Interpolation between Residual and Non-Residual Networks

Zonghan Yang [1] Yang Liu [1] Chenglong Bao [2] Zuoqiang Shi [3]

Abstract

Although ordinary differential equations (ODEs) provide insights for designing network architectures, its relationship with the non-residual convolutional neural networks (CNNs) is still unclear. In this paper, we present a novel ODE model by adding a damping term. It can be shown that the proposed model can recover both a ResNet and a CNN by adjusting an interpolation coefficient. Therefore, the damped ODE model provides a unified framework for the interpretation of residual and non-residual networks. The Lyapunov analysis reveals better stability of the proposed model, and thus yields robustness improvement of the learned networks. Experiments on a number of image classification benchmarks show that the proposed model substantially improves the accuracy of ResNet and ResNeXt over the perturbed inputs from both stochastic noise and adversarial attack methods. Moreover, the loss landscape analysis demonstrates the improved robustness of our method along the attack direction.

1. Introduction

Although deep learning has achieved remarkable success in many machine learning tasks, the theory behind it has still remained elusive. In recent years, developing new theories for deep learning has attracted increasing research interests. One important direction is to connect deep neural networks (DNNs) with differential equations (E, 2017) which have been largely explored in mathematics. This line of research mainly contains three perspectives: solving high dimensional differential equations with the help of DNNs due to its high expressive power (Han et al., 2018), discovering a differential equation that identifies the rule of the observed data based on the standard block of existing DNNs (Chen et al., 2018), and designing new architectures based on the numerical schemes of differential equations (Haber & Ruthotto, 2017; Lu et al., 2018; Zhu et al., 2018; Chang et al., 2018; Tao et al., 2018; Lu et al., 2019).

While each attempt in the above directions has strengthened the theoretical understanding of deep learning, there still remain many open questions. Among them, one important question is *what is the relationship between differential equations and non-residual convolutional neural networks*. Most prior studies have focused on associating residual networks (ResNets) (He et al., 2016) with differential equations (Lu et al., 2018; Chen et al., 2018), not only because ResNets are relatively easy to optimize and achieve better classification accuracy than CNNs, but also because the skipping connections among layers can be easily induced by the discretization of difference operators in differential equations. However, residual neural networks only account for a small fraction of the entire neural network family and have their own limitations. For example, Su et al. (2018) indicate that ResNets are more sensitive to the perturbation of the inputs and the shallow CNNs. As a result, it is important to move a further step to investigate the relationship between differential equations and non-residual convolutional neural networks.

In this paper, we present a new ordinary differential equation (ODE) that interpolates non-residual and residual CNNs. The ODE is controlled by an interpolation parameter λ ranging from 0 to ∞. It is equivalent to a residual network when λ is 0. On the contrary, the ODE amounts to a non-residual network when λ approaches to ∞. Hence, our work provides a unified framework for understanding both non-residual and residual neural networks from the perspective of ODE. The interpolation is able to improve over both non-residual and residual networks. Compared with non-residual networks, our ODE is much easier to optimize, especially for deep architectures. Compared with residual networks, we use the Lyapunov analysis to show that the interpolation results in improved robustness. To achieve the interpolation, a key difference of our work from existing methods is to discretize integral operators instead of difference operators to obtain neural networks. Experiments on image classification benchmarks show that our approach substantially improves

[1] Institute for Artificial Intelligence, Beijing National Research Center for Information Science and Technology, Department of Computer Science and Technology, Tsinghua University. [2] Yau Mathematical Sciences Center, Tsinghua University. [3] Department of Mathematical Sciences, Tsinghua University. Correspondence to: Chenglong Bao <clbao@mail.tsinghua.edu.cn>.

Proceedings of the 37^{th} International Conference on Machine Learning, Online, PMLR 119, 2020. Copyright 2020 by the author(s).

the accuracies of ResNet (He et al., 2016) and ResNeXt (Xie et al., 2017) when inputs are perturbed by both stochastic noise and adversarial attack methods. Furthermore, the visualization of the loss landscape of our model validates our Lyaponov analysis.

2. Related Work

Interpreting machine learning from the perspective of dynamic systems was firstly advocated by E (2017) and Haber & Ruthotto (2017). Recently, there have been many exciting works in this direction (Lu et al., 2018; Chen et al., 2018). We briefly review previous methods closely related to architecture design and model robustness.

ODE inspired architecture design Inspired by the relationship between ODE and neural networks, Lu et al. (2018) use a linear multi-step method to improve the model capacity of ResNet-like networks. Zhu et al. (2018) utilize the Runge-Kutta method to interpret and improve DenseNets and CliqueNets. Chang et al. (2018) and Haber & Ruthotto (2017) leverage the leap-frog method to design novel reversible neural networks. Tao et al. (2018) propose to model non-local neural networks with non-local differential equations. Lu et al. (2019) design a novel Tranformer-like architecture with Strang-Marchuk splitting scheme. Chen et al. (2018) show that blocks of a neural network can be instantiated by arbitrary ODE solvers, in which parameters can be directly optimized with the adjoint sensitivity method. Dupont et al. (2019) improve the expressive power of a neural ODE by mitigating the trajectory intersecting problem. Compared to the above works, our work provides a new ODE that unifies the analysis of residual and non-residual networks which leads to an interpolated architecture. The experiments validate the advantages of the proposed method using this framework.

ODE and model robustness A number of previous methods have also been proposed to improve adversarial robustness from the perspective of ODE. Zhang et al. (2019b) propose to use a smaller step factor in the Euler method for ResNet. Reshniak & Webster (2019) utilize an implicit discretization scheme for ResNet. Hanshu et al. (2019) propose to train a time-invariant neural ODE regularized by steady-state loss. Liu et al. (2019) and Wang et al. (2019) introduce stochastic noise to enhance its robustness inspired by stochastic differential equations. The aforementioned works have concentrated on improving numerical discretization schemes or introducing stochasticity for ODE modeling to gain robustness. From the Lyapunov stability perspective, Chang et al. (2019) propose to use anti-symmetric weight matrices to parametrize an RNN, which enhances its long-term dependency. Zhang et al. (2019a) also accelerate adversarial training by recasting it as a differential game from an ODE perspective. In this work, we provide the Lyaponov analysis of the proposed ODE model which shows the robustness improvements over ResNets in terms of local stability.

3. Methodology

In this section, we first introduce the background of the relationship between ODE and ResNets, and then the proposed ODE model and its stability analysis is present.

