Contributions on Deep Neural Networks with Toeplitz Matrices: Compression and Robustness

Alexandre Araujo

Université Paris-Dauphine, PSL Research University, CNRS, LAMSADE

Wavestone, Paris, France

Overview

Context & Background

Supervised Learning Algorithms

X					Y
$x_{1,1}$	$x_{1,2}$	$\cdots$	$x_{1,p-1}$	$x_{1,p}$	y_1
$x_{2,1}$	$x_{2,2}$	$\cdots$	$x_{2,p-1}$	$x_{2,p}$	y_2
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
$x_{n-1,1}$	$x_{n-1,2}$	$\cdots$	$x_{n-1,p-1}$	$x_{n-1,p}$	y_{n-1}
$x_{n,1}$	$x_{n,2}$	$\cdots$	$x_{n,p-1}$	$x_{n,p}$	y_n

Given a set of n **training examples** $\{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_n, y_n)\}$ where $\mathbf{x}_i$ is the feature vector of the i^{th} example, and y_i is the corresponding label.

Assumption: there is a function f matching any feature vector to its label.

The goal of a **learning algorithm** is to approximate f by a parameterized function f_θ. In order to measure how well the function fits, **a loss function** $\mathcal{L} : \mathcal{Y} \times \mathcal{Y} \rightarrow \mathbb{R}^+$ is defined. The standard method to learn the set of parameters θ is the **empirical risk minimization (ERM)**:

$$\hat{\theta}_{ERM} \triangleq \arg\min_{\theta} \frac{1}{n} \sum_{i=1}^{n} \mathcal{L}(f_\theta(\mathbf{x}_i), y_i)$$

Deep neural networks

Neural Neural can be analytically described as a composition of linear functions interlaced with non-linear functions:

Neural Network

A neural network of ℓ layers is defined as follows:

$$\mathcal{N}_\theta(\mathbf{x}) = \phi_{\mathbf{W}_\ell,\mathbf{b}_\ell} \circ \rho \circ \phi_{\mathbf{W}_{\ell-1},\mathbf{b}_{\ell-1}} \circ \rho \circ \cdots \circ \rho \circ \phi_{\mathbf{W}_1,\mathbf{b}_1}(\mathbf{x})$$

where for any i, $\phi_{\mathbf{W}_i,\mathbf{b}_i} \triangleq \mathbf{x} \mapsto \mathbf{W}_i\mathbf{x} + \mathbf{b}_i$, $\mathbf{x}_i \in \mathbb{R}^n$, $\mathbf{b}_i \in \mathbb{R}^m$, $\mathbf{W}_i \in \mathbb{R}^{m\times n}$, ρ some non linear functions and θ corresponds to the set of all parameters.

Evaluation of Neural Networks

- Classical evaluation with accuracy
- Robust evaluation against adversarial attacks

An **adversarial attack** refers to a small, imperceptible change of an input maliciously designed to fool the result of a machine learning algorithm.

Since the seminal work of Szegedy et al. (2014), numerous attack methods have been designed:

- **PGD** Madry et al. (2018)
- **C&W** Carlini and Wagner (2017)

Fully-Connected Neural Networks (neural networks defined with dense matrices) can have a very large number of parameters.

$\Rightarrow$ With MNIST dataset (LeCun et al. (1998)), a two-layers Fully-Connected neural network will have more than $\mathbf{6 \times 10^5}$ **parameters**.

Limits of Large Neural Networks

- They are hard to train
- They are subject to overfitting: they don't generalize well
- They are computationally expensive

$\Rightarrow$ To overcome these limitations, researchers have devised neural networks with **structured linear operations** in order to reduce the number of parameters needed.

A $n \times n$ structured matrix can be represented with less than n^2 parameters. In addition to offering a more compact representation, the structure of certain matrices can be leveraged to obtain better algorithms for matrix-vector product.

$$
\underbrace{\begin{pmatrix} a & & & \\ & b & & \\ & & c & \\ & & & d \end{pmatrix}}_{\text{diagonal}}
\underbrace{\begin{pmatrix} a & b & c & d \\ e & a & b & c \\ f & e & a & b \\ d & f & e & a \end{pmatrix}}_{\text{Toeplitz}}
\underbrace{\begin{pmatrix} ae & af & ag & ah \\ be & bf & bg & bh \\ ce & cf & cg & ch \\ de & df & dg & dh \end{pmatrix}}_{\text{Low Rank}}
\underbrace{\begin{pmatrix} a & a^2 & a^3 & a^4 \\ b & b^2 & b^3 & b^4 \\ c & c^2 & c^3 & c^4 \\ d & d^2 & d^3 & d^4 \end{pmatrix}}_{\text{Vandermonde}}
$$

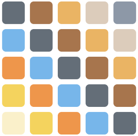
Figure 1: Examples of structured matrices.

