Journal of Mathematical Imaging and Vision (2020) 62:328–351
https://doi.org/10.1007/s10851-019-00922-y

Forward Stability of ResNet and Its Variants

Linan Zhang[1] · **Hayden Schaeffer**[1]

Received: 24 November 2018 / Accepted: 10 October 2019 / Published online: 1 November 2019

Abstract

The residual neural network (ResNet) is a popular deep network architecture which has the ability to obtain high-accuracy results on several image processing problems. In order to analyze the behavior and structure of ResNet, recent work has been on establishing connections between ResNets and continuous-time optimal control problems. In this work, we show that the post-activation ResNet is related to an optimal control problem with differential inclusions and provide continuous-time stability results for the differential inclusion associated with ResNet. Motivated by the stability conditions, we show that alterations of either the architecture or the optimization problem can generate variants of ResNet which improves the theoretical stability bounds. In addition, we establish stability bounds for the full (discrete) network associated with two variants of ResNet, in particular, bounds on the growth of the features and a measure of the sensitivity of the features with respect to perturbations. These results also help to show the relationship between the depth, regularization, and stability of the feature space. Computational experiments on the proposed variants show that the accuracy of ResNet is preserved and that the accuracy seems to be monotone with respect to the depth and various corruptions.

Keywords Deep feedforward neural networks · Residual neural networks · Stability · Differential inclusions · Optimal control problems

1 Introduction

Deep neural networks (DNNs) have been successful in several challenging data processing tasks, including but not limited to image classification, segmentation, speech recognition, and text analysis. The first convolutional neural network (CNN), which was used in the recognition of digits and characters, was the famous LeNet [25]. The LeNet architecture included two convolution layers and two fully connected layers. Part of the success of CNNs is their ability to capture spatially local and hierarchal features from images. In [22], the authors proposed a deeper CNN architecture, called AlexNet, which achieved record-breaking accuracy on the ILSVRC-2010 classification task [32]. In addition to the increased depth (i.e., the number of layers), AlexNet also used rectified linear unit (ReLU) as its activation function and overlapping max pooling to downsample the features between layers. Over the past few years, the most popular networks: VGG [35], GoogleNet [38], ResNet [16, 17], FractalNet [23], and DenseNet [18], continued to introduce new architectural structures and increase their depth. In each case, the depth of the network seems to contribute to the improved classification accuracy. In particular, it was shown in [16, 17] that deeper networks tended to improve classification accuracy on the common datasets (CIFAR 10, CIFAR 100, and ImageNet). It is not unusual for DNNs to have thousands of layers!

Although DNNs are widely successful in application, our understanding of their theoretical properties and behavior is limited. In this work, we develop connections between feedforward networks and optimal control problems. These connections are used to construct networks that satisfy some desired stability properties. To test the ideas, we will focus on the image classification problem. Let $\mathcal{D}$ be a set of images which are sampled from n distinct classes. The goal of the classification problem is to learn a function whose output $y \in \mathbb{R}^n$ predicts the correct label associated with the input image $x \in \mathcal{D}$. The jth component of y represents the probability of x being in Class j. It is worth noting that the image

✉ Linan Zhang
linanz@andrew.cmu.edu

Hayden Schaeffer
schaeffer@cmu.edu

[1] Department of Mathematical Sciences, Carnegie Mellon University, Pittsburgh, PA, USA

classification problem is an example of a high-dimensional problem that can be better solved by DNN than other standard approaches. One possible reason for this is that the mapping from images to labels represented by a neural network may generalize well to new data [1, 24].

As the network depth increases, several issues can occur during the optimization (of network parameters). Take for example the (supervised) image classification problem, where one learns a network by optimizing a cost function over a set of parameters. Since the parameters are high-dimensional and the problem is non-convex, one is limited in their choice of optimization algorithms [4]. In addition, the size of the training set can affect the quality and stability of the learned network [4]. The nonconvexity of the optimization problem may yield many local minimizers, and in [21] it was argued that sharp local minimizer could produce networks that are not as generalizable as the networks learned from flatter local minimizers. In [26], the authors showed that (visually) the energy landscape of ResNet and DenseNet is well-behaved and may be flatter than CNNs without shortcuts. Another potential issue with training parameters of deep networks involves exploding or vanishing gradients, which has been observed in various network architectures [2]. Some partial solutions have been given by using ReLU as the activation function [32] and by adding identity shortcuts [16, 17]. In addition, networks can be sensitive to the inputs in the sense that small changes may lead to misclassification [3, 13, 39]. This is one of the motivations for providing a quantitive measure of input-sensitivity in this work.

Recently, there have been several works addressing the architecture of neural networks as the forward flow of a dynamical system. By viewing a neural network as a dynamical system, one may be able to address issues of depth, scale, and stability by leveraging previous work and theory in differential equations. In [45], the connection between continuous dynamical systems and DNNs was discussed. In [14], the authors proposed several architectures for deep learning by imposing conditions on the weights in residual layers. The motivation for the architectures in [14] directly came from the ordinary differential equation (ODE) formulation of ResNets (when there is only one activation per residual layer). For example, they proposed using a Hamiltonian system, which should make the forward and back propagation stable in the sense that the norms of the features do not change. There could be more efficient ways to compute the back propagation of DNNs based on Hamiltonian dynamics, since the dynamics are time-reversible [5]. Reversible networks have several computationally beneficial properties [12]; however, layers such as batch normalization [19] may limit their use. The main idea of batch normalization is to normalize each training mini-batch by reducing its internal covariate shift, which does not preserve the Hamiltonian structure (at least directly). In a similar direction, ResNet-based architectures can be viewed as a control problem with the transport equation [27]. In [33], the authors designed networks using a symmetric residual layer which is related to parabolic and hyperbolic time-dependent partial differential equations, which produced similar results to the standard ResNet architecture. In [44], the authors formulated the population risk minimization problem in deep learning as a mean-field optimal control problem and proved optimality conditions of the Hamilton–Jacobi–Bellman type and the Pontryagin type. It is worth noting that some theoretical arguments connecting a ResNet with one convolution and one activation per residual layer to a first-order ODE are provided in [40].

In image classification, the last operation is typically an application of the softmax function so that the output of the network is a vector that represents the probability of an image being in each class; however, in [43] a harmonic extension is used. The idea in [43] is to learn an appropriate interpolant as the last layer, which may help to generalize the network to new data. In [30], the authors proposed a Lipschitz regularization term to the optimization problem and showed (theoretically) that the output of the regularized network converges to the correct classifier when the data satisfies certain conditions. In addition, there are several recent works that have made connections between optimization in deep learning and numerical methods for partial differential equations, in particular, the entropy-based stochastic gradient descent [6] and a Hamilton–Jacobi relaxation [7]. For a review of some other recent mathematical approaches to DNN, see [42] and the citations within.

1.1 Contributions of this Work

In this work, we connect the post-activation ResNet (Form (a) in [17]) to an optimal control problem with differential inclusions. We show that the differential system is well-posed and provide explicit stability bounds for the optimal control problem in terms of learnable parameters (i.e., the weights and biases). In particular, we provide a growth bound on the norm of the features and a bound on the sensitivity of the features with respect to perturbations in the inputs. These results hold in the continuous-time limit (i.e., when the depth of the network goes to infinity) and in the discrete setting where one includes all other operations such as batch normalization and pooling.

Since the stability results measure how sensitive the feature space is to perturbations on the input image, these results likely relate to the output accuracy. Based on the theory, we investigate two variants of ResNet that are developed in order to improve the two stability bounds. The variants are constructed by altering the architecture of the post-activation ResNet and the associated optimization problem used in the training phase. We show in the continuous-time limit and in the discrete network that the variants reduce the growth rate bounds by decreasing the constants in the stability conditions. In some cases, the constants become invariant to depth. Computational experiments on the proposed variants show that the accuracy of ResNet is preserved. It is also observed that for the image classification problem, ResNet and its variants monotonically improve accuracy by increasing depth, which is likely related to the well-posedness of the optimal control problem.

1.2 Overview

This paper is organized as follows. In Sect. 2, we analyze the forward stability of ResNet and its two variants in continuous-time by relating them to optimal control problems with differential inclusions. In Sect. 3, we prove the forward stability of the variants in the discrete setting, which includes the full network structure. In Sect. 4, experimental results are presented and show that the variants preserve the same accuracy as ResNet, with stronger stability bounds theoretically.

2 Continuous-Time ResNet System

The standard (post-activation) form of a residual layer can be written as an iterative update defined by:

$$x^{n+1} = \sigma(x^n - \tau\, A_2^n\, \sigma(A_1^n x^n + b_1^n) + \tau\, b_2^n), \tag{1}$$

where $x^n \in \mathbb{R}^d$ is a vector representing the features in Layer n, $A_i^n \in \mathbb{R}^{d\times d}$ (for $i = 1, 2$) are the weight matrices, $b_i^n \in \mathbb{R}^d$ (for $i = 1, 2$) are the biases, and σ is the rectified linear unit (ReLU). The parameter $\tau > 0$ can be absorbed into the weight matrix A_2^n; however, when scaled in this way, the iterative system resembles a forward Euler update applied to some differential equation. The connection between the residual layers (for a single activation function) and differential equations has been observed in [33].

By setting the (outer) activation in Eq. (1) to ReLU, one is imposing the "obstacle" $x \geq 0$ to the system; see for example [28, 34, 41] and the citations within. Letting $\tau \to 0_+$ in Eq. (1) leads to a differential inclusion:

$$-\frac{\mathrm{d}}{\mathrm{dt}}x(t) - A_2(t)\,\sigma(A_1(t)x(t) + b_1(t)) + b_2(t) \in \partial I_{\mathbb{R}^d_+}(x), \tag{2}$$

where $I_{\mathbb{R}^d_+}$ is the indicator function of the set $\mathbb{R}^d_+$ [see Eq. (38)]. It is possible to show that Eq. (1) is a consistent discretization of Eq. (2). Equation (1) is essentially the forward-backward splitting [11, 36], where the projection onto the "obstacle" is implicit and the force $A_2(t)\,\sigma(A_1(t)x(t) + b_1(t)) + b_2(t))$ is explicit.

2.1 Connection of Neural Networks to Optimal Control Problems

Let $\mathcal{D}$ be a dataset with C classes of images. Given an input image $x^0 \in \mathcal{D}$ to a nerual network, let $y \in \mathbb{R}^C$ be the one-hot encoding label vector associated with x^0, and let $x^N \in \mathbb{R}^C$ be the output of the network. The label vector y can be considered as the true distribution of x^0 over the C possible classes. To obtain a predicted distribution of x^0 from the network and compare it with y, typically one applies the softmax normalization function to the output x^N of the network, so that the loss to be minimized for each input $x^0 \in \mathcal{D}$ is $H(y, S(x^N))$, where H and S denote the cross entropy and softmax functions, respectively.

Let $\mathcal{I}$ be the index set for the layers in the network. Given an index $n \in \mathcal{I}$, let A^n and b^n be the weight and bias in Layer n, respectively (when applicable). To minimize the classification error of the network, one usually solves the following optimization problem:

$$\min_{\substack{A^n,\, b^n \\ \text{for } n \in \mathcal{I}}} \sum_{x^0 \in \mathcal{D}} H(y, S(x^N)) + \sum_{n \in \mathcal{I}} R_n(A^n), \tag{3}$$

where x^n satisfying Eq. (1) and R^n represents the regularizer for A^n.

The time parameter, $t > 0$, in Eq. (2) refers to the continuous analog of the depth of a neural network (without pooling layers). In the limit, as the depth of a neural network increases, one could argue that the behavior of the network (if scaled properly by τ) should mimic that of a continuous dynamical system. Thus, the training of the network, i.e., learning A^n and b^n given x^0 and x^N, is an optimal control problem. Therefore, questions on the stability of the forward propagation, in particular, do the features remain bounded and how sensitive are they to small changes in the input image, are also questions about the well-posedness of the continuous control problem.

2.2 Stability of Continuous-time ResNet

In this section, we will show that the continuous-time ResNet system is well-posed and that the forward propagation of the features is stable in the continuous-time. First, note that the function $I_{\mathbb{R}^d_+}$ is convex, and thus, its subdifferential $\partial I_{\mathbb{R}^d_+}(x)$ is monotone and is characterized by a normal cone:

$$\partial I_{\mathbb{R}^d_+}(x) = \mathcal{N}_{\mathbb{R}^d_+}(x) := \{\xi \in \mathbb{R}^d \,:\, \langle \xi, y - x\rangle \leq 0 \text{ for all } y \in \mathbb{R}^d_+\}.$$

By Remark 4, we have $\mathrm{prox}_{\gamma I_{\mathbb{R}^d_+}}(x) = \sigma(x)$. Therefore, Eq. (1) is indeed a discretization of Eq. (2), where the subdifferential of the indicator function is made implicit by the proximal operator (projection onto $\mathbb{R}^d_+$). We will use both the subdifferential and normal cone interpretation to make the arguments more direct.

Consider differential inclusions of the form:

$$-\frac{\mathrm{d}}{\mathrm{d}t}x(t) \in \mathcal{N}_{\mathbb{R}^d_+}(x(t)) + F(t, x(t)), \tag{4}$$

which have been studied within the context of optimal control and sweeping processes. The existence of solutions are given by Theorem 1 in [10] (see "Appendix C"). The continuous-time ResNet, characterized by Eq. (2), is a particular case of Eq. (4) with the forcing function F set to:

$$F(t, x(t)) := A_2(t)\, \sigma(A_1(t)x(t) + b_1(t)) - b_2(t).$$

Thus, Eq.(2) is equivalent to:

$$-\frac{\mathrm{d}}{\mathrm{d}t}x(t) \in \mathcal{N}_{\mathbb{R}^d_+}(x(t)) + A_2(t)\, \sigma(A_1(t)x(t) + b_1(t)) - b_2(t). \tag{5}$$

The following result shows that under certain conditions, Eq. (5) has a unique absolutely continuous solution in $\mathbb{R}^d_+$.

Theorem 1 (Continuous-time ResNet, existence of solutions) *Let $c > 0$, $x : \mathbb{R}_+ \to \mathbb{R}^d$, $A_i : \mathbb{R}_+ \to \mathbb{R}^{d\times d}$, $b_i : \mathbb{R}_+ \to \mathbb{R}^d$ (for $i = 1, 2$), and σ be the rectified linear unit. Assume that*

$$\|A_1(t)\|_{\ell^2(\mathbb{R}^d)}\, \|A_2(t)\|_{\ell^2(\mathbb{R}^d)} \le c$$

for all $t > 0$. Then for any $x_0 \in \mathbb{R}^d_+$, *the following dynamic process:*

$$\begin{cases} -\dfrac{\mathrm{d}}{\mathrm{d}t}x(t) \in \mathcal{N}_{\mathbb{R}^d_+}(x(t)) + A_2(t)\, \sigma(A_1(t)x + b_1(t)) - b_2(t) & \text{a.e. } t > 0 \\ x(0) = x_0 \end{cases} \tag{6}$$

has one and only one absolutely continuous solution $x \in \mathbb{R}^d_+$.

Theorem 1 shows that in the continuous-time case, there exists only one path in the feature space. Thus, as the number of residual layers increases in a network, we should expect the residual layers to approximate one consistent path from the input to the output. The requirement is that the matrices A_1 and A_2 are bounded in ℓ^2, which is often imposed in the training phase via the optimization problem (e.g., choosing a proper form of R_n in Eq. (3)). The stability bounds in the following theorems are derived from the subdifferential interpretation.

Theorem 2 (Continuous-time ResNet, stability bounds) *With the same assumptions as in Theorem 1, the unique absolutely continuous solution x to Eq. (2) is stable in the following sense:*

$$\begin{aligned} \|x(t)\|_2 \le{} & \|x(0)\|_2 \exp\left(\int_0^t \|A_1(s)\|_2\, \|A_2(s)\|_2 \,\mathrm{d}s\right) \\ & + \int_0^t \big(\|A_2(s)\|_2\, \|b_1(s)\|_2 \\ & + \|\sigma(b_2(s))\|_2\big) \exp\left(\int_s^t \|A_1(r)\|_2\, \|A_2(r)\|_2 \,\mathrm{d}r\right) \mathrm{d}s \end{aligned} \tag{7}$$

for all $t > 0$. In addition, if y is the unique absolutely continuous solution to Eq. (2) with input $y(0)$, then for all $t > 0$,

$$\begin{aligned} & \|x(t) - y(t)\|_2 \\ & \quad \le \|x(0) - y(0)\|_2 \exp\left(\int_0^t \|A_1(s)\|_2 \|A_2(s)\|_2 \,\mathrm{d}s\right). \end{aligned} \tag{8}$$

Equation (7) provides an upper-bound to the growth rate of the features in the continuous-time network, and Eq. (8) shows that the sensitivity of the network to perturbations depends on the size of the weight matrices. Without any additional assumptions on the weights A_i and/or biases b_i (for $i = 1, 2$) (except for uniform-in-time boundness), the solution to Eq. (2) and the perturbations can grow exponentially with respect to the depth. By testing a standard ResNet code,[1] we observed that without batch normalization, the norms of the features increase by a factor of 10 after about every 3–4 residual layers. Thus, in very deep networks there could be features with large values, which are typically not well-conditioned. It is interesting to note that with batch normalization, experiments show that the norms of the features grow but not as dramatically.

