Lipschitz constant estimation for 1D convolutional neural networks

Patricia Pauli   PATRICIA.PAULI@IST.UNI-STUTTGART.DE
University of Stuttgart, Institute for Systems Theory and Automatic Control, 70569 Stuttgart, Germany

Dennis Gramlich   DENNIS.GRAMLICH@IC.RWTH-AACHEN.DE
RWTH Aachen, Chair of Intelligent Control Systems, 52074 Aachen, Germany

Frank Allgöwer   FRANK.ALLGOWER@IST.UNI-STUTTGART.DE
University of Stuttgart, Institute for Systems Theory and Automatic Control, 70569 Stuttgart, Germany

Editors: N. Matni, M. Morari, G. J. Pappas

Abstract

In this work, we propose a dissipativity-based method for Lipschitz constant estimation of 1D convolutional neural networks (CNNs). In particular, we analyze the dissipativity properties of convolutional, pooling, and fully connected layers making use of incremental quadratic constraints for nonlinear activation functions and pooling operations. The Lipschitz constant of the concatenation of these mappings is then estimated by solving a semidefinite program which we derive from dissipativity theory. To make our method as efficient as possible, we exploit the structure of convolutional layers by realizing these finite impulse response filters as causal dynamical systems in state space and carrying out the dissipativity analysis for the state space realizations. The examples we provide show that our Lipschitz bounds are advantageous in terms of accuracy and scalability.

Keywords: Convolutional neural networks, robustness, dissipativity, incremental quadratic constraints

1. Introduction

Convolutional neural networks (CNNs) achieve excellent results in practical applications, wherein convolutional layers detect meaningful features in small sections of an input signal while necessitating significantly fewer parameters than fully connected layers. CNNs have hence become the state of the art in many machine learning applications. While 2D CNNs are prevalently used in image and video processing, applications of 1D CNNs include classification of medical data, health monitoring, fault detection in electrical machines, and audio processing (Kiranyaz et al., 2021; Oord et al., 2016). Safety-critical applications especially require neural networks (NNs) to perform robustly and reliably, for which the Lipschitz constant has become a generally accepted robustness measure (Szegedy et al., 2013). Thus, recently, efforts have been made to finding accurate upper bounds on the Lipschitz constant of NNs. For example, Latorre et al. (2020) formulated a polynomial optimization problem for Lipschitz constant estimation and in particular, they study Lipschitz bounds of general CNNs with respect to the ℓ_∞ norm. Combettes and Pesquet (2020) proposed bounds viewing activation functions as averaged operators and Fazlyab et al. (2019) derived a semidefinite program (SDP) based on incremental quadratic constraints (QCs) to over-approximate the nonlinear activation functions. However, both latter methods have limited scalability to NNs of practically relevant scale. One reason for this limitation in (Fazlyab et al., 2019) is the use of a sparse linear matrix inequality (LMI). Newton and Papachristodoulou (2021); Xue et al. (2022) suggested to exploit the

underlying structure of this LMI, i. e., its chordal sparsity pattern, to break down the corresponding LMI into multiple smaller LMIs. Similarly, we consider properties of individual layers, also yielding a set of smaller layer-wise LMIs instead of one large and sparse one, which is advantageous in terms of computational tractability of the underyling SDP.

Previous SDP-based approaches (Fazlyab et al., 2019; Xue et al., 2022) are formulated for fully connected NNs, yet they can be applied to CNNs by transforming the convolutional layers to fully connected layers. This corresponds to studying the Toeplitz matrices of the convolutional layers (Pauli et al., 2022a; Aquino et al., 2022), that, unfortunately, are highly redundant, i. e., they have a high degree of sparsity and repeated entries, which causes a significant computational overhead. Our approach of viewing convolutional layers as dynamical systems reduces this redundancy by exploiting the structure of CNNs in SDP-based Lipschitz constant estimation, which thus leads to better scalability. This observation that convolutional layers are dynamical systems is non-trivial to exploit, since other components of CNNs, such as pooling layers, are not linear time-invariant systems. We handle this heterogeneous feed-forward interconnection of systems (layers) by carrying out a dissipativity analysis for each layer type separately. Subsequently, we derive the Lipschitz bound of the input-output mapping based on the dissipativity properties of each individual layer and the underlying feed-forward interconnection of the CNN, like it is oftentimes done in multi-agent control (Arcak et al., 2016) or has been suggested for the robustness analysis of NNs (Aquino et al., 2022). Other than previous works, our approach considers pooling layers in the Lipschitz constant estimation, by deriving incremental QCs for them. In this work, we focus on 1D CNNs to lay out our novel dissipativity-based concept for the derivation of an SDP for Lipschitz constant estimation. However, using a compact 2D systems respresenation of 2D CNNs (Gramlich et al., 2023), the framework can be extended to the more popular class of 2D CNNs. Note further that in the same way (Fazlyab et al., 2019) can be applied to CNNs, our approach can be applied to 2D CNNs, necessitating to represent 2D convolutions as 1D convolutions.

Our main contributions can be summarized as follows. We provide a first result on how to exploit CNN structures in SDP-based Lipschitz constant estimation. To this end, we introduce a dissipativity-based method to derive layer-wise LMIs. Both our compact description of convolutions and the layer-wise LMIs improve the computational tractability over previous approaches. In Section 2, we formally state the problem and in Section 3, we outline the dissipativity-based approach for Lipschitz constant estimation for 1D CNNs. In doing so, we introduce the description of convolutional layers in state space, derive incremental QCs for the nonlinear activation functions and pooling layers, and subsequently, we establish an SDP to determine an upper bound on the Lipschitz constant for a given 1D CNN. Finally, in Section 4 we compare our approach to Fazlyab et al. (2019) in terms of computation time and accuracy and, in Section 5, we conclude the paper.

Notation: By $\mathbb{S}^n$ ($\mathbb{S}^n_+$), we denote the set of n-dimensional symmetric (positive definite) matrices. By $\mathbb{D}^n$ ($\mathbb{D}^n_+$), we denote the set of n-dimensional (positive definite) diagonal matrices, and by $\mathbb{N}_+$ the natural numbers without zero. $\mathcal{I}$ is a set of indices with elements $i \in \mathbb{N}_+$, and $|\mathcal{I}|$ gives the number of elements in the index set $\mathcal{I}$. The subscripts a, b in ν_a and ν_b label two instances of ν.

