Preventing Gradient Attenuation in Lipschitz Constrained Convolutional Networks

Qiyang Li*, Saminul Haque*, Cem Anil, James Lucas,
Roger Grosse, Jörn-Henrik Jacobsen

January 13, 2020

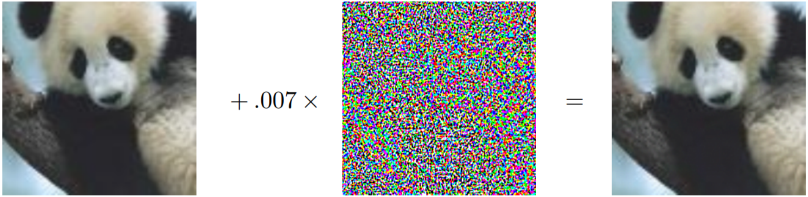
Introduction: Adversarial Examples

Given a classifier f that can correctly classify an input $\mathbf{x}$ ($\arg\max f(\mathbf{x}) = t$). An adversarial example is a perturbed input that fools the classifier:

$$\arg\max f(\mathbf{x} + \delta x) \neq t$$

Introduction: Adversarial Examples

- Image classification (Goodfellow et al., 2014)



$\boldsymbol{x}$	$\text{sign}(\nabla_{\boldsymbol{x}} J(\boldsymbol{\theta}, \boldsymbol{x}, y))$	$\boldsymbol{x} + \epsilon \text{sign}(\nabla_{\boldsymbol{x}} J(\boldsymbol{\theta}, \boldsymbol{x}, y))$
"panda"	"nematode"	"gibbon"
57.7% confidence	8.2% confidence	99.3 % confidence

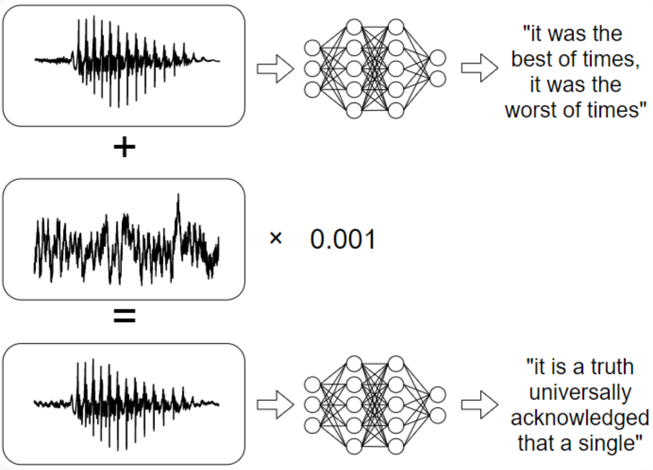
Introduction: Adversarial Examples

- Image classification (Goodfellow et al., 2014)
- Text classification (Ebrahimi et al., 2017)

South Africa's historic Soweto township marks its 100th birthday on Tuesday in a mood of optimism. 57% **World**
South Africa's historic Soweto township marks its 100th birthday on Tuesday in a moo**P** of optimism. 95% **Sci/Tech**
Chancellor Gordon Brown has sought to quell speculation over who should run the Labour Party and turned the attack on the opposition Conservatives. 75% **World**
Chancellor Gordon Brown has sought to quell speculation over who should run the Labour Party and turned the attack on the o**B**position Conservatives. 94% **Business**

Introduction: Adversarial Examples

- Image classification (Goodfellow et al., 2014)
- Text classification (Ebrahimi et al., 2017)
- Speech recognition (Qin et al., 2019)



Introduction: Defense to Adversarial Examples

- Simple heuristic such as training on adversarial examples can enhance the empirical robustness of the network.

Introduction: Defense to Adversarial Examples

- Simple heuristic such as training on adversarial examples can enhance the empirical robustness of the network.
- Lack of robustness guarantee runs the risk of being attacked in the future (`www.robust-ml.org/defenses`).