In practice, regularization is added to the optimization problem (often by penalizing the norms of the weight matrices) so that the trained network does not overfit the training data. In addition, Theorem 2 shows that for a deep network, the stability of the continuous-time dynamics depends on the norms of the weight matrices A_i. Thus, with sufficient regularization on the weights, the growth rate can be controlled to some extent.

2.3 Continuous-Time Stability of Variants of ResNet

There are multiple ways to control the feature-norms in deep ResNets. The results in Sect. 2.2 indicate that for a

[1] We used the open-sourced code from the TFLearn library on GitHub.

general residual layer, the regularization will control the growth rates. Alternatively, by changing the structure of the residual layer through constraints on A_i, the dynamics will emit solutions that satisfy smaller growth bound. In Sect. 4, computational experiments show that the variants produce similar accuracy results to the original ResNet [16] with provably tighter bounds.

We propose two variants on the residual layer, which improve the stability estimates from Sect. 2.2. The first form improves the feature-norm bound by imposing that $A_2(t) \in \mathbb{R}^{d\times d}_+$:

$$\begin{aligned} &-\frac{\mathrm{d}}{\mathrm{d}t}x(t) - A_2(t)\,\sigma(A_1(t)x(t)+b_1(t)) \\ &\quad + b_2(t) \in \partial I_{\mathbb{R}^d_+}(x) \quad \text{with } A_2(t) \in \mathbb{R}^{d\times d}_+. \end{aligned} \tag{9}$$

The network associated with residual layers characterized by Eq. (9) will be called **ResNet-D**. This is in reference to the decay of the system when the biases are identically zero for all time. When the biases are nonzero, one can show the following improved bound (as compared to Theorem 2).

Theorem 3 *With the same assumptions as in Theorem 1, the unique absolutely continuous solution x to Eq. (9) is stable in the following sense:*

$$\|x(t)\|_2 \le \|x(0)\|_2 + \int_0^t \|\sigma(b_2(s))\|_2 \,\mathrm{d}s \tag{10}$$

for all $t > 0$.

Theorem 3 shows that the continuous-time feature vector does not grow as quickly as the depth of the network increases. In order to improve Eq. (8), which measures the sensitivity of the features to changes in the inputs, we impose a symmetric structure to the weights:

$$-\frac{\mathrm{d}}{\mathrm{d}t}x(t) - A(t)^T\,\sigma(A(t)x(t)+b_1(t)) + b_2(t) \in \partial I_{\mathbb{R}^d_+}(x). \tag{11}$$

We refer to the network associated with residual layers characterized by Eq. (11) as **ResNet-S**. The forcing function:

$$F(t,x) = -A(t)^T\,\sigma(A(t)x(t)+b_1(t))$$

in Eq. (11) was proposed in [5, 33] and is motivated by parabolic differential equations. Similarly, Eq. (11) is the nonlinear parabolic differential equation which (under certain conditions) arises from an obstacle problem using the Dirichlet energy [28, 34, 41]. The following result shows that Eq. (11) improves the bounds in Theorem 2.

Theorem 4 *With the same assumptions as in Theorem 1, the unique absolutely continuous solution x to Eq. (11) is stable in the following sense:*

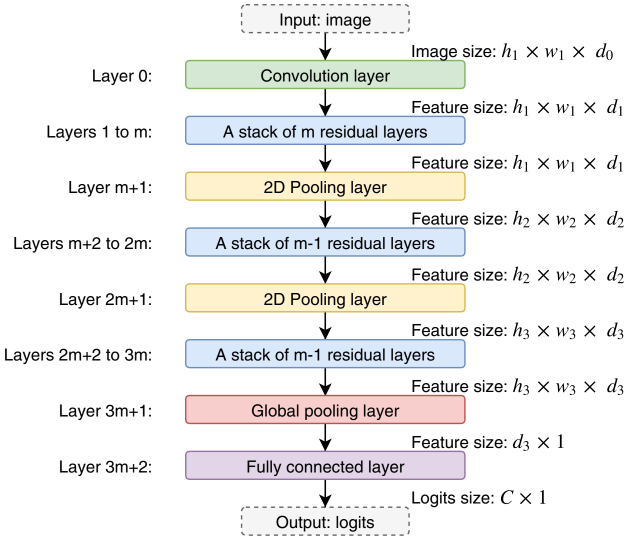

Input: image
Image size: $h_1 \times w_1 \times d_0$
Layer 0:
Convolution layer
Feature size: $h_1 \times w_1 \times d_1$
Layers 1 to m:
A stack of m residual layers
Feature size: $h_1 \times w_1 \times d_1$
Layer m+1:
2D Pooling layer
Feature size: $h_2 \times w_2 \times d_2$
Layers m+2 to 2m:
A stack of m-1 residual layers
Feature size: $h_2 \times w_2 \times d_2$
Layer 2m+1:
2D Pooling layer
Feature size: $h_3 \times w_3 \times d_3$
Layers 2m+2 to 3m:
A stack of m-1 residual layers
Feature size: $h_3 \times w_3 \times d_3$
Layer 3m+1:
Global pooling layer
Feature size: $d_3 \times 1$
Layer 3m+2:
Fully connected layer
Logits size: $C \times 1$
Output: logits

Fig. 1 Architecture of ResNet-D and ResNet-S for the image classification problem. The input image is of size $h_1 \times w_1 \times d_0$, and is contained in a dataset with C classes. The dimension of the features is changed through the network, where $h_{i+1} = \lceil h_i/2 \rceil$, $w_{i+1} = \lceil w_i/2 \rceil$, and $d_{i+1} = 2d_i$ (for $i = 1, 2$)

$$\|x(t)\|_2 \le \|x(0)\|_2 + \int_0^t \left\| \sigma\left(-A(s)^T\sigma(b_1(s)) + b_2(s)\right) \right\|_2 \mathrm{d}s \tag{12}$$

for all $t > 0$. In addition, if *y is the unique absolutely continuous solution to Eq. (11) with input y(0), then for all* $t > 0$,

$$\|x(t) - y(t)\|_2 \le \|x(0) - y(0)\|_2. \tag{13}$$

Equation (13) shows that the features are controlled by perturbations in the inputs, *regardless of the depth*.

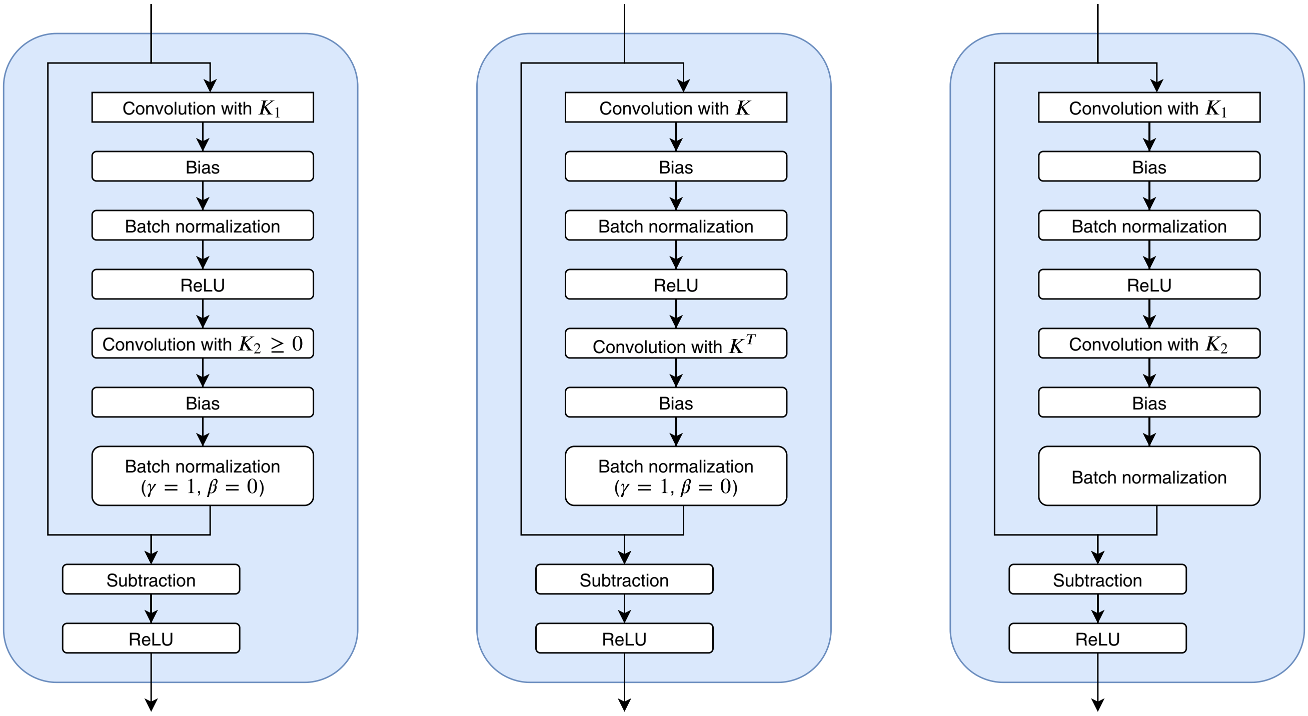
The proofs of Theorems 1–4 are provided in "Appendix B."

3 Discrete Stability of ResNet-D and ResNet-S

Since DNNs are discrete, in this section we provide discrete stability bounds on the features, similar to those in Sect. 2.

3.1 Architecture of ResNet-D and ResNet-S

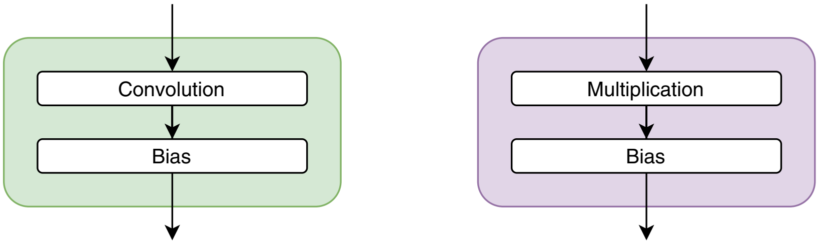
We will discuss the architecture used for the problem of image classification and the associated architecture of ResNet-D and ResNet-S. The base structure of the networks is shown in Fig. 1, which is a variant of the standard architecture for ResNets [16]. The input to a network is



(a) ResNet-D (Eq. (14)) **(b)** ResNet-S (Eq. (15)) **(c)** Original [18]

Fig. 2 The residual layers

Fig. 3 The linear layers



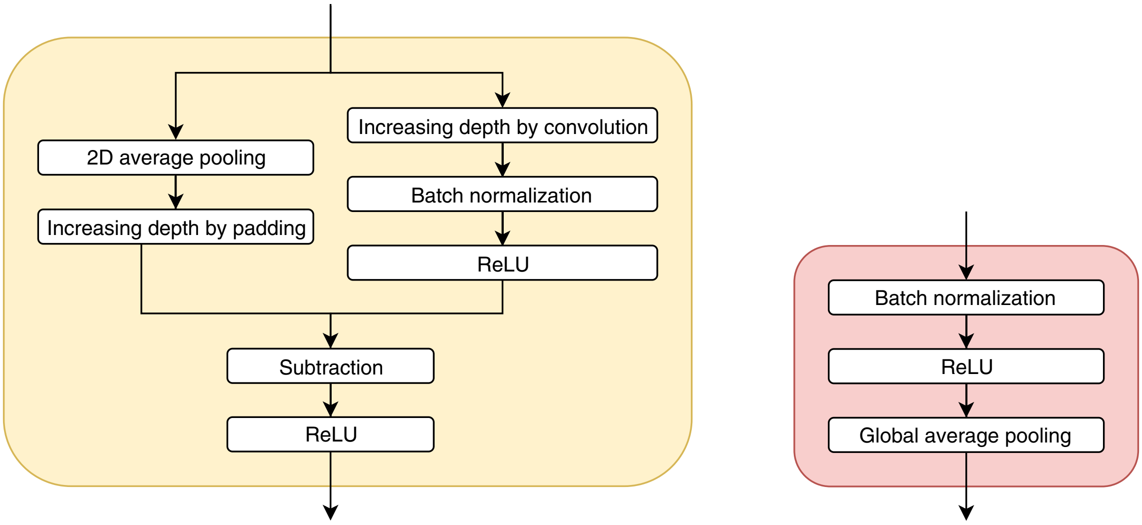
(a) Convolution (Eq. (16)) **(b)** Fully Connected (Eq. (17))

an image in $\mathbb{R}^{h_1 \times w_1 \times d_0}$, and the first layer in the network is a convolution layer (shown in Fig. 3a), which increases the depth of the input image to d_1. The convolution layer is followed by a stack of m residual layers, which take one of the two forms detailed in Fig. 2a, b. For comparison, we provide the original form of the residual layer in [16] in Fig. 2c. The residual block is followed by a 2D pooling layer (shown in Fig. 4a), which halves the resolution and doubles the depth of the incoming feature (i.e., $h_2 = \lceil h_1/2 \rceil$, $w_2 = \lceil w_1/2 \rceil$, and $d_2 = 2d_1$). The resulting feature is then processed by a stack of $m - 1$ residual layers, a 2D pooling layer (i.e., $h_3 = \lceil h_2/2 \rceil$, $w_3 = \lceil w_2/2 \rceil$, and $d_3 = 2d_2$), and another stack of $m - 1$ residual layers. Finally, we reduce the dimension of the resulting feature by adding a global average pooling layer (shown in Fig. 4b) and a fully connected layer (shown in Fig. 3b).

For simplicity of the subsequent analysis, we will use the vector form of the operations. Definitions are provided in "Appendix A." Let x^0 be the vector representing the input to the network, i.e., $x^0 \in \mathbb{R}^{h_1 w_1 d_0}$. The equations that characterize the layers in Figs. 2, 3 and 4 are defined as follows:

$$\text{the ResNet-D layer:} \quad x^{n+1} := \sigma(x^n - A_2^n \, \sigma(A_1^n x^n + b_1^n) + b_2^n) \text{ with } A_2^n \geq 0, \tag{14}$$

$$\text{the ResNet-S layer:} \quad x^{n+1} := \sigma(x^n - (A^n)^T \, \sigma(A^n x^n + b_1^n) + b_2^n), \tag{15}$$



(a) 2D Pooling (Eq. (18))

(b) Global Pooling (Eq. (19))

Fig. 4 The pooling layers

$$\text{the convolution layer:} \quad x^{n+1} := A^n x^n + b^n, \tag{16}$$

$$\text{the fully connected layer:} \quad x^{n+1} := W^n x^n + b^n, \tag{17}$$

$$\begin{aligned}\text{the 2D pooling layer:} \quad x^{n+1} &:= \sigma\big(E(P_2(x^n)) \\ &-\sigma\big((A^n)_{|s=2}\, x^n + b^n\big)\big),\end{aligned} \tag{18}$$

$$\text{the global pooling layer:} \quad x^{n+1} := P_g(\sigma(x^n)), \tag{19}$$

where x^n is the input to Layer n, A^n is the matrix associated with the 2D convolution operation with K^n in Layer n (when applicable), b^n is the bias in Layer n, and W^n is the weight matrix in the fully connected layer.

