i, \quad y_i = \alpha x_i.$$

Finally, for unknown neurons, $i \in \mathcal{I}^\pm$, we can write

$$(y_i - \alpha x_i)(y_i - \beta x_i) = 0,\ y_i \geq \beta x_i, \quad y_i \geq \alpha x_i.$$

A weighted combination of the above constraints yields

$$\sum_{i=1}^{n} \lambda_i(y_i - \alpha_i x_i)(y_i - \beta_i x_i) + \nu_i(y_i - \beta_i x_i) + \eta_i(y_i - \alpha_i x_i) \geq 0 \tag{50}$$

where $\alpha_i = \alpha + (\beta - \alpha)\mathbf{1}_{\mathcal{I}^+}(i)$, $\beta_i = \beta - (\beta - \alpha)\mathbf{1}_{\mathcal{I}^-}(i)$, $\nu_i \in \mathbb{R}_+$ for $i \notin \mathcal{I}^+$ and $\eta_i \in \mathbb{R}_+$ for $i \notin \mathcal{I}^-$. Furthermore, since $y_i = \max(\alpha x_i, \beta x_i)$ is slope-restricted on $[\alpha, \beta]$, we can write

$$-\sum_{i \neq j} \lambda_{ij}(y_j - y_i - \alpha(x_j - x_i))(y_j - y_i - \beta(x_j - x_i)) \geq 0. \tag{51}$$

Adding (50) and (51) and rearranging terms would yield the desired inequality. □

F. Proof of Theorem 1

Consider the identity $x^1 = \phi(W^0 x^0 + b^0)$. Using the assumption that ϕ satisfies the quadaratic constraint defined by $\mathcal{Q}_\phi$ on $\mathcal{Z}$, x^0, x^1 satisfy the QC

$$\begin{bmatrix} x^0 \\ x^1 \\ 1 \end{bmatrix}^\top \underbrace{\begin{bmatrix} W^0 & 0 & b^0 \\ 0 & I_{n_1} & 0 \\ 0 & 0 & 1 \end{bmatrix}^\top Q \begin{bmatrix} W^0 & 0 & b^0 \\ 0 & I_{n_1} & 0 \\ 0 & 0 & 1 \end{bmatrix}}_{M_{\mathrm{mid}}(Q)} \begin{bmatrix} x^0 \\ x^1 \\ 1 \end{bmatrix} \geq 0 \tag{52}$$

for any $Q \in \mathcal{Q}_\phi$ and all $x^0 \in \mathcal{X}$. By assumption $\mathcal{X}$ satisfies the QC defined by $\mathcal{P}_\mathcal{X}$, implying that for any $P \in \mathcal{P}_\mathcal{X}$

$$\begin{bmatrix} x^0 \\ x^1 \\ 1 \end{bmatrix}^\top \underbrace{\begin{bmatrix} I_{n_0} & 0 \\ 0 & 0 \\ 0 & 1 \end{bmatrix}^\top P \begin{bmatrix} I_{n_0} & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}}_{M_{\mathrm{in}}(P)} \begin{bmatrix} x^0 \\ x^1 \\ 1 \end{bmatrix} \geq 0 \tag{53}$$

for all $x^0 \in \mathcal{X}$. Suppose (28) holds for some $(P, Q) \in \mathcal{P}_\mathcal{X} \times \mathcal{Q}_\phi$. By left- and right- multiplying both sides of (28) by $[x^{0\top}\ x^{1\top}\ 1]$ and $[x^{0\top}\ x^{1\top}\ 1]^\top$, respectively, we obtain

$$\underbrace{\begin{bmatrix} x^0 \\ x^1 \\ 1 \end{bmatrix}^\top M_{\mathrm{in}}(P) \begin{bmatrix} x^0 \\ x^1 \\ 1 \end{bmatrix}}_{\geq 0 \text{ for all } x^0 \in \mathcal{X} \text{ by (53)}} + \underbrace{\begin{bmatrix} x^0 \\ x^1 \\ 1 \end{bmatrix}^\top M_{\mathrm{mid}}(Q) \begin{bmatrix} x^0 \\ x^1 \\ 1 \end{bmatrix}}_{\geq 0 \text{ for all } x^0 \in \mathcal{X} \text{ by (52)}}$$

$$+ \begin{bmatrix} x^0 \\ x^1 \\ 1 \end{bmatrix}^\top M_{\mathrm{out}}(S) \begin{bmatrix} x^0 \\ x^1 \\ 1 \end{bmatrix} \leq 0.$$