2. Problem statement

We consider a 1D CNN as shown in Fig. 1, consisting of convolutional layers $\mathcal{C}_i : \mathbb{R}^{c_{i-1} \times N_{i-1}} \to \mathbb{R}^{c_i \times N_i}$ with indices $i \in \mathcal{I}_C$, pooling layers $\mathcal{P}_i : \mathbb{R}^{c_{i-1} \times N_{i-1}} \to \mathbb{R}^{c_i \times N_i}$ with indices $i \in \mathcal{I}_P$, and fully connected layers $\mathcal{L}_i : \mathbb{R}^{n_{i-1}} \to \mathbb{R}^{n_i}$ with indices $i \in \mathcal{I}_F$, adding up to a total number of
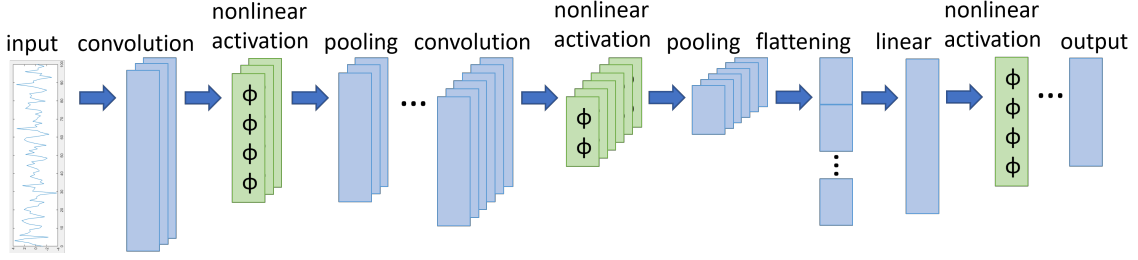
Figure 1: 1D CNN structure of (1) including convolutions, pooling and linear layers as well as nonlinear activation functions and a flattening layer.

$l = |\mathcal{I}_C| + |\mathcal{I}_P| + |\mathcal{I}_F|$ layers. Here, c_{i-1} and c_i denote the input and output channel sizes, N_{i-1} and N_i the input and output dimensions in pooling and covolutional layers, and n_{i-1} and n_i the input and output dimensions in fully connected layers, respectively. We study the CNN as a concatenation of the individual layers

$$\mathrm{CNN}_\theta = \mathcal{L}_l \circ \ldots \circ \mathcal{L}_{p+1} \circ \mathcal{F} \circ \mathcal{P}_p \circ \mathcal{C}_{p-1} \circ \ldots \circ \mathcal{P}_2 \circ \mathcal{C}_1, \qquad (1)$$

wherein the only formal restriction is the separation into two parts: (i) the part containing fully connected layers $\mathcal{I}_F = \{p+1, \ldots, l\}$ and (ii) the part consisting of convolutional and pooling layers $\mathcal{I}_C \cup \mathcal{I}_P = \{1, \ldots, p\}$ where p is the index of the last layer of part (ii), which can either be a convolutional or a pooling layer. Further, the transition between the two parts requires a flattening operation $\mathcal{F} : \mathbb{R}^{c_p \times N_p} \to \mathbb{R}^{n_p}$ of the output of the p-th layer, where $n_p = c_p N_p$.

To enable an efficient description of the layers constituting the CNN (1), we denote the signals between those layers by w^i, i.e., $w^i = \mathcal{C}_i(w^{i-1})$, $w^i = \mathcal{L}_i(w^{i-1})$, or $w^i = \mathcal{P}_i(w^{i-1})$ depending on the type of the i-th layer. Note that a signal w^i can either be an element of $\mathbb{R}^{c_i \times N_i}$ in the case of the convolutional part (ii) of the network, or an element of $\mathbb{R}^{n_i}$ in the case of the fully connected part (i) of the network. With this notation, we define each layer ($\mathcal{C}_i/\mathcal{P}_i/\mathcal{L}_i$) through the way it acts on its input w^{i-1}. A *convolutional* layer

$$\mathcal{C}_i : w_k^i = \phi_i \left(b_i + \sum_{j=0}^{\ell_i - 1} K_j^i w_{k-j}^{i-1} \right), \quad k = 0, \ldots, N_i - 1, \ \forall i \in \mathcal{I}_C, \qquad (2)$$

with convolution kernel $K_j^i \in \mathbb{R}^{c_i \times c_{i-1}}$, $j = 0, \ldots, \ell_i - 1$, kernel size ℓ_i, and bias $b_i \in \mathbb{R}^{c_i}$, as the first instance, performs a convolution on $w^{i-1} \in \mathbb{R}^{c_{i-1} \times N_{i-1}}$ and subsequently, applies an element-wise nonlinear activation function $\phi_i : \mathbb{R}^{c_i} \to \mathbb{R}^{c_i}$ to obtain the output $w^i \in \mathbb{R}^{c_i \times N_i}$, wherein we denote all entries along the channel dimension at propagation/time step k by $w_k^i \in \mathbb{R}^{c_i}$. Note that in (2) we define w_{k-j}^{i-1} as zero whenever $k - j \leq 0$, i.e., we apply zero padding.

In order to downsample a signal w^{i-1} convolutional layers are potentially followed by *pooling* layers. To this end, we consider average pooling layers and maximum pooling layers

$$\mathcal{P}_i^{\mathrm{av}} : w_k^i = \frac{1}{\ell_i} \sum_{j=1}^{\ell_i} w_{\ell_i(k-1)+j}^{i-1}, \quad \mathcal{P}_i^{\mathrm{max}} : w_k^i = \max_{j=1,\ldots,\ell_i} w_{\ell_i(k-1)+j}^{i-1}, \qquad \begin{matrix} k = 0, \ldots, N_i - 1, \\ \forall i \in \mathcal{I}_P, \end{matrix} \quad (3)$$

3

where ℓ_i is the size of the pooling kernel, which both are standard components in CNNs. Throughout this paper, we only consider the case that stride and kernel size are both ℓ_i, such that the output dimension is $N_i = N_{i-1}/\ell_i$, which requires that ℓ_i divides N_{i-1}. However, the approach can be extended to the general case of stride and pooling kernel size being different.