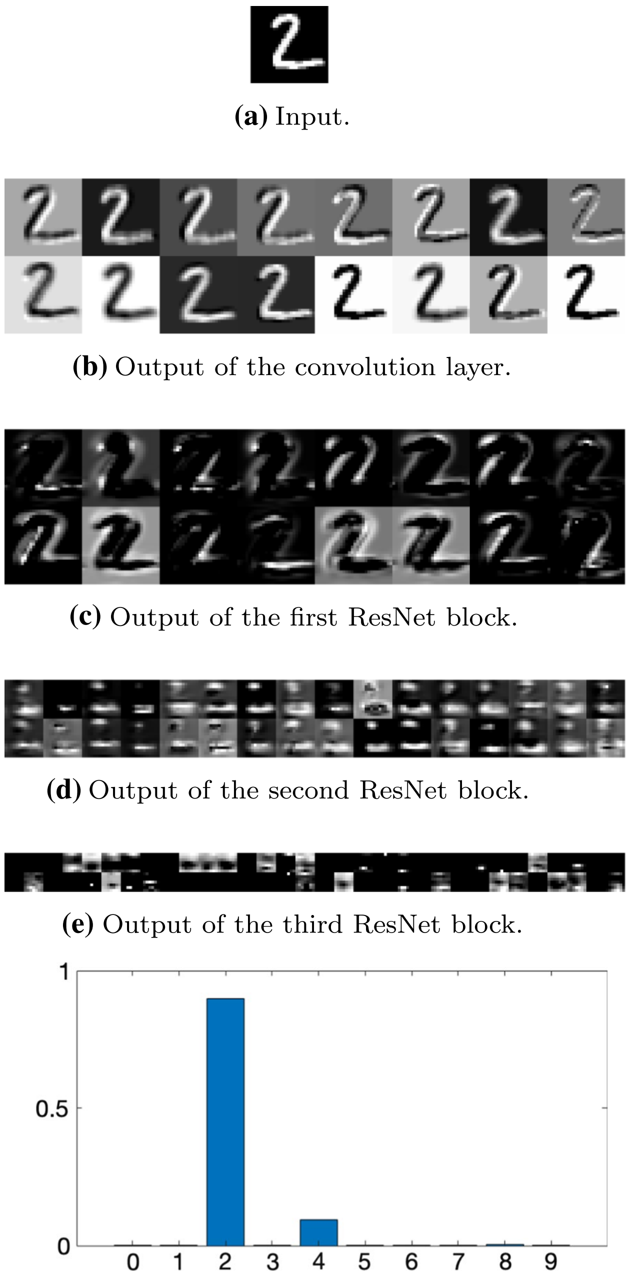
The forward propagation of the network is shown in Figs. 5 and 6, which display (channel-wise) the output feature of the indicated layer/block of ResNet-D and ResNet-S, respectively. As an example, the input image is a hand-written digit “2” from the MNIST dataset. The first convolution layer (Layer 1) returns low-level features of the digit (Figs. 5a, 6a). The low-level features are then processed by a stack of residual layers (Layers 1 to m), which yields mid-level features of the digit (Figs. 5b, 6b). The mid-level features are then downsampled by a 2D pooling layer (Layer $m+1$) and processed by a stack of residual layers (Layers $m+2$ to $2m$), which yields high-level features of the digit (Figs. 5c, 6c). Similarly, after a 2D pooling layer (Layer $2m+1$) and a stack of ResNet layers (Layers $2m+2$ to $3m$), the high-level features become linearly separable classifiers (Figs 5d, 6d). The global pooling layer (Layer $3m+1$) and the fully connected layer (Layer $3m+2$) convert the linearly separable classifiers to a vector that can be used to extract a predicted probability distribution of the input. For example, the predicted probability distributions in Figs. 5e and 6e are obtained by applying the softmax normalization function to the output of the fully connected layer, where the value of the ith bar represents the predicted probability that the input digit is i (for $i = 0, 1, \dots, 9$).

Note that the mid-level features resemble images filtered by edge detectors, similar to CNNs and the standard ResNet. Experimentally, we see that a ResNet-D layer produces a kernel K_1 which looks like a gradient stencil and a kernel K_2 which acts as a rescaled averaging filter. Thus, the first block in ResNet-D resembles a *nonlinear (possibly non-local) transport system*. The non-locality comes from the smoothing process determined by K_2. In ResNet-S, since the kernels K from the first residual block are gradient-like stencils, the first block in ResNet-S resembles a *nonlinear diffusive system*.

3.2 Forward Stability of ResNet-D and ResNet-S

The stability of forward propagation through ResNet-D and ResNet-S can determine both the sensitivity of the network to changes in the inputs and the level of consistency in various computations. If the norms of the weight matrices are small enough, then both the output of the network and changes in the features can be controlled by the inputs. In particular, we have the following (discrete) stability results for ResNet-D and ResNet-S.

Theorem 5 (Forward Stability, ResNet-D) *Consider a network defined in Fig. 1, where the ResNet layers are defined*



(a) Input.

(b) Output of the convolution layer.

(c) Output of the first ResNet block.

(d) Output of the second ResNet block.

(e) Output of the third ResNet block.

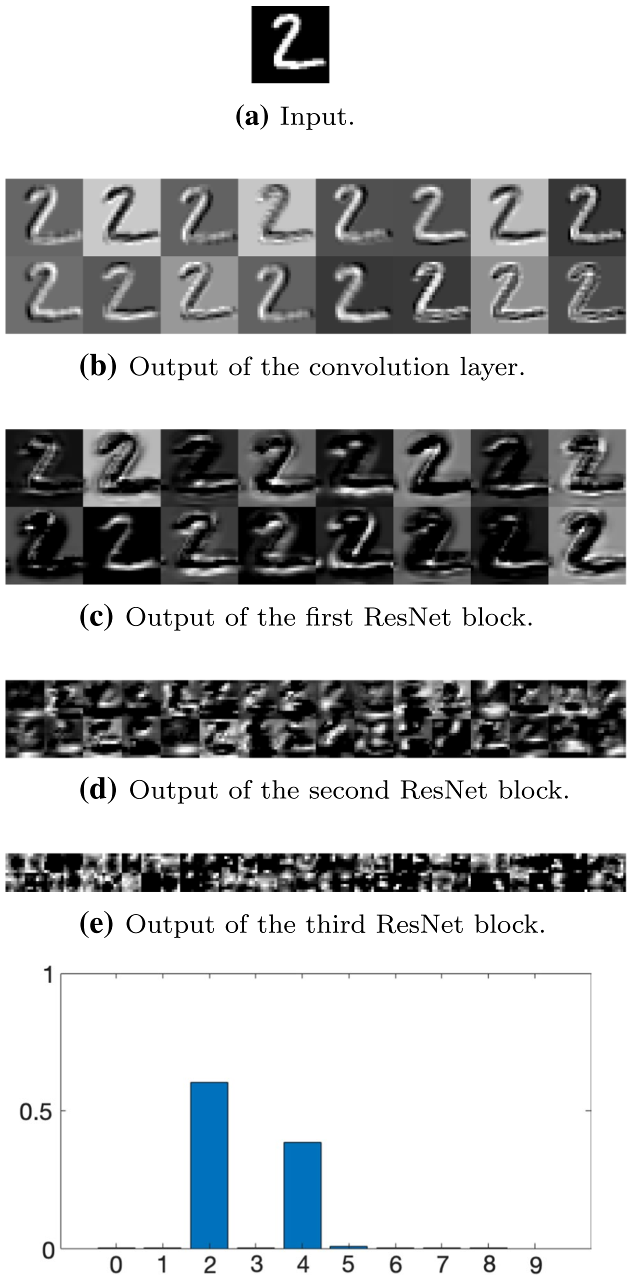
(f) Predicted probability distrubution.

Fig. 5 Forward propagation of ResNet-D

in Fig. 2a. Let x^0 be the vectorization of the input to the network, i.e., $x^0 \in \mathbb{R}^{h_1w_1d_0}$, and for each filter K^n in Layer n (when applicable), let A^n be the matrix associated with the 2D convolution operation with K^n. Assume that

$$\|A^0\|_{\ell^2(\mathbb{R}^{h_1w_1d_0})\to\ell^2(\mathbb{R}^{h_1w_1d_1})} \le 1, \tag{20a}$$

$$\|W^{3m+2}\|_{\ell^2(\mathbb{R}^{d_3})\to\ell^2(\mathbb{R}^{C})} \le 1. \tag{20b}$$



(a) Input.

(b) Output of the convolution layer.

(c) Output of the first ResNet block.

(d) Output of the second ResNet block.

(e) Output of the third ResNet block.

(f) Predicted probability distribution.

Fig. 6 Forward propagation of ResNet-S

Let x^n be the input to Layer n and x^N be the output of the network, where $N := 3m + 3$. Then the network is ℓ^2-stable in the sense that:

$$\|x^N\|_{\ell^2(\mathbb{R}^C)} \le \|x^0\|_{\ell^2(\mathbb{R}^{h_1w_1d_0})} + c(b^0, b^1, \dots, b^{N-1}), \tag{21}$$

where $c(b^0, b^1, \dots, b^{N-1})$ is a constant depending on the ℓ^2 norms of the biases in the network; see Eq. (45). if $y^0 \in \mathbb{R}^{h_1w_1d_0}$ is the vectorization of another input, then:

$$\begin{aligned}&\|x^N - y^N\|_{\ell^2(\mathbb{R}^C)}\\&\quad\le a(A^0, A^1, \dots, W^{N-1})\,\|x^0 - y^0\|_{\ell^2(\mathbb{R}^{h_1 w_1 d_0})},\end{aligned} \tag{22}$$

where $a(A^0, A^1, \dots, W^{N-1})$ is a constant depending on the ℓ^2 norms of the filters and weights in the network; see Eq. (51).

Note that the bounds on the growth of the features, Eq. (21), do not directly depend on the filters and weights, since the system (without the biases) decays. The sensitive bound, Eq. (22), depends on the ℓ^2 norms of the filters and weights, which are controlled by the regularizer [see Eq. (3)]. For ResNet-S, we can improve the constant in the sensitivity bound as follows.

Theorem 6 (Forward Stability, ResNet-S) *Consider a network defined in Fig. 1, where the residual layers are defined in Fig. 2b. Let x^0 be the vectorization of the input to the network, i.e., $x^0 \in \mathbb{R}^{h_1 w_1 d_0}$, and for each filter K^n in Layer n (when applicable), let A^n be the associated matrix. Assume that*

$$\|A^0\|_{\ell^2(\mathbb{R}^{h_1 w_1 d_0}) \to \ell^2(\mathbb{R}^{h_1 w_1 d_1})} \le 1, \tag{23a}$$

$$\|W^{3m+2}\|_{\ell^2(\mathbb{R}^{d_3}) \to \ell^2(\mathbb{R}^C)} \le 1, \tag{23b}$$

and that for each filter K^n in a residual layer:

$$\|A^n\|_{\ell^2} \le \sqrt{2}, \tag{24}$$

where $\|\cdot\|_{\ell^2}$ denotes the induced (matrix) norm. Let x^n be the input of Layer n and x^N be the output of the network, where $N := 3m + 3$. Then the network is ℓ^2-stable in the sense that:

$$\|x^N\|_{\ell^2(\mathbb{R}^C)} \le \|x^0\|_{\ell^2(\mathbb{R}^{h_1 w_1 d_0})} + c(b^0, b^1, \dots, b^{N-1}), \tag{25}$$

where $c(b^0, b^1, \dots, b^{N-1})$ is a constant depending on the ℓ^2 norms of the biases in the network; see Eq. (55). If $y^0 \in \mathbb{R}^{h_1 w_1 d_0}$ is the vectorization of another input, then:

$$\|x^N - y^N\|_{\ell^2(\mathbb{R}^C)} \le a(A^0, A^1, \dots, W^{N-1})\|x^0 - y^0\|_{\ell^2(\mathbb{R}^{h_1 w_1 d_0})}, \tag{26}$$

where $a(A^0, A^1, \dots, W^{N-1})$ is a constant independent of the depth of the residual block; see Eq. (57).

Equation (26) is useful since it implies that, as long as one constrains the norms of the filters such that $\|A^n\|_{\ell^2} \le \sqrt{2}$, the network will be stable for arbitrarily many residual layers.

The proofs of Theorems 5 and 6 are provided in "Appendix B."

Remark 1 The conclusions in Theorems 5 and 6 are still valid if the ℓ^2 norm in Eqs. (20) and (23) are replaced by the Frobenius norm, since given a matrix A, we have $\|A\|_{\ell^2} \le \|A\|_F$.

Remark 2 By Eq. (29), given an input feature x, the output y of the following concatenation of operations:

2D convolution → batch normalization

is obtained by:

$$y := \frac{\gamma(Ax - \mu)}{\sigma} + \beta, \tag{27}$$

where the constants μ and σ depend on the mini-batch containing x. For the forward propogation, if we set $\tilde{A} := \gamma A/\sigma$ and $\tilde{b} := -\gamma\mu/\sigma + \beta$, then Eq. (27) can be rewritten as:

$$y := \tilde{A}x + \tilde{b}.$$

To make sure that A satisfies the constraints of $\tilde{A}$, for example the correct sign in Eq. (15), we fix the parameters γ and β in the second batch normalization in the residual layers ($\gamma = 1$ and $\beta = 0$ for the second batch normalization in Fig. 2b).

4 Computational Experiments

We test the two proposed networks on the CIFAR-10 and CIFAR-100 datasets. The data are preprocessed and augmented as in [16].

All filters in the network are of size 3 × 3, and we assume that the input feature to each layer satisfies the periodic boundary condition. The width of each residual block (i.e., d_i in Fig. 1) is 16, 32, and 64, respectively. For the first convolution layer, the filters are initialized using the uniform scaling algorithm [37]; for the residual layers and the 2D pooling layers, the filters are initialized using the variance scaling algorithm [15]. The weight in the fully connected layer is initialized with values drawn from a normal distribution $\mathcal{N}(0, \sigma^2)$, where $\sigma = (d_3 C)^{-1}$, except that values with magnitude more than 2σ are discarded and redrawn (i.e., the truncated normal distribution). Note that in [9] and the citations within, it was shown that under certain conditions on a neural network, randomly initialized gradient descent applied to the associated optimization problem converges to a globally optimal solution at a linear convergence rate.

The biases in the network are initialized to be zero. The batch normalization parameters γ and β, if trained, are initialized to be 1 and 0, respectively. The regularizer R_n in

Table 1 Regularizers in the optimization problem defined in Eq. (3)

Layer(s)	ResNet-D	ResNet-S
The ResNet layers	$R_n = \frac{\alpha_n}{2} \sum_{i=1}^{2} \|\text{vec}(K_i^n)\|_{\ell^2}^2 + I_{K_2^n \geq 0}$	$R_n = \frac{\alpha_n}{2} \sum_{i=1}^{2} \|\text{vec}(K_i^n)\|_{\ell^2}^2$
The convolution/pooling layer	$R_n = \alpha_n \|\text{vec}(K^n)\|_{\ell^2}^2$	$R_n = \frac{\alpha_n}{2} \|\text{vec}(K^n)\|_{\ell^2}^2$
The fully connected layer	$R_n = \alpha_n \|\text{vec}(W^n)\|_{\ell^2}^2$	$R_n = \frac{\alpha_n}{2} \|\text{vec}(W^n)\|_{\ell^2}^2$

Definitions of the layers are provided in Eqs. (14)–(19). The indicator function $I_{K_2^n \geq 0}$ represents the constraint $K_2^n \geq 0$, and α_n are some nonnegative constants. The regularization parameter α_n is set to be 10^{-4} for all n

Eq. (3) is listed in Table 1. To impose the constraints in Theorems 5 and 6, we regularize the Frobenius norms of the filters and weights (see Remark 1). The element-wise constraints $A_2^n \geq 0$ in Eq. (14) are imposed directly by adding the indicator function $I_{K_2^n \geq 0}$ to the regularizer; that is, in each gradient descent step, K_2^n is projected onto the positive set.

The network is trained using the mini-batch gradient descent algorithm, with mini-batch size equal to 128 (i.e., 391 steps/epoch). The initial learning rate is 0.1, and is divided by 10 after every 32,000 training steps. The training process is terminated after 93,500 training steps. The network is validated on the test images after every 500 training steps.

Remark 3 Our focus in the experiments is to examine the stability of the variants of ResNet. We would like to demonstrate that the variants achieve similar accuracy. Thus, we fix the hyperparameters (including: depth of the network, learning rate and batch size in the optimization, etc.) and do not tune them to the data. Better results can be achieved with tuning and the inclusion of additional steps, e.g., bottleneck layers.

In Table 2, we list the depth of the network and the number of trainable parameters in the optimization problem with a few different values of m (where m is the size of the first residual block). Here, the depth of a network is considered to be the number of (unique) filters and weights in the network; for example, each ResNet-D layer (Fig. 2a) contains two filters, and each ResNet-S layer (Fig. 2b) contains only one filter.

4.1 Effect of Depth on Test Accuracy

We train the network with different depths and analyze the effect of the depth on test accuracy. The resulting test accuracies over the training steps are shown in Fig. 7. In particular, we calculate the average of the test accuracy in the last 5000 training steps and list the results in Table 3. It can be seen from Fig. 7 and Table 7 that the test accuracy of both ResNet-D and ResNet-S increases as the network goes deeper (without any hyperparameter tuning). This result is consistent with the observation in [16] that a deeper ResNet tends to have higher test accuracy. The monotone improvement of accuracy in depth is likely related to the well-posedness of the optimal control problem [Eq. (6)]. The classification accuracy of the original ResNet can be found in "Appendix D."