Therefore, the last term on the left-hand side must be nonpositive for all $x^0 \in \mathcal{X},\ x^1 = \phi(W^0 x^0 + b^0)$, or, equivalently

$$\begin{bmatrix} x^0 \\ x^1 \\ 1 \end{bmatrix}^\top \begin{bmatrix} I_{n_0} & 0 & 0 \\ 0 & W^{1\top} & 0 \\ 0 & b^{1\top} & 1 \end{bmatrix} S \begin{bmatrix} I_{n_0} & 0 & 0 \\ 0 & W^1 & b^1 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x^0 \\ x^1 \\ 1 \end{bmatrix} \leq 0.$$

Using the relations $x^0 = x$ and $f(x) = W^1 x^1 + b^1$, the above inequality is the desired inequality in (34). □

G. Proof of Theorem 2

Recall the definition $\mathcal{Z} = \{\mathbf{A}\mathbf{x} + \mathbf{b} \mid x \in \mathcal{X}\}$. Since ϕ satisfies the QC defined by $\mathcal{Q}_\phi$ on $\mathcal{Z}$, for any $Q \in \mathcal{Q}_\phi$, we have

$$\begin{bmatrix} \mathbf{x} \\ 1 \end{bmatrix}^\top \underbrace{\begin{bmatrix} \mathbf{A} & \mathbf{b} \\ \mathbf{B} & 0 \\ 0 & 1 \end{bmatrix}^\top Q \begin{bmatrix} \mathbf{A} & \mathbf{b} \\ \mathbf{B} & 0 \\ 0 & 1 \end{bmatrix}}_{M_{\mathrm{mid}}(Q)} \begin{bmatrix} \mathbf{x} \\ 1 \end{bmatrix} \geq 0 \text{ for all } x^0 \in \mathcal{X} \tag{54}$$

By assumption $\mathcal{X}$ satisfies the QC defined by $\mathcal{P}_\mathcal{X}$. Using the relation $x^0 = \mathbf{E}^0 \mathbf{x}$, for any $P \in \mathcal{P}_\mathcal{X}$ it holds that

$$\begin{bmatrix} \mathbf{x} \\ 1 \end{bmatrix}^\top \underbrace{\begin{bmatrix} \mathbf{E}^0 & 0 \\ 0 & 1 \end{bmatrix}^\top P \begin{bmatrix} \mathbf{E}^0 & 0 \\ 0 & 1 \end{bmatrix}}_{M_{\mathrm{in}}(P)} \begin{bmatrix} \mathbf{x} \\ 1 \end{bmatrix} \geq 0 \text{ for all } x^0 \in \mathcal{X}. \tag{55}$$

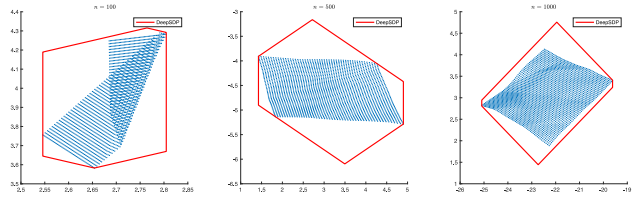

Fig. 10. Effect of the number of hidden neurons on the overapproximation quality of the SDP for a one-layer neural network with 100 (left), 500 (middle), and 1000 hidden nuerons (right). The activation function is ReLU. QCs for repeated nonlinearity are not included.