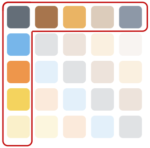
$\Rightarrow$ We focus on structured matrices from the **Toeplitz family**.

Focus on structured matrices from the Toeplitz family

More specifically: A Toeplitz matrix is a matrix with constant diagonal:

$$\begin{pmatrix} a & b & c & d \\ e & a & b & c \\ f & e & a & b \\ d & f & e & a \end{pmatrix}$$





$\Rightarrow$ A $n \times n$ Toeplitz matrix has $2n - 1$ unique values.

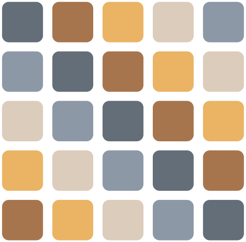
Focus on structured matrices from the Toeplitz family

For our contributions, we study:

- Circulant matrices
- Doubly-block Toeplitz matrices

Circulant Matrices

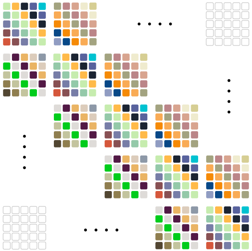
A $n \times n$ circulant matrix is a matrix where each row is a cyclic right shift of the previous one.



A circulant matrix

A block Toeplitz matrix is a matrix which contains **blocks that are repeated down the diagonals** of the matrix.

A **doubly-block Toeplitz matrix** is a block Toeplitz matrix where all blocks are also Toeplitz.



Doubly-block Toeplitz matrices

$\Rightarrow$ Doubly-block matrices are equivalent to the 2d convolution.

Our Contributions

We devised a compact architecture with Diagonal and Circulant Matrices

We define the expressive power of diagonal circulant neural networks.

We use diagonal circulant neural networks for compact large scale video classification.

Improving Robustness of Convolution Neural Networks with Doubly-Block Toeplitz matrices

We devise an upper bound on the singular values of convolutional layers.

We propose an efficient algorithm to compute this upper bound.

We propose a new regularization scheme to improve the robustness of Neural Networks.

Diagonal Circulant Neural Networks

Table of Contents

Circulant matrices for Deep Learning

A $n \times n$ circulant matrix $\mathbf{C}$ is a matrix where each row is a cyclic right shift of the previous one as illustrated below.

$$\mathbf{C} = \mathrm{circ}(\mathbf{c}) = \begin{pmatrix} c_0 & c_1 & c_2 & \cdots & c_{n-1} \\ c_{n-1} & c_0 & c_2 & & c_{n-2} \\ c_{n-2} & c_{n-1} & c_0 & & c_{n-3} \\ \vdots & & & \ddots & \vdots \\ c_1 & c_2 & c_3 & & c_0 \end{pmatrix}$$

Advantages:

- A $n \times n$ circulant matrix can be **compactly represented in memory** using only n real values.
- Multiplying a circulant matrix $\mathbf{C}$ by a vector **$\mathbf{x}$ can be done efficiently in the Fourier domain**

Limits:

- The product of circulant matrices is not expressive: circulant matrices are closed under product

Expressivity of the product of diagonal and circulant matrices

Theorem 1 (Reformulation from Huhtanen and Perämäki (2015))

For every matrix $\mathbf{A} \in \mathbb{C}^{n \times n}$*, for any* $\epsilon > 0$*, there exists a sequence of matrices* $\mathbf{B}_1 \cdots \mathbf{B}_{2n-1}$ *where* $\mathbf{B}_i$ *is a circulant matrix if* i *is odd, and a diagonal matrix otherwise, such that* $\|\mathbf{B}_1 \mathbf{B}_2 \cdots \mathbf{B}_{2n-1} - \mathbf{A}\| < \epsilon$*.*

- The decomposition needs more values that n^2
- The theorem does not provide any insights regarding the expressive power of m diagonal-circulant factors when m is much lower than $2n - 1$