4.2 Effect of Perturbation on Test Accuracy

We evaluate the trained networks on images with different types of perturbation. Given a test image x, its corrupted image is obtained via $x \mapsto x + \eta$, where two types of the additive noise η are considered:

$$\text{Unstructured:} \quad \eta \sim \mathcal{N}(0, \sigma^2), \tag{28a}$$

$$\text{Structured:} \quad \eta = \epsilon x_0, \tag{28b}$$

where x_0 is a fixed image chosen from the test images.

Table 2 Depth of the network and number of trainable parameters in the optimization problem

m	Depth	Trainable parameters
(a) ResNet-D on CIFAR-10		
3	18	0.223M
6	36	0.514M
9	54	0.805M
12	72	1.100M
(b) ResNet-S on CIFAR-10		
3	11	0.124M
6	20	0.270M
9	29	0.416M
12	38	0.561M

The depth counts the number of filters and weights in the network. The trainable parameters in Eq. (3) include all elements in the filters, weights, and biases, and the parameters in all batch normalization (if trained). The same network has 5760 more parameters on CIFAR-100

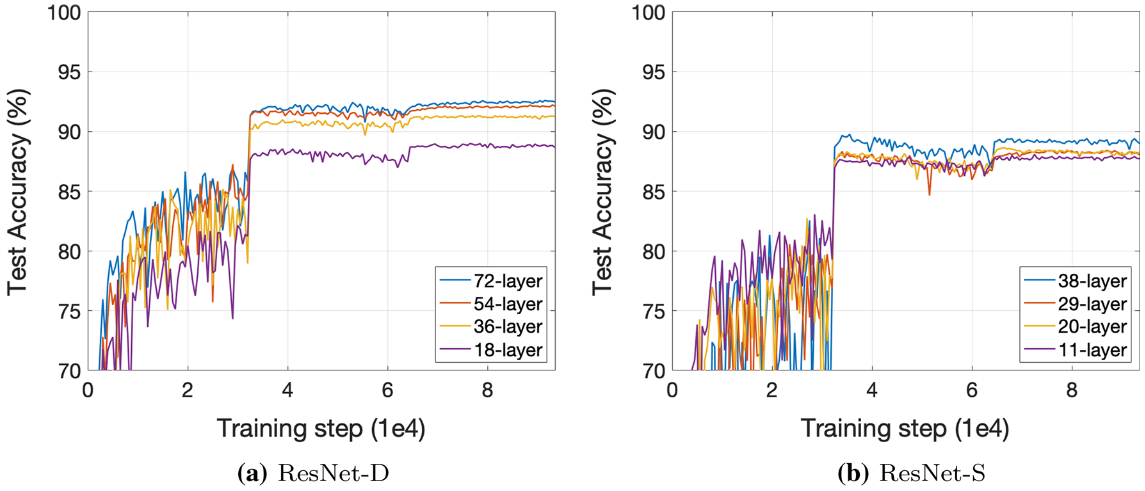


Fig. 7 Test accuracy of ResNet-D and ResNet-S on CIFAR-10. The test accuracy of both ResNet-D and ResNet-S tends to increase as the depth of the network increases

Table 3 Average of the test accuracy in the last 5000 training steps of ResNet-D and ResNet-S

Depth	Test accuracy (%)	
	CIFAR-10	CIFAR-100
(a) ResNet-D		
18	88.81	60.23
36	91.23	65.72
54	92.13	67.63
72	92.50	69.18
(b) ResNet-S		
11	87.78	58.17
20	88.16	61.79
29	88.18	62.14
38	89.09	63.17

In Table 4, we list the test accuracy of ResNet-D and ResNet-S on perturbed test images, which are evaluated using the learned parameters of the network from the last training step. Note that the networks are trained on the uncorrupted training set. The structured noise x_0 used in the experiments is shown in Fig. 8 and is added to the test images. Different values of σ and ϵ are used to vary the noise level of η. One can observe from Table 4a that when the noise level increases, the test accuracy of ResNet-D decreases. For low levels of perturbation, the accuracy remains high. We observe that deeper networks tend to have higher test accuracies after corruption of the test images. Similar conclusion can be drawn from Table 4b, in particular, that a deeper ResNet-S seems to be more robust to corrupted test data.

We illustrate the results in Figs. 9 and 10 using the trained 36-layer ResNet-D and 20-layer ResNet-S on three test images in CIFAR-10. The test images are labeled as "bird," "dog," and "horse," respectively. In Figs. 9 and 10, three test images and the corresponding corrupted images are shown, including the corresponding probability distributions predicted by the trained networks. One observation is that the probability of predicting the true label correctly tends to decrease as the corruption level increases. For example, consider the case where ResNet-S is applied to the "horse" image (the last two columns in Fig. 10). Figure 10a shows that the probability that the noise-free image x is a "horse" is 0.9985. When random noise η is added to x, i.e., $x \mapsto x + \eta$ with $\eta \sim \mathcal{N}(0, \sigma^2)$, the probability of correctly predicting $x + \eta$ to be a "horse" drops to 0.8750 and 0.7940 (for σ equal to 0.02 and 0.05, respectively). This is illustrated in Fig. 10b.

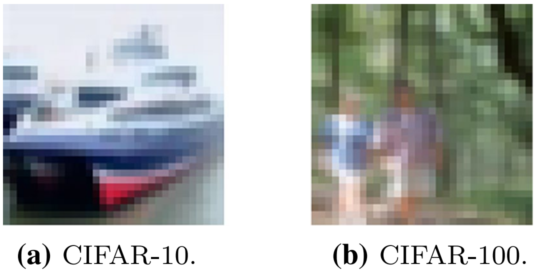
When the corruption level increases, the label with the second highest predicted probability may change. Take for example ResNet-S on the "dog" image (the middle two columns in Fig. 10). Let x be the original "dog" image. When random noise η is added to x, the second prediction made by the network changes from a "cat" (with probability 0.1410) to a "frog" (with probability 0.1717) (as σ increases from 0.02 to 0.05). This is within the stability bounds from Sect. 3. When we perturb a test image by another image (Fig. 8a), we observe similar stability results under this structured form of corruption. This is illustrated on the "bird" image in the first two columns of Fig. 9.

Equations (22) and (26) show that perturbation in the output depends on the perturbation in the input and the weight matrices in the network. In theory, if the norm of the additive noise to the input increases, perturbation in the output may be less controllable. Table 4 and Figs. 9 and 10 indicate that changes in the output may affect test accuracy.

Table 4 Test accuracy (%) with corrupted test images of ResNet-D and ResNet-S

Dataset	Depth	With no noise	With unstructured noise		With structure noise	
		$\sigma = \epsilon = 0$	$\sigma = 0.02$	$\sigma = 0.05$	$\epsilon = 0.25$	$\epsilon = 0.75$
(a) ResNet-D						
CIFAR-10	18	88.02	83.52	50.55	83.40	41.24
	36	90.74	85.48	56.70	84.54	35.18
	54	91.16	86.78	63.77	85.78	32.07
	72	91.79	86.95	61.73	86.95	40.25
CIFAR-100	18	59.04	46.15	16.01	55.68	29.92
	36	64.27	51.36	24.08	60.53	32.93
	54	66.78	52.82	23.08	62.55	33.17
	72	68.70	55.42	26.38	64.02	34.84
(b) ResNet-S						
CIFAR-10	11	87.73	82.43	52.44	82.24	33.66
	20	88.29	83.22	56.51	82.94	34.93
	29	88.05	83.68	55.82	83.50	36.80
	38	89.00	85.67	59.86	83.58	34.13
CIFAR-100	11	57.81	45.65	21.57	54.90	34.78
	20	61.01	47.18	21.09	57.93	37.17
	29	61.20	54.72	31.66	58.15	35.63
	38	65.05	53.56	22.99	61.02	36.66

Each network is trained on the uncorrupted training images of the dataset, and is evaluated using the learned parameters from the last training step on corrupted test images which are obtained via Eq. (28)



(a) CIFAR-10.

(b) CIFAR-100.

Fig. 8 The structured noise x_0 used in the experiments in Table 4. The use of x_0 is defined in Eq. (28b). **a** A test image in CIFAR-10 with label "ship." **b** A test image in CIFAR-100 with label "forest"

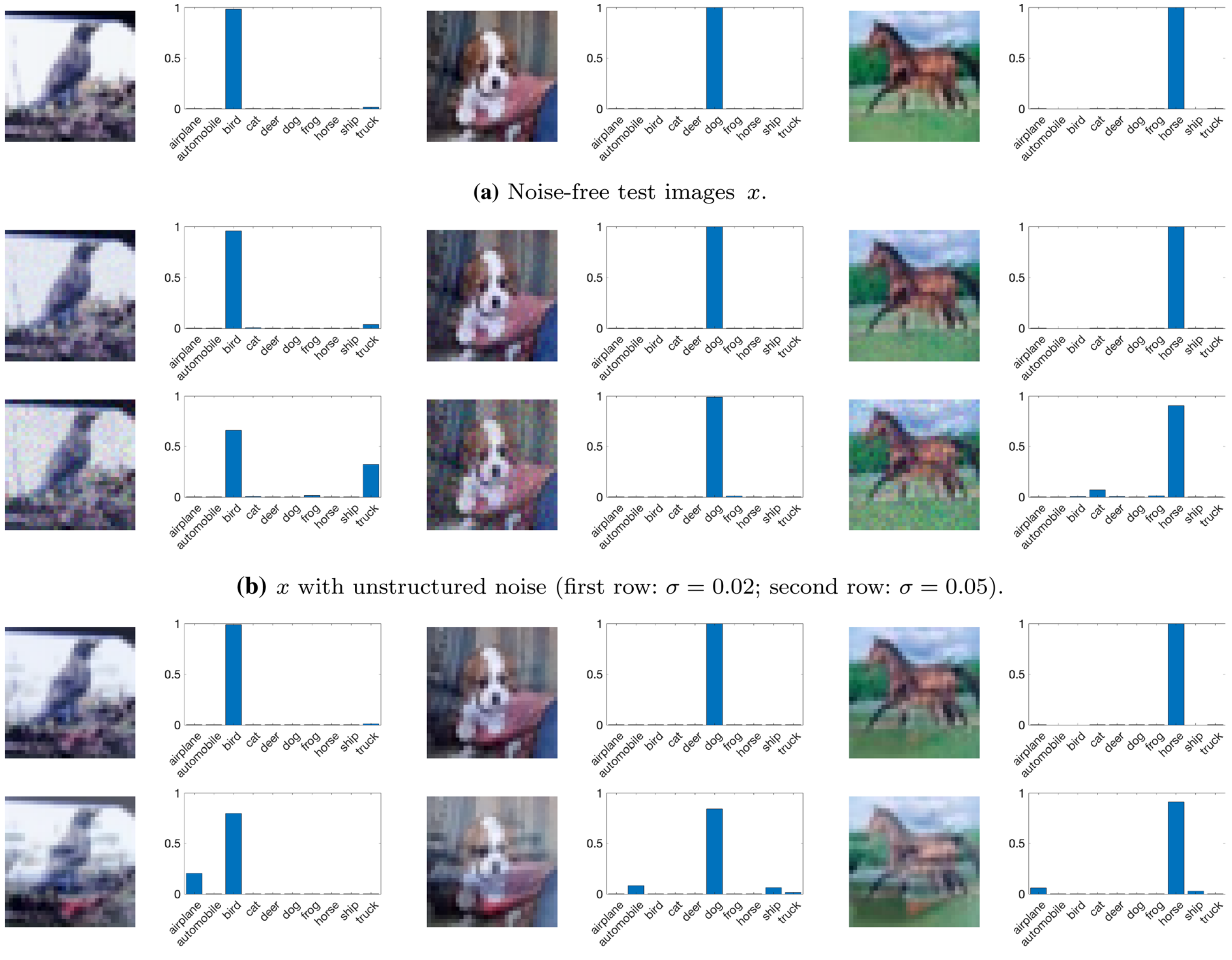
5 Discussion

We have provided a relationship between ResNet (or other networks with skip-connections) to an optimal control problem with differential inclusions and used this connection to gain some insights into the behavior of the network. We have shown that the system is well-posed and have provided growth bounds on the features. The continuous-time analysis is helpful in interpreting the success of networks with skip connections. For example, since the forward flow of well-posed dynamical systems will have regular paths between inputs and outputs, we should expect a similar result for very deep networks. This is likely a reason why DNNs with skip-connections generalize well, since similar inputs should follow similar paths and the skip-connections make the paths more regular.

In practice, ResNet and other DNNs have additional layers which are not currently captured by the optimal control formulation (for example, normalization and pooling). In this setting, we provided stability bounds for the entire network as a function of each of the layers' learnable parameters. In some cases, the network is stable regardless of its depth due to structural constraints or regularization. The constraints may also smooth the energy landscape so that the minimizers are flatter, which will be considered in future work.

It is also worth noting that ResNet and other DNNs are often "stabilized" by other operations. From experiments, one can observe that batch normalization has the additional benefit of controlling the norms of the features during forward propagation. Without batch normalization and without strong enough regularization, the features will grow unboundedly in the residual blocks. It would be interesting to analyze the role of different stabilizers in the network on the network's ability to generalize to new data.

Acknowledgements The authors acknowledge the support of AFOSR, FA9550-17-1-0125 and the support of NSF CAREER grant #1752116.



(a) Noise-free test images x.

(b) x with unstructured noise (first row: $\sigma = 0.02$; second row: $\sigma = 0.05$).

(c) x with structured noise (first row: $\epsilon = 0.25$; second row: $\epsilon = 0.75$).

Fig. 9 The trained 36-layer ResNet-D on corrupted test images from CIFAR-10. **a** Three noise-free test images x and the predicted probability distributions, **b** x with unstructured noise (i.e., $x + \eta$ with $\eta \sim \mathcal{N}(0, \sigma^2)$) and the predicted probability distributions, **c** x with structured noise (i.e., $x + \epsilon x_0$ with x_0 shown in Fig. 8a) and the predicted probability distributions

A DNN Operations in Vector Form

In this section, we provide definitions of a few DNN operations in vector form, as well as some basic properties.