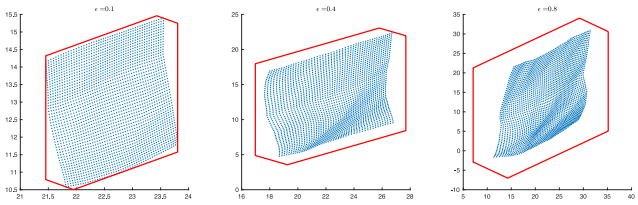

Fig. 11. Effect of ϵ (the ℓ_∞ norm of the input set) on the overapproximation quality of the SDP for $\epsilon = 0.1$ (left), $\epsilon = 0.4$ (middle), and $\epsilon = 0.8$ (right). The network architecture is 2-500-2 with ReLU activation functions. QCs for repeated nonlinearity are not included.

Suppose the LMI in (32) holds for some $(P, Q) \in \mathcal{P}_{\mathcal{X}} \times \mathcal{Q}_\phi$. By left- and right- multiplying both sides of (27) by $[\mathbf{x}^\top \; 1]$ and $[\mathbf{x}^\top \; 1]^\top$, respectively, we obtain

$$\underbrace{\begin{bmatrix}\mathbf{x}\\1\end{bmatrix}^\top M_{\text{in}}(P)\begin{bmatrix}\mathbf{x}\\1\end{bmatrix}}_{\geq 0 \text{ by } (55)} + \underbrace{\begin{bmatrix}\mathbf{x}\\1\end{bmatrix}^\top M_{\text{mid}}(Q)\begin{bmatrix}\mathbf{x}\\1\end{bmatrix}}_{\geq 0 \text{ by } (54)}$$
$$+ \begin{bmatrix}\mathbf{x}\\1\end{bmatrix}^\top M_{\text{out}}(S)\begin{bmatrix}\mathbf{x}\\1\end{bmatrix} \leq 0.$$

Therefore, the last quadratic term must be nonpositive for all $x^0 \in \mathcal{X}$, from where we can write

$$\begin{bmatrix}\mathbf{x}\\1\end{bmatrix}^\top \begin{bmatrix}\mathbf{E}^0 & 0\\ W^\ell\mathbf{E}^\ell & b^\ell\\ 0 & 1\end{bmatrix}^\top S \begin{bmatrix}\mathbf{E}^0 & 0\\ W^\ell\mathbf{E}^\ell & b^\ell\\ 0 & 1\end{bmatrix}\begin{bmatrix}\mathbf{x}\\1\end{bmatrix} \leq 0 \text{ for all } x^0 \in \mathcal{X}.$$

Using the relations $x^0 = \mathbf{E}^0\mathbf{x}$ and $f(x) = W^\ell\mathbf{E}^\ell\mathbf{x} + b^\ell$ from (31), the above inequality can be written as

$$\begin{bmatrix}x^0\\ f(x^0)\\1\end{bmatrix}^\top S \begin{bmatrix}x^0\\ f(x^0)\\1\end{bmatrix} \leq 0, \text{ for all } x^0 \in \mathcal{X}.$$

□

H. More Visualizations

In Fig. 10, we show the effect of the number of hidden neurons on the quality of approximation for a single-layer network, and in Fig. 11, we change the perturbation size.

REFERENCES

[1] J. Su, D. V. Vargas, and K. Sakurai, "One pixel attack for fooling deep neural networks," *IEEE Trans. Evol. Comput.*, vol. 23, no. 5, pp. 828–841, Oct. 2019.

[2] S.-M. Moosavi-Dezfooli, A. Fawzi, O. Fawzi, and P. Frossard, "Universal adversarial perturbations," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jul. 2017, pp. 43–49.

[3] C. Szegedy *et al.*, "Intriguing properties of neural networks," *Int. Conf. Learn. Representations*, 2014. [Online]. Available: http://arxiv.org/abs/1312.6199

[4] S. Zheng, Y. Song, T. Leung, and I. Goodfellow, "Improving the robustness of deep neural networks via stability training," in *Proc. IEEE Conf. Comput. Vis. pattern Recognit.*, 2016, pp. 4480–4488 .

[5] M. Bojarski *et al.*, "End to end learning for self-driving cars," 2016, *arXiv:1604.07316*.

[6] K. D. Julian, J. Lopez, J. S. Brush, M. P. Owen, and M. J. Kochenderfer, "Policy compression for aircraft collision avoidance systems," in *Proc. 35th Digit. Avionics Syst. Conf.*, 2016, pp. 1–10.

