Training deep residual networks for uniform approximation guarantees

Matteo Marchi MATMARCHI@UCLA.EDU
University of California, Los Angeles - California, USA

Bahman Gharesifard BAHMAN.GHARESIFARD@QUEENSU.CA
Queen's University, Kingston, Ontario, Canada

Paulo Tabuada TABUADA@EE.UCLA.EDU
University of California, Los Angeles - California, USA

Abstract

It has recently been shown that deep residual networks with sufficiently high depth, but bounded width, are capable of universal approximation in the supremum norm sense. Based on these results, we show how to modify existing training algorithms for deep residual networks so as to provide approximation bounds for the test error, in the supremum norm, based on the training error. Our methods are based on control-theoretic interpretations of these networks both in discrete and continuous time, and establish that it is enough to suitably constrain the set of parameters being learned in a way that is compatible with most currently used training algorithms.

Keywords: Deep residual networks, uniform approximation bounds, universal approximation.

1. Introduction

Deep learning (LeCun et al., 2015) has profoundly changed the way in which many engineering problems are solved, with computer vision being a particularly striking example. Deep neural networks can now perform complex tasks like gesture recognition (Oyedotun and Khashman, 2017), obstacle detection for self driving cars (Ramos et al., 2017), and many more (Voulodimos et al., 2018). Although similar benefits may be expected by integrating machine learning with control, we must contend with the safety critical nature of many control applications. Unfortunately, the statistical guarantees typically provided in machine learning, such as probably approximately correct learning bounds, cannot be directly used to establish formal guarantees of safety and performance of control loops. Despite these difficulties, several research efforts are underway to tackle this problem. For example, there are several recent papers studying control related tasks that employ data-driven controllers or perception maps (Dean et al., 2020; Cheng et al., 2019; Abbasi-Yadkori et al., 2019), and others that further study classical problems such as the Linear Quadratic Regulator (Dean et al., 2019) or the Kalman Filter (Tsiamis et al., 2020) in settings where the underlying parameters are to be learned as new information arrives, while still guaranteeing some level of performance. On the other hand, control-theoretic techniques, particularly ideas from robust and optimal control theory, have been useful in developing training algorithms for neural networks (Seidman et al., 2020; Li et al., 2017).

In this paper, we offer one additional contribution towards this effort. It was recently established in (Tabuada and Gharesifard, 2020), using control-theoretic techniques, that deep residual networks have the power of universal approximation with respect to the infinity norm. In other words, given a continuous function $f : E \to \mathbb{R}^n$ to be learned, defined on a compact set $E \subset \mathbb{R}^n$, and given a desired accuracy $\varepsilon \in \mathbb{R}^+$, there exists a deep residual network implementing the function ϕ satisfying $\sup_{x\in E} \|f(x) - \phi(x)\|_\infty \leq \varepsilon$. Such a result has obvious relevance in the context of a control loop since one can use well-established nonlinear control analysis techniques to study the effect of using ϕ instead of f by using the upper bound ε on the mismatch between f and ϕ. Unfortunately, the results in (Tabuada and Gharesifard, 2020) are not constructive and, in particular, they do not provide training procedures for deep residual networks that guarantee such bounds. Moreover, the results are established for the continuous limit of deep residual networks given by continuous-time control systems. These shortcomings are addressed in this paper. We show that most training algorithms, and gradient descent in particular, can be modified to offer similar approximation guarantees without the need to take the continuous limit.

At the technical level, we make the following two contributions: 1) we show how to modify training algorithms so that an upper bound on the infinity norm of the test error can be computed from an upper bound on the training error; 2) we show the training error can be made as small as desired by increasing depth (although we do not guarantee that any particular training algorithm can achieve it, as the training of deep networks is known to be a non-convex problem).

Interestingly, the guarantees provided in this paper are *deterministic* which contrasts with the mainstream approach in machine learning that typically only provides *probabilistic* guarantees (Shalev-Shwartz and Ben-David, 2014; Anthony and Bartlett, 2009). Underlying this difference are the assumptions made on the training data. Whereas classical learning theory assumes the training data to be generated according to some distribution, we make no assumptions on how it is generated. However, our bounds are based on how well the data covers the domain of the function to be learned. This is similar to robustness bounds in the Input to State Stability framework: nothing[1] is assumed about disturbances and the bound on the state depends on the concrete disturbance being applied. In our case, nothing is assumed about the training data and the bound depends on the actual data that was used for training.

2. Universal approximation in the uniform norm for deep residual networks

In this section we review the results of (Tabuada and Gharesifard, 2020) upon which the results of this paper are based.

2.1. Residual networks as control systems

One of the key insights exploited in (Tabuada and Gharesifard, 2020), is that residual neural networks can be thought of as the forward Euler discretization of continuous-time control systems. This observation, first made in (Weinan, 2017; Haber and Ruthotto, 2017; Lu et al., 2018), allows us to use control theoretic techniques to analyze deep residual networks.