Notations

Given a feature $x \in \mathbb{R}^{h \times w \times d}$, let x_i denote the ith channel of x, i.e.,

$$x = \left(x_1, x_2, \dots, x_d\right), \quad \text{with } x_i \in \mathbb{R}^{h \times w} \text{ for all } i = 1, 2, \dots, d,$$

and let $x_{i,j,k}$ denote the (i, j, k)th element in x. Given a feature $K \in \mathbb{R}^{n \times n \times d_1 \times d_2}$, let $K_{i,j}$ denote the (i, j)th subfilter of K, i.e.,

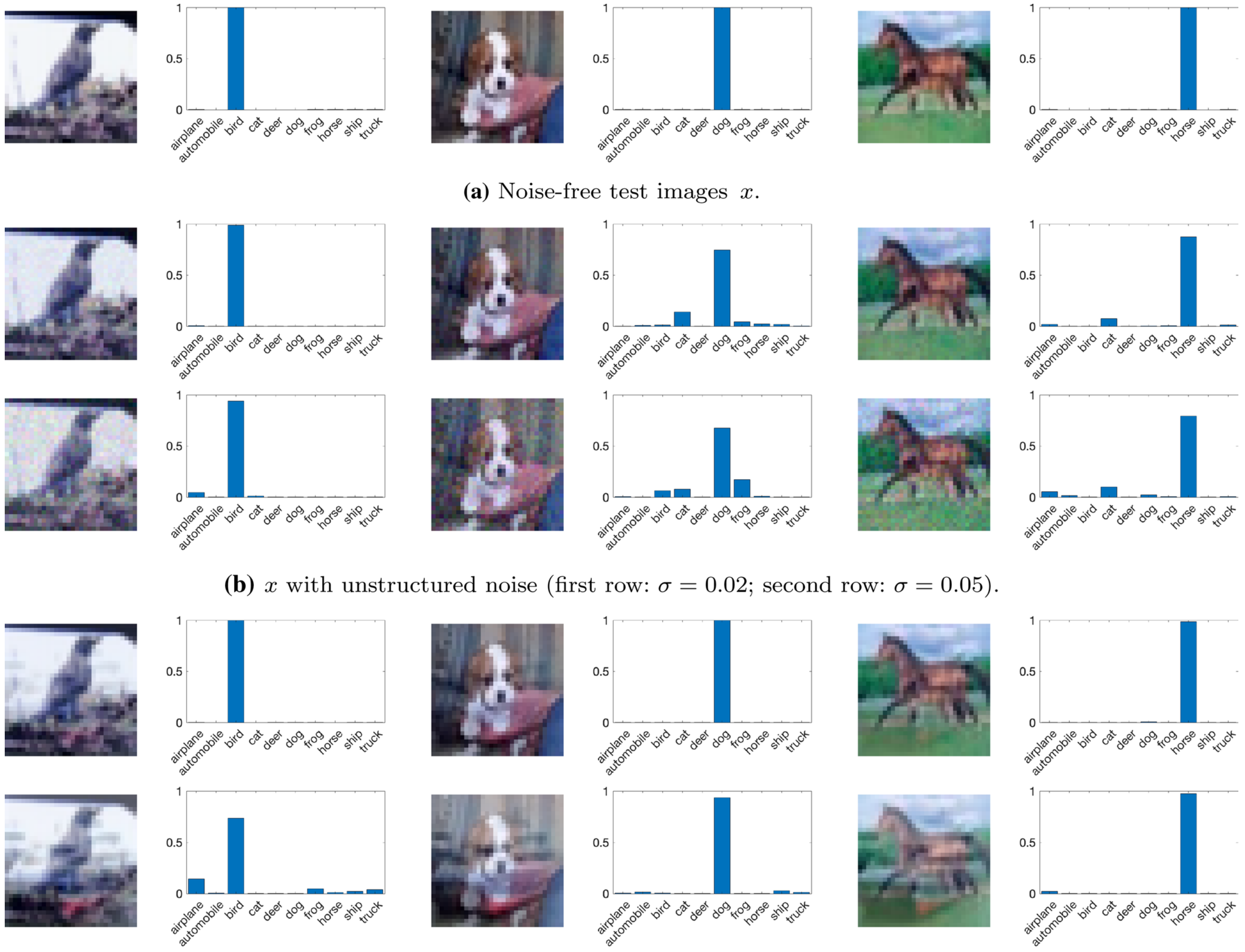
$$K := \begin{pmatrix} K_{1,1} & K_{1,2} & \cdots & K_{1,d_2} \\ K_{2,1} & K_{2,2} & \cdots & K_{2,d_2} \\ \vdots & \vdots & \ddots & \vdots \\ K_{d_1,1} & K_{d_1,2} & \cdots & K_{d_1,d_2} \end{pmatrix}$$

with $K_{i,j} \in \mathbb{R}^{n \times n}$ for all $i = 1, 2, \dots, d_1$ and $j = 1, 2, \dots, d_2$.

Definition 1 *Vectorization.* Let x be a feature in $\mathbb{R}^{h \times w \times d}$. The vectorization of x, denoted by $X := \text{vec}(x)$, is a vector in $\mathbb{R}^{hwd}$ such that

$$X_{(k-1)hw+(i-1)w+j} = x_{i,j,k},$$

for all $i = 1, 2, \dots, h$, $j = 1, 2, \dots, w$, and $k = 1, 2, \dots, d$.



(a) Noise-free test images x.

(b) x with unstructured noise (first row: $\sigma = 0.02$; second row: $\sigma = 0.05$).

(c) x with structured noise (first row: $\epsilon = 0.25$; second row: $\epsilon = 0.75$).

Fig. 10 The trained 20-layer ResNet-S on corrupted test images from CIFAR-10. **a** Three noise-free test images x and the predicted probability distributions, **b** x with unstructured noise (i.e., $x + \eta$ with $\eta \sim \mathcal{N}(0, \sigma^2)$) and the predicted probability distributions, **c** x with structured noise (i.e., $x + \epsilon x_0$ with x_0 shown in Fig. 8a) and the predicted probability distributions

Definition 2 *2D Convolution.* Let x be a feature in $\mathbb{R}^{h\times w\times d_1}$, K be a filter in $\mathbb{R}^{n\times n\times d_1\times d_2}$, and $y := K * x$ be a feature in $\mathbb{R}^{h\times w\times d_2}$. With $X = \text{vec}(x)$ and $Y = \text{vec}(y)$, one can derive a linear system $Y = AX$ which describes the forward operation of the 2D convolution $y = K * x$. The general form of A is:

$$A = \begin{pmatrix} A_{1,1} & A_{1,2} & \cdots & A_{1,d_1} \\ A_{2,1} & A_{2,2} & \cdots & A_{2,d_1} \\ \vdots & \vdots & \ddots & \vdots \\ A_{d_2,1} & A_{d_2,2} & \cdots & A_{d_2,d_1} \end{pmatrix},$$

where each $A_{i,j} \in \mathbb{R}^{hw\times hw}$ is a block-wise circulant matrix associated with subfilter $K_{j,i}$ (for all $i = 1, 2, \dots, d_1$ and $j = 1, 2, \dots, d_2$). The expression $Y = A_{|s=a}X$ denotes that the stride in the convolution $y = K * x$ is a.

Definition 3 *Adjoint of 2D Convolution.* Let x be a feature in $\mathbb{R}^{h\times w\times d_2}$ and K be a filter in $\mathbb{R}^{n\times n\times d_1\times d_2}$. The adjoint of the 2D convolution of x and K, denoted by $z := K^T * x$, is a feature in $\mathbb{R}^{h\times w\times d_1}$ such that $Z = A^T X$, where $X = \text{vec}(x)$, $Z = \text{vec}(z)$, A is the matrix associated with the 2D convolution operation with K defined in Definition 2, and A^T is the transpose of A in the matrix sense. The adjoint filter K^T is defined to be the filter whose matrix form is A^T.

Definition 4 Batch Normalization [19]. Let $\mathcal{B} := \{x^{(1)}, x^{(2)}, \dots, x^{(m)}\}$ be a batch of features. Batch normalization of $\mathcal{B}$ is defined as:

$$B(x^{(i)};\gamma,\beta) := \frac{\gamma(x^{(i)}-\mu)}{\sigma} + \beta, \quad i = 1,2,\dots,m, \tag{29}$$

where $\mu := \sum_{i=1}^{m} x^{(i)}/m$ and $\sigma^2 := \sum_{i=1}^{m}(x^{(i)}-\mu)^2/m$.

Definition 5 *Padding.* Let x be a feature in $\mathbb{R}^{h\times w\times d_1}$. The padding operator with parameter $d_2 > d_1$, denoted by $E : \mathbb{R}^{hwd_1} \to \mathbb{R}^{hwd_2}$, is defined as:

$$E(\mathrm{vec}(x);d_2) := \mathrm{vec}(y), \tag{30}$$

where y is a feature in $\mathbb{R}^{h\times w\times d_2}$ such that each channel $y_i \in \mathbb{R}^{h\times w}$ of y is defined as:

$$y_i := \begin{cases} x_{i-d}, & \text{if } d+1 \le i \le d+d_1 \text{ where } d := \lfloor (d_2-d_1)/2 \rfloor, \\ 0, & \text{otherwise,} \end{cases} \tag{31}$$

for $i = 1,2,\dots,d_2$.

Proposition 1 *The padding operator E has the following norm preserving property: if $x \in \mathbb{R}^{h\times w\times d_1}$ and $d_2 > d_1$, then*

$$\|E(\mathrm{vec}(x);d_2)\|_{\ell^p(\mathbb{R}^{hwd_2})} = \|\mathrm{vec}(x)\|_{\ell^p(\mathbb{R}^{hwd_1})} \tag{32}$$

for all $p \in [1,\infty]$.

Definition 6 *Pooling.* Let x be a feature in $\mathbb{R}^{h\times w\times d}$. The 2D average pooling operator with filter size 2×2 and stride size 2, denoted by $P_2 : \mathbb{R}^{hwd} \to \mathbb{R}^{\lceil h/2\rceil \lceil w/2d\rceil}$, is defined as:

$$P_2(\mathrm{vec}(x)) := \mathrm{vec}(y),$$

where y is a feature in $\mathbb{R}^{\lceil h/2\rceil\times\lceil w/2\rceil\times d}$ such that each channel $y_i \in \mathbb{R}^{\lceil h/2\rceil\times\lceil w/2\rceil}$ is defined as:

$$y_i := \frac{1}{4}\left(\begin{pmatrix} 1 & 1 \\ 1 & 1 \end{pmatrix} * x_i\right) \text{ with stride 2,} \quad i = 1,2,\dots,d, \tag{33}$$

where zero padding is used to perform the convolution. The global average pooling operator, $P_g : \mathbb{R}^{hwd} \to \mathbb{R}^d$, is defined as:

$$P_g(\mathrm{vec}(x)) := y,$$

where y is a vector in $\mathbb{R}^d$ such that each component y_k of y is defined as:

$$y_k := \frac{1}{hw}\sum_{i=1}^{h}\sum_{j=1}^{w} x_{i,j,k}, \quad k = 1,2,\dots,d.$$

Proposition 2 *The pooling operators P_2 and P_g are non-expansive in ℓ^2 in the sense that if $x \in \mathbb{R}^{h\times w\times d}$, then*

$$\|P_2(\mathrm{vec}(x))\|_{\ell^2(\mathbb{R}^{h_1w_1d})} \le \|\mathrm{vec}(x)\|_{\ell^2(\mathbb{R}^{hwd})}, \tag{34}$$

$$\|P_g(\mathrm{vec}(x))\|_{\ell^2(\mathbb{R}^{d})} \le \|\mathrm{vec}(x)\|_{\ell^2(\mathbb{R}^{hwd})}. \tag{35}$$

Definition 7 *Rectified Linear Unit.* The Rectified Linear Unit (ReLU) σ is an operation which is applied componentwise to any multi-dimensional feature x:

$$\sigma(x) = \max(x,0).$$

Proposition 3 *Let $n \in \mathbb{N}$ and $1 \le p \le \infty$. The rectified linear unit is non-expansive and 1-Lipschitz in $\ell^p(\mathbb{R}^n)$ in the sense that:*

$$\|\sigma(x)\|_{\ell^p(\mathbb{R}^n)} \le \|x\|_{\ell^p(\mathbb{R}^n)} \tag{36}$$

$$\|\sigma(x)-\sigma(y)\|_{\ell^p(\mathbb{R}^n)} \le \|x-y\|_{\ell^p(\mathbb{R}^n)} \tag{37}$$

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

Remark 4 Using ReLU as the activation function can be viewed as applying a proximal step in the dynamical system that defines the forward propagation. Let $I_{\mathbb{R}^d_+}$ be the indicator function of the set $\mathbb{R}^d_+$, which is defined as:

$$I_{\mathbb{R}^d_+}(x) := \begin{cases} 0, & \text{if } x \in \mathbb{R}^d_+, \\ \infty, & \text{if } x \notin \mathbb{R}^d_+. \end{cases} \tag{38}$$

The proximal operator associated with $I_{\mathbb{R}^d_+}$ is in fact ReLU, i.e.,

$$\begin{aligned} \mathrm{prox}_{\gamma I_{\mathbb{R}^d_+}}(x) &= \operatorname*{argmin}_{y\in\mathbb{R}^d} \ \gamma I_{\mathbb{R}^d_+}(x) + \frac{1}{2}\|x-y\|^2_{\ell^2(\mathbb{R}^d)} \\ &= \mathrm{proj}_{\mathbb{R}^d_+}(x) = \sigma(x), \end{aligned}$$

and is independent of $\gamma > 0$.

B Proofs of the Main Results

We provide the proofs of the results presented in this work.

Proof of Theorem 1 Take $(H, \|\cdot\|) = \left(\mathbb{R}^d, \|\cdot\|_{\ell^2(\mathbb{R}^d)}\right)$, $I = [0,\infty)$,

$$F(t,x(t)) := A_2(t)\,\sigma(A_1(t)x(t)+b_1(t)) - b_2(t), \tag{39}$$

and C be the multi-valued mapping such that $C(t) = \mathbb{R}^d_+$ for all $t \in [0,T]$. We will prove that conditions (i)-(iv) in Theorem 7 are satisfied. Without ambiguity, we write $\|\cdot\|_2$ for $\|\cdot\|_{\ell^2(\mathbb{R}^d)}$.

(i) For each $t \in [0, T]$, it is clear that $C(t)$ is a nonempty closed subset of H, and by [31], $C(t)$ is r-prox-regular.

(ii) Setting $v(t) = 0$ for all $t \in [0, T]$ yields Eq. (58).

(iii) Let $x, y : I \to \mathbb{R}^d$. By Eq. (39) and the assumptions that $\|A_1(t)\|_2 \, \|A_2(t)\|_2 \le c$ for all $t > 0$ and that σ is contractive (implied by Eq. (37)), we have:

$$\begin{aligned}&\|F(t,x(t)) - F(t,y(t))\|_2\\ &\quad= \|A_2(t)\,\sigma(A_1(t)x(t) + b_1(t))\\ &\qquad - A_2(t)\,\sigma(A_1(t)y(t) + b_1(t))\|_2\\ &\quad\le \|A_2(t)\|_2\;\|\sigma(A_1(t)x(t) + b_1(t))\\ &\qquad - \sigma(A_1(t)y(t) + b_1(t))\|_2\\ &\quad\le \|A_2(t)\|_2\;\|A_1(t)x(t) - A_1(t)y(t)\|_2\\ &\quad\le \|A_2(t)\|_2\;\|A_1(t)\|_2\;\|x(t) - y(t)\|_2\\ &\quad\le c\|x(t) - y(t)\|_2.\end{aligned}$$

(iv) Let $x : [0, T] \to \mathbb{R}^d$. A similar derivation as above yields:

$$\begin{aligned}&\|F(t,x(t))\|_2\\ &\quad= \|A_2(t)\,\sigma(A_1(t)x(t) + b_1(t)) - b_2(t)\|_2\\ &\quad\le \|A_2(t)\|_2\;\|\sigma(A_1(t)x(t) + b_1(t))\|_2 + \|b_2(t)\|_2\\ &\quad\le \|A_2(t)\|_2\;\|A_1(t)x(t) + b_1(t)\|_2 + \|b_2(t)\|_2\\ &\quad\le \|A_2(t)\|_2\;\big(\|A_1(t)\|_2\|x(t)\|_2 + \|b_1(t)\|_2\big) + \|b_2(t)\|_2\\ &\quad\le \beta(t)\big(1 + \|x(t)\|_2\big),\end{aligned}$$

where

$$\beta(t) := \max\left\{c, \|A_2(t)\|_2\;\|b_1(t)\|_2 + \|b_2(t)\|_2\right\}.$$

Therefore, by Theorem 7, there exists a unique absolutely continuous solution x to Eq. (6) for almost every $x_0 \in \mathbb{R}^d_+$. In particular, by Remark 5, the solution x satisfies that $x(t) \in \mathbb{R}^d_+$ for all $t > 0$. □

Proof of Theorem 2 Fix $t > 0$. Taking the inner product of Eq. (2) with x yields:

$$\begin{aligned}&x(t)^T\frac{\mathrm{d}}{\mathrm{dt}}x(t) + x(t)^T A_2(t)\,\sigma(A_1(t)x(t) + b_1(t))\\ &\quad - x(t)^T b_2(t) = x(t)^T p_x(t)\end{aligned}$$

for some $p_x(t) \in -\partial I_{\mathbb{R}^d_+}(x)$. Note that $0 \in \mathbb{R}^d_+$ and $0 \in \partial I_{\mathbb{R}^d_+}(x)$. Thus, by monotonicity of the subdifferential, we have:

$$x(t)^T p_x(t) = (x(t) - 0)^T (p_x(t) - 0) \le 0,$$

which implies that:

$$\begin{aligned}&x(t)^T\frac{\mathrm{d}}{\mathrm{dt}}x(t) + x(t)^T A_2(t)\,\sigma(A_1(t)x(t) + b_1(t))\\ &\quad - x(t)^T b_2(t) \le 0.\end{aligned}$$

Therefore, after re-arranging terms, we have:

$$\begin{aligned}&\frac{\mathrm{d}}{\mathrm{dt}}\left(\frac{\|x(t)\|_2^2}{2}\right) = x(t)^T\frac{\mathrm{d}}{\mathrm{dt}}x(t)\\ &\quad\le -x(t)^T A_2(t)\,\sigma(A_1(t)x(t) + b_1(t)) + x(t)^T b_2(t).\end{aligned}$$

By Theorem 1, $x(t) \in \mathbb{R}^d_+$ for a.e. $t > 0$, and thus, the inner product $x(t)^T b_2(t)$ is bounded above by the positive part of $b_2(t)$; that is,

$$x(t)^T b_2(t) \le x(t)^T \sigma(b_2(t)) \le \|x(t)\|_2\;\|\sigma(b_2(t))\|_2.$$

Therefore, since ReLU is contractive and $\sigma(0) = 0$, we have:

$$\begin{aligned}&\frac{\mathrm{d}}{\mathrm{dt}}\left(\frac{\|x(t)\|_2^2}{2}\right)\\ &\quad\le \|A_2(t)\|_2\;\|x(t)\|_2\;\|\sigma(A_1(t)x(t) + b_1(t))\|_2\\ &\qquad + \|x(t)\|_2\;\|\sigma(b_2(t))\|_2\\ &\quad\le \|A_2(t)\|_2\;\|x(t)\|_2\;\|A_1(t)x(t) + b_1(t)\|_2\\ &\qquad + \|x(t)\|_2\;\|\sigma(b_2(t))\|_2\\ &\quad\le \|A_1(t)\|_2\;\|A_2(t)\|_2\;\|x(t)\|_2^2\\ &\qquad + \big(\|A_2(t)\|_2\|b_1(t)\|_2 + \|\sigma(b_2(t))\|_2\big)\|x(t)\|_2.\end{aligned}$$

Applying Theorem 8 with $u = \|x\|_2^2/2$, $f = 2\|A_1\|_2\;\|A_2\|_2$, $g = \sqrt{2}\big(\|A_2\|_2\|b_1\|_2 + \|\sigma(b_2)\|_2\big)$, $c = \|x(0)\|_2^2/2$, $t_0 = 0$, and $\alpha = 1/2$ yields:

$$\begin{aligned}\|x(t)\|_2 \le{}& \|x(0)\|_2\,\exp\left(\int_0^t \|A_1(s)\|_2\;\|A_2(s)\|_2\,\mathrm{d}s\right)\\ &+ \int_0^t \big(\|A_2(s)\|_2\;\|b_1(s)\|_2\\ &+\|\sigma(b_2(s))\|_2\big)\,\exp\left(\int_s^t \|A_1(r)\|_2\;\|A_2(r)\|_2\,\mathrm{d}r\right)\mathrm{d}s,\end{aligned}$$

which proves Eq. (7).