3.1. Background

Considering the ordinary differential equation:

$$\frac{\mathrm{d}\mathbf{x}(t)}{\mathrm{d}t} = f(\mathbf{x}(t), t), \ \mathbf{x}(0) = \mathbf{x}_0, \tag{1}$$

where $\mathbf{x} : [0, T] \to \mathbb{R}^d$ represents the state of the system. Given the discretization step Δt and define $t_n = n\Delta t$, the forward Euler method of Eq. (1) becomes

$$\mathbf{x}(t_{n+1}) = \mathbf{x}(t_n) + \Delta t f(\mathbf{x}(t_n), t_n). \tag{2}$$

Let $\mathbf{x}_n = \mathbf{x}(t_n)$, $\Delta t = 1$, it recovers a residual block:

$$\mathbf{x}_{n+1} = \mathbf{x}_n + f_n(\mathbf{x}_n), \tag{3}$$

and f_n is the n-th layer operation in ResNets. Thus, the output of network is equivalent to the evolution of the state variable at terminal time T, i.e. $\mathbf{x}(T) = \mathbf{x}_N$ is the output of last layer in a ResNet if assuming $N = T/\Delta t$.

The dynamic formulation of ResNets (see Eq. (1)) was initially established in (E, 2017). It inspired many interesting neural network architectures by using different discretization methods the first order derivative in Eq. (1) such as linear multi-step network (Lu et al., 2018) and Runge-Kutta network (Zhu et al., 2018). From Eq. (1), the skip connection from the current step $\mathbf{x}_n$ to the next step estimation $\mathbf{x}_{n+1}$ always exists no matter which kind of discretization is applied. Thus, a feedforward CNN without skip connection can not be directly explained under this framework which inspired current work. In the next section, we introduce a damped ODE which bridges the non-residual CNNs and ResNets.

3.2. The Proposed ODE Model

Based on the ODE formulation , we add a damping term to the model (1) and leads to the following model:

$$\frac{\mathrm{d}\mathbf{x}(t)}{\mathrm{d}t} = -\lambda\mathbf{x}(t) + \rho(\lambda) f(\mathbf{x}(t), t), \tag{4}$$

starting from $\mathbf{x}(0) = \mathbf{x}_0$. The constant $\lambda \in [0, +\infty)$ is the called interpolation coefficient and $\rho : [0, +\infty) \mapsto [0, +\infty)$ is the weight function. The following proposition shows that the model shown in Eq. (4) has a closed form solution.

Proposition 3.1. *For any $T > 0$, the solution of the ODE* (4) *is*

$$\mathbf{x}(T) = e^{-\lambda T}\left(\mathbf{x}_0 + \rho(\lambda)\int_0^T e^{\lambda t} f(\mathbf{x}(t), t)\mathrm{d}t\right). \quad (5)$$

Proof. Multiplying both sides by $e^{\lambda t}$, it has

$$\frac{\mathrm{d}(e^{\lambda t}\mathbf{x}(t))}{\mathrm{d}t} = e^{\lambda t}\frac{\mathrm{d}\mathbf{x}(t)}{\mathrm{d}t} + \lambda e^{\lambda t}\mathbf{x}(t) = \rho(\lambda)e^{\lambda t} f(\mathbf{x}(t), t).$$

Integrating within $[0, T]$ yields

$$e^{\lambda T}\mathbf{x}(T) - \mathbf{x}(0) = \rho(\lambda)\int_0^T e^{\lambda t} f(\mathbf{x}(t), t)\mathrm{d}t, \quad (6)$$

which induces the equality (5). □

Following from the proposition 3.1 and the notations in section 3.1, the iterative formula of $\mathbf{x}_n$ is

$$\mathbf{x}_{n+1} = e^{-\lambda\Delta t}\mathbf{x}_n + e^{-\lambda t_{n+1}}\rho(\lambda)\int_{t_n}^{t_{n+1}} e^{\lambda t} f(\mathbf{x}(t), t)\mathrm{d}t. \quad (7)$$

Assuming $f(\mathbf{x}(t), t) = f(\mathbf{x}_n, t_n)$ for all $t \in [t_n, t_{n+1})$, the iterative scheme in Eq. (7) reduces to

$$\mathbf{x}_{n+1} = e^{-\lambda\Delta t}\mathbf{x}_n + \frac{1 - e^{-\lambda\Delta t}}{\lambda}\rho(\lambda) f_n(\mathbf{x}_n), \quad (8)$$

where $f_n(\mathbf{x}_n) = f(\mathbf{x}(t_n), t_n)$ is the convolutions in n-th layer. Now, we are ready to analyze Eq. (8) by choosing an appropriate weight function $\rho(\lambda)$. When the weight function $\rho(\lambda)$ satisfies

$$\rho(\lambda) \to 1, \lambda \to 0^+ \text{ and } \rho(\lambda) \sim \lambda, \lambda \to +\infty, \quad (9)$$

the output of n-th layer is

$$\mathbf{x}_{n+1} = \begin{cases} \mathbf{x}_n + f_n(\mathbf{x}_n), & \text{if } \lambda \to 0^+, \\ \Delta t f_n(\mathbf{x}_n), & \text{if } \lambda \to +\infty. \end{cases} \quad (10)$$

The above equation clearly shows that our model recovers ResNets when the interpolation parameter λ approaches 0 and the non-residual CNNs when it approaches $+\infty$. Therefore, the ODE shown in Eq. (4) bridges the residual and non-residual CNNs and inspires the design of new architectures of neural networks.

3.3. Interpolated Network Design

Based on the unified ODE model shown in Eq. (4), two types of $\rho(\lambda)$ are chosen and the corresponding network architectures are proposed. Considering the case when λ is small, we choose $\rho(\lambda) = 1$ and substitute the damping factor $e^{-\lambda\Delta t}$ by its first order approximation:

$$e^{-\lambda\Delta t} \approx 1 - \lambda\Delta t. \quad (11)$$

Then, from Eq. (8), the output of n-th layer is

$$\mathbf{x}_{n+1} = (1 - \lambda\Delta t)\mathbf{x}_n + \Delta t f_n(\mathbf{x}_n). \quad (12)$$

To guarantee the positiveness of λ, we add the ReLU function to the interpolation parameter λ and absorb the Δt into it. Thus the n-th layer of the network is

$$\mathbf{x}_{n+1} = (1 - \mathrm{ReLU}(\lambda_n))\mathbf{x}_n + f_n(\mathbf{x}_n). \quad (13)$$

Each λ_n is a trainable parameter for the n-th layer. It is known that the forward Euler discretization is stable when $\lambda\Delta t \in (0, 2)$, i.e. $\lambda \in (0, 2/\Delta t)$. As Δt in a continuous-time dynamic system is small, the stable range of λ can be viewed as a relaxation of $(0, +\infty)$, which coincides with the boundary condition in Eq. (9).