Let us consider a deep residual network modeled by the discrete-time control system:

$$z(k+1) = z(k) + s(k)\Sigma(W(k)z(k) + b(k)), \tag{1}$$

where $k \in \mathbb{N}_0$ indexes the layers of the network, $z(k) \in \mathbb{R}^n$ models the contents of the n neurons in layer k, $(s(k), W(k), b(k)) \in \mathbb{R} \times \mathbb{R}^{n\times n} \times \mathbb{R}^n$ are the weights (interpreted as inputs), and

1. Except for being essentially bounded.

$\Sigma(z) = (\sigma(z_1), \ldots, \sigma(z_n))$ is defined by the scalar activation function $\sigma : \mathbb{R} \to \mathbb{R}$. In particular, we consider deep residual networks that have fixed width equal to n. To help the reader distinguish between discrete-time models and continuous-time models, we reserve z for the state of discrete-time models and x for the state of continuous-time models. Before introducing the continuous analogue of (1), we recall the notion of flow. The flow $Z^k : \mathbb{R}^n \to \mathbb{R}^n$ of (1) under the input $(s, W, b) : \{0, 1, \ldots, \ell\} \to \mathbb{R} \times \mathbb{R}^{n\times n} \times \mathbb{R}^n$ is the map Z^k taking the state $z \in \mathbb{R}^n$ to the state $Z^k(z)$ reached from z at time $k \in \{0, 1, \ldots, \ell + 1\}$ by the solution of (1) under the input (s, W, b).

Discrete-time control systems of the form (1) can be viewed as the time discretization of the continuous-time control system:

$$\dot{x}(t) = s(t)\Sigma(W(t)x(t) + b(t))\,, \tag{2}$$

where $x(t)$ and $(s(t), W(t), b(t))$ take values in the same sets as in (1), except they are indexed by $t \in \mathbb{R}_0^+$ modeling continuous time. The weights are now functions defined on $[0, \tau]$ that we interpret as open-loop inputs parameterized by time $t \in [0, \tau]$. Similarly to the discrete-time case, we define the flow $X^t : \mathbb{R}^n \to \mathbb{R}^n$ induced by (2) and by a choice of inputs $(s, W, b) : [0, \tau] \to \mathbb{R} \times \mathbb{R}^{n\times n} \times \mathbb{R}^n$ to be the function maping the state $x \in \mathbb{R}^n$ to the state $X^t(x)$ reached at time $t \in [0, \tau]$ from x by the solution of (2) under the input (s, W, b).

2.2. Controllability

The interpretation of deep residual networks as continuous-time control systems allows the problem of training a network to be recast as the problem of designing an open-loop control input.

Let us assume that we seek to train a network so as to learn a continuous function $f : E \to \mathbb{R}^n$ where $E \subset \mathbb{R}^n$ is a compact set. We are given a collection of samples of this function, i.e., a collection of d pairs $(x^i, f(x^i))$, with $i \in \{1, \ldots, d\}$, and our objective is to choose a time $\tau \in \mathbb{R}^+$ and a control input $(s, W, b) : [0, \tau] \to \mathbb{R} \times \mathbb{R}^{n\times n} \times \mathbb{R}^n$ so that the resulting flow X^t satisfies $X^\tau(x^i) = f(x^i)$ for all $i \in \{1, \ldots, d\}$. We emphasize that, independently of the number of samples d, we seek a *single* control input. In other words, we seek a single input to concurrently control d copies of the control system (2), each initialized at one of the points x^i in the sample set. To make this idea formal, we introduce the ensemble control system:

$$\dot{X}(t) = \big[s(t)\Sigma(W(t)X_{\bullet 1}(t) + b(t))|s(t)\Sigma(W(t)X_{\bullet 2}(t) + b(t))| \ldots |s(t)\Sigma(W(t)X_{\bullet d}(t) + b(t))\big]\,, \tag{3}$$

where $X_{\bullet i}$ represents the i-th column of $X(t) \in \mathbb{R}^{n\times d}$. If we define $X^{\text{init}} = \big[x^1|x^2| \ldots |x^d\big]$ and $X^{\text{fin}} = \big[f(x^1)|f(x^2)| \ldots |f(x^d)\big]$, we can express the network training problem as the design of a time $\tau \in \mathbb{R}^+$ and an input $(s, W, b) : [0, \tau] \to \mathbb{R} \times \mathbb{R}^{n\times n} \times \mathbb{R}^n$ so that the flow X^t of (3) satisfies $X^\tau(X^{\text{init}}) = X^{\text{fin}}$. Note that the ensemble control system (3), is formed by d exact copies of (2) whereas the literature on ensemble control, e.g., (Li and Khaneja, 2006; Helmke and Schönlein, 2014; Brockett, 2007), mostly deals with ensembles of different control systems, with the exception of (Agrachev and Sarychev, 2020) where a setting similar to the one here is considered.

To summarize, the ability to train a network relies on the controllability of the ensemble control system (3). Let us recall the notion of controllability.

Definition 1 *Control system* (3) *is said to be controllable on a submanifold M of $\mathbb{R}^{n\times d}$ if, given any points $X^{\text{init}}, X^{\text{fin}} \in M$ there exist $\tau \in \mathbb{R}^+$ and a control input $(s, W, b) : [0, \tau] \to \mathbb{R} \times \mathbb{R}^{n\times n} \times \mathbb{R}^n$ so that the flow X^t of* (3) *under said input satisfies $X^\tau(X^{\text{init}}) = X^{\text{fin}}$.*

In order to state one of the key results in (Tabuada and Gharesifard, 2020), showing that controllability holds on an open, dense, and connected subset of $\mathbb{R}^{n\times d}$, we introduce a mild assumption on the activation function σ stated in terms of its derivative denoted by $D\sigma$:

Assumption 1 *We assume that $D\sigma \geq 0$, $\overline{D\sigma} := \sup_{x\in\mathbb{R}} D\sigma < \infty$, and that there exists $k \in \mathbb{N}_0$ such that $\xi = D^k\sigma$ is injective and satisfies a quadratic differential equation $D\xi = a_0 + a_1\xi + a_2\xi^2$ with $a_2 \neq 0$.*

Assumption 1 is quite mild. It is satisfied, e.g., by the logistic function, hyperbolic tangent, tangent, and soft plus. Moreover, it also holds for the ReLU by regarding this function as the limit of the soft plus function, see (Tabuada and Gharesifard, 2020) for more details.