Defense ↑↓	Venue ↑↓	Dataset	Threat Model	Natural Accuracy	Claims	Analyses
Mitigating Adversarial Effects Through Randomization (Xie et al.) (code)	ICLR 2018	ImageNet	$\ell_\infty(\epsilon = 10/255)$	99.2% accuracy (on images originally classified correctly by underlying model)	86% accuracy (on images originally classified correctly)	• 0% accuracy [ACW18] (code)
Thermometer Encoding: One Hot Way To Resist Adversarial Examples (Buckman et al.) (code)	ICLR 2018	CIFAR-10	$\ell_\infty(\epsilon = 8/255)$	90% accuracy	79% accuracy	• 30% accuracy [ACW18] (code)
Countering Adversarial Images using Input Transformations (Guo et al.) (code)	ICLR 2018	ImageNet	$\ell_2(\epsilon = 0.06)$	75% accuracy	70% accuracy on ImageNet with average normalized ℓ_2 perturbation of 0.06	• 0% accuracy [ACW18] (code)
Stochastic Activation Pruning for Robust Adversarial Defense (Dhillon et al.) (code)	ICLR 2018	CIFAR-10	$\ell_\infty(\epsilon = 4/255)$	83% accuracy	51% accuracy	• 0% accuracy [ACW18] (code)

Introduction: Certifying Adversarial Robustness

- A function $f : \mathbb{R}^n \to \mathbb{R}^m$ is K-Lipschitz if and only if

$$||f(\mathbf{x}_1) - f(\mathbf{x}_2)||_2 \leq K||\mathbf{x}_1 - \mathbf{x}_2||_2, \forall \mathbf{x}_1, \mathbf{x}_2 \in \mathbb{R}^n.$$

  We denote $\mathrm{Lip}(f)$ as the smallest value K such that f is K-Lipschitz. This is called the **Lipschitz constant** of f.
- f is robust against a L_2 perturbation around an input example $\mathbf{x}$ within a radius of ϵ as long as

$$\sqrt{2}K\epsilon < y_t - \max_{i \neq t} y_i$$

  where $f(\mathbf{x}) = [y_1, y_2, \cdots, y_m]$ is the output of the classifier, y_t is the logit of the correct label.

Introduction: Training Provably Robust Classifier

- Provably robust classifier f^* can be obtained by solving an optimization problem over 1-Lipschitz functions using hinge loss.

$$f^* = \operatorname*{arg\,max}_{f:\mathrm{Lip}(f)\leq 1} \mathbb{E}_{\mathbf{x}\sim\mathcal{D}} \min\left(0, y_t - \max_{i\neq t} y_i - \sqrt{2}\epsilon\right)$$

Introduction: Training Provably Robust Classifier

- Provably robust classifier f^* can be obtained by solving an optimization problem over 1-Lipschitz functions using hinge loss.

$$f^* = \underset{f:\mathrm{Lip}(f)\leq 1}{\arg\max}\ \mathbb{E}_{\mathbf{x}\sim\mathcal{D}} \min\left(0, y_t - \max_{i\neq t} y_i - \sqrt{2}\epsilon\right)$$

- Need a way to parameterize functions with Lipschitz constraint!

Introduction: Estimating Wasserstein Distance

- The 1-Wasserstein distance between two distributions can also be casted as an optimization problem over 1-Lipschitz functions:

$$W_1(P_1, P_2) = \sup_{f:\text{Lip}(f)\leq 1} \left(\mathop{\mathbb{E}}_{\mathbf{x}\sim P_1}[f(\mathbf{x})] - \mathop{\mathbb{E}}_{\mathbf{x}\sim P_2}[f(\mathbf{x})] \right).$$

Introduction: Estimating Wasserstein Distance

- The 1-Wasserstein distance between two distributions can also be casted as an optimization problem over 1-Lipschitz functions:

$$W_1(P_1, P_2) = \sup_{f:\mathrm{Lip}(f)\leq 1} \left(\mathop{\mathbb{E}}_{\mathbf{x}\sim P_1}[f(\mathbf{x})] - \mathop{\mathbb{E}}_{\mathbf{x}\sim P_2}[f(\mathbf{x})] \right).$$

- Need a way to parameterize functions with Lipschitz constraint!