The second choice of the weight function is $\rho(\lambda) = \lambda + 1$ which satisfies the assumption in Eq. (9). Using the same approximation in Eq. 11, the scheme in Eq. (7) reduces to

$$\mathbf{x}_{n+1} = (1 - \lambda\Delta t)\mathbf{x}_n + (1 + \lambda\Delta t) f_n(\mathbf{x}_n). \quad (14)$$

Similar as the first choice, the second interpolated network is given by

$$\mathbf{x}_{n+1} = (1 - \mathrm{ReLU}(\lambda_n))\mathbf{x}_n + (1 + \mathrm{ReLU}(\lambda_n)) f_n(\mathbf{x}_n). \quad (15)$$

It is easy to know the interpolated networks shown in Eq. (13) and (15) recover a non-residual CNN if $\lambda_n = 1$ and a Residual network if $\lambda_n = 0$. As claimed in (He et al., 2016; Li et al., 2018), the identity shortcut connection helps mitigate gradient vanish problem and makes the loss landscape more smooth. It is natural that when $\lambda \to 0$ in Eq. (13), the optimization process of the interpolated model is much better than the non-residual CNN case with the same number of layers.

3.4. Interpolated Network Improves Robustness

Despite the high accuracy of ResNets, it is sensitive to the small perturbation of inputs due to the existence of adversarial examples. That is, for a fragile neural network, minor perturbation can accumulate dramatically with respect to layer propagation, resulting in giant shift of prediction. In this section, we show the improvment of the proposed interpolated networks over ResNets. The added damping term in our model weakens the amplitude of the solution of the original ODE. As a result, adding a damping term to the ODE model damps the error propagation process of ResNet, which improves model robustness.

In the following context, we show that robustness improvement of our proposed networks by using the stability analysis of the ODE.

Definition 3.2. *Let $\mathbf{x}^*$ be an equilibrium point of the ODE model (1). Then $\mathbf{x}^*$ is called asymptotically locally stable if there exists $\delta > 0$ such that $\lim_{t\to+\infty} \|\mathbf{x}_t - \mathbf{x}^*\| = 0$ for all starting points $\mathbf{x}_0$ within $\|\mathbf{x}_0 - \mathbf{x}^*\| \leq \delta$.*

Therefore, the perturbation around equilibrium $\mathbf{x}^*$ does not change the output the network if $\mathbf{x}^*$ is asymptotically locally stable. The next proposition from (Lyapunov, 1992; Chen, 2001) presents a classical method that checks the stability of nonlinear system around the equilibrium when f is time invariant. It is noted that this time invariant assumption may hold as the learned filters in the deep layers converges.

Proposition 3.3. *The equilibrium $\mathbf{x}^*$ of the ODE model*

$$\frac{\mathrm{d}\mathbf{x}}{\mathrm{d}t} = f(\mathbf{x}(t)) \tag{16}$$

is asymptotically locally stable if and only if $\mathrm{Re}(\nu) < 0$ where ν is the eigenvalue of $\partial_{\mathbf{x}} f(\mathbf{x}^*)$ which is the Jacobi matrix of f at $\mathbf{x}^*$.

Considering the damped ODE

$$\frac{\mathrm{d}\mathbf{x}}{\mathrm{d}t} = -\lambda\mathbf{x}(t) + \rho(\lambda) f(\mathbf{x}(t)), \tag{17}$$

the Jacobi matrix at the equilibrium $\mathbf{x}^*$ is

$$J_\lambda(\mathbf{x}^*) = \rho(\lambda)\partial_{\mathbf{x}} f(\mathbf{x}^*) - \lambda.$$

Then, the eigenvalues $\hat{\nu}$ of $J_\lambda(\mathbf{x}^*)$ are

$$\rho(\lambda)\nu - \lambda \tag{18}$$

where ν is the eigenvalue of $\partial_{\mathbf{x}} f(\mathbf{x}^*)$. When $\rho(\lambda) = 1$, we know

$$\mathrm{Re}(\hat{\nu}) = \mathrm{Re}(\nu) - \lambda < \mathrm{Re}(\nu).$$

By choosing positive λ properly, we know the ODE in Eq. (17) is asymptotically locally stable at $\mathbf{x}^*$. In general, we know

$$\mathrm{Re}(\hat{\nu}) < \mathrm{Re}(\nu) \Leftrightarrow \rho(\lambda) < 1 + \frac{\lambda}{\mathrm{Re}(\nu)},$$

which coincides with our assumption in Eq. (9). The above analysis shows that the stationary point of our proposed damped ODE model is more likely to be locally stable, and thus improve the its robustness when the input has be perturbed. In the experiments, our loss landscape visualization further validates this analysis.

4. Experiments

4.1. Setup

We evaluate our proposed model on CIFAR-10 and CIFAR-100 benchmarks, training and testing with the originally given dataset. Following (He et al., 2016), we adopt the simple data augmentation technique: padding 4 pixels on each side of the image and sampling a 32×32 crop from it or its horizontal flip. For ResNet experiments, we select the pre-activated version of ResNet-110 and ResNet-164 as baseline architectures. For ResNeXt experiments, we select ResNeXt-29, 8×64d as baseline from (Xie et al., 2017).

We apply Eq. (13) to ResNet-110, ResNet-164 and ResNeXt, and refer to them as In-ResNet-110, In-ResNet-164, and In-ResNeXt. We also apply Eq. (15) to ResNet-110 and ResNet-164, referring to them as λ-In-ResNet-110, λ-In-ResNet-164, and λ-In-ResNeXt.

The parameters λ_n of our interpolation models are initialized by randomly sampling from $\mathcal{U}[0.2, 0.25]$ in (λ-)In-ResNet-110 and (λ-)In-ResNeXt, and $\mathcal{U}[0.1, 0.2]$ in In-ResNet-164. The initialization of other parameters in ResNet and ResNeXt follows (He et al., 2016) and (Xie et al., 2017), respectively.

For all of the experiments, we use SGD optimizer with batch size = 128. For ResNet and (λ-)In-ResNet experiments, we train for 160 (300) epochs for the CIFAR-10 (-100) benchmark; the learning rate starts with 0.1, and is divided it by 10 at 80 (150) and 120 (225) epochs. We apply weight decay of 1e-4 and momentum of 0.9. For ResNeXt and (λ-)In-ResNeXt experiments, the learning rate starts at 0.05, and is divided it by 10 at 150 and 225 epochs. We apply weight decay of 5e-4 and momentum of 0.9.