Finally, a CNN typically holds *fully connected* layers, which we define as mappings

$$\mathcal{L}_i : w^i = \phi_i(W_i w^{i-1} + b_i) \quad \forall i \in \mathcal{I}_F \backslash \{l\}, \quad \mathcal{L}_l : w^l = W_l w^{l-1} + b_l \tag{4}$$

with weights $W_i \in \mathbb{R}^{n_i \times n_{i-1}}$, biases $b_i \in \mathbb{R}^{n_i}$ and activation functions $\phi_i : \mathbb{R}^{n_i} \to \mathbb{R}^{n_i}$.

The CNN $f_\theta(w^0) = w^l$ is thus characterized by $\theta = \{\{K^i, b_i\}_{i \in \mathcal{I}_C}, \{W_i, b_i\}_{i \in \mathcal{I}_F}\}$ and the chosen pooling layers and nonlinear activation functions. The aim of this paper is to find a Lipschitz certificate γ for 1D CNNs.

Problem 1 *For a given 1D convolutional neural network* CNN_θ *with fixed parameter* θ, *find an upper bound on the Lipschitz constant, i. e., find a value* $\gamma > 0$ *such that*

$$\| \mathrm{CNN}_\theta(w_a^0) - \mathrm{CNN}_\theta(w_b^0)\|_2 \leq \gamma \|w_a^0 - w_b^0\|_2 \qquad \forall w_a^0, w_b^0 \in \mathbb{R}^{N_0}. \tag{5}$$

3. Dissipativity-based Lipschitz constant estimation for 1D CNNs

To solve Problem 1, we establish dissipativity properties for the individual layers, i. e., we find Q_i, S_i, R_i that satisfy the incremental quadratic constraints

$$s(\Delta w^{i-1}, \Delta w^i) := \begin{bmatrix} \underline{w}_a^i - \underline{w}_b^i \\ \underline{w}_a^{i-1} - \underline{w}_b^{i-1} \end{bmatrix}^\top \begin{bmatrix} Q_i & S_i \\ S_i^\top & R_i \end{bmatrix} \begin{bmatrix} \underline{w}_a^i - \underline{w}_b^i \\ \underline{w}_a^{i-1} - \underline{w}_b^{i-1} \end{bmatrix} \geq 0 \ \ \forall \underline{w}_a^{i-1}, \underline{w}_b^{i-1} \in \mathbb{R}^{n_{i-1}} \tag{6}$$

for each $\mathcal{C}_i$-, $\mathcal{L}_i$-, and $\mathcal{P}_i$-layer, where $\underline{w}^{i-1}$ denotes a column-wise stacked vector of dimension $n_i = c_i N_i$ if it comes from $\mathbb{R}^{c_i \times N_i}$ and corresponds to w^{i-1} if it comes from $\mathbb{R}^{n_i}$, and where $\Delta w^i = w_a^i - w_b^i$. To further estimate the Lipschitz constant of the input-output mapping of the 1D CNN (1), we connect the different layers through their dissipativity properties, considering the CNN's cascaded interconnection structure. To account for this structure, we introduce the block-tridiagonal coupling matrix (Kanellopoulos et al., 2019)

$$H := \begin{bmatrix} \gamma^2 I - R_1 & -S_1^\top & 0 & \dots & 0 \\ -S_1 & -Q_1 - R_2 & -S_2^\top & \ddots & \vdots \\ 0 & -S_2 & -Q_2 - R_3 & \ddots & 0 \\ \vdots & \ddots & \ddots & \ddots & -S_l^\top \\ 0 & \dots & 0 & -S_l & -Q_l - I \end{bmatrix} \succeq 0, \tag{7}$$

where the layer indices $\{1, \dots, l\} = \mathcal{I}_C \cup \mathcal{I}_P \cup \mathcal{I}_F$ count through all $|\mathcal{I}_C|$ convolutional layers, $|\mathcal{I}_P|$ pooling layers, and $|\mathcal{I}_F|$ fully connected layers.

Theorem 1 *Let* CNN_θ *and* $\gamma > 0$ *be given. If there exist* $Q_i \in \mathbb{S}^{n_i}$, $S_i \in \mathbb{R}^{n_i \times n_{i-1}}$, $R_i \in \mathbb{S}^{n_{i-1}}$ *such that* (6) *is satisfied for all* $i = 1, \dots, l$ *layers and if in addition* (7) *holds, then the* CNN_θ *is* γ-*Lipschitz continuous.*

Proof We left and right multiply (7) by $\begin{bmatrix} \Delta\underline{w}^{0\top} & \Delta\underline{w}^{1\top} & \ldots & \Delta\underline{w}^{l\top} \end{bmatrix}$ and its transpose, respectively, and obtain

$$\gamma^2 \Delta\underline{w}^{0\top} \Delta\underline{w}^0 \underbrace{-s(\Delta w^0, \Delta w^1)}_{\leq 0} - \ldots \underbrace{-s(\Delta w^{l-1}, \Delta w^l)}_{\leq 0} - \Delta\underline{w}^{l\top} \Delta\underline{w}^l \geq 0,$$

The supply functions $s(\Delta w^{i-1}, \Delta w^i)$ are nonnegative by (6) which implies $\gamma^2 \|\Delta w^0\|_2^2 - \|\Delta w^l\|_2^2 \geq 0$, i.e., γ-Lipschitz continuity of CNN_θ. ∎

In the following Sections 3.1 to 3.3, we discuss incremental QCs for the different layer types and in Section 3.4, we establish an SDP that renders Theorem 1 computational, providing an accurate upper bound on the Lipschitz constant of a given 1D CNN.

3.1. Incremental quadratic constraints for convolutional layers

Previous approaches to analyze the Lipschitz constant of CNNs using LMIs and incremental QCs rely on rewriting convolutional layers as sparse fully connected layers using Toeplitz matrices (Pauli et al., 2022a; Aquino et al., 2022). In the present section, we introduce a compact and non-sparse state space representation for convolutional layers which significantly reduces the size of the convolutional layer description, yielding a scalable SDP-based approach for Lipschitz constant estimation.