Background: Parameterizing Functions with Lipschitz Constraint

- **Idea:** Dynamically scale down f by $\mathrm{Lip}(f)$
- **Problem:** The exact Lipschitz constant of an arbitrarily defined network is extremely challenging to compute.
- **The Naive Solution:** Use the multiplication of Lipschitz constant for each layer as an upperbound of the global Lipschitz constant.

$$\mathrm{Lip}(f \circ g) \leq \mathrm{Lip}(f)\,\mathrm{Lip}(g)$$

Let's pause for a second

- What if $\mathrm{Lip}(f \circ g) = \mathrm{Lip}(f)\,\mathrm{Lip}(g)$ always hold? Can we design the architecture of the networks such that this always holds?

Let's pause for a second

- What if $\text{Lip}(f \circ g) = \text{Lip}(f)\,\text{Lip}(g)$ always hold? Can we design the architecture of the networks such that this always holds?

Let's pause for a second

- What if $\mathrm{Lip}(f \circ g) = \mathrm{Lip}(f)\,\mathrm{Lip}(g)$ always hold? Can we design the architecture of the networks such that this always holds?
- Gradient norm preservation!

Background: Gradient Norm Preservation

- A function $f : \mathbb{R}^n \mapsto \mathbb{R}^m$ is gradient norm preserving if

$$\left\| J^T \mathbf{g} \right\|_2 = ||\mathbf{g}||_2, \forall \mathbf{g} \in \mathcal{G}.$$

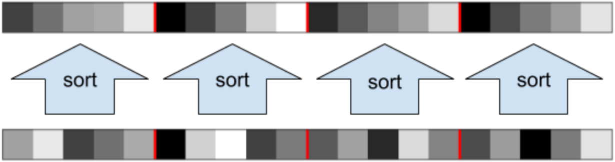
  where J is the input-output Jacobian of f, $\mathcal{G}$ is the set of all possible gradient vectors from the output, which we assume to be $\mathbb{R}^m$.
- A few important properties of GNP functions:
  1. The composition of two GNP functions is GNP.
  2. GNP functions have Lipschitz constant of exactly 1.

Background: GNP Activation

- Almost all commonly used activation functions are non-GNP!

Background: GNP Activation

- Almost all commonly used activation functions are non-GNP!
- **Solution:** GroupSort (Anil et al., 2018)



Background: GNP Linear is Orthogonal

- If $f : \mathbb{R}^n \mapsto \mathbb{R}^m$ is linear and GNP, then

$$\mathbf{y} = f(\mathbf{x}) = \mathbf{H}\mathbf{x}$$

  with $\mathbf{H}$ being an matrix with orthonormal rows.
- Without the loss of generality, we only consider the case where $\mathbf{H}$ is an orthogonal matrix.
- Parameterization of/training with orthogonal matrices has been well-studied in the literature.
  1. Björck-Bowie iterative algorithm (Björck and Bowie, 1971).
  2. Matrix exponential of skew symmetric matrix (Lezcano-Casado and Martínez-Rubio, 2019).
  3. Composition of householder matrices (Householder, 1958).
  4. ...

What about Convolutions?

- Convolutions are linear. Enforcing GNP on the convolution layer requires the matrix representation of it to be orthogonal.

What about Convolutions?

- Convolutions are linear. Enforcing GNP on the convolution layer requires the matrix representation of it to be orthogonal.
- The orthogonality of a convolutional kernel subject to **cyclic padding** is **invariant** to input size if we assume the input size (s) is large relative to the kernel size ($s \geq 2k - 1$).

What about Convolutions?

- Convolutions are linear. Enforcing GNP on the convolution layer requires the matrix representation of it to be orthogonal.
- The orthogonality of a convolutional kernel subject to **cyclic padding** is **invariant** to input size if we assume the input size (s) is large relative to the kernel size ($s \geq 2k - 1$).
- This allows us to study orthogonal convolutions **independent** of the input size.

What about Convolutions?

