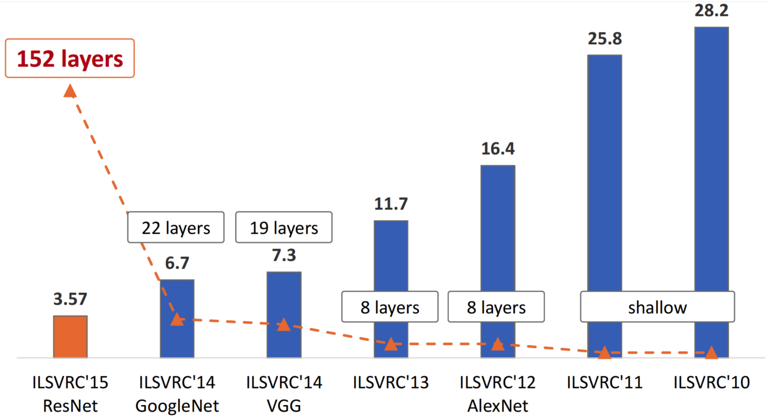
Interpolation between Residual And Non-Residual Networks

Zonghan Yang, Yang Liu, Chenglong Bao, Zuoqiang Shi

Tsinghua University

August 21, 2020

Deep Learning, Deep Network



- The deeper the network, the better the performance[1];
- What about going beyond finite layers? **ODE perspective!**

[1] Kien et. al., Iris recognition with off-the-shelf CNN features: A deep learning perspective. IEEE Access

ODE Perspective of Neural Networks

- Considering the ordinary differential equation:

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

- Applying the forward Euler discretization scheme:

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

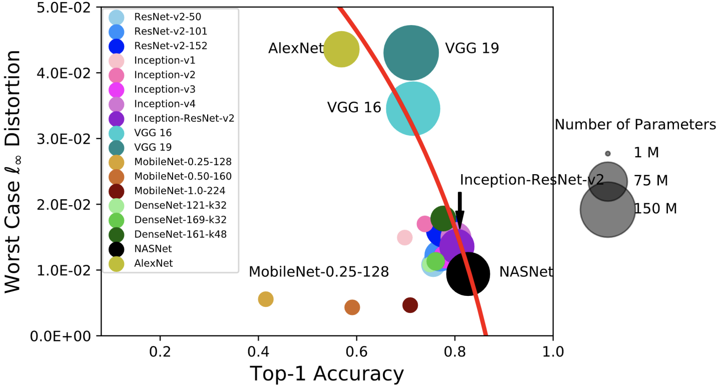
- Let $\mathbf{x}_n = \mathbf{x}(t_n)$, $\Delta t = 1$ ...

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

- ... and it recovers a residual network.
- Such connection relies on the residual connections!

Our Goal

- ODE perspective for **non-residual** networks?
- Non-residual CNNs sometimes enjoy better robustness[2]:



- Need better ODE to unify residual and non-residual networks!

[2] Su et. al., Is Robustness the Cost of Accuracy? – A Comprehensive Study on the Robustness of 18 Deep Image Classification Models. ECCV 2018

ODE perspective for Model Robustness

Several related work ...

- Decrease step size: Zhang et al., IJCAI 2019;
- Implicit numerical scheme: Reshniak et al., arxiv 2019;
- Time-invariant neural ODE: Yan et al., ICLR 2020;
- Neural Stochastic Differential Equation: Liu et al., arxiv 2019;
- Ensemble of noise-injected ResNet: Wang et al., NeurIPS 2019;
- Adv. training as differential game: Zhang et al., NeurIPS 2019.

Most of them focus on numerical scheme or stochasticity!

We propose a novel ODE model that unifies residual and non-residual networks, which improves model robustness.

Proposed ODE Model

- Adding a damping term to the original ODE:

$$\frac{\mathrm{d}\mathbf{x}(t)}{\mathrm{d}t} = -\lambda \mathbf{x}(t) + \rho(\lambda) f(\mathbf{x}(t), t), \;\; \mathbf{x}(0) = \mathbf{x}_0. \tag{4}$$

- Constant $\lambda \in [0, +\infty)$: interpolation coefficient;
- $\rho : [0, +\infty) \mapsto [0, +\infty)$: weight function.