Theorem 2 (Tabuada and Gharesifard (2020)) *Let $N \subset \mathbb{R}^{n\times d}$ be the set defined by:*

$$N = \left\{ F \in \mathbb{R}^{n\times d} \mid \prod_{1\leq i<j\leq d} (F_{\ell i} - F_{\ell j}) = 0,\ \ell \in \{1,\ldots,n\} \right\}. \tag{4}$$

Suppose that Assumption 1 holds. If $n > 1$, then the ensemble control system (3) is controllable on the submanifold $M = \mathbb{R}^{n\times d}\backslash N$.

Theorem 2 shows that given any finite set of samples defining X^{init} and X^{fin}, as described above, only two situations can occur: 1) either $X^{\text{init}}, X^{\text{fin}} \in M$ and $X^\tau(X^{\text{init}}) = X^{\text{fin}}$ or; 2) $X^{\text{init}} \notin M$ or $X^{\text{fin}} \notin M$ and $\|X^\tau(X^{\text{init}}) - X^{\text{fin}}\| \leq \varepsilon$ for any chosen norm $\|\cdot\|$ and accuracy $\varepsilon \in \mathbb{R}^+$. The latter case holds in virtue of M being an open and dense subset of $\mathbb{R}^{n\times d}$. In other words, deep residual networks can memorize exactly almost any finite set of samples. Moreover, those finite sets that cannot be exactly memorized can be approximated to an arbitrary accuracy.

2.3. Uniform approximation

The second main result in (Tabuada and Gharesifard, 2020) extends Theorem 2 from finite sets to the whole domain of the function to be learned. This is done by using a deep residual network with $2n$ neurons to learn a function $f : E \to \mathbb{R}^n$, with $E \subset \mathbb{R}^n$ a compact set; the reason for doubling of the number of neurons has to do with an embedding of continuous maps into flows that is used to establish this result; even though the number of neurons can be reduced, we use a deep residual network with $2n$ neurons in this work. To recall this result more precisely, since f and the flow X^t have different domains and co-domains, we introduce a linear injection $\alpha : \mathbb{R}^n \to \mathbb{R}^{2n}$ and a linear projection $\beta : \mathbb{R}^{2n} \to \mathbb{R}^n$ and measure the error between f and the learned function $\beta \circ X^\tau \circ \alpha$ by:

$$\|f - \beta \circ X^\tau \circ \alpha\|_{L^\infty(E)} := \sup_{x\in E} \|f(x) - \beta \circ X^\tau \circ \alpha(x)\|_\infty.$$

Theorem 3 *Let $n > 1$ and suppose that Assumption 1 holds. Then, for every continuous function $f : \mathbb{R}^n \to \mathbb{R}^n$, for every compact set $E \subset \mathbb{R}^n$, and for every $\varepsilon \in \mathbb{R}^+$ there exist a time $\tau \in \mathbb{R}^+$, a linear injection $\alpha : \mathbb{R}^n \to \mathbb{R}^{2n}$, a linear projection $\beta : \mathbb{R}^{2n} \to \mathbb{R}^n$, and an input $(s, W, b) : [0,\tau] \to \mathbb{R} \times \mathbb{R}^{2n\times 2n} \times \mathbb{R}^{2n}$ so that the flow $X^t : \mathbb{R}^{2n} \to \mathbb{R}^{2n}$ of (2) with state space $\mathbb{R}^{2n}$ under the said input satisfies:*

$$\|f - \beta \circ X^\tau \circ \alpha\|_{L^\infty(E)} \leq \varepsilon.$$

By interpreting α and β as linear layers, the first and last, respectively, we conclude that deep residual networks can approximate any continuous function arbitrarily well with respect to the supremum norm. In (Tabuada and Gharesifard, 2020), α is the fixed function $\alpha(x) = (x, x)$ and β is the linear function $\beta(x, y) = x + Ay$ where A has to be appropriately chosen for each f being approximated. In the remainder of the paper we work with a different model where β is the fixed function $\beta(x, y) = x$ and the function implemented by a deep residual network is:

$$\beta \circ X^\tau \circ \alpha(x) + Ax,$$

where the sum $\beta \circ X^\tau \circ \alpha(x) + Ax$ can be implemented by summing the output of a deep residual network implementing $\beta \circ X^\tau \circ \alpha(x)$ with A acting on the input of such network. This is akin to using a skip connection (He et al., 2016) from the input of the network all the way to the last layer.

3. Main results

3.1. Approximation bounds for training algorithms

In this section, we show how the approximation error between a function to be learned and the function implemented by a deep residual network can be bounded by the training error. Monotonicity plays a key role in our approach and thus we start by reviewing this concept.