- Convolutions are linear. Enforcing GNP on the convolution layer requires the matrix representation of it to be orthogonal.
- The orthogonality of a convolutional kernel subject to **cyclic padding** is **invariant** to input size if we assume the input size (s) is large relative to the kernel size ($s \geq 2k-1$).
- This allows us to study orthogonal convolutions **independent** of the input size.
- **Definition**: A kernel size k convolution is orthogonal if

$$\mathcal{T}_{2k-1}^T \mathcal{T}_{2k-1} = \mathbf{I}$$

where $\mathcal{T}_s$ is the matrix representation of the convolution operation with an input size of s

Parameterization of Orthogonal Convolution

An orthogonal convolution kernel can be parameterized as a composition of orthogonal convolution kernels with kernel sizes of 1 and 2 (Kautsky and Turcajová, 1994):

$$\mathbf{A} = \mathbf{H} \square \begin{bmatrix} \mathbf{P}_1 & (\mathbf{I} - \mathbf{P}_1) \end{bmatrix} \square \cdots \square \begin{bmatrix} \mathbf{P}_{k-1} & (\mathbf{I} - \mathbf{P}_{k-1}) \end{bmatrix}$$

where $\mathbf{H} \in O(n)$ is an orthogonal matrix of $n \times n$, each of $\mathbf{P}_i$ is a symmetric projector ($\mathbf{P}_i = \mathbf{P}_i^T = \mathbf{P}_i^2$) and $\square$ represents the composition of convolutions.

A Surprise from the Topology Standpoint

$$\mathbf{A} = \mathbf{H} \square \begin{bmatrix} \mathbf{P}_1 & (\mathbf{I} - \mathbf{P}_1) \end{bmatrix} \square \cdots \square \begin{bmatrix} \mathbf{P}_{k-1} & (\mathbf{I} - \mathbf{P}_{k-1}) \end{bmatrix}$$

- **Theorem**: The space of 1-D orthogonal convolution kernel has $2(k-1)n + 2$ (disconnected) **connected components** (n is the channel size).
- If we use gradient-based optimization procedure in this space, it can **never** escape the connected component it is initialized in.

Background: Symmetric Projector

- Symmetric projectors ($\mathbf{P} = \mathbf{P}^2 = \mathbf{P}^T$) represent orthogonal projections onto linear subspaces.

Background: Symmetric Projector

- Symmetric projectors ($\mathbf{P} = \mathbf{P}^2 = \mathbf{P}^T$) represent orthogonal projections onto linear subspaces.
- $\mathbf{P}_1$ and $\mathbf{P}_2$ belong to the same connected component iff $\mathrm{rank}(\mathbf{P}_1) = \mathrm{rank}(\mathbf{P}_2)$.

Background: Symmetric Projector

- Symmetric projectors ($\mathbf{P} = \mathbf{P}^2 = \mathbf{P}^T$) represent orthogonal projections onto linear subspaces.
- $\mathbf{P}_1$ and $\mathbf{P}_2$ belong to the same connected component iff $\text{rank}(\mathbf{P}_1) = \text{rank}(\mathbf{P}_2)$.
- The $n \times n$ symmetric projector space has $n + 1$ connected components!

Example: Orthogonal Convolution Kernels with $k = 2$, $n = 2$

$$\mathbf{A} = \mathbf{H} \square \begin{bmatrix} \mathbf{P}_1 & (\mathbf{I} - \mathbf{P}_1) \end{bmatrix} \square \cdots \square \begin{bmatrix} \mathbf{P}_{k-1} & (\mathbf{I} - \mathbf{P}_{k-1}) \end{bmatrix}$$

Example: Orthogonal Convolution Kernels with $k = 2$, $n = 2$

$$\mathbf{A} = \mathbf{H}\square \begin{bmatrix} \mathbf{P} & (\mathbf{I} - \mathbf{P}) \end{bmatrix}$$

Example: Orthogonal Convolution Kernels with $k = 2$, $n = 2$

$$\mathbf{A} = \begin{bmatrix} \mathbf{P} & (\mathbf{I} - \mathbf{P}) \end{bmatrix}$$

Example: Orthogonal Convolution Kernels with $k = 2$, $n = 2$

- $\mathbf{P} = \begin{bmatrix} x & y \\ y & z \end{bmatrix}$.