A discrete-time state space representation of the finite impulse response (FIR) filter (2) is, for example, given by

$$x_{k+1}^i = A_i x_k^i + B_i w_k^{i-1}, \quad y_k^i = C_i x_k^i + D_i w_k^{i-1} + b_i, \quad w_k^i = \phi(y_k^i) \quad \forall i \in \mathcal{I}_C$$

with state $x_k \in \mathbb{R}^{n_{x_i}}$ and state dimension $n_{x_i} = (\ell_i - 1) c_{i-1}$. Here, the matrices A_i, B_i, C_i, D_i are

$$A_i = \begin{bmatrix} 0 & I \\ 0 & 0 \end{bmatrix}, \ B_i = \begin{bmatrix} 0 \\ I \end{bmatrix}, \ C_i = \begin{bmatrix} K_{\ell_i-1}^i & \ldots & K_1^i \end{bmatrix}, \ D_i = K_0^i$$

or similarity transformations $(EAE^{-1}, EB, CE^{-1}, D)$ thereof for some invertible $E \in \mathbb{R}^{n_{x_i} \times n_{x_i}}$. Note that the formulation of a state space representation of (2) requires causality. In standard CNN literature, 1D convolutions are given by an acausal FIR filter $y_k^i = b_i + \sum_{j=-\tilde{\ell}_i}^{\tilde{\ell}_i} K_j^i w_{k-j}^{i-1}$. A change of indices, however, yields (2). One clear advantage of representing the convolutional layer compactly in state space rather than by a Toeplitz matrix is that it is independent of the signal's input dimension N_{i-1} and scales with the channel sizes c_{i-1} and c_i and the kernel size ℓ_i of the convolution filter instead. This means that the computational expense of analyzing a convolutional layer is the same for inputs of arbitrary input dimension N_{i-1}.

In addition to the convolution operation, a convolutional layer (2) consists of a nonlinear activation function. Commonly used activation functions, such as ReLU, $\tanh$ and sigmoid, are slope-restricted which can be captured by an incremental QC as follows.

Lemma 2 (Fazlyab et al. (2019); Pauli et al. (2022c)) *Let* $\varphi : \mathbb{R} \to \mathbb{R}$ *be slope-restricted on* $[0, 1]$. *Then, for any* $\Lambda_i \in \mathbb{D}_+^{c_i}$ *the vector-valued function* $\phi(\nu^i) = \begin{bmatrix} \varphi(\nu_1^i) & \ldots & \varphi(\nu_{c_i}^i) \end{bmatrix}^\top : \mathbb{R}^{c_i} \to \mathbb{R}^{c_i}$ *satisfies*

$$\begin{bmatrix} \phi(\nu_a^i) - \phi(\nu_b^i) \\ \nu_a^i - \nu_b^i \end{bmatrix}^\top \begin{bmatrix} -2\Lambda_i & \Lambda_i \\ \Lambda_i & 0 \end{bmatrix} \begin{bmatrix} \phi(\nu_a^i) - \phi(\nu_b^i) \\ \nu_a^i - \nu_b^i \end{bmatrix} \geq 0 \quad \forall \nu_a^i, \nu_b^i \in \mathbb{R}^{c_i}. \tag{8}$$

5

By virtue of Lemma 2, we replace constraints $w_a^i - w_b^i = \phi(\nu_a^i) - \phi(\nu_b^i)$ with the inequality (8) which relaxes the nonlinear constraint. This kind of approach is common practice in robust control (Scherer and Weiland, 2000), where control engineers routinely interpret nonlinearities, such as slope-restricted activation functions, as uncertainties and use QCs to perform a convex relaxation of the constraints imposed by these nonlinearities to render a problem computationally tractable.

Using these QCs and the state space realization of the convolution, we introduce the concept of layer-wise incremental dissipativity for an individual convolutional layer (2). We denote the incremental difference of the unrolled signal by $\Delta w^i = w_a^i - w_b^i$ and the incremental difference at propagation step k by $\Delta_w^i = w_{a,k}^i - w_{b,k}^i$.

Definition 3 (Incremental dissipativity of convolutional layers) *We call the i-th convolutional layer $\mathcal{C}_i$ incrementally (Q, S, R)-dissipative if any input-output pair $\{\Delta w^{i-1}, \Delta w^i\}$ stemming from (2) satisfies*

$$s(\Delta w^{i-1}, \Delta w^i) = \sum_{k=0}^{N_i-1} \Delta_w^{i\top} \widetilde{Q}_i \Delta_w^i + 2\Delta_w^{i\top} \widetilde{S}_i \Delta_w^{i-1} + \Delta_w^{i-1\top} \widetilde{R}_i \Delta_w^{i-1} \geq 0 \quad \forall N_i \in \mathbb{N}_+ \quad (9)$$

with some matrices $\widetilde{Q}_i \in \mathbb{S}^{c_i}$, $\widetilde{S}_i \in \mathbb{R}^{c_i \times c_{i-1}}$, $\widetilde{R}_i \in \mathbb{S}^{c_{i-1}}$.

Such incremental dissipativity certificates can capture, e.g., Lipschitz continuity ($\widetilde{Q}_i = -I$, $\widetilde{R}_i = \gamma^2 I$, $\widetilde{S}_i = 0$), slope restriction ($\widetilde{Q}_i = 0$, $\widetilde{R}_i = 2\Lambda_i$, $\widetilde{S}_i = -\Lambda_i$) or other behavioral properties such as incremental passivity ($\widetilde{Q}_i = \widetilde{R}_i = 0$, $\widetilde{S}_i = I$) (Hill and Moylan, 1976). Our goal is to find some $\widetilde{Q}_i$, $\widetilde{S}_i$, and $\widetilde{R}_i$ such that the i-th convolutional layer satisfies (9). The following lemma provides us with a sufficient condition for dissipativity of convolutional layers $\mathcal{C}_i$.

Lemma 4 *If for some matrices $\widetilde{Q}_i \in \mathbb{S}^{c_i}$, $\widetilde{S}_i \in \mathbb{R}^{c_i \times c_{i-1}}$, $\widetilde{R}_i \in \mathbb{S}^{c_{i-1}}$ there exist $P_i \in \mathbb{S}_+^{n_{x_i}}$, $\Lambda_i \in \mathbb{D}_+^{c_i}$ such that*

$$\begin{bmatrix} P_i - A_i^\top P_i A_i & -A_i^\top P_i B_i & -C_i^\top \Lambda_i \\ -B_i^\top P_i A_i & \widetilde{R}_i - B_i^\top P_i B_i & \widetilde{S}_i^\top - D_i^\top \Lambda_i \\ -\Lambda_i C_i & \widetilde{S}_i - \Lambda_i D_i & 2\Lambda_i + \widetilde{Q}_i \end{bmatrix} \succeq 0 \quad (10)$$

holds, then the i-th convolutional layer $\mathcal{C}_i$ is (Q, S, R)-dissipative.