Proposition

For any $T > 0$, the solution of the ODE (4) is

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

Unification of Residual and Non-Residual Networks

- Assuming $f(\mathbf{x}(t), t) = f(\mathbf{x}_n, t_n)$ for all $t \in [t_n, t_{n+1})$, the iterative scheme[3] is

$$\mathbf{x}_{n+1} = e^{-\lambda \Delta t}\mathbf{x}_n + \frac{1 - e^{-\lambda \Delta t}}{\lambda}\rho(\lambda) f_n(\mathbf{x}_n). \tag{6}$$

- When the weight function $\rho(\lambda)$ satisfies

$$\rho(\lambda) \to 1, \lambda \to 0^+ \text{ and } \rho(\lambda) \sim \lambda, \lambda \to +\infty, \tag{7}$$

- the output of n-th layer is

$$\mathbf{x}_{n+1} = \begin{cases} \mathbf{x}_n + f_n(x_n), & \text{if } \lambda \to 0^+, \\ \Delta t f_n(x_n), & \text{if } \lambda \to +\infty. \end{cases} \tag{8}$$

[3] This is in fact the 1^{st} order ETD scheme.

Damping Term Leads to Robustness

Proposition

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

$$\frac{\mathrm{d}\mathbf{x}}{\mathrm{d}t} = f(\mathbf{x}(t)) \tag{9}$$

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 $\mathbf{x}^*$.

- The proposed ODE model has more locally stable equilibrium points, as the Jacobi matrix is reduced by the damping term.
- More locally stable points $\leftrightarrow$ more robust data points!

Interpolated Neural Network Design

- Instantiate with $\rho(\lambda) = 1$, $e^{-\lambda \Delta t} \approx 1 - \lambda \Delta t$:

$$\mathbf{x}_{n+1} = (1 - \lambda \Delta t)\mathbf{x}_n + \Delta t f_n(\mathbf{x}_n). \tag{10}$$

- Equip with learnable positive λ, yielding **In-ResNet**:

$$\mathbf{x}_{n+1} = (1 - \mathrm{ReLU}(\lambda_n))\mathbf{x}_n + f_n(\mathbf{x}_n). \tag{11}$$

- Instantiate with $\rho(\lambda) = \lambda + 1$, $e^{-\lambda \Delta t} \approx 1 - \lambda \Delta t$:

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

- and further reduces to **$\boldsymbol{\lambda}$-In-ResNet**:

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

Accuracy and Robustness Results

Benchmark	Model	Acc.	Noise	FGSM	IFGSM	PGD
CIFAR-10	ResNet-110	**93.58**	53.70	41.98	5.93	5.60
	In-ResNet-110	92.28	**72.67**	**55.24**	**32.05**	**31.74**
	λ-In-ResNet-110	92.15	72.35	50.84	30.72	30.45
	ResNet-164	**94.46**	56.51	44.37	8.19	7.77
	In-ResNet-164	92.69	**72.05**	**51.84**	**27.43**	**26.95**
	λ-In-ResNet-164	92.55	71.88	50.53	26.50	26.04
CIFAR-100	ResNet-110	**72.73**	25.76	**18.74**	2.18	2.11
	In-ResNet-110	70.55	34.63	**18.74**	4.92	4.81
	λ-In-ResNet-110	70.39	**34.69**	18.40	**5.17**	**5.00**
	ResNet-164	**76.06**	26.95	**23.58**	3.45	3.31
	In-ResNet-164	72.94	**35.12**	22.30	6.59	6.34
	λ-In-ResNet-164	73.22	34.58	22.50	**6.64**	**6.46**

Table 1: Accuracy and robustness results.

- Accuracy slightly drops but robustness is largely improved!

Learned Interpolation Coefficients

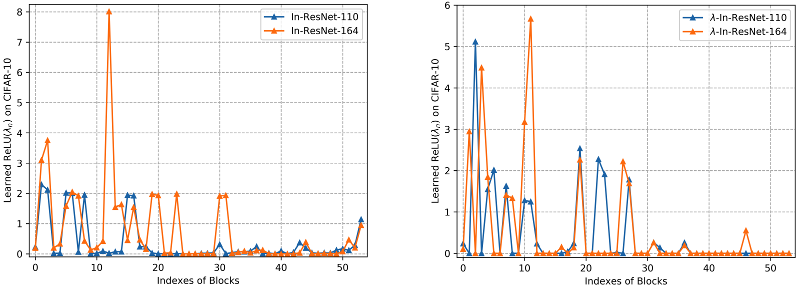
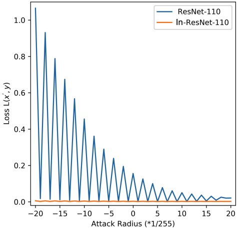


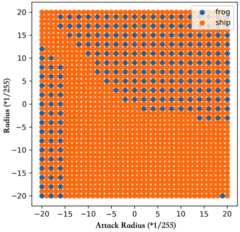
Figure 1: Learned interpolation coefficients on CIFAR-10.

- Most of the learned interpolation coefficients lie in $[0, 2]$;
- Stable range for the forward numerical scheme!