Relation between diagonal circulant matrices and low rank matrices

Theorem 2 (Rank-based circulant decomposition)

Let $\mathbf{A} \in \mathbb{C}^{n \times n}$ be a matrix of rank at most k. Assume that n can be divided by k. For any $\epsilon > 0$, there exists a sequence of $4k+1$ matrices $\mathbf{B}_1, \ldots, \mathbf{B}_{4k+1}$, where $\mathbf{B}_i$ is a circulant matrix if i is odd, and a diagonal matrix otherwise, such that $\|\mathbf{B}_1\mathbf{B}_2 \ldots \mathbf{B}_{4k+1} - \mathbf{A}\| < \epsilon$

$\Rightarrow$ If the number of diagonal-circulant factors is set to a value K, we can represent all linear transform whose rank is $\frac{K-1}{4}$.

We replace the weight matrices of Fully-Connected layers by a product of Diagonal and Circulant matrices :

Fully-Connected layer

$$\mathbf{x} \mapsto \mathbf{W}\mathbf{x} + \mathbf{b}$$

where $\mathbf{x} \in \mathbb{R}^n$, $\mathbf{b} \in \mathbb{R}^n$, $\mathbf{W} \in \mathbb{R}^{n \times n}$.

Diagonal-Circulant layer

$$\mathbf{x} \mapsto \left[\prod_{i=0}^{k} \mathbf{D}_i \mathbf{C}_i \right] \mathbf{x} + \mathbf{b}$$

where $\mathbf{x} \in \mathbb{R}^n$, $\mathbf{b} \in \mathbb{R}^n$, $\mathbf{D}_i \in \mathbb{R}^{n \times n}$ is a diagonal matrix, $\mathbf{C}_i \in \mathbb{R}^{n \times n}$ is a circulant matrix, k is a user defined parameter.

Theorem 3 (Rank-based expressive power of DCNNs)

Let $\mathcal{N}$ be a deep ReLU network of width n, depth L and a total rank k[1]. Let $\mathcal{X} \subset \mathbb{C}^n$ be a bounded set. Then, for any $\epsilon > 0$, there exists a DCNN with ReLU activation $\mathcal{N}'$ of width n such that $\|\mathcal{N}(\mathbf{x}) - \mathcal{N}'(\mathbf{x})\| < \epsilon$ for all $\mathbf{x} \in \mathcal{X}$ and the depth of $\mathcal{N}'$ is bounded by $9k$.

By combining Theorem 3 and the universal approximation theorem of Neural Network, we have:

Corollary 4

Bounded width DCNNs are ***universal approximators***

[1] The sum of ranks of the weight matrices

Table of Contents

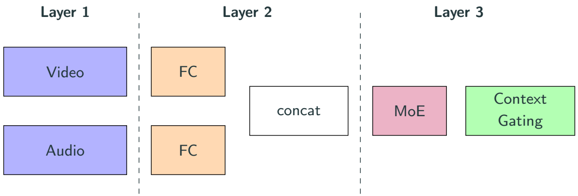
Experimental Setup

Dataset: ***YouTube-8M***

8 millions embedded audio & video frames

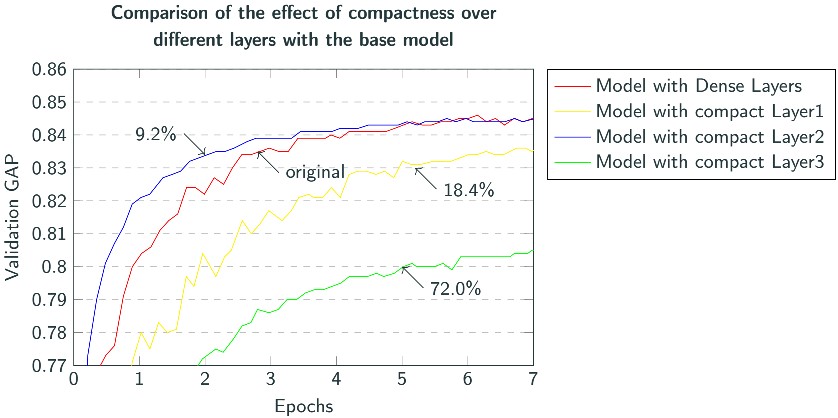
3200 classes

State-of-the-art architecture for video classification (Miech et al. (2017)).



⇒ This architecture has 5.7 millions parameters.

