13th ICCCNT 2022

IEEE 13th INTERNATIONAL CONFERENCE ON COMPUTING, COMMUNICATION AND NETWORKING TECHNOLOGIES

Lipschitz Bound Analysis of Neural Networks

Paper ID: 58

Sarosij Bose[*]

[*]Department of Computer Science and Engineering, University of Calcutta.

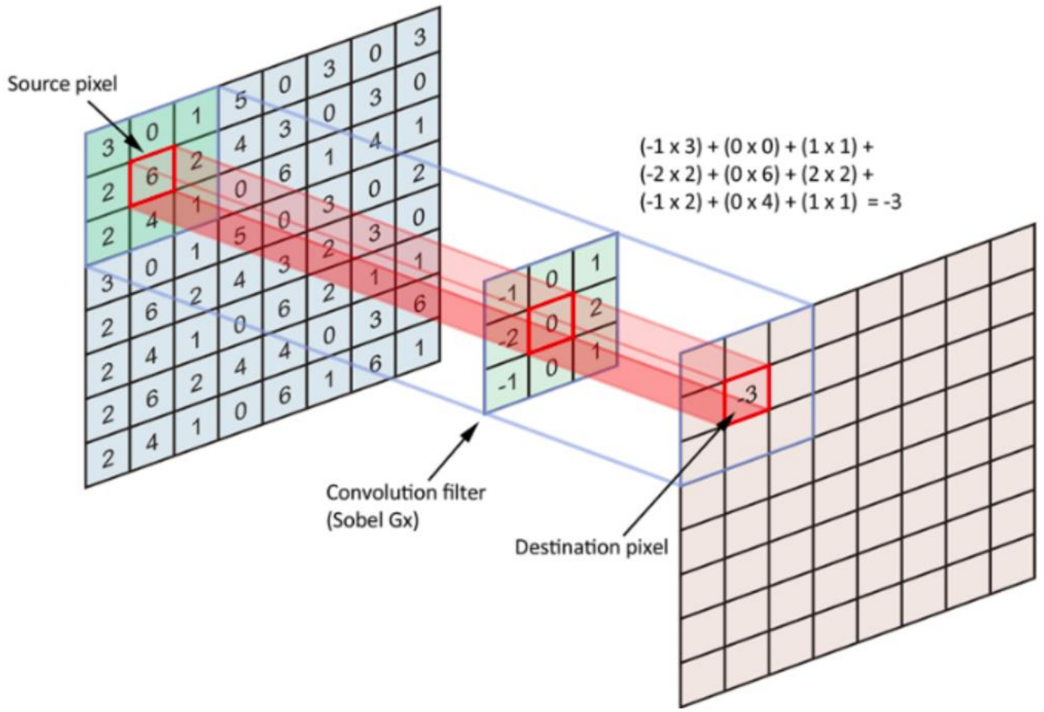
Introduction

Convolutions: An Overview

Convolutions in deep learning have been a goto choice for people to learn latent features of images.

Initial layers learn low level features from images like edges and corners and the deeper we go in the CNN the richer the feature maps become.

Convolution can be viewed as a kernel sliding over a matrix.



Source pixel
(-1 x 3) + (0 x 0) + (1 x 1) +
(-2 x 2) + (0 x 6) + (2 x 2) +
(-1 x 2) + (0 x 4) + (1 x 1) = -3
Convolution filter (Sobel Gx)
Destination pixel

[2]

Introduction

Toeplitz Matrices

How is it linked to convolution?
The convolution operator can be rerpresented as a stacked sequence of doubly block Toeplitz matrices. [1]

Why is it needed?
Toeplitz matrices help us convert the convolution operation into a matrix vector product format. Hence it can be thought of as a fully connected network.

$$\begin{bmatrix} t_0 & t_{-1} & t_{-2} & \cdots & t_{-(n-1)} \\ t_1 & t_0 & t_{-1} & & \\ t_2 & t_1 & t_0 & & \vdots \\ \vdots & & & \ddots & \\ t_{n-1} & & & \cdots & t_0 \end{bmatrix}$$