Definition 4 (First-orthant partial order on $\mathbb{R}^n$) *The first-orthant partial order $\preceq$ on $\mathbb{R}^n$ is defined by $x \preceq y$ if and only if $x_i \leq y_i$ for all $i \in \{1, \ldots, n\}$, where $\leq$ denotes the usual total order on $\mathbb{R}$.*

Monotonicity of a map can now be introduced by making use of the preceding partial order. Although monotonicity can be defined with respect to other orders, the first-orthant order will simplify the analysis.

Definition 5 (Monotone map) *We say that a function $\phi : \mathbb{R}^n \to \mathbb{R}^m$ is monotone if for any $x, y \in \mathbb{R}^n$:*

$$x \preceq y \implies \phi(x) \preceq \phi(y) \, .$$

We say that a flow Z^k is monotone if the map $Z^k : \mathbb{R}^n \to \mathbb{R}^n$ is monotone for each $k \in \mathbb{N}$.

Given a set $E \subset \mathbb{R}^n$, we denote, respectively, by $\sup E$ and $\inf E$ the least upper bound of all the elements in E and the greatest lower bound of all the elements in E with respect to the order $\preceq$. Moreover, given two points $x, z \in \mathbb{R}^n$ with $x \preceq z$, we denote by $[x, y]$ the set defined by:

$$[x, z] = \{y \in \mathbb{R}^n \mid x \preceq y \preceq z\}.$$

Flows of deep residual networks can be made monotone by enforcing certain constraints on the inputs.

Proposition 6 *Consider the discrete-time control system (1) modeling a deep residual network and assume Assumption 1 holds. Any flow Z^k induced by (1) using an input $(s, W, b) : \{0, 1, \ldots, \ell\} \to \mathbb{R} \times \mathbb{R}^{n\times n} \times \mathbb{R}^n$ satisfying:*

$$s(k)w_{ij}(k) \geq 0, \quad 1 + s(k)w_{ii}(k)\overline{D\sigma} \geq 0, \qquad \forall i, j \in \{1, 2, \ldots, n\}, i \neq j, \quad k \in \{0, 1, \ldots, \ell\}, \tag{5}$$

is monotone.

Proof Let $k \in \mathbb{N}_0$ and consider the map $\phi^k : \mathbb{R}^n \to \mathbb{R}^n$ defined by

$$\phi^k(z) = z + s(k)\Sigma(W(k)z + b(k))\,.$$

Note that for all $i, j \in \{1, 2, \dots, n\}$ with $i \neq j$, we have that

$$\frac{\partial \phi^k_i}{\partial x_j}(x) = s(k)D\sigma(W(k)x + b(k))w_{ij}(k)\,.$$

Since $D\sigma \geq 0$ and $s(k)w_{ij}(k) \geq 0$, by assumption, we have $\frac{\partial \phi^k_i}{\partial x_j} \geq 0$ for all $i \neq j$. Moreover, for $i = j$ we have:

$$\frac{\partial \phi^k_i}{\partial x_i}(x) = 1 + s(k)D\sigma(W(k)x + b(k))w_{ii}(k)\,.$$

It now follows from the assumption $1 + s(k)w_{ii}(k)\overline{D\sigma} \geq 0$, alongside with the fact that $\overline{D\sigma} \geq 0$, that $\frac{\partial \phi^k_i}{\partial x_j} \geq 0$ holds for $i = j$. Hence, by (Hirsch and Smith, 2006, Lemma 5.1), we conclude that ϕ^k is monotone.

Finally, since:

$$Z^k = \phi^{k-1} \circ \phi^{k-2} \circ \cdots \circ \phi^0\,,$$

and the composition of monotone functions is monotone, Z^k is monotone. ■

We now prove one of the main results of this paper, a deterministic bound for the approximation error based on the training error. Although reminiscent of a generalization bound that holds for **any** set of sample points, the bound in the following result depends on the **specific** set of samples.

Theorem 7 *Consider the discrete-time control system (1) modeling a deep residual network and assume that Assumption 1 holds. Let $f : \mathbb{R}^n \to \mathbb{R}^n$ be a continuous function, and let $E \subset \mathbb{R}^n$ be a compact set. For any finite set $E_{\text{samples}} \subset \mathbb{R}^n$ satisfying $E \subseteq [\inf E_{\text{samples}}, \sup E_{\text{samples}}]$, let $\delta \in \mathbb{R}^+$ be the smallest number satisfying:*

$$\forall x \in E,\ \exists \underline{x}, \overline{x} \in E_{\text{samples}}, \quad \|\overline{x} - \underline{x}\|_\infty \leq \delta \text{ and } \underline{x} \preceq x \preceq \overline{x}.$$

For any flow Z^k induced by (1) and by an input $(s, W, b) : \{0, 1, \dots, \ell\} \to \mathbb{R} \times \mathbb{R}^{2n\times 2n} \times \mathbb{R}^{2n}$ satisfying the constraints (5), and for any linear map $A : \mathbb{R}^n \to \mathbb{R}^n$, the following bound holds:

$$\|f - (\beta \circ Z^{\ell+1} \circ \alpha + A)\|_{L^\infty(E)} \leq 3\|f - (\beta \circ Z^{\ell+1} \circ \alpha + A)\|_{L^\infty(E_{\text{samples}})} + 2\omega_f(\delta) + 2\|A\|\delta, \quad (6)$$

where ω_f is the modulus of continuity of f, $\alpha : \mathbb{R}^n \to \mathbb{R}^{2n}$ is given by $\alpha(x) = (x, x)$, and $\beta : \mathbb{R}^{2n} \to \mathbb{R}^n$ is given by $\beta(x, y) = x$.