Example: Orthogonal Convolution Kernels with $k = 2$, $n = 2$

- $\mathbf{P} = \begin{bmatrix} x & y \\ y & z \end{bmatrix}$.
  - $\text{rank}(\mathbf{P}) = 0 \to \mathbf{P} = 0$

Example: Orthogonal Convolution Kernels with $k = 2$, $n = 2$

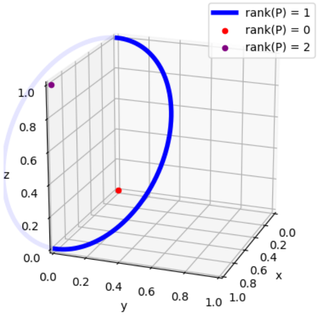
- $\mathbf{P} = \begin{bmatrix} x & y \\ y & z \end{bmatrix}$.
  - $\text{rank}(\mathbf{P}) = 0 \to \mathbf{P} = 0$
  - $\text{rank}(\mathbf{P}) = 1 \to x = 1 - z, y = \pm\sqrt{(1-z)z}, 0 \leq z \leq 1$

Example: Orthogonal Convolution Kernels with $k = 2$, $n = 2$

- $\mathbf{P} = \begin{bmatrix} x & y \\ y & z \end{bmatrix}$.
  - $\text{rank}(\mathbf{P}) = 0 \to \mathbf{P} = 0$
  - $\text{rank}(\mathbf{P}) = 1 \to x = 1 - z, y = \pm\sqrt{(1-z)z}, 0 \leq z \leq 1$
  - $\text{rank}(\mathbf{P}) = 2 \to \mathbf{P} = \mathbf{I}$

Example: Orthogonal Convolution Kernels with $k = 2$, $n = 2$

- $\mathbf{P} = \begin{bmatrix} x & y \\ y & z \end{bmatrix}$.
  - $\text{rank}(\mathbf{P}) = 0 \to \mathbf{P} = 0$
  - $\text{rank}(\mathbf{P}) = 1 \to x = 1 - z, y = \pm\sqrt{(1-z)z}, 0 \leq z \leq 1$
  - $\text{rank}(\mathbf{P}) = 2 \to \mathbf{P} = \mathbf{I}$



A Surprise from the Topology Standpoint

$$\mathbf{A} = \mathbf{H} \square \begin{bmatrix} \mathbf{P}_1 & (\mathbf{I} - \mathbf{P}_1) \end{bmatrix} \square \cdots \square \begin{bmatrix} \mathbf{P}_{k-1} & (\mathbf{I} - \mathbf{P}_{k-1}) \end{bmatrix}$$

- **Theorem**: The space of 1-D orthogonal convolution kernel has $2(k-1)n + 2$ (disconnected) **connected components** (n is the channel size).
- If we use gradient-based optimization procedure in this space, it can **never** escape the connected component it is initialized in.

Doubling the Channel Size Circumvents the Issue

- **Observation**: For all orthogonal convolutions $\mathbf{A}^{(n)}$ with channel sizes of n and arbitrary ranks for $\mathbf{P}_i^{(n)}$, there always exists an orthogonal convolution $\mathbf{A}^{(2n)}$ with a channel size of $2n$ and $\operatorname{rank}\left(\mathbf{P}_i^{(2n)}\right) = n$ such that

$$\left(\mathbf{A}^{(2n)}\mathbf{x}\right)_{1:n} = \left(\mathbf{A}^{(n)}\mathbf{x}_{1:n}\right), \forall \mathbf{x} \in \mathbb{R}^{2n}$$

Doubling the Channel Size Circumvents the Issue

- **Observation**: For all orthogonal convolutions $\mathbf{A}^{(n)}$ with channel sizes of n and arbitrary ranks for $\mathbf{P}_i^{(n)}$, there always exists an orthogonal convolution $\mathbf{A}^{(2n)}$ with a channel size of $2n$ and rank $\left(\mathbf{P}_i^{(2n)}\right) = n$ such that

$$\left(\mathbf{A}^{(2n)}\mathbf{x}\right)_{1:n} = \left(\mathbf{A}^{(n)}\mathbf{x}_{1:n}\right), \forall \mathbf{x} \in \mathbb{R}^{2n}$$

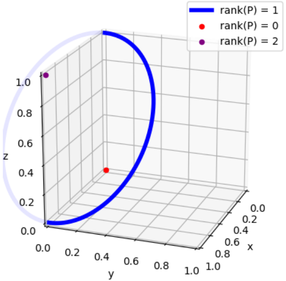
- This means that we can effectively double the dimensions in every hidden layer of an neural network to enable it express all connected components of the original network in just one connected component!