The proof follows along the lines of typical arguments used in dissipativity analysis (Goodwin and Sin, 2014) and robust control (Scherer and Weiland, 2000) and is for completeness included in Appendix 1 in Pauli et al. (2022b).

A convolutional layer that satisfies (9) also satisfies (6), choosing the parameterization

$$Q_i = \mathrm{blkdiag}(\widetilde{Q}_i, \ldots, \widetilde{Q}_i), \ S_i = \mathrm{blkdiag}(\widetilde{S}_i, \ldots, \widetilde{S}_i), \ R_i = \mathrm{blkdiag}(\widetilde{R}_i, \ldots, \widetilde{R}_i),$$

where the matrices $\widetilde{Q}_i$, $\widetilde{S}_i$, $\widetilde{R}_i$ appear N_i times as entries of block-diagonal matrices, which means that (10) ensures the incremental QC (6) for convolutional layers $\mathcal{C}_i$.

3.2. Incremental quadratic constraints for fully connected layers

Similarly, we next consider fully connected layers (4) of the CNN individually and address incremental dissipativity for these layers.

Definition 5 (Incremental dissipativity of fully connected layers) *We call the i-th fully connected layer $\mathcal{L}_i$ incrementally (Q, S, R)-dissipative if any input-output pair $\{\Delta w^{i-1}, \Delta w^i\}$ stemming from (4) satisfies*

$$s(\Delta w^{i-1}, \Delta w^i) = \Delta w^{i^\top} Q_i \Delta w^i + 2\Delta w^{i^\top} S_i \Delta w^{i-1} + \Delta w^{i-1^\top} R_i \Delta w^{i-1} \geq 0 \qquad (11)$$

with some matrices $Q_i \in \mathbb{S}^{n_i}$, $S_i \in \mathbb{R}^{n_i \times n_{i-1}}$, $R_i \in \mathbb{S}^{n_{i-1}}$.

Lemma 6 *If for some matrices $Q_i \in \mathbb{S}^{n_i}$, $S_i \in \mathbb{R}^{n_i \times n_{i-1}}$, $R_i \in \mathbb{S}^{n_{i-1}}$ there exist $\Lambda_i \in \mathbb{D}_+^{n_i}$ such that*

$$\begin{bmatrix} R_i & S_i^\top - W_i^\top \Lambda_i \\ S_i - \Lambda_i W_i & 2\Lambda_i + Q_i \end{bmatrix} \succeq 0 \qquad (12)$$

holds, then the i-th fully connected layer $\mathcal{L}_i$ is (Q, S, R)-dissipative.

The proof follows as a special case of the proof of Lemma 4 with $A_i = 0$, $B_i = 0$, $C_i = 0$, and $D_i = W_i$, as well as $c_i = n_i$, $N_i = 1$ and, accordingly, the LMI (12) ensures (Q, S, R)-dissipativity (11) / the incremental QC (6) for fully connected layers (4).

3.3. Incremental quadratic constraints for pooling layers

Besides the relaxation of activation functions (Fazlyab et al., 2019), we newly introduce incremental QCs to handle maximum and average pooling layers. These pooling layers downsample a signal channel-wise, i. e., multiple successive time/propagation steps of the input signal are mapped to one time/propagation step of the output signal. Such operations cannot easily be captured in a state space formulation which is why we represent them via incremental QCs instead. Alternatively, linear pooling layers, such as average pooling, can be unrolled to fully-connected layers, again yielding a sparse and redundant formulation thereof (Fazlyab et al., 2019; Pauli et al., 2022a). The use of incremental QCs suggested in this paper is not only more efficient but also allows to account for nonlinear maximum pooling layers, which is not possible in (Fazlyab et al., 2019).

Average pooling $\mathcal{P}_i^{\text{av}}$ as defined in (3) is applied channel-wise such that for the j-th channel, we can define

$$w_{jk}^i = \frac{1}{\ell_i} \sum_{s=1}^{\ell_i} w_{j,\ell_i(k-1)+s}^{i-1} =: \text{Pool}^{\text{av}}(v_{jk}^i), \quad k = 0, \dots, N_i - 1,$$

where $v_{jk}^i = \begin{bmatrix} w_{j,\ell_i(k-1)+1}^{i-1} & \cdots & w_{j,\ell_i k}^{i-1} \end{bmatrix}^\top \in \mathbb{R}^{\ell_i}$.

Proposition 7 *The average pooling operation* Pool^{av} *is Lipschitz continuous with* $\mu_i = \frac{1}{\sqrt{\ell_i}}$, *i. e.,*

$$|w_{a,jk}^i - w_{b,jk}^i|^2 \leq \mu_i^2 \|v_{a,jk}^i - v_{b,jk}^i\|^2 \quad j = 1, \dots, c_i, \ k = 1, \dots, N_i \quad \forall v_{a,jk}^i, v_{b,jk}^i \in \mathbb{R}^{\ell_i}. \ (13)$$

The proof can be found in Appendix 2 in Pauli et al. (2022b). While (13) only considers Lipschitz continuity of one channel, the following incremental QC holds for all channels, for which we define the vertically flattened vector

$$\underline{v}_k^i = \begin{bmatrix} w_{1,\ell_i(k-1)+1}^{i-1} & \cdots & w_{c_i,\ell_i(k-1)+1}^{i-1} & w_{1,\ell_i(k-1)+2}^{i-1} & \cdots & w_{c_i,\ell_i(k-1)+2}^{i-1} & \cdots & w_{1,\ell_i k}^{i-1} & \cdots & w_{c_i,\ell_i k}^{i-1} \end{bmatrix}^\top,$$

that collects entries from all $c_i = c_{i-1}$ channels and from all ℓ_i propagation steps (of the $(i-1)$-th signal) that are combined in the pooling operation corresponding to propagation step k (of the i-th signal).