Proof We first show the injection α is monotone. For any $x, y \in \mathbb{R}^n$ we have:

$$x \preceq y \implies \alpha(x) = (x, x) \preceq (y, y) = \alpha(y)\,.$$

Similarly, we show that the projection β is monotone:

$$\forall w, x, y, z \in \mathbb{R}^n : \quad (x, z) \preceq (y, w) \implies \beta(x, z) = x \preceq y = \beta(y, w)\,.$$

Then, since $Z^{\ell+1}$ is monotone, by virtue of Proposition 6, and the composition of monotone functions is monotone, we have that $\beta \circ Z^{\ell+1} \circ \alpha$ is monotone.

Let $\phi = \beta \circ Z^{\ell+1} \circ \alpha$, then:

$$\|f - (\beta \circ Z^{\ell+1} \circ \alpha + A)\|_{L^\infty(E_{\text{samples}})} = \|(f - A) - \phi\|_{L^\infty(E_{\text{samples}})} = \|\tilde{f} - \phi\|_{L^\infty(E_{\text{samples}})}, \quad (7)$$

where $\tilde{f} = f - A$. Then, since $\tilde{f}$ is a continuous function and ϕ is monotone, we can apply Lemma A.3 in (Tabuada and Gharesifard, 2020) to obtain:

$$\begin{aligned} \|f - (\beta \circ Z^{\ell+1} \circ \alpha + A)\|_{L^\infty(E)} &= \|\tilde{f} - \phi\|_{L^\infty(E)} \\ &\le 3\|\tilde{f} - \phi\|_{L^\infty(E_{\text{samples}})} + 2\omega_{\tilde{f}}(\delta) \\ &\le 3\|f - (\beta \circ Z^{\ell+1} \circ \alpha + A)\|_{L^\infty(E_{\text{samples}})} + 2\omega_{\tilde{f}}(\delta). \end{aligned}$$

To conclude the proof, it remains to be shown that $\omega_{\tilde{f}}(\delta) \le \omega_f(\delta) + \|A\|\delta$. This can be seen by observing that for any $x, y \in \mathbb{R}^n$, we have:

$$\begin{aligned} \|\tilde{f}(x) - \tilde{f}(y)\| &= \|f(x) + f(y) - A(x - y)\| \\ &\le \|f(x) + f(y)\| + \|A\|\|x - y\| \\ &\le \omega_f(\|x - y\|) + \|A\|\|x - y\|, \end{aligned}$$

yielding the claim. ■

In Theorem 7, the linear map A is arbitrary. However, to obtain low training error, A has to be chosen appropriately since it depends on the function f being learned. In fact, we will see in Section 3.2 that for each f there exists an A so that the training error can be made arbitrarily small. In practice, A is also be learned from data. The bound (6) can be used as a stopping criterion for the training. Since $\|f - (\beta \circ Z^{\ell+1} \circ \alpha + A)\|_{L^\infty(E_{\text{samples}})}$ is known, some knowledge of an upper bound for ω_f will directly give us an upper bound for the approximation error. Hence, the question arises as to how to train deep residual networks so that the assumptions of Theorem 7 hold. This can be done by training a deep residual network with any of the usual iterative optimization techniques (such as gradient descent), as long the parameters s, W are suitably constrained to satisfy (5). We now make this idea precise.

Consider a deep residual network described by (1) with $2n$ neurons and $\ell + 1$ layers enhanced with a layer implementing the linear map $\alpha(x) = (x, x)$, functioning as layer 0, with a layer implementing the sum of the linear map $\beta(x, y) = x$ with the linear map A of network's input, functioning as layer $\ell + 2$ (A can be viewed as a skip connection (He et al., 2016) from the input to the output layer). Note that while layer 0 is fixed, the parameters A in layer $\ell + 2$ will also be learned. If we denote by $\theta(k)$, $k \in \{1, 2, \dots, \ell + 2\}$ the parameters of each layer, we have:

$$\theta(k) = (s(k), W(k), b(k)) \in \mathbb{R} \times \mathbb{R}^{2n \times 2n} \times \mathbb{R}^{2n}, \quad k \in \{1, 2, \dots, \ell + 1\}$$

$$\theta(\ell + 2) = A \in \mathbb{R}^{n \times n}.$$

To ensure that the parameters $\theta(k)$, $k \in \{1, 2, \dots, \ell + 1\}$, satisfy the constraints (5), we project them to the closest point in the set defined by (5) using the projection $\text{proj} : \mathbb{R}^{4n^2+2n+1} \to$

$\mathbb{R}^{4n^2+2n+1}$ defined by the solution of a quadratic optimization problem:

$$\operatorname{proj}(\theta(k)) = \begin{cases} \arg\min_{\theta' \in \mathbb{R}^{4n^2+1}} & \|\theta' - \theta(k)\|^2 \\ \text{s.t.} & s' w'_{ij} \geq 0, \quad i, j \in \{1, \ldots, 2n\}, i \neq j \ , \\ & 1 + s' w'_{ii} \overline{D\sigma} \geq 0, \quad i \in \{1, \ldots, 2n\} \end{cases} \tag{8}$$

where $\theta' = (s', W', b')$ is the variable we optimize over (note that b' does not appear in the constraints, so can always be taken as $b' = b(k)$). Since the constraints (5) are independent for each layer, the parameters of each layer can be independently projected. Although problem (8) is nonconvex, many efficient heuristics exist for solving quadratic programs, see (Park and Boyd, 2017) and references therein. Furthermore, for Theorem 7 to hold, it is enough to solve the feasibility problem of (8), which is doable in polynomial time.