Effect of Diagonal-Circulant layers



⇒ 9.2% compression rate without loss in accuracy

⇒ 72% compression rate with a loss of 4 points in accuracy

Improving Robustness of Convolution Neural Networks with Doubly-Block Toeplitz matrices

Table of Contents

Defending against Adversarial Attacks

An **Adversarial Attack** aims to find the worst perturbation τ with $\|\tau\|_p \leq \epsilon$ in such a way that the Neural Network misclassifies. Therefore, an attacker aims to find the solution to the following problem:

$$\tau_\theta^{\mathrm{adv}}(\mathbf{x}) \triangleq \max_{\|\tau\|_p \leq \epsilon} \mathcal{L}(\mathcal{N}_\theta(\mathbf{x}+\tau), y)$$

Goodfellow et al. (2015) have proposed **Adversarial Training** which follows **ERM** training over adversarially-perturbed samples

$$\arg\min_\theta \frac{1}{n} \sum_{i=1}^{n} \mathcal{L}(\mathcal{N}_\theta(\mathbf{x}_i + \tau_\theta^{\mathrm{adv}}(\mathbf{x}_i)), y_i)$$

Farnia et al. (2019) have shown that the adversarial generalization error depends on the Lipschitz constant of the network.

$\Rightarrow$ Reducing the Lipschitz constant of the Neural Network improves the robustness against adversarial attacks.

Lipschitz constant of a Neural Network

The **Lipschitz constant** w.r.t ℓ_2 of a function is the smallest constant K such that:

$$\|f(\mathbf{x}) - f(\mathbf{y})\|_2 \leq K\|\mathbf{x} - \mathbf{y}\|_2$$

Let us denote $\mathrm{Lip}(f) = K$ or that f is K-Lipschitz.

The Lipschitz constant of the composition of multiple functions can be upper bounded by the product of the Lipschitz constant of each function.

Remarks:

- For a linear function, the Lipschitz constant also corresponds to the maximal singular value.
- Usual non-linear functions used in Neural Networks (*e.g.* ReLU) are 1-Lipschitz

Therefore, we can upper bound the Lipschitz constant of a Neural Network $\mathcal{N}$ as follows:

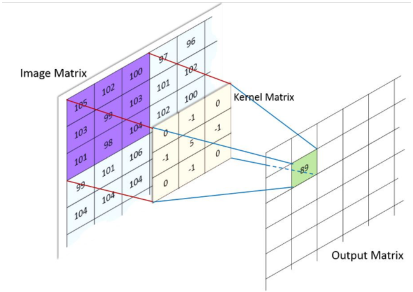
$$\mathrm{Lip}(\mathcal{N}_\theta) \leq \prod_{i=0}^{\ell} \mathrm{Lip}(\phi_{\mathbf{W}_i,\mathbf{b}_i}) = \prod_{i=0}^{\ell} \sigma_1(\mathbf{W}_i)$$

$\Rightarrow$ This bound is hard to compute

Table of Contents

Convolution as matrix-multiplication

A discrete convolution between a signal $\mathbf{x}$ and a kernel $\mathbf{k}$ can be expressed as a product between the vectorization of $\mathbf{x}$ and a doubly-block Toeplitz matrix $\mathbf{M}$, whose coefficients have been chosen to match the convolution $\mathbf{x} * \mathbf{k}$.

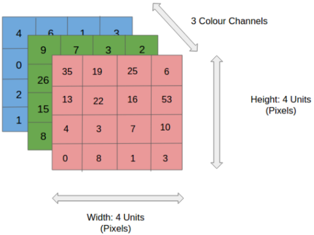


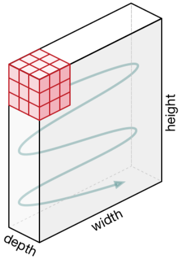
Convolution between a 2-dimensional image and a 2 dimensional kernel

The convolution is equivalent to a matrix-vector product between a **doubly-block Toeplitz matrix** and the vectorize image.

Convolution as matrix-multiplication

In practice, the image has multiple **channels** (*e.g.* RGB). We refer to the number of input channels *cin* and the number of output channels *cout*.





A multi-channel convolution is equivalent to a matrix-vector product where the matrix is a **block matrix** where each block is doubly-block Toeplitz matrix.