We focus on two types of performances: optimization difficulty and model robustness. For optimization difficulty, we test our model on the CIFAR testing dataset. For model robustness, we evaluate the accuracy of our model over the perturbed inputs, details of which are given in the next section. For each experiment, we conduct 5 runs with different random seeds and report the averaged result to reduce the impact of random variations. The standard deviations of reported results can be found in Appendix D.

4.2. Measuring Robustness

In this section we introduce the two types of perturbation methods that we use: stochastic noise perturbations and adversarial attacks. For stochastic noise, we leverage the stochastic noise groups in CIFAR-10-C and CIFAR-100-C dataset (Hendrycks & Dietterich, 2019) for testing. The four groups of stochastic noise are impulse noise, speckle noise, Gaussian noise, and shot noise. For adversarial attacks, we consider three classical methods: Fast Gradient Sign Method (FGSM), Iterated Fast Gradient Sign Method (IFGSM), and Projected Gradient Descent (PGD). For a given data point $(\mathbf{x}, y)$:

- FGSM induces the adversarial example $\mathbf{x}'$ by moving

Benchmark	Model	Impulse	Speckle	Gaussian	Shot	Avg.
CIFAR-10	ResNet-110	56.38	59.12	43.82	55.47	53.70
	In-ResNet-110	**66.32**	**76.81**	**71.01**	**76.55**	**72.67**
	λ-In-ResNet-110	65.67	76.59	70.72	76.40	72.35
	ResNet-164	60.88	61.77	45.66	57.75	56.51
	In-ResNet-164	**67.95**	75.96	68.95	75.31	**72.05**
	λ-In-ResNet-164	65.72	**76.27**	**69.74**	**75.80**	71.88
	ResNeXt	55.12	58.21	39.14	52.06	51.13
	In-ResNeXt	**55.26**	**59.87**	**39.75**	**54.12**	**52.25**
	λ-In-ResNeXt	51.27	57.20	37.23	51.25	49.24
CIFAR-100	ResNet-110	25.36	29.69	20.16	27.81	25.76
	In-ResNet-110	32.00	**38.81**	30.00	37.71	34.63
	λ-In-ResNet-110	**32.15**	38.77	**30.02**	**37.82**	**34.69**
	ResNet-164	27.55	30.90	20.40	28.97	26.95
	In-ResNet-164	**33.05**	**39.50**	**29.77**	**38.17**	**35.12**
	λ-In-ResNet-164	32.92	38.79	29.08	37.53	34.58
	ResNeXt	**26.83**	28.29	17.09	25.67	24.47
	In-ResNeXt	25.85	29.90	18.59	27.72	25.52
	λ-In-ResNeXt	25.33	**31.18**	**19.88**	**28.75**	**26.29**

Table 1. Accuracy over the stochastic noise groups from CIFAR-10-C and CIFAR-100-C datasets, corresponded with perturbed CIFAR-10 and CIFAR-100 images from four types of stochastic noise, respectively. All of the results reported are averaged from 5 runs.

Model	CIFAR-10	CIFAR-100
ResNet-110	93.58	72.73
In-ResNet-110	92.28	70.55
λ-In-ResNet-110	92.15	70.39
ResNet-164	94.46	76.06
In-ResNet-164	92.69	72.94
λ-In-ResNet-164	92.55	73.22
ResNeXt	96.35	81.63
In-ResNeXt	96.48	81.64
λ-In-ResNeXt	96.22	81.29

Table 2. Accuracy over CIFAR-10 and CIFAR-100 testing data, representing optimization difficulty of each model. All of the results reported are averaged from 5 runs.

with step size of ϵ at each component of the gradient descent direction, namely

$$\mathbf{x}' = \mathbf{x} + \epsilon \cdot \mathrm{sign}(\nabla_{\mathbf{x}} \mathcal{L}(\mathbf{x}, y)). \qquad (19)$$

- IFGSM performs FGSM with step size of α, and clips the perturbed images within $[\mathbf{x} - \epsilon, \mathbf{x} + \epsilon]$ iteratively, namely

$$\mathbf{x}^{(m+1)} = \mathrm{Clip}_{\mathbf{x},\epsilon} \left\{ \mathbf{x}^{(m)} + \alpha \cdot \mathrm{sign}(\nabla_{\mathbf{x}} \mathcal{L}(\mathbf{x}^{(m)}, y)) \right\}, \qquad (20)$$

  where $m = 1, 2, \cdots, M$, $\mathbf{x}^{(0)} = \mathbf{x}$, and $\mathbf{x}^{(M)}$ is the induced adversarial image. In our experiments, we set $\alpha = 2/255$ and iteration times $M = 20$.

- PGD attack is the same with IFGSM, except that the $\mathbf{x}^{(0)} = \mathbf{x} + \delta$ with $\delta \sim \mathcal{U}[-\epsilon, \epsilon]$.

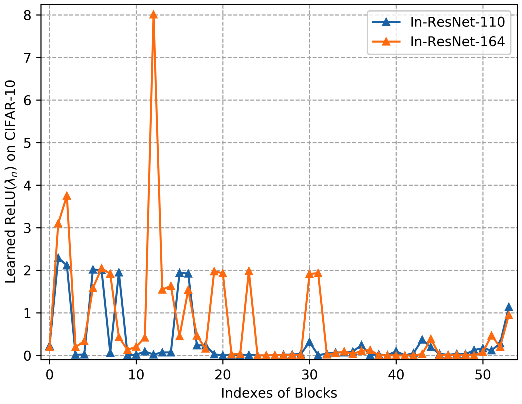


Figure 1. Learned interpolation coefficients in In-ResNet-110 and In-ResNet-164 models trained on CIFAR-10 benchmarks.

4.3. Results

Optimization difficulty Table 2 shows the results of In-ResNet-110 and In-ResNet-164 as well as the baselines over CIFAR-10 and CIFAR-100 testing set. On one hand, it can be seen that for (λ-)In-ResNet-110 and (λ-)In-ResNet-164, there is accuracy drop within 3 percent compared with the ResNet baselines. This agrees with the fact that the interpolation model may be harder to optimize than ResNet. However, the performance of the interpolation models are still much better than that of the deep non-residual CNN models.