Let now Θ be the full set of parameters of the network $\Theta = (\theta(1), \ldots, \theta(\ell+1), \theta(\ell+2))$. Any iterative training algorithm can be written in the form:

$$\Theta_{i+1} = \psi(\Theta_i),$$

for a suitably defined ψ (that we assume here to encode all the information about the problem, such as the available training data), and modified to:

$$\begin{aligned} \widetilde{\Theta}_{i+1} &= \psi(\Theta_i) \\ \Theta_{i+1} &= (\operatorname{proj}(\tilde{\theta}_{i+1}(1)), \operatorname{proj}(\tilde{\theta}_{i+1}(2)), \ldots, \operatorname{proj}(\tilde{\theta}_{i+1}(\ell+1)), \tilde{\theta}_{i+1}(\ell+2)), \end{aligned}$$

so that the parameters θ at each iteration satisfy the constraints (5). Even though we do not plan to dwell on this topic here, it is worth pointing out that the projected gradient descent has convergence properties similar to those of normal gradient descent (Attouch et al., 2013). In fact, constraining the weights in neural networks is a recurring idea in the literature, see, e.g., (Chorowski and Zurada, 2014; Daniels and Velikova, 2010).

As a concluding remark, it is interesting to note that penalizing the norm of A during training, (that would typically be done for purposes of regularization), can be justified on the basis of Theorem 7, where $\|A\|$ appears in the final approximation guarantee, indicating a potential tradeoff between training and test errors.

3.2. Revisiting the universal approximation properties of deep residual networks

The bound (6) provided in Theorem 7 provides information about the approximation error based on the training error. However, we do not know if low training error is achievable when the constraints (5) on the inputs are enforced. We will show this to be the case by building upon the results in (Tabuada and Gharesifard, 2020). We first introduce a variant of Corollary 4.5 in (Tabuada and Gharesifard, 2020).

Theorem 8 *Let $n > 1$ and suppose that Assumption 1 holds. Then, for every continuous function $f : \mathbb{R}^n \to \mathbb{R}^n$, for every compact set $E \subset \mathbb{R}^n$, and for every $\varepsilon \in \mathbb{R}^+$ there exist a time $\tau \in \mathbb{R}^+$, a linear map $A : \mathbb{R}^n \to \mathbb{R}^n$, and an input $(s, W, b) : [0, \tau] \to \mathbb{R} \times \mathbb{R}^{2n \times 2n} \times \mathbb{R}^{2n}$ satisfying:*

$$s(t) w_{ij}(t) \geq 0 \qquad \forall i, j \in \{1, 2, \ldots, 2n\}, i \neq j, t \in [0, \tau], \tag{9}$$

so that the flow $X^t : \mathbb{R}^{2n} \to \mathbb{R}^{2n}$ of (2) *with state space $\mathbb{R}^{2n}$ under the said input satisfies:*

$$\|f - (\beta \circ X^\tau \circ \alpha + A)\|_{L^\infty(E)} \leq \varepsilon,$$

where $\alpha : \mathbb{R}^n \to \mathbb{R}^{2n}$ is given by $\alpha(x) = (x, x)$, and $\beta : \mathbb{R}^{2n} \to \mathbb{R}^n$ is given by $\beta(x, y) = x$.

Note that this theorem differs from Theorem 3 by the additional constraints (9) on the inputs. However, the proof of Theorem 3 only requires minor modifications to account for (9).

Proof The same arguments used in (Tabuada and Gharesifard, 2020) will suffice to prove this result, provided that there exist choices of inputs (s, W, b) satisfying the constraints (9) that induce the same set of vector fields used in the proofs of Proposition A.4 and Theorem 4.4 of (Tabuada and Gharesifard, 2020). The vector fields used in Proposition A.4 of this reference are:

$$\begin{cases} X_i^+ = \sigma(c)\frac{\partial}{\partial x_i}, & X_i^- = -X_i^+ \\ Y_i^+ = \sigma(x_i)\frac{\partial}{\partial x_i}, & Y_i^- = -\sigma(x_i)\frac{\partial}{\partial x_i} \end{cases}, \quad i \in \{1, 2, \ldots, 2n\},$$

for some c such that $\sigma(c) \neq 0$. Respectively, these can be realized in our case by the choices $(s, W, b) = (\pm 1, 0, ce_i)$ and $(s, W, b) = (\pm 1, E_{ii}, ce_i)$, where e_i is the vector with 1 in its i-th entry and 0 in every other entry, and E_{ij} is the matrix with 1 in its ij-th entry and 0 in all other entries (note that the constraints (9) only apply to off-diagonal entries). An additional class of vector fields that is used in Theorem 4.4 of (Tabuada and Gharesifard, 2020) is given by

$$Z_{ij} = \sigma(x_j)\frac{\partial}{\partial x_i}, \quad i, j \in \{1, \ldots, 2n\}.$$

Note that these vector fields are realizable in our case by the choice $(s, W, b) = (1, E_{ij}, 0)$.