[7] W. Xiang *et al.*, "Verification for machine learning, autonomy, and neural networks survey," 2018, *arXiv:1810.01989*.

[8] I. J. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and harnessing adversarial examples," *Int. Conf. Learn. Representations*, 2015. [Online]. Available: http://arxiv.org/abs/1412.6572

[9] A. Kurakin, I. Goodfellow, and S. Bengio, "Adversarial examples in the physical world," *ICLR Workshop*, 2017. [Online]. Available: https://arxiv.org/abs/1607.02533

[10] N. Papernot, P. McDaniel, S. Jha, M. Fredrikson, Z. B. Celik, and A. Swami, "The limitations of deep learning in adversarial settings," in *Proc. IEEE Eur. Symp. Secur. Privacy*, 2016, pp. 372–387.

[11] S.-M. Moosavi-Dezfooli, A. Fawzi, and P. Frossard, "Deepfool: A simple and accurate method to fool deep neural networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 2574–2582.

[12] O. Bastani, Y. Ioannou, L. Lampropoulos, D. Vytiniotis, A. Nori, and A. Criminisi, "Measuring neural net robustness with constraints," in *Proc. Adv. Neural Inf. Process. Syst.*, 2016, pp. 2613–2621.

[13] S. Dutta, S. Jha, S. Sankaranarayanan, and A. Tiwari, "Output range analysis for deep feedforward neural networks," in *NASA Formal Methods Symp.*, Berlin, Germany: Springer, 2018, pp. 121–138.

[14] A. Lomuscio and L. Maganti, "An approach to reachability analysis for feed-forward relu neural networks," 2017, *arXiv:1706.07351*.

[15] V. Tjeng, K. Xiao, and R. Tedrake, "Evaluating robustness of neural networks with mixed integer programming," *Int. Conf. Learn. Representations*, 2019. [Online]. Available: https://openreview.net/forum?id=HyGIdiRqtm

[16] E. Wong and J. Z. Kolter, "Provable defenses against adversarial examples via the convex outer adversarial polytope," *Int. Conf. Machine Learn.*, PMLR, 2018, pp. 5286–5295.

[17] K. Dvijotham, R. Stanforth, S. Gowal, T. A. Mann, and P. Kohli, "A Dual Approach to scalable verification of deep networks," in *UAI*, vol. 1, p. 2, 2018.

[18] H. Salman, G. Yang, H. Zhang, C.-J. Hsieh, and P. Zhang, "A convex relaxation barrier to tight robustness verification of neural networks," in *Adv. Neural Inf. Process. Syst.*, 2019, pp. 9832–9842.

[19] L. Pulina and A. Tacchella, "Challenging SMT solvers to verify neural networks," *AI Commun.*, vol. 25, no. 2, pp. 117–135, 2012.

[20] R. Ivanov, J. Weimer, R. Alur, G. J. Pappas, and I. Lee, "Verisig: Verifying safety properties of hybrid systems with neural network controllers," in *Proc. 22nd ACM Int. Conf. Hybrid Syst., Comput. Control* , 2019, pp. 169–178.

[21] W. Xiang, H.-D. Tran, and T. T. Johnson, "Output reachable set estimation and verification for multilayer neural networks," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 29, no. 11, pp. 5777–5783, 2018.

[22] M. Mirman, T. Gehr, and M. Vechev, "Differentiable abstract interpretation for provably robust neural networks," in *Int. Conf. Mach. Learn.*, 2018, pp. 3575–3583.

[23] T. Gehr, M. Mirman, D. Drachsler-Cohen, P. Tsankov, S. Chaudhuri, and M. Vechev, "Ai2: Safety and robustness certification of neural networks with abstract interpretation," in *Proc. IEEE Symp. Secur. Privacy*, 2018, pp. 3–18.