Robustness against stochastic noise Table 1 shows the ac-

Benchmark	Model	FGSM			IFGSM			PGD		
		1/255	2/255	4/255	1/255	2/255	4/255	1/255	2/255	4/255
CIFAR-10	ResNet-110	58.59	41.48	29.45	39.45	5.93	0.06	38.91	5.60	0.06
	In-ResNet-110	**71.97**	**55.24**	**38.26**	65.70	**32.05**	**5.14**	65.66	**31.74**	**5.01**
	λ-In-ResNet-110	71.06	50.84	30.05	**65.93**	30.72	3.52	**65.81**	30.45	3.41
	ResNet-164	63.32	44.37	30.21	46.79	8.19	0.09	46.43	7.77	0.07
	In-ResNet-164	**70.88**	**51.84**	**32.81**	**64.34**	**27.43**	**2.27**	**64.20**	**26.95**	**2.15**
	λ-In-ResNet-164	70.01	50.53	31.77	63.33	26.50	2.01	63.19	26.04	1.91
CIFAR-100	ResNet-110	28.01	**18.74**	**14.12**	15.05	2.18	0.28	14.69	2.11	0.26
	In-ResNet-110	32.24	**18.74**	11.84	23.44	4.92	**0.55**	23.22	4.81	**0.53**
	λ-In-ResNet-110	**32.79**	18.40	11.24	**24.17**	**5.17**	0.53	**24.03**	**5.00**	0.51
	ResNet-164	35.15	**23.58**	**17.04**	21.23	3.45	0.29	20.78	3.31	0.22
	In-ResNet-164	37.21	22.30	13.93	28.05	6.59	**0.73**	27.75	6.34	**0.67**
	λ-In-ResNet-164	**37.37**	22.50	13.94	**28.25**	**6.64**	0.69	**28.03**	**6.46**	0.64

Table 3. Accuracy over perturbed CIFAR-10 and CIFAR-100 images from FGSM, IFGSM, and PGD adversarial attacks with different attack radii. All of the results reported are averaged over 5 runs.

curacies of all models over the perturbed CIFAR-10 and CIFAR-100 images from four types of stochastic noise. Our In-ResNet-110 and In-ResNet-164 models achieve substantial improvement over the ResNet-110 and ResNet-164 baselines. For perturbed CIFAR-10 images, accuracy of (λ-)In-ResNet-110 and (λ-)In-ResNet-164 are over 15 % higher than ResNet-110 and ResNet-164 baselines on average. For perturbed CIFAR-100 images, accuracy of (λ-)In-ResNet-110 and (λ-)In-ResNet-164 are over 5% higher than ResNet-110 and ResNet-164 baselines on average. In-ResNeXt models improves the accuracy of the perturbed images over ResNeXt as well.

Robustness against adversarial attacks Table 3 shows the accuracies of all models over the perturbed CIFAR-10 and CIFAR-100 images from FGSM, IFGSM, and PGD attacks at different attack radii of $1/255$, $2/255$, and $4/255$. Most of the robustness results of our (λ-)In-ResNet-110 and (λ-)In-ResNet-164 models are higher than those of the ResNet-110 and ResNet-164 models, which is empirically consistent with our Lyapunov analysis. Especially on CIFAR-10 benchmark, our In-ResNet-110 and In-ResNet-164 models obtain significant robustness improvement against the strong IFGSM and PGD attacks at the radii of $1/255$ and $2/255$.

Learned interpolation coefficients To get a better understanding of the interpolation model, we plot the interpolation coefficients $\{\mathrm{ReLU}(\lambda_n)\}$ in In-ResNet-110 and In-ResNet-164 models trained on CIFAR-10 benchmarks. As shown in Fig 1, most of the interpolation coefficients lie within the range $[0,\ 1]$, suggesting an interpolating behaviour. According to Eq. (13), interpolation coefficients lying within $[1,\ 2]$ represent negative skip connections, with the absolute weight scale of less than 1. Very few of the interpolation coefficients are larger than 2, which is in line with the stability range of forward Euler scheme. In general, 79.6%(72.2%) of λ_n's in In-ResNet-110(164) are larger than 0.01, which accounts for the significance in robustness. More visualizations of learned interpolation coefficients can be found in Appendix A.

Loss landscape analysis As is given by the Lyapunov analysis, the robustness improvement is theoretically provided in that the damped models enjoy more locally stable points than the original ones. To further verify this, we visualize the loss landscapes of In-ResNet-110 and ResNet-110 models trained on CIFAR-10 benchmark along the attack direction. For a instance $(\mathbf{x}, y)$, we plot the loss function $L(\mathbf{x}, y)$ of along the FGSM attack direction. We also select a random orthogonal direction from the FGSM attack one and plot the model predictions of each grids. The unit of each axis in the figures is at the scale of $1/255$. To better analyze model robustness, we select the data instance $(\mathbf{x}, y)$ for the CIFAR-10-C dataset, namely $(\mathbf{x}^{'}, y)$, where $\mathbf{x}^{'}$ is $\mathbf{x}$ with injected stochastic noise.

Figure 2 illustrates the loss landscapes of the ResNet-110 and In-ResNet-110 models along the FGSM attack direction. We select two input data instances: for Figure 2-{(a)-(c)}, the input is the 3-th image in the shot noise group of the CIFAR-10-C dataset, the ground-truth label of which is ship; for Figure 2-{(d)-(f)}, the input is the 8-th image in the speckle noise group of the CIFAR-10-C dataset, the ground-truth label of which is horse. For the first input example, ResNet-110 and In-ResNet-110 both make the correct prediction; for the second input example, they both make the wrong one. It can be seen that the added damping term have damped the loss landscape along the FGSM attack direction, resulting in a much weaker amplitude (the first example), or even turned the amplifying loss landscape of ResNet-110 into a damping one of In-ResNet-110 (the second example). Whether ResNet-110 and In-ResNet-110

Model	Acc.	noise	FGSM	IFGSM	PGD
ResNet-110	**93.58**	53.70	41.48	5.93	5.60
In-ResNet-110	92.28	**72.67**	**55.24**	**32.05**	**31.74**
In-ResNet-sig-110	93.49	55.04	44.65	6.29	5.94
In-ResNet-gating-110	93.46	54.53	41.25	5.65	5.33
In-ResNet-gating-sig-110	90.68	68.04	46.17	21.89	21.65

Table 4. Accuracy and robustness of In-ResNet-110, In-ResNet-sig-110, In-ResNet-gating-110, and In-ResNet-gating-sig-110 models, as well as the ResNet-110 baseline on CIFAR-10 benchmarks. "Acc." denotes the accuracy over CIFAR-10 testing set. "noise" denotes the **average** accuracy of the four stochastic noise groups from CIFAR-10-C. "FGSM", "IFGSM", and "PGD" represent accuracy under the corresponding attacks at the attack radius of $2/255$. All of the results reported are averaged over 5 runs.

both make the correct prediction or the wrong one, it is clear that the In-ResNet-110 model enjoys better robustness than ResNet-110, which agrees with our Lyapunov analysis that the damping term has introduced more locally stable points.