This ensures that Proposition A.4, Theorem 4.4, and in particular Corollary 4.5 of (Tabuada and Gharesifard, 2020) hold even when the inputs of the control system (2) are constrained according to (9), proving this result. ■

We can now use the previous result to establish that the training error can be made as small as desired provided the network depth is large enough. This will be done by invoking Theorem 8 to obtain a continuous-time input leading to low training error on the model (2) and then using forward Euler discretization to obtain a discrete-time input leading to low training error in the model (1).

Theorem 9 *Consider the discrete-time control system* (1) *modeling a deep residual network and assume that Assumption 1 holds. Then, for every continuous function $f : \mathbb{R}^n \to \mathbb{R}^n$ with $n > 1$, for every compact set $E \subset \mathbb{R}^n$, and for every $\varepsilon \in \mathbb{R}^+$ there exist a time $\ell \in \mathbb{N}$, a linear map $A : \mathbb{R}^n \to \mathbb{R}^n$, and an input $(s, W, b) : \{0, 1, \ldots, \ell\} \to \mathbb{R} \times \mathbb{R}^{2n\times 2n} \times \mathbb{R}^{2n}$ satisfying* (5) *so that the flow $Z^k : \mathbb{R}^{2n} \to \mathbb{R}^{2n}$ of* (1) *with state space $\mathbb{R}^{2n}$ under the said input satisfies:*

$$\|f - (\beta \circ Z^{\ell+1} \circ \alpha + A)\|_{L^\infty(E)} \leq \varepsilon, \tag{10}$$

where $\alpha : \mathbb{R}^n \to \mathbb{R}^{2n}$ is given by $\alpha(x) = (x, x)$ and $\beta : \mathbb{R}^{2n} \to \mathbb{R}^n$ is given by $\beta(x, y) = x$.

Proof . According to Theorem 8, there exist a linear map $A : \mathbb{R}^n \to \mathbb{R}^n$ and a piecewise constant control input[2] (s, W, b) for (2) satisfying (9) and inducing a flow X^t so that $\|f - h\|_{L^\infty(E)} \leq \frac{\varepsilon}{2}$ with

2. The controllability results in (Tabuada and Gharesifard, 2020) only rely on being able to switch between a finite set of vector fields which is achieved by piecewise constant inputs.

h defined by $h(x) = \beta \circ X^\tau \circ \alpha(x) + Ax$. Consider now the function $g(x) = \beta \circ Z^{\ell+1} \circ \alpha(x) + Ax$ where Z^k is the flow induced by (1) under some input. Then, for any $x \in E$:

$$\begin{aligned}
\|f(x) - g(x)\|_\infty &\leq \|f(x) - h(x)\|_\infty + \|h(x) - g(x)\|_\infty \\
&\leq \frac{\varepsilon}{2} + \|\beta(X^\tau \circ \alpha(x) - Z^{\ell+1} \circ \alpha(x))\|_\infty \\
&\leq \frac{\varepsilon}{2} + \|X^\tau(x,x) - Z^{\ell+1}(x,x)\|_\infty, \qquad (11)
\end{aligned}$$

where the second inequality follows from linearity of β and the third inequality follows from β being Lipschitz continuous with Lipschitz constant equal to 1.

Let (s, W, b) be the continuous-time input for (2) that results in the continuous-time flow X^t. We now construct a discrete-time input (s', W', b') for (1) so that the resulting flow Z^k satisfies:

$$\|X^\tau(x,x) - Z^{\ell+1}(x,x)\|_\infty \leq \frac{\varepsilon}{2}. \qquad (12)$$

By Euler forward integration, see (Atkinson, 2008), there exists a sufficiently small $T \in \mathbb{R}^+$ so that the flow of (1) under the input $(s'(k), W'(k), b'(k)) = (Ts(kT), W(kT), b(kT))$, $k \in \{1, 2, \ldots, \lfloor \tau/T \rfloor\}$, satisfies (12) with $\ell = \lfloor \tau/T \rfloor$, provided the solution of (2) has bounded second derivative and the right-hand side of (2) is Lipschitz continuous in the state variable. Since the input (s, W, b) is piecewise constant, the solution of (2) can be seen as the composition of analytic flows (the right-hand side of (2) is analytic for constant inputs in virtue of Assumption 1) functions, one flow per each constant component of the input. Since the solution is defined on the compact $[0, \tau]$, its second derivate is bounded.

By combining (11) with (12) we obtain inequality (10), and hence to conclude the proof it suffices to show that the inputs (s', W', b') satisfy (5). The first requirement in (5):

$$s'(k) w'_{ij}(k) \geq 0,$$

is immediately satisfied, since $\forall t \in [0, \tau]$ we have that $s(t) w_{ij}(t) \geq 0$. In order to show that the second requirement in (5) is satisfied, we first let

$$\bar{s} = \max_{[0,t]} |s(t)|, \quad \bar{w}_{ij} = \max_{[0,t]} |w_{ij}(t)|,$$

and select T such that:

$$T \leq \min_{i,j \in \{0,\ldots,n\}} \left(\bar{s}\bar{w}_{ij}\overline{D\sigma}\right)^{-1}.$$

Therefore, we have that

$$1 + Ts(kT) w_{ii}(kT) \overline{D\sigma} = 1 + s'(k) w'_{ii}(k) \overline{D\sigma} \geq 0.$$

■

4. Conclusions

In this paper we showed how to modify existing training algorithms for deep residual networks so that approximation bounds can be given in the supremum norm. These results are different from the typical approximation guarantees in the literature in that they are deterministic and are based on the sample set used for training. They are applicable to scenarios where the domain of the function to be learned is known and can be appropriately sampled. Although not all applications satisfy these requirements, we regard these results as useful first steps to obtain hard guarantees with a view towards integrating deep networks within a control loop.