Lemma 8 *For all $T_i \in \mathbb{S}_+^{c_i}$, the average pooling operation $\mathcal{P}_i^{\mathrm{av}}$ fulfills*

$$\begin{bmatrix} w_{a,k}^i - w_{b,k}^i \\ \underline{v}_{a,k}^i - \underline{v}_{b,k}^i \end{bmatrix}^\top \left[\begin{array}{c|ccc} -T_i & 0 & \ldots & 0 \\ \hline 0 & \mu_i^2 T_i & & \\ \vdots & & \ddots & 0 \\ 0 & & 0 & \mu_i^2 T_i \end{array} \right] \begin{bmatrix} w_{a,k}^i - w_{b,k}^i \\ \underline{v}_{a,k}^i - \underline{v}_{b,k}^i \end{bmatrix} \geq 0, \quad \begin{array}{c} k = 1, \ldots, N_i, \\ \forall \underline{v}_{a,k}^i, \underline{v}_{b,k}^i \in \mathbb{R}^{\ell_i c_i}. \end{array} \quad (14)$$

This implies that all $i \in \mathcal{I}_P^{\mathrm{av}}$ average pooling layers are (Q, S, R)-dissipative, i. e., they satisfy (6) *with $Q_i = -\mathrm{blkdiag}(T_i, \ldots, T_i) \in \mathbb{S}^{n_i}$, $S_i = 0$, $R_i = \mu_i^2 \,\mathrm{blkdiag}(T_i, \ldots, T_i) \in \mathbb{S}^{n_{i-1}}$.*

Proof The positive semidefinite matrix T_i has a factorization $T_i = L_i^\top L_i$, $L_i \in \mathbb{R}^{c_i \times c_i}$, which we use to define $\tilde{\underline{v}}_k^i := \mathrm{blkdiag}(L_i, \ldots, L_i)\underline{v}_k^i$, $k = 1, \ldots, N_i$. As average pooling is a linear operator, L_i can be pulled out of the pooling operation, i. e., $\tilde{w}_k^i = L_i w_k^i$. Hence, (14) is equivalent to

$$\begin{bmatrix} \tilde{w}_{a,k}^i - \tilde{w}_{b,k}^i \\ \tilde{\underline{v}}_{a,k}^i - \tilde{\underline{v}}_{b,k}^i \end{bmatrix}^\top \begin{bmatrix} -I_{c_i} & 0 \\ 0 & \mu_i^2 I_{\ell_i c_i} \end{bmatrix} \begin{bmatrix} \tilde{w}_{a,k}^i - \tilde{w}_{b,k}^i \\ \tilde{\underline{v}}_{a,k}^i - \tilde{\underline{v}}_{b,k}^i \end{bmatrix} \geq 0, \quad k = 1, \ldots, N_i, \quad \forall \tilde{\underline{v}}_{a,k}^i, \tilde{\underline{v}}_{b,k}^i \in \mathbb{R}^{\ell_i c_i},$$

which is the channel-wise stacked up version of the Lipschitz condition (13). This can easily be seen by reordering the entries of the flattened vectors $\tilde{\underline{v}}_{a,k}^i$ and $\tilde{\underline{v}}_{b,k}^i$. Stacking up the incremental QC (14) for all $k = 1, \ldots, N_i$ propagation steps yields a special case of (6). ∎

An alternative commonly used pooling operation is *maximum pooling $\mathcal{P}_i^{\mathrm{max}}$* as defined in (3). This nonlinear operation is 1-Lipschitz in case the kernel size and the stride coincide.

Lemma 9 *For all $\Sigma_i \in \mathbb{D}_+^{c_i}$, the maximum pooling operation $\mathcal{P}_i^{\mathrm{max}}$ fulfills*

$$\begin{bmatrix} w_{a,k}^i - w_{b,k}^i \\ \underline{v}_{a,k}^i - \underline{v}_{b,k}^i \end{bmatrix}^\top \left[\begin{array}{c|ccc} -\Sigma_i & 0 & \ldots & 0 \\ \hline 0 & \Sigma_i & & \\ \vdots & & \ddots & \\ 0 & & & \Sigma_i \end{array} \right] \begin{bmatrix} w_{a,k}^i - w_{b,k}^i \\ \underline{v}_{a,k}^i - \underline{v}_{b,k}^i \end{bmatrix} \geq 0, \quad \begin{array}{c} k = 1, \ldots, N_i, \\ \forall \underline{v}_{a,k}^i, \underline{v}_{b,k}^i \in \mathbb{R}^{\ell_i c_i}. \end{array} \quad (15)$$

This implies that all $i \in \mathcal{I}_P^{\mathrm{max}}$ maximum pooling layers are (Q, S, R)-dissipative, i. e., they satisfy (6) *with $Q_i = -\mathrm{blkdiag}(\Sigma_i, \ldots, \Sigma_i) \in \mathbb{D}^{n_i}$, $S_i = 0$, $R_i = \mathrm{blkdiag}(\Sigma_i, \ldots, \Sigma_i) \in \mathbb{D}^{n_{i-1}}$.*

Proof Condition (15) is the stacked up version of the incremental QCs

$$\sigma_i |w_{a,jk}^i - w_{b,jk}^i|^2 \leq \sigma_i \|v_{a,jk}^i - v_{b,jk}^i\|^2, \quad j = 1, \ldots, c_i, \ k = 1, \ldots, N_i, \quad \forall v_{a,jk}^i, v_{b,jk}^i \in \mathbb{R}^{\ell_i},$$

which holds true for all nonnegative scalars $\sigma_i \geq 0$ by 1-Lipschitz continuity of the maximum pooling operation. The multiplier is then $\Sigma_i = \mathrm{diag}(\sigma_1, \ldots, \sigma_{c_i})$. Accordingly to the proof of Lemma 8, we can stack up (15) for all $k = 1, \ldots, N_i$ propagation steps to obtain a special case of (6). ∎

The incremental QC (14) describing average pooling allows full matrices T_i as multipliers, whereas maximum pooling requires Σ_i in (15) to be diagonal. This result corresponds to the main results in (Kulkarni and Safonov, 2002) who address that only linear operations maintain incremental positivity. Note that Fazlyab et al. (2019) wrongly proposed full matrices Λ_i as multipliers in (8) which was later corrected by Pauli et al. (2022c).