4.4. Comparison among In-ResNet Variants

While Eq. (13) depicts the In-ResNet structure, in this section, we propose several variants of In-ResNet and compare their performances. To facilitate the discussion, the In-ResNet can be written in the general form:

$$\mathbf{x}_{n+1} = (1 - \mathrm{act}(d(\mathbf{x}_n)))\mathbf{x}_n + \Delta t f_n(\mathbf{x}_n), \qquad (21)$$

where $d(x_n)$ is the function determining the interpolation coefficients. act is the activation function. For In-ResNet, the $d(x_n)$ is a learnable scalar parameter λ_n; act is ReLU function. We propose several In-ResNet variants:

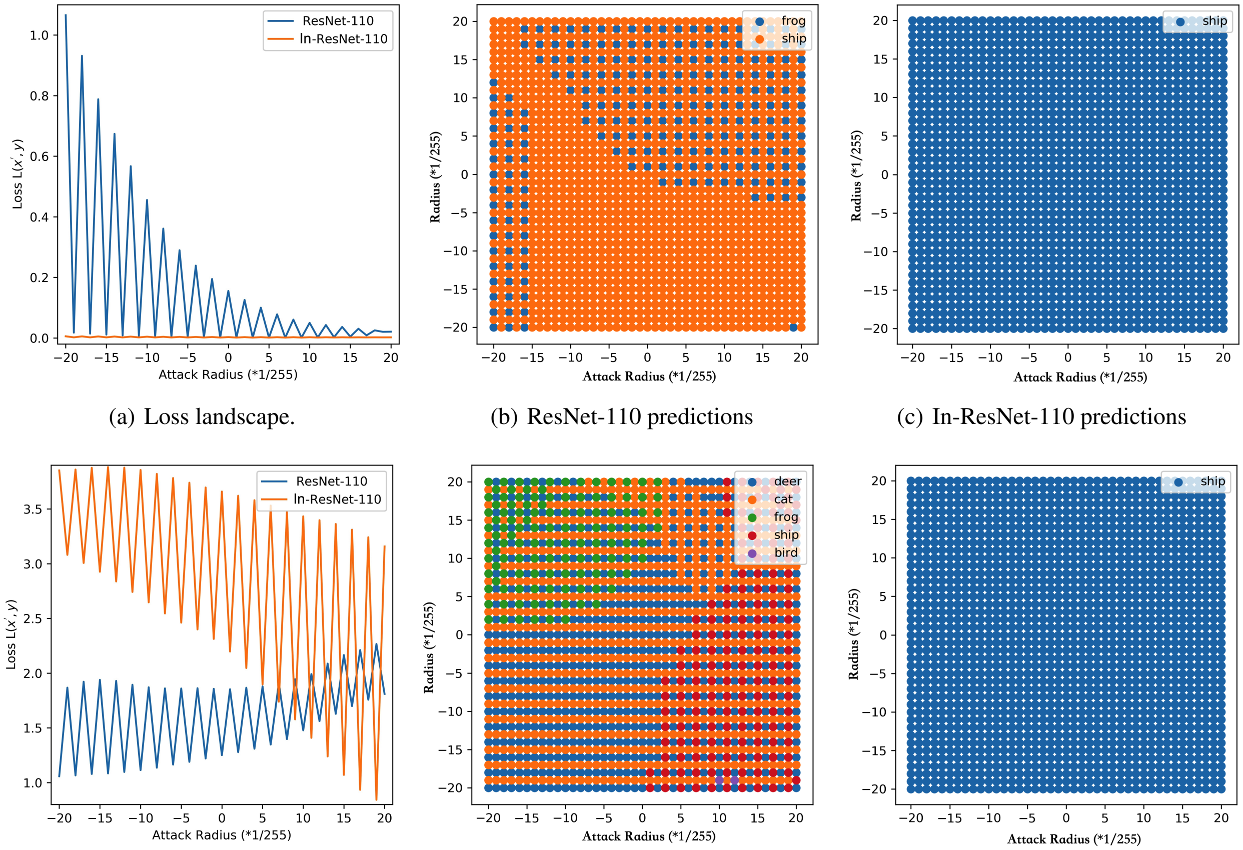
- $d(\mathbf{x}_n) = \lambda_n$, $\mathrm{act} = \mathrm{sigmoid}$: we replace the activation function to be $\mathrm{sigmoid}$, which restricts the interpolation coefficients to be within $[0,\ 1]$, and thus guarantees that the learned model is an interpolation. We refer to it as In-ResNet-sig.
- $d(\mathbf{x}_n) = W_d\mathbf{x}_n + b_d$, $\mathrm{act} = \mathrm{ReLU}$: we let the learnable scalar parameters determined by a linear transformation from input x_n, yielding a gating mechanism. We refer to it as In-ResNet-gating.
- $d(\mathbf{x}_n) = W_d\mathbf{x}_n + b_d$, $\mathrm{act} = \mathrm{sigmoid}$: based on the previous variant, we further replace the activation function to be $\mathrm{sigmoid}$. It is noteworthy that this variant is the shortcut-only gating mechanism discussed in (He et al., 2016). We refer to it as In-ResNet-gating-sig.

We use In-ResNet-110 as the basic In-ResNet model and experiment on CIFAR-10 benchmark to compare their performance. The accuracy and robustness results are reported averagely from 5 runs, shown in Table 4. We elaborately tune the initialization intervals and report the model with the largest sum of the accuracy over both the CIFAR-10 testing set and the noise groups in the CIFAR-10-C dataset. It can be seen that In-ResNet-110 leads to the largest robustness improvements over ResNet-110 baseline, with a relatively small accuracy drop. The In-ResNet-sig-110 model achieves better accuracy result than In-ResNet-110, however, its performance on robustness improvements are marginal. This is because the learned interpolation coefficients in In-ResNet-sig-110 are close to 0, resulting in nearly identity skip-connections. Similarly, the performance of In-ResNet-gating-110 is very close to ResNet-110 baseline due to the degeneration of its damped skip-connections. The In-ResNet-gating-sig-110 model also improves over the ResNet-110 baseline with a large margin in terms of robustness performance. The improvement, however, is less significant than our In-ResNet-110 model. The accuracy of the In-ResNet-gating-sig-110 model also lags behind In-ResNet-110, which may attribute to the extra optimization difficulty introduced by the gating mechanism.