Generating functions of Toeplitz and block Toeplitz matrices

A $n \times n$ Toeplitz matrix $\mathbf{A}$ is fully determined by a two-sided sequence of scalars: $\{a_h\}_{h \in N}$, whereas a $nm \times nm$ block Toeplitz matrix $\mathbf{B}$ is fully determined by a two-sided sequence of blocks $\{\mathbf{B}_h\}_{h \in N}$, where $N = \{-n+1, \dots, n-1\}$ and where each block $\mathbf{B}_h$ is a $m \times m$ matrix.

$$\mathbf{A} = \begin{pmatrix} a_0 & a_1 & \cdots & a_{n-1} \\ a_{-1} & a_0 & \ddots & \vdots \\ \vdots & \ddots & a_0 & a_1 \\ a_{-n+1} & \cdots & a_{-1} & a_0 \end{pmatrix} \qquad \mathbf{B} = \begin{pmatrix} \mathbf{B}_0 & \mathbf{B}_1 & \cdots & \mathbf{B}_{n-1} \\ \mathbf{B}_{-1} & \mathbf{B}_0 & \ddots & \vdots \\ \vdots & \ddots & \mathbf{B}_0 & \mathbf{B}_1 \\ \mathbf{B}_{-n+1} & \cdots & \mathbf{B}_{-1} & \mathbf{B}_0 \end{pmatrix}.$$

We also write:

$$\mathbf{A} = (a_{j-i})_{i,j \in \{0,\dots,n-1\}} \qquad \mathbf{B} = (\mathbf{B}_{j-i})_{i,j \in \{0,\dots,n-1\}}$$

Generating functions of Toeplitz and block Toeplitz matrices

Let us define a complex-valued function and a matrix-valued function which are the inverse Fourier Transform of the sequences $\{a_h\}_{h\in N}$ and $\{\mathbf{B}\}_{h\in N}$ as follows:

$$f(\omega) = \sum_{h\in N} a_h e^{\mathbf{i}h\omega} \qquad F(\omega) = \sum_{h\in N} \mathbf{B}_h e^{\mathbf{i}h\omega}$$

One can recover these two sequences using the standard Fourier transform:

$$a_h = \frac{1}{2\pi}\int_0^{2\pi} e^{-\mathbf{i}h\omega} f(\omega) d\omega \qquad \mathbf{B}_h = \frac{1}{2\pi}\int_0^{2\pi} e^{-\mathbf{i}h\omega} F(\omega) d\omega.$$

From there, we can define an operator $\mathbf{T}$ mapping integrable 2π-periodic functions to Toeplitz or block Toeplitz matrices:

$$\mathbf{T}(g) \triangleq \left(\frac{1}{2\pi}\int_0^{2\pi} e^{-\mathbf{i}(i-j)\omega} g(\omega)\, d\omega\right)_{i,j\in\{0,\ldots,n-1\}}.$$

Because doubly-block Toeplitz matrices are **block Toeplitz** where each block is a **Toeplitz matrix**, we can extend the reasoning with a 2 dimensional function $f : \mathbb{R}^2 \rightarrow \mathbb{C}$.

The block Toeplitz can be written as follows:

$$\mathbf{D}(f) = \left(\mathbf{D}_{i,j}(f)\right)_{i,j \in \{0,\ldots,n-1\}}$$

and each block $\mathbf{D}_{i,j}$ is defined as:

$$\mathbf{D}_{i,j}(f) = \left(\frac{1}{4\pi^2}\int_{[0,2\pi]^2} e^{-\mathbf{i}((i-j)\omega_1+(k-l)\omega_2)} f(\omega_1,\omega_2)\, d(\omega_1,\omega_2)\right)_{k,l \in \{0,\ldots,m-1\}}.$$

The operator $\mathbf{D}$ which maps a function $f : \mathbb{R}^2 \rightarrow \mathbb{C}$ to a doubly-block Toeplitz matrix of size $nm \times nm$.