3.4. Semidefinite program for Lipschitz constant estimation

In this section, we propose an SDP to certify an upper bound on the Lipschitz constant for a given 1D CNN, that renders Theorem 1 computational.

Pooling layers are incrementally (Q, S, R)-dissipative by design according to Lemma 8 and Lemma 9, respectively, and in addition Lemma 4 and Lemma 6 provide us LMIs to enforce dissipativity onto convolutional and fully connected layers such that all $\mathcal{C}_i$-, $\mathcal{P}_i$-, and $\mathcal{F}_i$-layers in (1) are incrementally (Q, S, R)-dissipative. Our goal is to determine an accurate Lipschitz bound of the input-output mapping defined by a given 1D CNN, for which the dissipativity properties of the individual layers are a utility. Hence, in all $i \in \mathcal{I}_C \cup \mathcal{I}_F$, we consider Q_i, S_i, and R_i as decision variables that provide us degrees of freedom in the optimization. To reduce the number of decision variables, which is computationally favorable, we parameterize Q_i, S_i, and R_i in such a way that H as defined in (7) is equal to zero. This yields $R_1 = Q_0 = \gamma^2 I$, $S_i = 0$, and $R_i = -Q_{i-1}$, $i = 1, \ldots, l$, $Q_l = -I$. Further, we note that in pooling layers Q_i and R_i are both parameterized by the same matrix, such that we use this information to specify the structure of $\widetilde{Q}_{i-1}$ of the preceding convolutional layer, which yields Corollary 10.

Corollary 10 *Let* CNN_θ *and* $\gamma > 0$ *be given and let all activation functions be slope-restricted on* $[0, 1]$. *Further, we choose* $\widetilde{Q}_0 = -\tilde{\gamma}^2 I$, *where* $\gamma = \tilde{\gamma} \prod_{s \in \mathcal{I}_P^{\mathrm{av}}} \mu_s$. *If there exist*

(i) $\widetilde{Q}_i \in \mathbb{S}^{c_i}$ *(*$\widetilde{Q}_i \in \mathbb{D}^{c_i}$ *if a convolutional layer is followed by a maximum pooling layer),* $P_i \in \mathbb{S}_+^{n_{x_i}}$, *and* $\Lambda_i \in \mathbb{D}_+^{c_i}$ *such that*

$$\begin{bmatrix} P_i - A_i^\top P_i A_i & -A_i^\top P_i B_i & -C_i^\top \Lambda_i \\ -B_i^\top P_i A_i & -\widetilde{Q}_{i-1} - B_i^\top P_i B_i & -D_i^\top \Lambda_i \\ -\Lambda_i C_i & -\Lambda_i D_i & 2\Lambda_i + \widetilde{Q}_i \end{bmatrix} \succeq 0 \quad \forall i \in \mathcal{I}_C, \qquad (16)$$

(ii) $Q_i \in \mathbb{S}^{n_i}$ *and* $\Lambda_i \in \mathbb{D}_+^{n_i}$ *such that*

$$\begin{bmatrix} -Q_{i-1} & -W_i^\top \Lambda_i \\ -\Lambda_i W_i & 2\Lambda_i + Q_i \end{bmatrix} \succeq 0 \quad \forall i = \mathcal{I}_F \backslash \{l\}, \qquad \begin{bmatrix} -Q_{l-1} & -W_l^\top \\ -W_l & I \end{bmatrix} \succeq 0, \qquad (17)$$

then the CNN_θ *is* γ-*Lipschitz continuous.*

A complete proof is given in Appendix 3 in Pauli et al. (2022b). For a given CNN, we can find an upper bound on the Lipschitz constant by solving the semidefinite program

$$\min_{\gamma^2, Q, \Lambda, P} \gamma^2 \quad \mathrm{s.\,t.} \quad (16), (17), \qquad (18)$$

where $\Lambda = \{\Lambda_i\}_{i \in \mathcal{I}_C \cup \mathcal{I}_F \backslash \{l\}}$, $Q = \{Q_i\}_{i \in \mathcal{I}_C \cup \mathcal{I}_F \backslash \{l\}}$, $P = \{P_i\}_{i \in \mathcal{I}_C}$.

Remark 11 *The matrices* Q_i *carry information on the layer-wise worst-case gain and worst-case direction of the corresponding input to the layer. In case a maximum pooling layer exists, we lose the information of the worst-case direction through the parameterization of* Q_i *as a diagonal matrix.*

Remark 12 *For fully connected NNs, the set of layer-wise LMIs (17) is equivalent to the LMI in Fazlyab et al. (2019) by chordal sparsity (Xue et al., 2022). For CNNs our approach is slightly less accurate than Fazlyab et al. (2019), but unlike Fazlyab, we can provide Lipschitz constant estimates that hold for all input dimensions. In addition, our method utilizes multipliers* Λ_i *of dimension* c_i, *whereas using Toeplitz matrices and the approach in Fazlyab et al. (2019) requires the use of larger multipliers of size* $c_i N_i$ *for each layer, i.e., more degrees of freedom.*
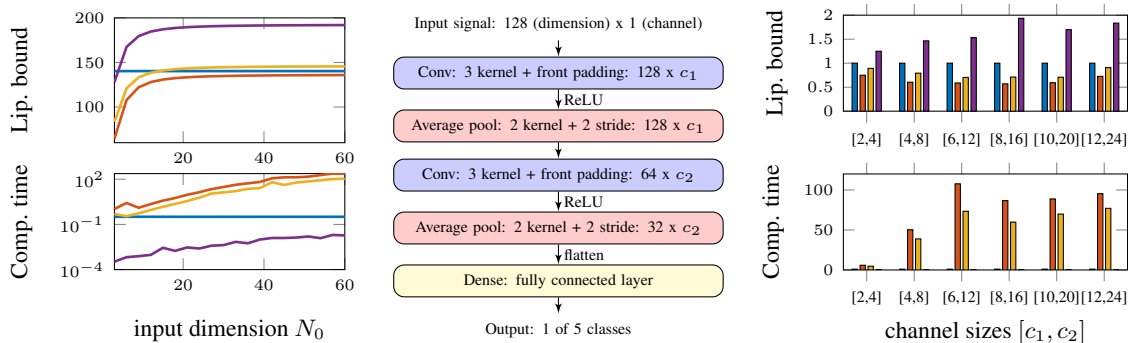
Figure 2: Comparison of our method (–), LipSDP-Neuron (–), LipSDP-Layer (–), and the spectral norm product (–). Left: Lipschitz bounds and computation time for fully convolutional neural network over input dimension, middle: architecture of CNN, right: Normalized Lipschitz bounds and normalized computation times over increasing channel sizes.