4.5. Trade-off between Optimization and Robustness

As is shown in Table 2, while (λ-)In-ResNet enjoys better robustness, it suffers from optimization difficulty: an accuracy degeneration around 2% is caused by our (λ-)In-ResNet model. In this section, we show that the initialization of λ_n is of great importance to the optimization process. We use In-ResNet-110 model and λ-In-ResNet-110 model trained on CIFAR-10 benchmark as the basic model, initializing λ_n by randomly sampling from $\mathcal{U}[x, y]$. For the basic model, we have that $\mathcal{U}[x, y] = \mathcal{U}[0.2, 0.25]$. We try the following initialization schemes as well: $\mathcal{U}[x, y] = \mathcal{U}[0, 0.1]$, $\mathcal{U}[0.1, 0.2]$, $\mathcal{U}[0.2, 0.25]$, $\mathcal{U}[0.25, 0.3]$, and $\mathcal{U}[0.3, 0.4]$. The accuracy and robustness results are reported averagely from 5 runs, shown in Table 5.

From the experimental results, we can see that the performance of (λ-)In-ResNet is sensitive to the initialization of λ_n. On one hand, as the initialization becomes larger, the model robustness goes up. This agrees with our Lyapunov analysis, as the larger initialization of λ_n's tends to help model to converge to the larger final λ_n's, yielding larger damping terms and better robustness. One the other hand, larger initialization leads to worse accuracy results. Espe-



(a) Loss landscape. (b) ResNet-110 predictions (c) In-ResNet-110 predictions

(d) Loss landscape. (e) ResNet-110 predictions (f) In-ResNet-110 predictions

Figure 2. The input data instance is the 3-th/8-th image in the shot/speckle noise group of the CIFAR-10-C dataset for (a)-(c)/(d)-(f), the ground truth label of which is ship/horse. For (a)-(c)/(d)-(f), ResNet-110 and In-ResNet-110 both make the correct / wrong prediction. (a) and (c) depict the loss landscape of ResNet-110 and In-ResNet-110 along the FGSM attack direction. {(b) and (e)} / {(c) and (f)} illustrate model predictions of {ResNet-110} / {In-ResNet-110} at each grids determined by the FGSM attack direction and a random orthogonal direction.

Model	Initialization	Acc.	noise	FGSM	IFGSM	PGD
ResNet	-	**93.58**	53.70	41.48	5.93	5.60
In-ResNet	$\mathcal{U}[0.00, 0.10]$	**93.51**	55.15	46.74	8.39	7.96
	$\mathcal{U}[0.10, 0.20]$	93.25	62.88	49.58	16.89	16.46
	$\mathcal{U}[0.20, 0.25]$	92.28	72.67	55.24	32.05	31.74
	$\mathcal{U}[0.25, 0.30]$	91.63	76.20	55.79	36.53	36.28
	$\mathcal{U}[0.30, 0.40]$	90.62	**79.35**	**55.95**	**41.07**	**40.84**
λ-In-ResNet	$\mathcal{U}[0.00, 0.10]$	**93.41**	54.18	42.28	6.78	6.48
	$\mathcal{U}[0.10, 0.20]$	92.86	63.58	46.07	16.99	16.60
	$\mathcal{U}[0.20, 0.25]$	92.15	72.35	50.84	30.72	30.45
	$\mathcal{U}[0.25, 0.30]$	91.30	75.65	53.29	36.90	36.74
	$\mathcal{U}[0.30, 0.40]$	90.17	**79.66**	**55.03**	**41.06**	**40.94**

Table 5. Accuracy and robustness results of In-ResNet-110 and λ-In-ResNet-110 with different initialization schemes. "Acc." denotes the accuracy over CIFAR-10 testing set. "noise" denotes the **average** accuracy of the four stochastic noise groups from CIFAR-10-C. "FGSM", "IFGSM", and "PGD" represent model accuracy under the corresponding attacks at the radius of $2/255$. All of the results reported are averaged over 5 runs except for $\mathcal{U}[0.3, 0.4]$: they are averaged over 4(2) runs, as 1(3) out of 5 runs for In-ResNet-110 (λ-In-ResNet-110) failed with a final accuracy of 10% on CIFAR-10 test set.

Model	Acc.	noise	FGSM	IFGSM	PGD
ResNet-110	93.58	53.70	41.48	5.93	5.60
ResNet-110, ens	**95.03**	55.70	43.99	6.26	5.93
In-ResNet-110	92.28	72.67	55.24	32.05	31.74
In-ResNet-110, ens	94.03	**75.86**	**58.42**	**34.44**	**34.03**
λ-In-ResNet-110	92.15	72.35	50.84	30.72	30.45
λ-In-ResNet-110, ens	94.00	75.29	53.66	32.95	32.77
ResNet-164	94.46	56.51	44.37	8.19	7.77
ResNet-164, ens	**95.44**	58.76	46.54	8.53	8.14
In-ResNet-164	92.69	72.05	51.84	27.43	26.95
In-ResNet-164, ens	94.26	**75.26**	**54.72**	**28.97**	**28.51**
λ-In-ResNet-164	92.55	71.88	50.53	26.50	26.04
λ-In-ResNet-164, ens	94.20	74.97	53.17	27.74	27.30

Table 6. Comparison between the accuracy and robustness results of the ensemble model over 5 different runs and those of the single model (scores are averaged). "Acc." denotes the accuracy over CIFAR-10 testing set. "noise" denotes the **average** accuracy of the four stochastic noise groups from CIFAR-10-C. "FGSM", "IFGSM", and "PGD" represent model accuracy under the corresponding attacks at the radius of $2/255$.

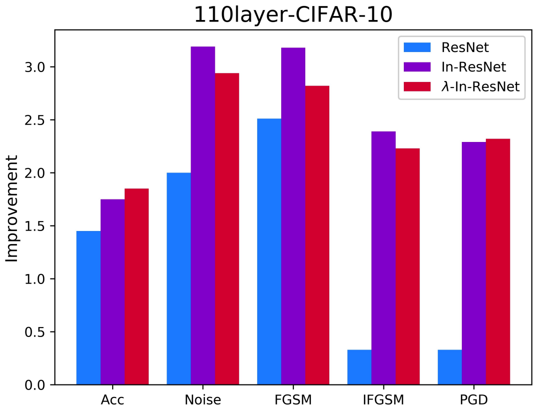


Figure 3. The accuracy **improvements** over single models for the ensemble ResNet-110, In-ResNet-110 and λ-In-ResNet-110 over CIFAR-10 dataset. Both of the ensemble of our models have more significant accuracy improvements than the ensemble of the baseline ResNet-110 model.

cially for $\mathcal{U}[0.30, 0.40]$, 1(3) out of 5 runs of In-ResNet-110 (λ-In-ResNet-110) fails the optimization with a final accuracy of 10%. This can be interpreted that the damped shortcuts hamper information propagation and lead to optimization difficulty (He et al., 2016). More results on CIFAR-100 benchmark can be found in Appendix B.