Theorem 5 (Bound on the maximal singular value of a Doubly-Block Toeplitz Matrix)

Let $\mathbf{D}(f) \in \mathbb{R}^{nm \times nm}$ *be a doubly-block Toeplitz matrix generated by the function* f*, then:*

$$\sigma_1\left(\mathbf{D}(f)\right) \leq \sup_{\omega_1, \omega_2 \in [0, 2\pi]^2} \left| f(\omega_1, \omega_2) \right|$$

where the function $f : \mathbb{R}^2 \rightarrow \mathbb{C}$*, is a multivariate trigonometric polynomial of the form:*

$$f(\omega_1, \omega_2) \triangleq \sum_{h_1 \in N} \sum_{h_2 \in M} d_{h_1, h_2} e^{\mathbf{i}(h_1 \omega_1 + h_2 \omega_2)},$$

where d_{h_1, h_2} *is the* ${h_2}^{\mathrm{th}}$ *scalar of the* ${h_1}^{\mathrm{th}}$ *block of the doubly-Toeplitz matrix* $\mathbf{D}(f)$*, and where* $M = \{-m+1, \ldots, m-1\}$*.*

Bound Singular Values of Convolution

Theorem 6 (Bound on the maximal singular value on the convolution operation)

Let us define doubly-block Toeplitz matrices $\mathbf{D}(f_{11}), \dots, \mathbf{D}(f_{cin \times cout})$ *where* $f_{ij} : \mathbb{R}^2 \to \mathbb{C}$ *is a generating function. Construct a matrix* $\mathbf{M}$ *with* $cin \times n^2$ *rows and* $cout \times n^2$ *columns such as*

$$\mathbf{M} \triangleq \begin{pmatrix} \mathbf{D}(f_{11}) & \cdots & \mathbf{D}(f_{1,cout}) \\ \vdots & & \vdots \\ \mathbf{D}(f_{cin,1}) & \cdots & \mathbf{D}(f_{cin,cout}) \end{pmatrix}.$$

Then, with f_{ij} *a multivariate polynomial, we have:*

$$\sigma_1(\mathbf{M}) \leq \sqrt{\sum_{i=1}^{cout} \sup_{\omega_1, \omega_2 \in [0,2\pi]^2} \sum_{j=1}^{cin} |f_{ij}(\omega_1, \omega_2)|^2}.$$

In the following, for a given convolution layer parametrized by $\mathbf{W}$, we will call this bound LipBound($\mathbf{W}$)

Table of Contents

To improve the robustness of Neural Networks, we propose the following objective function:

$$\arg\min_{\theta} \frac{1}{n} \sum_{i=1}^{n} \mathcal{L}(\mathcal{N}_\theta(\mathbf{x}_i + \tau_\theta^{\mathrm{adv}}(\mathbf{x}_i)), y_i) + \lambda \sum_{j=0}^{\ell} \log(\mathrm{LipBound}(\mathbf{W}_j))$$

where λ is used-defined parameter which controls the regularization.

Computing the bound

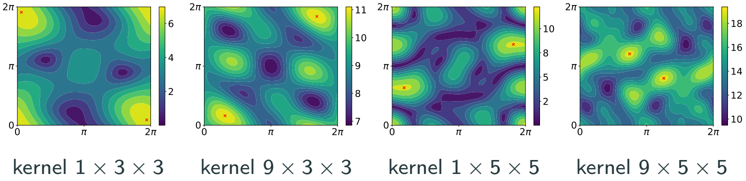


Figure 4: Contour plot of multivariate trigonometric polynomials.

Computing LipBound implies to compute the maximum modulus of a 2-dimensional trigonometric polynomial on $[0, 2\pi]^2$.

- This problem has been known to be NP-hard (Pfister and Bresler (2018))
- However, trigonometric polynomials defined by usual convolutional kernels have a low degree (between 1 and 3)
- A simple grid search algorithm is efficient and can be implemented on GPU

Empirical Results – CIFAR-10 Dataset

Dataset: CIFAR-10

50K images

10 classes

	Accuracy	PGD-ℓ_∞ 0.03	C&W-ℓ_2 0.6	C&W-ℓ_2 0.8
Baseline	**0.953**	0.000	0.002	0.000
AT	0.864	0.426	0.477	0.334
AT+LipReg	0.808	**0.457**	**0.547**	**0.438**
Diff	-0.056	$+0.031$	$+0.07$	$+0.104$

Table 1: This table shows the Accuracy under ℓ_2 and ℓ_∞ attacks of CIFAR10 dataset. We use λ equals to 0.008.

Conclusion & Future Work

Conclusion & Future work

Diagonal Circulant Neural Network

- We proposed the use of a matrix decomposition into diagonal and circulant matrices in Deep Learning settings
- We applied have applied this structure for large scale video classification
- We showed that this method allows a good compression rate without an important impact on the accuracy.