Next, let x and y be the unique absolutely continuous solutions to Eq. (2), with different initial values $x(0)$ and $y(0)$. Then:

$$\begin{aligned}&\left(\frac{\mathrm{d}}{\mathrm{dt}}x(t) - \frac{\mathrm{d}}{\mathrm{dt}}y(t)\right) + A_2(t)\left(\sigma(A_1(t)x(t)+b_1(t))\right.\\ &\quad\left.-\sigma(A_1(t)y(t)+b_1(t))\right) = p_x(t) - p_y(t)\end{aligned}$$

for some $p_x(t) \in -\partial I_{\mathbb{R}^d_+}(x)$ and $p_y(t) \in -\partial I_{\mathbb{R}^d_+}(y)$. By monotonicity of the subdifferentials, we have:

$$(x(t)-y(t))^T(p_x(t)-p_y(t)) \leq 0,$$

which implies that:

$$\begin{aligned}&\frac{\mathrm{d}}{\mathrm{dt}}\left(\frac{\|x(t)-y(t)\|_2^2}{2}\right)\\ &\quad= (x(t)-y(t))^T\left(\frac{\mathrm{d}}{\mathrm{dt}}x(t) - \frac{\mathrm{d}}{\mathrm{dt}}y(t)\right)\\ &\quad\leq -(x(t)-y(t))^T A_2(t)\left(\sigma(A_1(t)x(t)+b_1(t))\right.\\ &\qquad\left.-\sigma(A_1(t)y(t)+b_1(t))\right)\\ &\quad\leq \|A_2(t)\|_2\,\|x(t)-y(t)\|_2\,\|\sigma(A_1(t)x(t)+b_1(t))\\ &\qquad-\sigma(A_1(t)y(t)+b_1(t))\|_2.\end{aligned}$$

Therefore, since ReLU is contractive and $\sigma(0)=0$, we have:

$$\begin{aligned}&\frac{\mathrm{d}}{\mathrm{dt}}\left(\frac{\|x(t)-y(t)\|_2^2}{2}\right)\\ &\quad\leq \|A_2(t)\|_2\,\|x(t)-y(t)\|_2\,\|A_1(t)(x(t)-y(t))\|_2\\ &\quad\leq \|A_1(t)\|_2\,\|A_2(t)\|_2\,\|x(t)-y(t)\|_2^2.\end{aligned}$$

Applying Theorem 8 with $u=\|x\|_2^2/2$, $f=2\|A_1\|_2\,\|A_2\|_2$, $g=0$, $c=\|x(0)\|_2^2/2$, $t_0=0$, and $\alpha=1$ yields:

$$\begin{aligned}&\|x(t)-y(t)\|_2\\ &\quad\leq \|x(0)-y(0)\|_2\,\exp\left(\int_0^t \|A_1(s)\|_2\,\|A_2(s)\|_2\,\mathrm{d}s\right),\end{aligned}$$

which proves Eq. (8). □

Proof of Theorem 3 Taking the inner product of Eq. (9) with x yields:

$$\begin{aligned}&x(t)^T\frac{\mathrm{d}}{\mathrm{dt}}x(t) + x^T A_2(t)\sigma(A_1(t)x(t)+b_1(t))\\ &\quad- x^T b_2(t) = x(t)^T p_x(t)\end{aligned}$$

for some $p_x(t) \in -\partial I_{\mathbb{R}^d_+}(x)$. Using the same argument as in the proof of Theorem 2, we have:

$$\begin{aligned}&\frac{\mathrm{d}}{\mathrm{dt}}\left(\frac{\|x(t)\|_2^2}{2}\right) = x(t)^T\frac{\mathrm{d}}{\mathrm{dt}}x(t)\\ &\quad\leq -x(t)^T A_2(t)\,\sigma(A_1(t)x(t)+b_1(t)) + x(t)^T\sigma(b_2(t)).\end{aligned}$$

By Remark 5, $x \in \mathbb{R}^d_+$, and by assumption, $A_2(t) \geq 0$. Thus:

$$x(t)^T A_2(t)\,\sigma(A_1(t)x(t)+b_1(t)) \geq 0,$$

which implies that

$$\frac{\mathrm{d}}{\mathrm{dt}}\left(\frac{\|x(t)\|_2^2}{2}\right) \leq x(t)^T\sigma(b_2(t)) \leq \|x(t)\|_2\|\sigma(b_2(t))\|_2.$$

Applying Theorem 8 with $u=\|x\|_2^2/2$, $f=0$, $g=\sqrt{2}\|\sigma(b_2)\|_2$, $c=\|x(0)\|_2^2/2$, $t_0=0$, and $\alpha=1/2$ yields:

$$\|x(t)\|_2 \leq \|x(0)\|_2 + \int_0^t \|\sigma(b_2(s))\|_2\,\mathrm{d}s,$$

which proves Eq. (10). □

To prove Theorem 4, we will first show an auxiliary result.

Lemma 1 *Let $b \in \mathbb{R}^d$ and define the function $G : \mathbb{R}^d \to \mathbb{R}^d$ by:*

$$G(x) := \sigma(x+b),$$

where σ is ReLU. Then G is monotone in ℓ^2, i.e.,

$$(x-y)^T(G(x)-G(y)) \geq 0$$

for all $x, y \in \mathbb{R}^d$.

Proof This is an immediate consequence of the fact that σ is monotone:

$$\begin{aligned}&(x-y)^T(G(x)-G(y))\\ &\quad= (x-y)^T(\sigma(x+b)-\sigma(y+b))\\ &\quad= ((x+b)-(y-b))^T(\sigma(x+b)-\sigma(y+b)) \geq 0.\end{aligned}$$

□

Proof of Theorem 4 Taking the inner product of Eq. (11) with x yields:

$$\begin{aligned}&x(t)^T\frac{\mathrm{d}}{\mathrm{dt}}x(t) + (A_1(t)x(t))^T A_2(t)\,\sigma(A_1(t)x(t)+b_1(t))\\ &\quad- x(t)^T b_2(t) = x(t)^T p_x(t)\end{aligned}$$

for some $p_x(t) \in -\partial I_{\mathbb{R}^d_+}(x)$. Using the same argument as in the proof of Theorem 2, we have:

$$\frac{\mathrm{d}}{\mathrm{dt}}\left(\frac{\|x(t)\|_2^2}{2}\right) \le -(A(t)x(t))^T\,\sigma(A(t)x(t) + b_1(t)) + x(t)^T b_2(t).$$

Define the function $G : \mathbb{R}^d \to \mathbb{R}^d$ by:

$$G(x(t)) := \sigma(x(t) + b_1(t)).$$

By Lemma 1:

$$\begin{aligned} &-(A(t)x(t))^T\,\sigma(A(t)x(t) + b_1(t)) \\ &\quad = -(A(t)x(t) - 0)^T(G(A(t)x(t)) - G(0)) \\ &\qquad - (A(t)x(t))^T G(0) \\ &\quad \le -(A(t)x(t))^T\sigma(b_1(t)). \end{aligned}$$

Therefore,

$$\begin{aligned} &\frac{\mathrm{d}}{\mathrm{dt}}\left(\frac{\|x(t)\|_2^2}{2}\right) \\ &\quad \le -(A(t)x(t))^T\sigma(b_1(t)) + x(t)^T b_2(t) \\ &\quad \le \|x(t)\|_2 \left\|\sigma\left(-A(t)^T\sigma(b_1(t)) + b_2(t)\right)\right\|_2. \end{aligned}$$

Applying Theorem 8 with $u = \|x\|_2^2/2$, $f = 0$, $g = \sqrt{2}\sigma\left(-A^T\sigma(b_1) + b_2\right)$, $c = \|x(0)\|_2^2/2$, $t_0 = 0$, and $\alpha = 1/2$ yields:

$$\|x(t)\|_2 \le \|x(0)\|_2 + \int_0^t \left\|\sigma\left(-A(s)^T\sigma(b_1(s)) + b_2(s)\right)\right\|_2 \mathrm{d}s,$$

which proves Eq. (12).

Next, let x and y be the unique absolutely continuous solutions to Eq. (2), with different initial values $x(0)$ and $y(0)$. Using the same argument as in the proof of Theorem 2 yields:

$$\begin{aligned} &\frac{\mathrm{d}}{\mathrm{dt}}\left(\frac{\|x(t) - y(t)\|_2^2}{2}\right) \\ &\quad = (x(t) - y(t))^T\left(\frac{\mathrm{d}}{\mathrm{dt}}x(t) - \frac{\mathrm{d}}{\mathrm{dt}}y(t)\right) \\ &\quad \le -(A(t)x(t) - A(t)y(t))^T\left(\sigma(A(t)x(t) + b_1(t))\right. \\ &\qquad \left. -\sigma(A(t)y(t) + b_1(t))\right) \le 0, \end{aligned}$$

where the last inequality is due to monotonicity of G. This proves Eq. (13). □

Proof of Theorem 5, part 1 We will show that

$$\|x^{n+1}\|_2 \le \|x^n\|_2 + c_n$$

for all $n = 1, 2, \dots, 3m + 3$, where $c_n \ge 0$ is independent of the x^n.

For the convolution layer (Layer 0), we show that:

$$\|x^1\|_{\ell^2(\mathbb{R}^{h_1 w_1 d_1})} \le \|x^0\|_{\ell^2(\mathbb{R}^{h_1 w_1 d_0})} + \|b^0\|_{\ell^2(\mathbb{R}^{h_1 w_1 d_1})} \tag{40}$$

provided that $\|A^0\|_{\ell^2(\mathbb{R}^{h_1 w_1 d_0}) \to \ell^2(\mathbb{R}^{h_1 w_1 d_1})} \le 1$. By Eq. (16), we have:

$$\begin{aligned} \|x^1\|_{\ell^2(\mathbb{R}^{h_1 w_1 d_1})} &\le \|A^0 x^0\|_{\ell^2(\mathbb{R}^{h_1 w_1 d_1})} + \|b^0\|_{\ell^2(\mathbb{R}^{h_1 w_1 d_1})} \\ &\le \|A^0\|_{\ell^2(\mathbb{R}^{h_1 w_1 d_0}) \to \ell^2(\mathbb{R}^{h_1 w_1 d_1})} \|x^0\|_{\ell^2(\mathbb{R}^{h_1 w_1 d_0})} \\ &\quad + \|b^0\|_{\ell^2(\mathbb{R}^{h_1 w_1 d_1})} \\ &\le \|x^0\|_{\ell^2(\mathbb{R}^{h_1 w_1 d_0})} + \|b^0\|_{\ell^2(\mathbb{R}^{h_1 w_1 d_1})}. \end{aligned}$$

For the first stack of ResNet layers (Layer n with $n = 1, 2 \dots, m$), we show that:

$$\|x^{n+1}\|_{\ell^2(\mathbb{R}^{h_1 w_1 d_1})} \le \|x^n\|_{\ell^2(\mathbb{R}^{h_1 w_1 d_1})} + \|b_2^n\|_{\ell^2(\mathbb{R}^{h_1 w_1 d_1})}. \tag{41}$$

Fix $i \in [h_1 w_1 d_1]$. By Eq. (15), we have:

$$0 \le x_i^{n+1} = \sigma(x_i^n - a_i^n\,\sigma(A_1^n x^n + b_1^n) + (b_2^n)_i),$$

where a_i^n denotes the ith row of A_2^n and $(b_2^n)_i$ denotes the ith element of b_2^n. Consider two cases. If $x_i^n - a_i^n\,\sigma(A_1^n x^n + b_1^n) + (b_2^n)_i < 0$, then $x_i^{n+1} = 0$. Otherwise, since $a_i^n \ge 0$ component-wise, it holds that

$$0 \le x_i^{n+1} = x_i^n - a_i^n\sigma(A_1^n x^n + b_1^n) + (b_2^n)_i \le x_i^n + (b_2^n)_i.$$

Therefore,

$$\|x^{n+1}\|_{\ell^2(\mathbb{R}^{h_1 w_1 d_1})} \le \|x^n\|_{\ell^2(\mathbb{R}^{h_1 w_1 d_1})} + \|b_2^n\|_{\ell^2(\mathbb{R}^{h_1 w_1 d_1})}.$$

Analysis for the remaining ResNet layers, Layers $m + 2$ to $2m$ and Layers $2m + 2$ to $3m$, is the same.

For the first 2d pooling layer (Layer n with $n = m + 1$), we show that:

$$\|x^{n+1}\|_{\ell^2(\mathbb{R}^{h_2 w_2 d_2})} \le \|x^n\|_{\ell^2(\mathbb{R}^{h_1 w_1 d_1})}. \tag{42}$$

Observe from Figs. 1, 2, 3 and 4 that $x^j \ge 0$ component-wise for all $j = 2, 3, \dots, 3m + 2$. Since both $E(P_2(x^n))$ and $\sigma\left((A^n)_{|s=2}\, x^n + b^n\right)$ are component-wise nonnegative, by Eq. (18), we have the following component-wise inequality:

$0 \leq x^{n+1} \leq E(P_2(x^n)),$

and thus by Eqs. (32) and (34):

$$\begin{aligned}\|x^{n+1}\|_{\ell^2(\mathbb{R}^{h_2w_2d_2})} &\leq \|E(P_2(x^n))\|_{\ell^2(\mathbb{R}^{h_2w_2d_2})}\\ &= \|P_2(x^n)\|_{\ell^2(\mathbb{R}^{h_2w_2d_1})} \leq \|x^n\|_{\ell^2(\mathbb{R}^{h_1w_1d_1})}.\end{aligned}$$

Analysis for the second 2D pooling layer, Layer $2m+1$, is the same.