[24] T.-W. Weng *et al.*, "Towards fast computation of certified robustness for RELU networks," *ICML*, pp. 5273–5282, 2018. [Online]. Available: http://proceedings.mlr.press/v80/weng18a.html

[25] H. Zhang, T.-W. Weng, P.-Y. Chen, C.-J. Hsieh, and L. Daniel, "Efficient neural network robustness certification with general activation functions," in *Adv. Neural Inf. Process. Syst.*, 2018, pp. 4939–4948.

[26] M. Hein and M. Andriushchenko, "Formal guarantees on the robustness of a classifier against adversarial manipulation," in *Proc. Adv, Neural Info. Process. Syst.*, 2017, pp. 2266–2276.

[27] S. Wang, K. Pei, J. Whitehouse, J. Yang, and S. Jana, "Efficient formal safety analysis of neural networks," in *Adv. Neural Info. Process. Syst.*, 2018, pp. 6367–6377.
[28] S. Wang, K. Pei, J. Whitehouse, J. Yang, and S. Jana, "Formal security analysis of neural networks using symbolic intervals," in *27th USENIX Secur. Symp.*, (Baltimore, MD), Aug. 2018, pp. 1599–1614.
[29] V. Yakubovich, "S-procedure in nonlinear control theory," *Vestnick Leningrad Univ. Math.*, vol. 4, pp. 73–93, 1997.
[30] R. Ehlers, "Formal verification of piece-wise linear feed-forward neural networks," in *Int. Symp. Autom. Technol. Verification Anal.*, Berlin, Germany: Springer, 2017, pp. 269–286.
[31] X. Huang, M. Kwiatkowska, S. Wang, and M. Wu, "Safety verification of deep neural networks," in *Int. Conf. Comput. Aided Verification*, Berlin, Germany: Springer, 2017, pp. 3–29.
[32] G. Katz, C. Barrett, D. L. Dill, K. Julian, and M. J. Kochenderfer, "Reluplex: An efficient SMT solver for verifying deep neural networks," in *Int. Conf. Comput. Aided Verification*, Berlin, Germany: Springer, 2017, pp. 97–117.
[33] A. Raghunathan, J. Steinhardt, and P. Liang, "Certified defenses against adversarial examples," *Int. Conf. Learn. Representations*, 2018. [Online]. Available: https://openreview.net/forum?id=Bys4ob-Rb
[34] A. Raghunathan, J. Steinhardt, and P. S. Liang, "Semidefinite relaxations for certifying robustness to adversarial examples," in *Proc. Adv. Neural Info. Process. Syst.*, 2018, pp. 10900–10910.
[35] M. Fazlyab, M. Morari, and G. J. Pappas, "Probabilistic verification and reachability analysis of neural networks: Convex relaxations," in *Proc. IEEE Conf. Decis. Control (CDC)*, 2019, pp. 2726–2731, doi: 10.1109/CDC40024.2019.9029310.
[36] Y. Nesterov, *Introductory Lectures Convex Optimization: A. Basic Course*, vol. 87. New York, NY, USA: Springer, 2013.
[37] A. Megretski and A. Rantzer, "System analysis via integral quadratic constraints," *IEEE Trans. Autom. Control*, vol. 42, no. 6, pp. 819–830, Jun. 1997.
[38] H. K. Khalil and J. W. Grizzle, *Nonlinear Systems*, vol. 3. Upper Saddle River, NJ, USA: Prentice-Hall, 2002.
[39] G. Zames and P. Falb, "Stability conditions for systems with monotone and slope-restricted nonlinearities," *SIAM J. Control*, vol. 6, no. 1, pp. 89–108, 1968.
[40] F. D'amato, M. A. Rotea, A. Megretski, and U. Jönsson, "New results for analysis of systems with repeated nonlinearities," *Automatica*, vol. 37, no. 5, pp. 739–747, 2001.
[41] V. V. Kulkarni and M. G. Safonov, "All multipliers for repeated monotone nonlinearities," *IEEE Trans. Autom. Control*, vol. 47, no. 7, pp. 1209–1212, Jul. 2002.
[42] S. Gowal *et al.*, "On the effectiveness of interval bound propagation for training verifiably robust models," 2018, *arXiv:1810.12715*.
[43] C.-H. Cheng, G. Nührenberg, and H. Ruess, "Maximum resilience of artificial neural networks," *in Int. Symp. Autom. Technol. Verification Anal.*, Berlin, Germany: Springer, 2017, pp. 251–268.
[44] S. Boyd, L. El Ghaoui, E. Feron, and V. Balakrishnan, *Linear Matrix Inequalities in System and Control Theory*, vol. 15. Philadelphia, PA, USA: SIAM, 1994.
[45] A. Ben-Tal, L. El Ghaoui, and A. Nemirovski, *Robust Optimization*, vol. 28. Princeton, NJ, USA: Princeton Univ. Press, 2009.
[46] M. ApS, MOSEK Optimization Toolbox for MATLAB Manual. Version 8.1., 2017.
[47] I. CVX Research, "CVX: Matlab Software for Disciplined Convex Programming, Version 2.0.," 2012. [Online]. Available: http://cvxr.com/cvx