Doubling the Channel Size Circumvents the Issue

- $n = 1, \mathbf{P}^{(2)} = \begin{bmatrix} x & y \\ y & z \end{bmatrix}.$

Doubling the Channel Size Circumvents the Issue

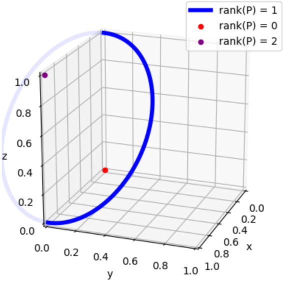
- $n = 1, \mathbf{P}^{(2)} = \begin{bmatrix} x & y \\ y & z \end{bmatrix}$.
- Assume that we parameterize in the connected component where $\text{rank}(\mathbf{P}^{(2)}) = 1$ (in blue on the left).



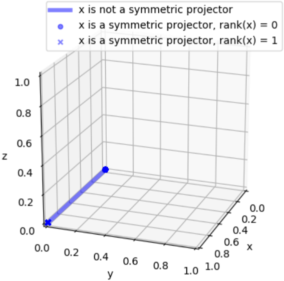
$\mathbf{P}^{(2)}$

Doubling the Channel Size Circumvents the Issue

- $n = 1, \mathbf{P}^{(2)} = \begin{bmatrix} x & y \\ y & z \end{bmatrix}$.
- Assume that we parameterize in the connected component where $\text{rank}(\mathbf{P}^{(2)}) = 1$ (in blue on the left).
- x can now reach both connected components of $\mathbf{P}^{(1)}$: 0 and 1 (as shown on the right).



$\mathbf{P}^{(2)}$



x

Parameterization Summary

- 1-D orthogonal convolution can be parameterized in the following connected component:

$$\mathbf{A} = \mathbf{H} \square \begin{bmatrix} \mathbf{P}_1 & (\mathbf{I} - \mathbf{P}_1) \end{bmatrix} \square \cdots \square \begin{bmatrix} \mathbf{P}_{k-1} & (\mathbf{I} - \mathbf{P}_{k-1}) \end{bmatrix}$$

  where $\mathbf{P}_i \in \mathbb{R}^{2n \times 2n}$ are symmetric projectors with $\mathrm{rank}(\mathbf{P}_i) = n$, $\mathbf{H} \in O(2n)$.
- Similar parameterization also exists in 2-D orthogonal convolutions where the same trick can also be used.

How does our GNP convolution compare to other convolutions?

- OSSN: One-sided spectral normalization (Gouk et al., 2018)
- RKO: Reshaped kernel method (Cisse et al., 2017)
- SVCM: Singular value clipping and masking (Sedghi et al., 2019)
- Evaluated on provable adversarial robustness on MNIST and CIFAR-10.

Dataset			OSSN	RKO	SVCM	Ours
MNIST ($\epsilon = 1.58$)	**Small**	Clean	96.86 ± 0.13	97.28 ± 0.08	97.24 ± 0.09	**97.54** $\pm$ 0.06
		Robust	42.95 ± 1.09	43.58 ± 0.44	28.94 ± 1.58	**45.84** $\pm$ 0.90
	Large	Clean	98.31 ± 0.03	98.44 ± 0.05	97.93 ± 0.05	**98.77** $\pm$ 0.05
		Robust	53.77 ± 1.02	55.18 ± 0.46	38.00 ± 1.82	**56.66** $\pm$ 0.23
CIFAR-10 ($\epsilon = 36/255$)	**Small**	Clean	62.18 ± 0.66	61.77 ± 0.63	62.39 ± 0.46	**64.53** $\pm$ 0.30
		Robust	48.03 ± 0.54	47.46 ± 0.53	47.59 ± 0.56	**50.01** $\pm$ 0.21
	Large	Clean	67.51 ± 0.47	70.01 ± 0.26	69.65 ± 0.38	**72.41** $\pm$ 0.22
		Robust	53.64 ± 0.49	55.76 ± 0.16	53.61 ± 0.51	**58.72** $\pm$ 0.23

How does our GNP convolution compare to other convolutions?