References

Yasin Abbasi-Yadkori, Nevena Lazic, and Csaba Szepesvári. Model-free linear quadratic control via reduction to expert prediction. In *The 22nd International Conference on Artificial Intelligence and Statistics*, pages 3108–3117. PMLR, 2019.

Andrei Agrachev and Andrey Sarychev. Control in the spaces of ensembles of points. *SIAM Journal on Control and Optimization*, 58(3):1579–1596, 2020.

Martin Anthony and Peter L Bartlett. *Neural network learning: Theoretical foundations*. cambridge university press, 2009.

Kendall E Atkinson. *An introduction to numerical analysis*. John wiley & sons, 2008.

Hedy Attouch, Jérôme Bolte, and Benar Fux Svaiter. Convergence of descent methods for semi-algebraic and tame problems: proximal algorithms, forward–backward splitting, and regularized gauss–seidel methods. *Mathematical Programming*, 137(1-2):91–129, 2013.

Roger W Brockett. Optimal control of the Liouville equation. *AMS IP Studies in Advanced Mathematics*, 39:23, 2007.

Richard Cheng, Gábor Orosz, Richard M Murray, and Joel W Burdick. End-to-end safe reinforcement learning through barrier functions for safety-critical continuous control tasks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 3387–3395, 2019.

Jan Chorowski and Jacek M Zurada. Learning understandable neural networks with nonnegative weight constraints. *IEEE transactions on neural networks and learning systems*, 26(1):62–69, 2014.

Hennie Daniels and Marina Velikova. Monotone and partially monotone neural networks. *IEEE Transactions on Neural Networks*, 21(6):906–917, 2010.

Sarah Dean, Horia Mania, Nikolai Matni, Benjamin Recht, and Stephen Tu. On the sample complexity of the linear quadratic regulator. *Foundations of Computational Mathematics*, pages 1–47, 2019.

Sarah Dean, Nikolai Matni, Benjamin Recht, and Vickie Ye. Robust guarantees for perception-based control. In *Learning for Dynamics and Control*, pages 350–360, 2020.

E. Haber and L. Ruthotto. Stable architectures for deep neural networks. *Inverse Problems*, 34(1), 2017.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

Uwe Helmke and Michael Schönlein. Uniform ensemble controllability for one-parameter families of time-invariant linear systems. *Systems & Control Letters*, 71:69–77, 2014.

Morris W Hirsch and Hal Smith. Monotone dynamical systems. In *Handbook of differential equations: ordinary differential equations*, volume 2, pages 239–357. Elsevier, 2006.

Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *nature*, 521(7553):436–444, 2015.

Jr-Shin Li and Navin Khaneja. Control of inhomogeneous quantum ensembles. *Physical review A*, 73(3):030302, 2006.

Qianxiao Li, Long Chen, Cheng Tai, and E Weinan. Maximum principle based algorithms for deep learning. *The Journal of Machine Learning Research*, 18(1):5998–6026, 2017.

Y. Lu, A. Zhong, Q. Li, and B. Dong. Beyond finite layer neural networks: Bridging deep architectures and numerical differential equations. In *International Conference on Machine Learning*, pages 3276–3285, 2018.

Oyebade K Oyedotun and Adnan Khashman. Deep learning in vision-based static hand gesture recognition. *Neural Computing and Applications*, 28(12):3941–3951, 2017.

Jaehyun Park and Stephen Boyd. General heuristics for nonconvex quadratically constrained quadratic programming. *arXiv preprint arXiv:1703.07870*, 2017.

Sebastian Ramos, Stefan Gehrig, Peter Pinggera, Uwe Franke, and Carsten Rother. Detecting unexpected obstacles for self-driving cars: Fusing deep learning and geometric modeling. In *2017 IEEE Intelligent Vehicles Symposium (IV)*, pages 1025–1032. IEEE, 2017.

Jacob H Seidman, Mahyar Fazlyab, Victor M Preciado, and George J Pappas. Robust deep learning as optimal control: Insights and convergence guarantees. *arXiv preprint arXiv:2005.00616*, 2020.

Shai Shalev-Shwartz and Shai Ben-David. *Understanding machine learning: From theory to algorithms*. Cambridge university press, 2014.

Paulo Tabuada and Bahman Gharesifard. Universal approximation power of deep neural networks via nonlinear control theory. *arXiv preprint arXiv:2007.06007v2*, 2020.

Anastasios Tsiamis, Nikolai Matni, and George Pappas. Sample complexity of Kalman filtering for unknown systems. In *Learning for Dynamics and Control*, pages 435–444. PMLR, 2020.

Athanasios Voulodimos, Nikolaos Doulamis, Anastasios Doulamis, and Eftychios Protopapadakis. Deep learning for computer vision: A brief review. *Computational intelligence and neuroscience*, 2018, 2018.

E. Weinan. A proposal on machine learning via dynamical systems. *Communications in Mathematics and Statistics*, 5, 2017.