For the global pooling layer (Layer n with $n = 3m+1$), we show that:

$$\|x^{n+1}\|_{\ell^2(\mathbb{R}^{d_3})} \leq \|x^n\|_{\ell^2(\mathbb{R}^{h_3w_3d_3})}. \tag{43}$$

By Eqs. (35) and (36), the functions P_g and ReLU are non-expansive in ℓ^2, and thus:

$$\begin{aligned}\|x^{n+1}\|_{\ell^2(\mathbb{R}^{d_3})} &= \|P_g(\sigma(x^n))\|_{\ell^2(\mathbb{R}^{d_3})}\\ &\leq \|\sigma(x^n)\|_{\ell^2(\mathbb{R}^{h_3w_3d_3})} \leq \|x^n\|_{\ell^2(\mathbb{R}^{h_3w_3d_3})}.\end{aligned}$$

For the fully connected layer (Layer $n = N-1$ with $N = 3m+3$), we show that:

$$\|x^N\|_{\ell^2(\mathbb{R}^C)} \leq \|x^{N-1}\|_{\ell^2(\mathbb{R}^{d_3})} + \|b^{N-1}\|_{\ell^2(\mathbb{R}^C)} \tag{44}$$

provided that $\|W^{N-1}\|_{\ell^2(\mathbb{R}^{d_3})\to\ell^2(\mathbb{R}^C)} \leq 1$. Analysis for the fully connected layer is the same as the analysis for the convolution layer. By Eq. (17), we have:

$$\begin{aligned}\|x^N\|_{\ell^2(\mathbb{R}^C)} &\leq \|W^{N-1}x^{N-1}\|_{\ell^2(\mathbb{R}^C)} + \|b^{N-1}\|_{\ell^2(\mathbb{R}^C)}\\ &\leq \|W^{N-1}\|_{\ell^2(\mathbb{R}^{d_3})\to\ell^2(\mathbb{R}^C)}\|x^{N-1}\|_{\ell^2(\mathbb{R}^{d_3})}\\ &\quad + \|b^{N-1}\|_{\ell^2(\mathbb{R}^C)}\\ &\leq \|x^{N-1}\|_{\ell^2(\mathbb{R}^{d_3})} + \|b^{N-1}\|_{\ell^2(\mathbb{R}^C)}.\end{aligned}$$

Combining Eqs. (40)–(44) yields:

$$\|x^N\|_{\ell^2(\mathbb{R}^C)} \leq \|x^0\|_{\ell^2(\mathbb{R}^{h_1w_1d_0})} + c(b^0, b^1, \ldots, b^{N-1}),$$

where $c(b^0, b^1, \ldots, b^{N-1})$ is a constant depending on the ℓ^2 norms of the biases in the network:

$$c(b^0, b^1, \ldots, b^{N-1}) := \sum_{n=0}^{3m+2} \|b_2^n\|_{\ell^2}. \tag{45}$$

This proves Eq. (21). □

Proof of Theorem 5, part 2 We will show that

$$\|x^{n+1} - y^{n+1}\|_{\ell^2} \leq a_n\|x^n - y^n\|_{\ell^2}$$

for all $n = 1, 2, \ldots, 3m+3$, where $a_n \geq 0$ is independent of the x^n and y^n.

For the convolution layer (Layer 0), we have, by Eqs. (16) and (20a), that:

$$\begin{aligned}&\|x^1 - y^1\|_{\ell^2(\mathbb{R}^{h_1w_1d_1})}\\ &\quad= \|A^0x^0 - A^0y^0\|_{\ell^2(\mathbb{R}^{h_1w_1d_1})}\\ &\quad\leq \|A^0\|_{\ell^2(\mathbb{R}^{h_1w_1d_0})\to\ell^2(\mathbb{R}^{h_1w_1d_1})}\|x^0 - y^0\|_{\ell^2(\mathbb{R}^{h_1w_1d_0})}\\ &\quad\leq \|x^0 - y^0\|_{\ell^2(\mathbb{R}^{h_1w_1d_0})}.\end{aligned} \tag{46}$$

For the first stack of residual layers (Layer n with $n = 1, 2\ldots, m$), we have, by Eq. (14), that:

$$\begin{aligned}&\|x^{n+1} - y^{n+1}\|_{\ell^2(\mathbb{R}^{h_1w_1d_1})}\\ &\quad= \|\sigma(x^n - A_2^n\,\sigma(A_1^nx^n + b_1^n) + b_2^n)\\ &\qquad - \sigma(y^n - A_2^n\,\sigma(A_1^ny^n + b_1^n) + b_2^n)\|_{\ell^2(\mathbb{R}^{h_1w_1d_1})}\\ &\quad\leq \|(x^n - A_2^n\,\sigma(A_1^nx^n + b_1^n))\\ &\qquad - (y^n - A_2^n\,\sigma(A_1^ny^n + b_1^n))\|_{\ell^2(\mathbb{R}^{h_1w_1d_1})}\\ &\quad\leq \|x^n - y^n\|_{\ell^2(\mathbb{R}^{h_1w_1d_1})} + \|A_2^n\|_{\ell^2(\mathbb{R}^{h_1w_1d_1})}\|\sigma(A_1^nx^n + b_1^n)\\ &\qquad - \sigma(A_1^ny^n + b_1^n)\|_{\ell^2(\mathbb{R}^{h_1w_1d_1})}\\ &\quad\leq \|x^n - y^n\|_{\ell^2(\mathbb{R}^{h_1w_1d_1})}\\ &\qquad + \|A_2^n\|_{\ell^2(\mathbb{R}^{h_1w_1d_1})}\|A_1^nx^n - A_1^ny^n\|_{\ell^2(\mathbb{R}^{h_1w_1d_1})}\\ &\quad\leq \left(1 + \|A_1^n\|_{\ell^2(\mathbb{R}^{h_1w_1d_1})}\|A_2^n\|_{\ell^2(\mathbb{R}^{h_1w_1d_1})}\right)\|x^n - y^n\|_{\ell^2(\mathbb{R}^{h_1w_1d_1})},\end{aligned} \tag{47}$$

where we have used the fact that ReLU is 1-Lipschitz in ℓ^2 (see Eq. (37)). Analysis for the remaining residual layers, Layers $m+2$ to $2m$ and Layers $2m+2$ to $3m$, is the same.

For the first 2d pooling layer (Layer n with $n = m+1$), we have, by Eq. (18), that:

$$\begin{aligned}&\|x^{n+1} - y^{n+1}\|_{\ell^2(\mathbb{R}^{h_2w_2d_2})}\\ &\quad= \Big\|\sigma\big(E(P_2(x^n)) - \sigma\big((A^n)_{|s=2}\,x^n + b^n\big)\big)\\ &\qquad -\sigma\big(E(P_2(y^n)) - \sigma\big((A^n)_{|s=2}\,y^n + b^n\big)\big)\Big\|_{\ell^2(\mathbb{R}^{h_2w_2d_2})}\\ &\quad\leq \|E(P_2(x^n)) - E(P_2(y^n))\|_{\ell^2(\mathbb{R}^{h_2w_2d_2})}\\ &\qquad + \|\sigma\big((A^n)_{|s=2}\,x^n + b^n\big)\\ &\qquad - \sigma\big((A^n)_{|s=2}\,y^n + b^n\big)\|_{\ell^2(\mathbb{R}^{h_2w_2d_2})}\\ &\quad\leq \|x^n - y^n\|_{\ell^2(\mathbb{R}^{h_2w_2d_2})}\\ &\qquad + \|(A^n)_{|s=2}\,x^n - \ (A^n)_{|s=2}\,y^n\|_{\ell^2(\mathbb{R}^{h_2w_2d_2})}\\ &\quad\leq \left(1 + \|A^n\|_{\ell^2(\mathbb{R}^{h_1w_1d_1})\to\ell^2(\mathbb{R}^{h_2w_2d_2})}\right)\|x^n - y^n\|_{\ell^2(\mathbb{R}^{h_1w_1d_1})},\end{aligned} \tag{48}$$

where we have used the fact that padding and 2d average pooling are linear operators and are non-expansive in ℓ^2 (see Properties 1 and 2). Analysis for the second 2D pooling layer, Layer $2m+1$, is the same.

For the global pooling layer (Layer n with $n = 3m+1$), we have, by Eq. (19), that:

$$\begin{aligned}\|x^{n+1} - y^{n+1}\|_{\ell^2(\mathbb{R}^{d_3})} &= \|P_g(\sigma(x^n)) - P_g(\sigma(y^n))\|_{\ell^2(\mathbb{R}^{h_3 w_3 d_3})} \\ &\leq \|\sigma(x^n) - \sigma(y^n)\|_{\ell^2(\mathbb{R}^{h_3 w_3 d_3})} \\ &\leq \|x^n - y^n\|_{\ell^2(\mathbb{R}^{h_3 w_3 d_3})},\end{aligned} \tag{49}$$

where we have used the fact that global average pooling is a linear operator and is non-expansive in ℓ^2.

For the fully connected layer (Layer $n = 3m+2 = N-1$), we have, by Eqs. (17) and (20b), that

$$\begin{aligned}\|x^N - y^N\|_{\ell^2(\mathbb{R}^C)} &= \|W^{N-1}x^{N-1} - W^{N-1}y^{N-1}\|_{\ell^2(\mathbb{R}^C)} \\ &\leq \|W^{N-1}\|_{\ell^2(\mathbb{R}^{d_3})\to\ell^2(\mathbb{R}^C)}\|x^{N-1} - y^{N-1}\|_{\ell^2(\mathbb{R}^{d_3})} \\ &\leq \|x^{N-1} - y^{N-1}\|_{\ell^2(\mathbb{R}^{d_3})}.\end{aligned} \tag{50}$$

Combining Eqs. (46)–(50) yields:

$$\begin{aligned}&\|x^N - y^N\|_{\ell^2(\mathbb{R}^C)} \\ &\quad\leq a(A^0, A^1, \dots, W^{N-1})\, \|x^0 - y^0\|_{\ell^2(\mathbb{R}^{h_1 w_1 d_0})},\end{aligned}$$

where $a(A^0, A^1, \dots, W^{N-1})$ is a constant depending on the ℓ^2 norms of the filters and weights in the network:

$$a(A^0, A^1, \dots, W^{N-1}) := \prod_{n=1}^{3m} \left(1 + \|A_1^n\|_{\ell^2}\|A_2^n\|_{\ell^2}\right). \tag{51}$$

This proves Eq. (22). □

To prove Theorem 6, we will first show an auxiliary result.

Lemma 2 Let $A \in \mathbb{R}^{d\times d}$ *and* $b \in \mathbb{R}^d$. *Define the function* $F : \mathbb{R}^d \to \mathbb{R}^d$ *by:*

$$F(x) := x - A^T\sigma(Ax+b),$$

where σ is ReLU. If $\|A\|_{\ell^2(\mathbb{R}^d)} \leq \sqrt{2}$, then F is non-expansive in ℓ^2, i.e.,

$$\|F(x) - F(y)\|_{\ell^2(\mathbb{R}^d)} \leq \|x - y\|_{\ell^2(\mathbb{R}^d)}$$

for all $x, y \in \mathbb{R}^d$.

Proof First note that the activation function $\sigma : \mathbb{R}^d \to \mathbb{R}^d$ is applied component-wise. The function is of bounded variation and has a derivative in the measure sense. Fix an index $i \in [d]$ and consider the ith component F_i of F:

$$F_i : \mathbb{R}^d \to \mathbb{R}, \quad F_i(x) := x_i - (A^T)_{i,:}\,\sigma(Ax+b)_i,$$

where $(A^T)_{i,:}$ denotes the ith row of A^T. Its derivative ∇F_i is defined almost everywhere:

$$\nabla F_i : \mathbb{R}^d \to \mathbb{R}^{1\times d}, \quad \nabla F_i(x) := (e_i)^T - A^T\,\nabla\sigma(Ax+b)\,A_{i,:},$$

where e_i is the ith standard basis in $\mathbb{R}^d$ and $A_{i,:}$ denotes the ith row of A. For any $x, y \in \mathbb{R}^d$, applying the fundamental theorem of calculus yields:

$$\begin{aligned}F_i(x) - F_i(y) &= \int_0^1 \Big((e_i)^T - A^T\,\nabla\sigma(A((1-s)y \\ &\qquad +sx)+b)\,A_{i,:}\Big)(x-y)\,\mathrm{d}s \\ &= x_i - y_i - A^T\left(\int_0^1 \nabla\sigma(A((1-s)y \right. \\ &\qquad \left. +sx)+b)\,\mathrm{d}s\right)A_{i,:}(x-y),\end{aligned}$$

and thus:

$$F(x) - F(y) = \left(I - A^T D(x,y) A\right)(x-y),$$

where $D(x,y) \in \mathbb{R}^{d\times d}$ is the diagonal matrix defined as:

$$D(x,y) := \int_0^1 \nabla\sigma(A((1-s)y+sx)+b)\,\mathrm{d}s.$$

Since ReLU is non-decreasing with derivative bounded in magnitude by 1, we have $0 \leq D(x,y)_{ii} \leq 1$ for all $i = 1, 2, \dots, d$. Therefore, the ℓ^2 norm is equivalent to:

$$\begin{aligned}&\|F(x) - F(y)\|_{\ell^2(\mathbb{R}^d)} \\ &\quad= \|I - A^T D(x,y) A\|_{\ell^2(\mathbb{R}^d)}\,\|x-y\|_{\ell^2(\mathbb{R}^d)}.\end{aligned}$$

If $\|A\|_{\ell^2(\mathbb{R}^d)} \leq \sqrt{2}$, then $0 \leq \lambda_{\max}(A^T D(x,y) A) \leq 2$, and thus:

$$\|I - A^T D(x,y) A\|^2_{\ell^2(\mathbb{R}^d)} \leq 1,$$

which implies that F is non-expansive in ℓ^2. □

Proof of Theorem 6, part 1 By the proof of Theorem 5 (part 1), the following bound:

$$\|x^{n+1}\|_{\ell^2} \leq \|x^n\|_{\ell^2} + c_n \tag{52}$$

holds for the convolution layer, the pooling layers, and the fully connected layer, where $c_n \geq 0$ is independent of the x^n. We will show that Eq. (52) also hold for ResNet-S layers.

For the first stack of residual layers (Layer n with $n = 1, 2 \ldots, m$), we use an alternative approach and show that:

$$\begin{aligned}&\|x^{n+1}\|_{\ell^2(\mathbb{R}^{h_1 w_1 d_1})}\\&\quad\leq \|x^n\|_{\ell^2(\mathbb{R}^{h_1 w_1 d_1})} + \sqrt{2}\|b_1^n\|_{\ell^2(\mathbb{R}^{h_1 w_1 d_1})} + \|b_2^n\|_{\ell^2(\mathbb{R}^{h_1 w_1 d_1})}\end{aligned} \tag{53}$$

provided that $\|A^n\|_{\ell^2(\mathbb{R}^{h_1 w_1 d_1})} \leq \sqrt{2}$. By Eqs. (15) and (37), we have:

$$\begin{aligned}&\|x^{n+1}\|_{\ell^2(\mathbb{R}^{h_1 w_1 d_1})}\\&\quad\leq \|x^n - (A^n)^T \sigma(A^n x^n + b_1^n)\|_{\ell^2(\mathbb{R}^{h_1 w_1 d_1})}\\&\qquad+ \|b_2^n\|_{\ell^2(\mathbb{R}^{h_1 w_1 d_1})}.\end{aligned}$$

Define $F_n : \mathbb{R}^{h_1 w_1 d_1} \to \mathbb{R}^{h_1 w_1 d_1}$ by:

$$F_n(x) := x - (A^n)^T \sigma(A^n x + b_1^n). \tag{54}$$

By Lemma 2, if $\|A^n\|_{\ell^2(\mathbb{R}^{h_1 w_1 d_1})} \leq \sqrt{2}$, then F_n is non-expansive in ℓ^2. Therefore,

$$\begin{aligned}&\|x^{n+1}\|_{\ell^2(\mathbb{R}^{h_1 w_1 d_1})}\\&\quad\leq \|F_n(x^n)\|_{\ell^2(\mathbb{R}^{h_1 w_1 d_1})} + \|b_2^n\|_{\ell^2(\mathbb{R}^{h_1 w_1 d_1})}\\&\quad\leq \|F_n(x^n) - F_n(0)\|_{\ell^2(\mathbb{R}^{h_1 w_1 d_1})}\\&\qquad+ \|F_n(0)\|_{\ell^2(\mathbb{R}^{h_1 w_1 d_1})} + \|b_2^n\|_{\ell^2(\mathbb{R}^{h_1 w_1 d_1})}\\&\quad\leq \|x^n\|_{\ell^2(\mathbb{R}^{h_1 w_1 d_1})} + \|(A^n)^T \sigma(b_1^n)\|_{\ell^2(\mathbb{R}^{h_1 w_1 d_1})}\\&\qquad+ \|b_2^n\|_{\ell^2(\mathbb{R}^{h_1 w_1 d_1})}\\&\quad\leq \|x^n\|_{\ell^2(\mathbb{R}^{h_1 w_1 d_1})} + \sqrt{2}\|b_1^n\|_{\ell^2(\mathbb{R}^{h_1 w_1 d_1})}\\&\qquad+ \|b_2^n\|_{\ell^2(\mathbb{R}^{h_1 w_1 d_1})}.\end{aligned}$$

Analysis for the remaining residual layers, Layers $m + 2$ to $2m$ and Layers $2m + 2$ to $3m$, is the same.