- Estimations of Wasserstein distance between the data and generator distributions of STL-10 GAN and CIFAR-10 GAN.
- Each estimate is a **strict lower bound**: higher value indicates a tighter estimate.

		OSSN	**RKO**	**Ours**
STL-10	**MaxMin**	7.39 ± 0.31	8.95 ± 0.12	$\mathbf{9.91} \pm 0.11$
	ReLU	7.06 ± 0.72	7.82 ± 0.21	$\mathbf{8.28} \pm 0.19$
CIFAR-10	**MaxMin**	3.29 ± 0.05	4.95 ± 0.08	$\mathbf{5.34} \pm 0.07$
	ReLU	3.07 ± 0.12	4.20 ± 0.06	$\mathbf{4.39} \pm 0.07$

SOTA Comparison

Dataset		Ours-Large	FC-3	KW-Large (Wong et al., 2018)
MNIST ($\epsilon = 1.58$)	Clean	**98.77** $\pm$ 0.05	98.71 $\pm$ 0.02	88.12
	Robust	**56.66** $\pm$ 0.23	54.46 $\pm$ 0.30	44.53
CIFAR-10 ($\epsilon = 36/255$)	Clean	**72.41** $\pm$ 0.22	62.60 $\pm$ 0.39	59.76
	Robust	**58.72** $\pm$ 0.23	49.97 $\pm$ 0.35	50.60

- Our GNP convolutional network outperforms the state-of-the-art on certifying deterministic provable adversarial robustness under L_2 metric.

Take-aways

1. **Gradient norm preserving (GNP)** ConvNets are effective for (1) provably robust image classification, and (2) Wasserstein distance estimation on image domain.
2. Although the orthogonal convolution space is **disconnected**, the issue can be largely mitigated by doubling the channel size and choosing the appropriate connected component to optimize in.
3. We design the **first** GNP ConvNet. It outperforms the state-of-the-art on **deterministic provable adversarial robustness** with L_2 metric on MNIST and CIFAR10.

References I

Anil, C., Lucas, J., and Grosse, R. (2018). Sorting out lipschitz function approximation. *arXiv preprint arXiv:1811.05381*.

Björck, Å. and Bowie, C. (1971). An iterative algorithm for computing the best estimate of an orthogonal matrix. *SIAM Journal on Numerical Analysis*, 8(2):358–364.

Cisse, M., Bojanowski, P., Grave, E., Dauphin, Y., and Usunier, N. (2017). Parseval networks: Improving robustness to adversarial examples. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 854–863. JMLR. org.

Ebrahimi, J., Rao, A., Lowd, D., and Dou, D. (2017). Hotflip: White-box adversarial examples for text classification. *arXiv preprint arXiv:1712.06751*.

Goodfellow, I. J., Shlens, J., and Szegedy, C. (2014). Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*.

References II

Gouk, H., Frank, E., Pfahringer, B., and Cree, M. (2018). Regularisation of neural networks by enforcing Lipschitz continuity. *arXiv preprint arXiv:1804.04368*.

Householder, A. S. (1958). Unitary triangularization of a nonsymmetric matrix. *Journal of the ACM (JACM)*, 5(4):339–342.

Kautsky, J. and Turcajová, R. (1994). A matrix approach to discrete wavelets. In *Wavelet Analysis and Its Applications*, volume 5, pages 117–135. Elsevier.

Lezcano-Casado, M. and Martínez-Rubio, D. (2019). Cheap orthogonal constraints in neural networks: A simple parametrization of the orthogonal and unitary group. *arXiv preprint arXiv:1901.08428*.

References III

Qin, Y., Carlini, N., Goodfellow, I., Cottrell, G., and Raffel, C. (2019). Imperceptible, robust, and targeted adversarial examples for automatic speech recognition. *arXiv preprint arXiv:1903.10346*.

Sedghi, H., Gupta, V., and Long, P. M. (2019). The singular values of convolutional layers. In *International Conference on Learning Representations*.

Wong, E., Schmidt, F., Metzen, J. H., and Kolter, J. Z. (2018). Scaling provable adversarial defenses. In *Advances in Neural Information Processing Systems*, pages 8400–8409.