4. Simulation results

In the following, we compare our approach for Lipschitz constant estimation to the approaches LipSDP-Neuron and LipSDP-Layer suggested by Fazlyab et al. (2019) and to the trivial bound obtained from multiplication of spectral norms of weight matrices (Szegedy et al., 2013)

4.1. Fully convolutional neural networks

We set up a fully convolutional neural network with three convolutional layers, no pooling layers, and no fully connected layers with channel dimensions $[1, 3, 5, 10]$ and kernel sizes $[3, 3, 3]$ and randomly generate a set of convolution kernels for the CNN. In Fig. 2 on the left, we see the Lipschitz bounds for different input signal dimensions N_0 and the corresponding computation times. We observe that for small input dimensions the Lipschitz bounds LipSDP-Neuron and LipSDP-Layer are superior to our approach. However, for larger input dimensions the computation time increases drastically and, in this example, our method outperforms LipSDP-Layer for input dimensions $N_0 > 15$.

4.2. Deep CNNs

For the CNN shown in Fig. 2 in the middle, we generate six different CNNs with different channel sizes c_1 and c_2 and compare the Lipschitz bounds and the computation times of our method, LipSDP-Neuron, LipSDP-Layer, and the spectral norm product. While the computational overhead of LipSDP-Neuron/Layer is huge the Lipschitz bound is comparable to the one obtained using our approach which is clearly more accurate than the spectral norm bound. We note that, unlike LipSDP-Neuron/-Layer, our approach scales to significantly larger CNNs than shown in Fig. 2.

5. Conclusion and future work

In this paper, we proposed an efficient SDP to determine accurate upper bounds on the Lipschitz constant for 1D CNNs based on dissipativity theory. To do so, we derived incremental QCs for pooling layers and activation functions and further chose a compact state space representation for convolutional layers. A future direction is to directly exploit the structure of a 2D convolution for improved scalability.

Acknowledgments

References

Bernardo Aquino, Arash Rahnama, Peter Seiler, Lizhen Lin, and Vijay Gupta. Robustness against adversarial attacks in neural networks using incremental dissipativity. *IEEE Control Systems Letters*, 6:2341–2346, 2022.

Murat Arcak, Chris Meissen, and Andrew Packard. *Networks of dissipative systems: compositional certification of stability, performance, and safety*. Springer, 2016.

Patrick L Combettes and Jean-Christophe Pesquet. Lipschitz certificates for layered network structures driven by averaged activation operators. *SIAM Journal on Mathematics of Data Science*, 2 (2):529–557, 2020.

Mahyar Fazlyab, Alexander Robey, Hamed Hassani, Manfred Morari, and George Pappas. Efficient and accurate estimation of Lipschitz constants for deep neural networks. *Advances in Neural Information Processing Systems*, 32, 2019.

Graham C Goodwin and Kwai Sang Sin. *Adaptive filtering prediction and control*. Courier Corporation, 2014.

Dennis Gramlich, Patricia Pauli, Carsten W. Scherer, Frank Allgöwer, and Christian Ebenbauer. Convolutional neural networks as 2-D systems. *arXiv preprint arXiv:2303.03042*, 2023.

David Hill and Peter Moylan. The stability of nonlinear dissipative systems. *IEEE Transactions on Automatic Control*, 21(5):708–711, 1976. doi: 10.1109/TAC.1976.1101352.

Aris Kanellopoulos, Kyriakos G Vamvoudakis, and Vijay Gupta. Decentralized verification for dissipativity of cascade interconnected systems. In *58th Conference on Decision and Control (CDC)*, pages 3629–3634. IEEE, 2019.

Serkan Kiranyaz, Onur Avci, Osama Abdeljaber, Turker Ince, Moncef Gabbouj, and Daniel J Inman. 1d convolutional neural networks and applications: A survey. *Mechanical systems and signal processing*, 151:107398, 2021.

Vishwesh V Kulkarni and Michael G Safonov. Incremental positivity nonpreservation by stability multipliers. *IEEE Transactions on Automatic Control*, 47(1):173–177, 2002. doi: 10.1109/9. 981740.

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

Matthew Newton and Antonis Papachristodoulou. Exploiting sparsity for neural network verification. In *Learning for Dynamics and Control*, pages 715–727. PMLR, 2021.

Aaron van den Oord, Sander Dieleman, Heiga Zen, Karen Simonyan, Oriol Vinyals, Alex Graves, Nal Kalchbrenner, Andrew Senior, and Koray Kavukcuoglu. Wavenet: A generative model for raw audio. *arXiv preprint arXiv:1609.03499*, 2016.

Patricia Pauli, Niklas Funcke, Dennis Gramlich, Mohamed Amine Msalmi, and Frank Allgöwer. Neural network training under semidefinite constraints. In *2022 IEEE 61st Conference on Decision and Control (CDC)*, pages 2731–2736. IEEE, 2022a.

Patricia Pauli, Dennis Gramlich, and Frank Allgöwer. Lipschitz constant estimation for 1D convolutional neural networks. *arXiv preprint arXiv:2211.15253*, 2022b.

Patricia Pauli, Anne Koch, Julian Berberich, Paul Kohler, and Frank Allgöwer. Training robust neural networks using Lipschitz bounds. *IEEE Control Systems Letters*, 6:121–126, 2022c.

Carsten Scherer and Siep Weiland. Linear matrix inequalities in control. *Lecture Notes, Dutch Institute for Systems and Control, Delft, The Netherlands*, 3(2), 2000.

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

Anton Xue, Lars Lindemann, Alexander Robey, Hamed Hassani, George J Pappas, and Rajeev Alur. Chordal sparsity for lipschitz constant estimation of deep neural networks. In *2022 IEEE 61st Conference on Decision and Control (CDC)*, pages 3389–3396. IEEE, 2022.