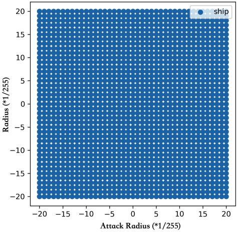
Loss Landscape Analysis with CIFAR-C Input



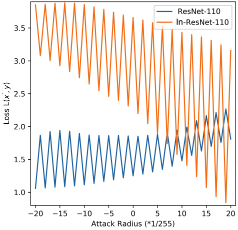
(a) Loss landscape.



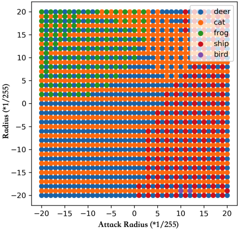
(b) ResNet predictions



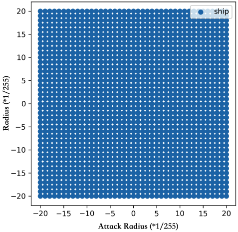
(c) In-ResNet predictions



(d) Loss landscape.



(e) ResNet predictions



(f) In-ResNet predictions

Comparison with In-ResNet Variants

- General form of In-ResNet:

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

- In-ResNet variants:

	$d(\mathbf{x}_n) = \lambda_n$	$d(\mathbf{x}_n) = W_d \mathbf{x}_n + b_d$
$\mathrm{act} = \mathrm{ReLU}$	In-ResNet	In-ResNet-gating
$\mathrm{act} = \mathrm{sigmoid}$	In-ResNet-sig	In-ResNet-gating-sig

- Comparison of accuracy and robustness:

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

Comparison with In-ResNet Variants

- General form of In-ResNet:

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

- In-ResNet variants:

	$d(\mathbf{x}_n) = \lambda_n$	$d(\mathbf{x}_n) = W_d\mathbf{x}_n + b_d$
$\mathrm{act} = \mathrm{ReLU}$	In-ResNet	In-ResNet-gating
$\mathrm{act} = \mathrm{sigmoid}$	In-ResNet-sig	In-ResNet-gating-sig

- Comparison of accuracy and robustness:

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

Comparison with In-ResNet Variants

- General form of In-ResNet:

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

- In-ResNet variants:

	$d(\mathbf{x}_n) = \lambda_n$	$d(\mathbf{x}_n) = W_d\mathbf{x}_n + b_d$
$\mathrm{act} = \mathrm{ReLU}$	In-ResNet	In-ResNet-gating
$\mathrm{act} = \mathrm{sigmoid}$	In-ResNet-sig	In-ResNet-gating-sig

- Comparison of accuracy and robustness:

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

Trade-off between Robustness and Accuracy

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**

- Uniformly sampling as λ_n's initialization from different $\mathcal{U}[x, y]$;
- Becoming non-residual: accuracy drops and robustness rises[4].

[4] 1(3) out of 5 fails for (λ-)In-ResNet with $\mathcal{U}[0.30, 0.40]$.

Ensemble Leads to Robustness Improvement

- Model ensemble over 5 different runs:

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

- Robustness of all the models rises ...
- ... what about robustness improvement?
- Compare the **difference** between robustness of the ensemble model and robustness of the single model ...

Ensemble Leads to Robustness Improvement

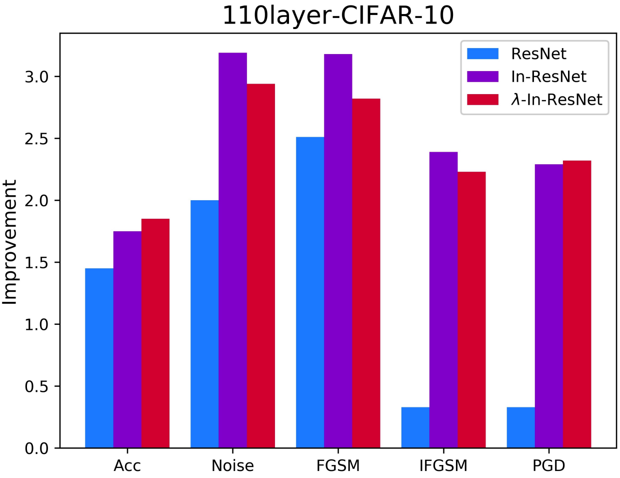


Figure 2: Comparison of robustness improvements.

Takeaways

- Current Neural ODE model relies on residual connection;
- We propose a damped ODE model and unify residual and non-residual networks from Neural ODE perspective;
- Theory and experimental results show the robustness improvement of our model.

- Paper: `https://arxiv.org/abs/2006.05749`
- Code: `https://github.com/minicheshire/InResNet`

Thank you for your attention!