Combining Eqs. (40), (53), and (42)–(44) yields:

$$\|x^N\|_{\ell^2(\mathbb{R}^C)} \leq \|x^0\|_{\ell^2(\mathbb{R}^{h_1 w_1 d_0})} + c(b^0, b^1, \ldots, b^{N-1}),$$

where $c(b^0, b^1, \ldots, b^{N-1})$ is a constant depending on the ℓ^2 norms of the biases in the network:

$$\begin{aligned}c\big(\{b^n\}_{n=0}^N\big) &:= \|b^0\|_{\ell^2(\mathbb{R}^{h_1 w_1 d_1})} + \|b^{N-1}\|_{\ell^2(\mathbb{R}^C)}\\&\quad+ \sum_{i=1}^{3m} \Big(\sqrt{2}\|b_1^n\|_{\ell^2} + \|b_2^n\|_{\ell^2}\Big).\end{aligned} \tag{55}$$

This proves Eq. (25). □

Proof of Theorem 6, part 2 The proof is similar to the proof of Theorem 5 (part 2), except for the residual layers. We will show that the following bound:

$$\|x^{n+1} - y^{n+1}\|_{\ell^2} \leq a_n \|x^n - y^n\|_{\ell^2}$$

also holds for the residual layers, where $a_n \geq 0$ is independent of the x^n and y^n.

For the first stack of residual layers (Layer n with $n = 1, 2 \ldots, m$), we have, by Eq. (15), that:

$$\begin{aligned}&\|x^{n+1} - y^{n+1}\|_{\ell^2(\mathbb{R}^{h_1 w_1 d_1})}\\&\quad= \|\sigma(x^n - (A^n)^T \sigma(A^n x^n + b_1^n) + b_2^n)\\&\qquad- \sigma(y^n - (A^n)^T \sigma(A^n y^n + b_1^n) + b_2^n)\|_{\ell^2(\mathbb{R}^{h_1 w_1 d_1})}\\&\quad\leq \|F_n(x^n) - F_n(y^n)\|_{\ell^2(\mathbb{R}^{h_1 w_1 d_1})},\end{aligned}$$

where the function $F_n : \mathbb{R}^{h_1 w_1 d_1} \to \mathbb{R}^{h_1 w_1 d_1}$ is defined in Eq. (54). By Lemma 2, F_n is non-expansive in ℓ^2 if $\|A^n\|_{\ell^2(\mathbb{R}^{h_1 w_1 d_1})} \leq \sqrt{2}$. Thus,

$$\|x^{n+1} - y^{n+1}\|_{\ell^2(\mathbb{R}^{h_1 w_1 d_1})} \leq \|x^n - y^n\|_{\ell^2(\mathbb{R}^{h_1 w_1 d_1})}, \tag{56}$$

Analysis for the remaining residual layers, Layers $m + 2$ to $2m$ and Layers $2m + 2$ to $3m$, is the same.

Combining Eqs. (46), (56), and (48)–(50) yields:

$$\begin{aligned}&\|x^N - y^N\|_{\ell^2(\mathbb{R}^C)}\\&\quad\leq a(A^0, A^1, \ldots, W^{N-1})\|x^0 - y^0\|_{\ell^2(\mathbb{R}^{h_1 w_1 d_0})},\end{aligned}$$

where $a(A^0, A^1, \ldots, W^{N-1})$ is a constant depending on the ℓ^2 norms of the filters and weights in the network:

$$a(A^0, A^1, \ldots, W^{N-1}) := \big(1 + \|A^{m+1}\|_{\ell^2}\big)\big(1 + \|A^{2m+1}\|_{\ell^2}\big). \tag{57}$$

This proves Eq. (26). □

C Auxiliary Results

To be self-contained, we include some results in differential inclusions and differential equations that we used in the main text.

Table 5 Classification error of the post-activation ResNet on CIFAR-10 (Table 6 from [16])

Depth	Trainable parameters	Test accuracy (%)
20	0.27M	91.25
32	0.46M	92.49
44	0.66M	92.83
56	0.85M	93.03

Definition 8 (*page 350*, [10]) For a fixed $r > 0$, the set S is said to be r-prox-regular if, for any $x \in S$ and any $\xi \in \mathcal{N}^{\mathcal{L}}_S(x)$ such that $\|\xi\| < 1$, one has $x = \operatorname{proj}_S(x + r\xi)$, where $\mathcal{N}^{\mathcal{L}}$ denotes the limiting normal cone (see [29]).

Theorem 7 (Theorem 1, [10]) *Let H be a Hilbert space with the associated norm $\|\cdot\|$. Assume that $C : [0,T] \to H$ with $T > 0$ is a set-valued map which satisfies the following two conditions*:

1. for each $t \in [0,T]$, $C(t$ *is a nonempty closed subset of* $)$ H *which is r-prox-regular*;
2. there exists an absolutely continuous function $v : [0,T] \to \mathbb{R}$ *such that for any* $y \in H$ *and* $s, t \in [0,T]$,
$$|\operatorname{dist}(y, C(t)) - \operatorname{dist}(y, C(s))| \leq |v(t) - v(s)|. \tag{58}$$

Let $F : [0,T] \times H \to H$ be a separately measurable map on [0, T] such that

3. *for every $\eta > 0$ there exists a nonnegative function $k_\eta \in L^1(\underline{[0,T]}, \mathbb{R})$ $\underline{\text{such that}}$ for all $t \in [0,T]$ and for any $(x,y) \in \overline{B(0,\eta)} \times \overline{B(0,\eta)}$*,
$$\|F(t,x) - F(t,y)\| \leq k_\eta(t)\|x - y\|,$$
where $\overline{B(0,\eta)}$ stands for the closed ball of radius η centered at $0 \in H$;
4. there exists a nonnegative function $\beta \in L^1([0,T], \mathbb{R})$ *such that for all* $t \in [0,T]$ *and for all* $x \in \cup_{s \in [0,T]} C(s)$,
$$\|F(t,x)\| \leq \beta(t)(1 + \|x\|).$$

Then, for any$x_0 \in C(T_0)$, where $0 \leq T_0 < T$, the following perturbed sweeping process

$$\begin{cases} -\dfrac{\mathrm{d}}{\mathrm{d}t}x(t) \in \mathcal{N}_{C(t)}(x(t)) + F(t, x(t)) \quad \text{a.e. } t \in [0,T] \\ x(T_0) = x_0 \end{cases}$$

has a unique absolutely continuous solution x, and the solution x satisfies

$$\left\|\frac{\mathrm{d}}{\mathrm{d}t}x(t) + F(t,x(t))\right\| \leq (1 + a)\beta(t) + \left|\frac{\mathrm{d}}{\mathrm{d}t}v(t)\right| \quad \text{a.e. } t \in [0,T],$$
$$\|F(t,x(t))\| \leq (1 + a)\beta(t) \quad \text{a.e. } t \in [0,T],$$

where

$$a := \|x_0\| + \exp\left(2\int_{T_0}^{T} \beta(s)\,\mathrm{d}s\right)\int_{T_0}^{T}\left(2\beta(s)(1 + \|x_0\|) + \left|\frac{\mathrm{d}}{\mathrm{d}t}v(s)\right|\right)\mathrm{d}s.$$

Remark 5 (Remark 2.1, [20]) If x is a solution to Eq. (6) defined on $[T_0, \infty)$, then $x(t) \in C(t)$ for all $t \in [T_0, \infty)$.

The following theorem states a nonlinear generalization of Gronwall's inequality.

Theorem 8 (Theorem 21, [8]) *Let u be a nonnegative function that satisfies the integral inequality*

$$u(t) \leq c + \int_{t_0}^{t} f(s)u(s) + g(s)u^\alpha(s)\,\mathrm{d}s,$$

where $c \geq 0$, $\alpha \geq 0$, f and g are continuous nonnegative functions for $t \geq t_0$.

1. For $0 \leq \alpha < 1$, we have:
$$u(t)^{1-\alpha} \leq c^{1-\alpha}\exp\left((1-\alpha)\int_{t_0}^{t} f(s)\,\mathrm{d}s\right) + (1-\alpha)\int_{t_0}^{t} g(s)\exp\left((1-\alpha)\int_{s}^{t} f(r)\,\mathrm{d}r\right)\mathrm{d}s.$$
2. For $\alpha = 1$, we have:
$$u(t) \leq c\exp\left((1-\alpha)\int_{t_0}^{t} f(s) + g(s)\,\mathrm{d}s\right).$$

D Classification Accuracy of ResNet

For comparison, we include the computational results of the post-activation ResNet (see Fig. 2c) in Table 5. Implementation details can be found in [16].

References

1. Bengio, Y.: Learning deep architectures for AI. Found. Trends. Mach. Learn. **2**(1), 1–127 (2009)
2. Bengio, Y., Simard, P., Frasconi, P.: Learning long-term dependencies with gradient descent is difficult. IEEE Trans. Neural Netw. **5**(2), 157–166 (1994)
3. Biggio, B., Corona, I., Maiorca, D., Nelson, B., Šrndić, N., Laskov, P., Giacinto, G., Roli, F.: Evasion attacks against machine learning at test time. In: Joint European Conference on Machine Learning and Knowledge Discovery in Databases, Springer, pp. 387–402 (2013)
4. Bottou, L., Curtis, F.E., Nocedal, J.: Optimization methods for large-scale machine learning. SIAM Rev. **60**(2), 223–311 (2018)
5. Chang, B., Meng, L., Haber, E., Ruthotto, L., Begert, D., Holtham, E.: Reversible architectures for arbitrarily deep residual neural networks. In: Thirty-Second AAAI Conference on Artificial Intelligence (2018)
6. Chaudhari, P., Choromanska, A., Soatto, S., LeCun, Y., Baldassi, C., Borgs, C., Chayes, J., Sagun, L., Zecchina, R.: Entropy-SGD: biasing gradient descent into wide valleys. ArXiv e-prints (2016)
7. Chaudhari, P., Oberman, A., Osher, S., Soatto, S., Carlier, G.: Deep relaxation: partial differential equations for optimizing deep neural networks. Res. Math. Sci. **5**(3), 30 (2018)
8. Dragomir, S.S.: Some Gronwall Type Inequalities and Applications. Nova Science Publishers, New York (2003)
9. Du, S.S., Zhai, X., Poczos, Barnabas, S., Aarti: gradient descent provably optimizes over-parameterized neural networks. ArXiv e-prints (2018)
10. Edmond, J.F., Thibault, L.: Relaxation of an optimal control problem involving a perturbed sweeping process. Math. Program. Ser. B **104**, 347–373 (2005)
11. Goldstein, T., Studer, C., Baraniuk, R.: A field guide to forward-backward splitting with a FASTA implementation. ArXiv e-prints (2014)
12. Gomez, A. N., Ren, M., Urtasun, R., Grosse, R. B.: The reversible residual network: backpropagation without storing activations. In: Advances in Neural Information Processing Systems, pp. 2214–2224 (2017)
13. Goodfellow, I.J., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., Bengio, Y.: Generative adversarial nets. In: Advances in Neural Information Processing Systems, pp. 2672–2680 (2014)
14. Haber, E., Ruthotto, L.: Stable architectures for deep neural networks. Inverse Probl. **34**(1), 014004 (2017)
15. He, K., Zhang, X., Ren, S., Sun, J.: Delving deep into rectifiers: surpassing human-level performance on imagenet classification. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 1026–1034 (2015)
16. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 770–778 (2016)
17. He, K., Zhang, X., Ren, S., Sun, J.: Identity mappings in deep residual networks. In: European Conference on Computer Vision, Springer, pp. 630–645 (2016)
18. Huang, G., Liu, Z., Van Der Maaten, L., Weinberger, K. Q.: Densely connected convolutional networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 4700–4708 (2017)
19. Ioffe, S., Szegedy, C.: Batch normalization: accelerating deep network training by reducing internal covariate shift. ArXiv e-prints (2015)
20. Kamenskii, M., Makarenkov, O., Wadippuli, L.N., de Fitte, P.R.: Global stability of almost periodic solutions to monotone sweeping processes and their response to non-monotone perturbations. Nonlinear Anal. Hybrid Syst. **30**, 213–224 (2018)
21. Keskar, N. S., Mudigere, D., Nocedal, J., Smelyanskiy, M., Tang, P.T.P.: On large-batch training for deep learning: generalization gap and sharp minima. In: International Conference on Learning Representations (2017)
22. Krizhevsky, A., Sutskever, I., Hinton, G.E.: ImageNet classification with deep convolutional neural networks. In: Advances in Neural Information Processing Systems, pp. 1097–1105 (2012)
23. Larsson, G., Maire, M., Shakhnarovich, G.: FractalNet: Ultra-deep neural networks without residuals. ArXiv e-prints (2016)
24. LeCun, Y., Bengio, Y., Hinton, G.: Deep learning. Nature **521**(7553), 436 (2015)
25. LeCun, Y., Boser, B., Denker, J.S., Henderson, D., Howard, R.E., Hubbard, W., Jackel, L.D.: Backpropagation applied to handwritten zip code recognition. Neural Comput. **1**(4), 541–551 (1989)
26. Li, H., Xu, Z., Taylor, G., Studer, C., Goldstein, T.: Visualizing the loss landscape of neural nets. In: Advances in Neural Information Processing Systems, pp. 6389–6399 (2018)
27. Li, Z., Shi, Z.: Deep residual learning and PDEs on manifold. arXiv preprint arXiv:1708.05115 (2017)
28. Lions, P.-L., Mercier, B.: Splitting algorithms for the sum of two nonlinear operators. SIAM J. Numer. Anal. **16**(6), 964–979 (1979)
29. Mordukhovich, B.S., Shao, Y.: Nonsmooth sequential analysis in asplund spaces. Trans. Am. Math. Soc. **348**, 1235–1280 (1996)
30. Oberman, A. M., Calder, J.: Lipschitz regularized deep neural networks converge and generalize. ArXiv e-prints (2018)
31. Poliquin, R.A., Rockafellar, R.T.: Prox-regular functions in variational analysis. Trans. Am. Math. Soc. **348**(5), 1805–1838 (1996)
32. Russakovsky, O., Deng, J., Hao, S., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., Berg, A.C., Fei-Fei, L.: ImageNet large scale visual recognition challenge. Int. J. Comput. Vis. (IJCV) **115**(3), 211–252 (2015)
33. Ruthotto, L., Haber, E.: Deep neural networks motivated by partial differential equations. ArXiv e-prints (2018)
34. Schaeffer, H.: A penalty method for some nonlinear variational obstacle problems. Commun. Math. Sci. **16**(7), 1757–1777 (2018)
35. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. ArXiv e-prints (2014)
36. Singer, Y., Duchi, J.C.: Efficient learning using forward–backward splitting. In: Advances in Neural Information Processing Systems, vol. 22, Curran Associates, Inc., pp. 495–503 (2009)
37. Sussillo, D., Abbott, L.F.: Random walk initialization for training very deep feedforward networks. ArXiv e-prints (2014)
38. Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., Rabinovich, A.: Going deeper with convolutions. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 1–9 (2015)
39. Szegedy, C., Zaremba, W., Sutskever, I., Bruna, J., Erhan, D., Goodfellow, I., Fergus, R.: Intriguing properties of neural networks. ArXiv e-prints (2013)
40. Thorpe, M., van Gennip, Y.: Deep limits of residual neural networks. ArXiv e-prints (2018)
41. Tran, G., Schaeffer, H., Feldman, W.M., Osher, S.J.: An l^1 penalty method for general obstacle problems. SIAM J. Appl. Math. **75**(4), 1424–1444 (2015)
42. Vidal, R., Bruna, J., Giryes, R., Soatto, S.: Mathematics of deep learning. ArXiv e-prints (2017)
43. Wang, B., Luo, X., Li, Z., Zhu, W., Shi, Z., Osher, S.: Deep neural nets with interpolating function as output activation. In: Advances in Neural Information Processing Systems, pp. 743–753 (2018)

44. Weinan, E., Han, J., Li, Q.: A mean-field optimal control formulation of deep learning. Res. Math. Sci. **6**(1), 10 (2019)
45. Weinan, E.: A proposal on machine learning via dynamical systems. Commun. Math. Stat. **5**(1), 1–11 (2017)

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Linan Zhang Linan Zhang is a Ph.D. candidate in the Department of Mathematical Sciences at the Carnegie Mellon University. She received B.Sc. in Mathematical Sciences from Worcester Polytechnic Institute.

Hayden Schaeffer Hayden Schaeffer is an Assistant Professor in the Department of Mathematics at CMU. He received an NSF CAREER Award and an AFOSR Young Investigator Award. Previously, he was an NSF Mathematical Sciences Postdoctoral Research Fellow, a von Karmen Instructor at Caltech, a UC President's Postdoctoral Fellow at UC Irvine, and a Collegium of University Teaching Fellow at UCLA. His education includes a Ph.D. and Masters in mathematics from UCLA and a B.A. from Cornell University.