Lipschitz Bound of Convolutional Layers

- We introduced a new bound on the Lipschitz constant of convolutional layers that is both accurate and efficient to compute;
- We used this bound to regularize the Lipschitz constant of neural networks;
- We showed that it increases the robustness of the trained networks to adversarial attacks;

Recent works (Virmaux and Scaman (2018); Fazlyab et al. (2019); Latorre et al. (2020)) have tried to devise algorithms to compute the Lipschitz constant of a Neural Network but these techniques are difficult to implement for neural networks with more than one or two layers.

Question

Can we leverage the block-Toeplitz structure of convolution to devise fast and accurate algorithm to compute the Lipschitz constant of Neural Networks ?

Thank You

Appendix

Efficient Matrix-vector product with Circulant Matrices

A circulant matrix $\mathbf{C} \in \mathbb{R}^{n \times n}$ such as $\mathbf{C} = \mathrm{circ}(\mathbf{c})$, with $\mathbf{c} \in \mathbb{R}^n$ can be diagonalized by the Discrete Fourier Transform:

$$\mathbf{C} = \mathbf{W}^{-1} \Lambda \mathbf{W}$$

where $\mathbf{W} = \frac{1}{\sqrt{n}} \left(\omega^{jk} \right)_{j,k=0,\dots,n-1}$ with ω being the n^{th} root of unity, Λ is a diagonal matrix with the eigenvalues of the matrix $\mathbf{C}$ and the eigenvalues of the matrix $\mathbf{C}$ can correspond to $\mathbf{Wc}$.

Therefore, thanks to the convolution theorem, matrix-vector multiplication can be done efficiently with the **Fast Fourier Transform** as follows:

$$\mathbf{Cx} = \mathrm{IDFT}(\mathrm{DFT}(\mathbf{c}) * \mathrm{DFT}(\mathbf{x}))$$

where the multiplication is performed elements-wise.

References

Carlini, N. and Wagner, D. (2017). Towards evaluating the robustness of neural networks. In 2017 IEEE Symposium on Security and Privacy (SP), pages 39–57. IEEE.

Farnia, F., Zhang, J., and Tse, D. (2019). Generalizable adversarial training via spectral normalization. In International Conference on Learning Representations.

Fazlyab, M., Robey, A., Hassani, H., Morari, M., and Pappas, G. (2019). Efficient and accurate estimation of lipschitz constants for deep neural networks. In Wallach, H., Larochelle, H., Beygelzimer, A., d Alché-Buc, F., Fox, E., and Garnett, R., editors, Advances in Neural Information Processing Systems 32, pages 11423–11434. Curran Associates, Inc.

Goodfellow, I., Shlens, J., and Szegedy, C. (2015). Explaining and harnessing adversarial examples. In International Conference on Learning Representations.

References

Huhtanen, M. and Perämäki, A. (2015). Factoring matrices into the product of circulant and diagonal matrices. Journal of Fourier Analysis and Applications, 21(5):1018–1033.

Latorre, F., Rolland, P., and Cevher, V. (2020). Lipschitz constant estimation for neural networks via sparse polynomial optimization. In International Conference on Learning Representations.

LeCun, Y., Bottou, L., Bengio, Y., and Haffner, P. (1998). Gradient-based learning applied to document recognition. Proceedings of the IEEE, 86(11):2278–2324.

Madry, A., Makelov, A., Schmidt, L., Tsipras, D., and Vladu, A. (2018). Towards deep learning models resistant to adversarial attacks. In International Conference on Learning Representations.

Miech, A., Laptev, I., and Sivic, J. (2017). Learnable pooling with context gating for video classification. CoRR, abs/1706.06905.

References

Pfister, L. and Bresler, Y. (2018). Bounding multivariate trigonometric polynomials with applications to filter bank design. arXiv preprint arXiv:1802.09588.

Szegedy, C., Zaremba, W., Sutskever, I., Bruna, J., Erhan, D., Goodfellow, I., and Fergus, R. (2014). Intriguing properties of neural networks. In International Conference on Learning Representations.

Virmaux, A. and Scaman, K. (2018). Lipschitz regularity of deep neural networks: analysis and efficient estimation. In Bengio, S., Wallach, H., Larochelle, H., Grauman, K., Cesa-Bianchi, N., and Garnett, R., editors, Advances in Neural Information Processing Systems 31, pages 3835–3844. Curran Associates, Inc.