Mahyar Fazlyab (Student Member, IEEE) received the Ph.D. degree in electrical and systems engineering from the University of Pennsylvania, Philadelphia, PA, USA, in 2018.

He was a Postdoctoral Fellow with the ESE Department, UPenn from 2018 to 2020. He will be joining the Department of Electrical and Computer Engineering and Mathematical Institute for Data Science (MINDS), Johns Hopkins University, Baltimore, MD, USA, as an Assistant Professor, in 2021. His research interests are in the topics intersection of optimization, control, and machine learning.

Dr. Fazlyab won the Joseph and Rosaline Wolf Best Doctoral Dissertation Award in 2019, awarded by the Department of Electrical and Systems Engineering, University of Pennsylvania.

Manfred Morari (Fellow, IEEE) received the Diploma from ETH Zürich, Zürich, Switzerland, in 1974, and the Ph.D. degree from the University of Minnesota, Minneapolis, MN, USA, in 1977, both in chemical engineering.

He was a Professor and the Head of the Department of Information Technology and Electrical Engineering, ETH Zürich. He was the McCollumCorcoran Professor of chemical engineering and the Executive Officer of control and dynamical systems with the California Institute of Technology (Caltech), Pasadena, CA, USA. He was a Professor with the University of Wisconsin, Madison, WI, USA. He is currently with the University of Pennsylvania, Philadelphia, PA, USA. He has supervised more than 80 Ph.D. students.

Dr. Morari is a fellow of AIChE, IFAC, and the U.K. Royal Academy of Engineering. He is a member of the U.S. National Academy of Engineering. He was a recipient of numerous awards, including Eckman, Ragazzini, and Bellman Awards from the American Automatic Control Council (AACC); Colburn, Professional Progress, and CAST Division awards from the American Institute of Chemical Engineers (AIChE); Control Systems Award and Bode Lecture Prize from IEEE; Nyquist Lectureship and Oldenburger Medal from the American Society of Mechanical Engineers (ASME); and the IFAC High Impact Paper Award. He was the President of the European Control Association. He served on the technical advisory boards of several major corporations.

George J. Pappas (Fellow, IEEE) received the Ph.D. degree in electrical engineering and computer sciences from the University of California, Berkeley, CA, USA, in 1998.

He is currently the Joseph Moore Professor and Chair of the Department of Electrical and Systems Engineering, University of Pennsylvania, Philadelphia, PA, USA. He also holds a secondary appointment with the Department of Computer and Information Sciences and the Department of Mechanical Engineering and Applied Mechanics. He is a member of the GRASP Lab and the PRECISE Center. He had previously served as the Deputy Dean for Research with the School of Engineering and Applied Science. His research interests include control theory and, in particular, hybrid systems, embedded systems, cyberphysical systems, and hierarchical and distributed control systems, with applications to unmanned aerial vehicles, distributed robotics, green buildings, and biomolecular networks.

Dr. Pappas was a recipient of various awards, such as the Antonio Ruberti Young Researcher Prize, the George S. Axelby Award, the Hugo Schuck Best Paper Award, the George H. Heilmeier Award, the National Science Foundation PECASE Award, and numerous Best Student Paper awards at ACC, CDC, and ICCPS.