4.6. Effect of Model Ensemble

It is known that an ensemble model is more robust than a single model (Wang et al., 2019). To further improve accuracy and robustness, we perform model ensemble over the 5 different runs of baseline and our models. Table 6 shows the comparison between ensemble models and single models for ResNet-110, In-ResNet-110 and λ-ResNet-110 over CIFAR-10 dataset. It can be seen that all of the ensemble models are more robust and more accurate than the corresponding single models.

We also plot the accuracy **improvements** over single models for the ensemble ResNet-110, In-ResNet-110 and λ-In-ResNet-110. As shown in 3, both of the ensemble of our models have more significant accuracy improvements than the ensemble of the baseline ResNet-110 model. This can be attributed to the performance difference among different runs of our model due to optimization difficulty. More results and visualizations of the effect of ensemble method can be found in Appendix C.

5. Conclusion

While the relationship between ODEs and non-residual networks remains unclear, in this paper, we present a novel ODE model by adding a damping term. By adjusting the interpolation coefficient, the proposed model unifies the interpretation of both residual and non-residual networks. Lyapunov analysis and experimental results on CIFAR-10 and CIFAR-100 benchmarks reveals better robustness of the proposed interpolated networks against both stochastic noise and several adversarial attack methods. Loss landscape analysis reveals the improved robustness of our method along the attack direction. Furthermore, experiments show that the performance of proposed model is sensitive to the initialization of the interpolation coefficients, demonstrating trade-off between optimization difficulty and robustness. The significance of the design of interpolated networks is shown by comparing several model variants. Future work includes determining the interpolated coefficients as a black-box process and leveraging data augmentation techniques to improve our models.

Acknowledgements

We thank all the anonymous reviewers for their suggestions. Yang Liu is supported by the National Key R&D Program of China (No. 2017YFB0202204), National Natural Science Foundation of China (No. 61925601, No. 61761166008), and Huawei Technologies Group Co., Ltd. Chenglong Bao is supported by National Natural Sciences Foundation of China (No. 11901338) and Tsinghua University Initiative Scientific Research Program. Zuoqiang Shi is supported by National Natural Sciences Foundation of China (No. 11671005). This work is also supported by Beijing Academy of Artificial Intelligence.

References

Chang, B., Meng, L., Haber, E., Ruthotto, L., Begert, D., and Holtham, E. Reversible architectures for arbitrarily deep residual neural networks. In *AAAI*, 2018.

Chang, B., Chen, M., Haber, E., and Chi, E. H. Antisymmetricrnn: A dynamical system view on recurrent neural networks. In *ICLR*, 2019.

Chen, G. Stability of nonlinear systems. *Wiley Encyclopedia of Electrical and Electronics Engineering*, 2001.

Chen, R. T. Q., Rubanova, Y., Bettencourt, J., and Duvenaud, D. Neural Ordinary Differential Equations. In *NeurIPS*, 2018.

Dupont, E., Doucet, A., and Teh, Y. W. Augmented neural odes. In *NeurIPS*, pp. 3134–3144, 2019.

E, W. A proposal on machine learning via dynamical systems. *Communications in Mathematics and Statistics*, 5 (1):1–11, 2017.

Haber, E. and Ruthotto, L. Stable architectures for deep neural networks. *Inverse Problems*, 34(1):014004, 2017.

Han, J., Jentzen, A., and E, W. Solving high-dimensional partial differential equations using deep learning. *PNAS*, 115(34):8505–8510, 2018.

Hanshu, Y., Jiawei, D., Vincent, T., and Jiashi, F. On robustness of neural ordinary differential equations. In *ICLR*, 2019.

He, K., Zhang, X., Ren, S., and Sun, J. Identity mappings in deep residual networks. In *ECCV*, pp. 630–645. Springer, 2016.

Hendrycks, D. and Dietterich, T. Benchmarking neural network robustness to common corruptions and perturbations. *ICLR*, 2019.

Li, H., Xu, Z., Taylor, G., Studer, C., and Goldstein, T. Visualizing the loss landscape of neural nets. In *NeurIPS*, pp. 6389–6399, 2018.

Liu, X., Si, S., Cao, Q., Kumar, S., and Hsieh, C.-J. Neural sde: Stabilizing neural ode networks with stochastic noise. *arXiv:1906.02355*, 2019.

Lu, Y., Zhong, A., Li, Q., and Dong, B. Beyond Finite Layer Neural Networks: Bridging Deep Architectures and Numerical Differential Equations. In *ICML*, 2018.

Lu, Y., Li, Z., He, D., Sun, Z., Dong, B., Qin, T., Wang, L., and Liu, T.-Y. Understanding and improving transformer from a multi-particle dynamic system point of view. *arXiv:1906.02762*, 2019.

Lyapunov, A. M. The general problem of the stability of motion. *International journal of control*, 55(3):531–534, 1992.

Reshniak, V. and Webster, C. Robust learning with implicit residual networks. *arXiv:1905.10479*, 2019.

Su, D., Zhang, H., Chen, H., Yi, J., Chen, P.-Y., and Gao, Y. Is robustness the cost of accuracy?–a comprehensive study on the robustness of 18 deep image classification models. In *ECCV*, pp. 631–648, 2018.

Tao, Y., Sun, Q., Du, Q., and Liu, W. Nonlocal neural networks, nonlocal diffusion and nonlocal modeling. In *NeurIPS*, pp. 496–506, 2018.

Wang, B., Yuan, B., Shi, Z., and Osher, S. J. Enresnet: Resnet ensemble via the feynman-kac formalism. In *NeurIPS*, 2019.

Xie, S., Girshick, R., Dollár, P., Tu, Z., and He, K. Aggregated residual transformations for deep neural networks. In *CVPR*, pp. 1492–1500, 2017.

Zhang, D., Zhang, T., Lu, Y., Zhu, Z., and Dong, B. You only propagate once: Painless adversarial training using maximal principle. In *NeurIPS*, 2019a.

Zhang, J., Han, B., Wynter, L., Low, K. H., and Kankanhalli, M. Towards robust resnet: A small step but a giant leap. In *IJCAI*, 2019b.

Zhu, M., Chang, B., and Fu, C. Convolutional Neural Networks combined with Runge-Kutta Methods. *arXiv.org*, February 2018.