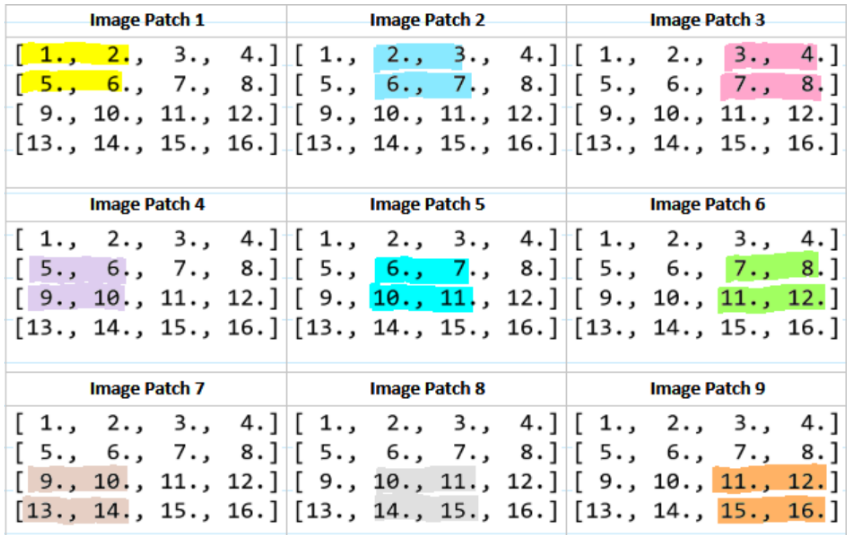
The general structure of a Toeplitz matrix

$$\begin{bmatrix} s0 \\ s1 \\ s2 \\ s3 \\ s4 \\ \ldots \end{bmatrix} = \begin{bmatrix} g0 & 0 & 0 & 0 \\ g1 & g0 & 0 & 0 \\ g2 & g1 & g0 & 0 \\ g3 & g2 & g1 & g0 \\ g4 & g3 & g2 & g1 \\ & \ldots & & \end{bmatrix} \times \begin{bmatrix} f0 \\ f1 \\ f2 \\ f3 \end{bmatrix}$$

Using the Toeplitz matrix of the kernel for matrix-vector implementation of convolution

Introduction

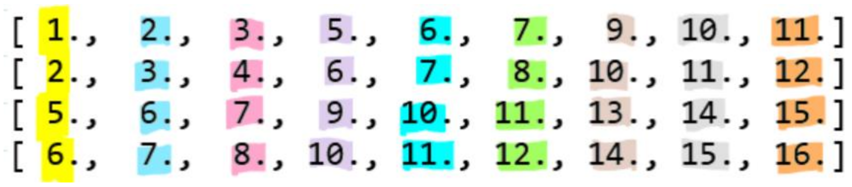
Unrolling Convolution[3]



```
Image Patch 1              Image Patch 2              Image Patch 3
[ 1.,  2.,  3.,  4.]       [ 1.,  2.,  3.,  4.]       [ 1.,  2.,  3.,  4.]
[ 5.,  6.,  7.,  8.]       [ 5.,  6.,  7.,  8.]       [ 5.,  6.,  7.,  8.]
[ 9., 10., 11., 12.]       [ 9., 10., 11., 12.]       [ 9., 10., 11., 12.]
[13., 14., 15., 16.]       [13., 14., 15., 16.]       [13., 14., 15., 16.]

Image Patch 4              Image Patch 5              Image Patch 6
[ 1.,  2.,  3.,  4.]       [ 1.,  2.,  3.,  4.]       [ 1.,  2.,  3.,  4.]
[ 5.,  6.,  7.,  8.]       [ 5.,  6.,  7.,  8.]       [ 5.,  6.,  7.,  8.]
[ 9., 10., 11., 12.]       [ 9., 10., 11., 12.]       [ 9., 10., 11., 12.]
[13., 14., 15., 16.]       [13., 14., 15., 16.]       [13., 14., 15., 16.]

Image Patch 7              Image Patch 8              Image Patch 9
[ 1.,  2.,  3.,  4.]       [ 1.,  2.,  3.,  4.]       [ 1.,  2.,  3.,  4.]
[ 5.,  6.,  7.,  8.]       [ 5.,  6.,  7.,  8.]       [ 5.,  6.,  7.,  8.]
[ 9., 10., 11., 12.]       [ 9., 10., 11., 12.]       [ 9., 10., 11., 12.]
[13., 14., 15., 16.]       [13., 14., 15., 16.]       [13., 14., 15., 16.]
```



```
[ 1.,  2.,  3.,  5.,  6.,  7.,  9., 10., 11.]
[ 2.,  3.,  4.,  6.,  7.,  8., 10., 11., 12.]
[ 5.,  6.,  7.,  9., 10., 11., 13., 14., 15.]
[ 6.,  7.,  8., 10., 11., 12., 14., 15., 16.]
```

Flattening weight vector

$$\begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix}$$

```
[1., 2., 3., 4.]
```

Unrolled Input * Flattened Vector

```
[ 44.,  54.,  64.,  84.,  94., 104., 124., 134., 144.]
```

Rolling back the output:

```
[ 44.,  54.,  64.]
[ 84.,  94., 104.]
[124., 134., 144.]
```

Methodology

Trivial Bound

The trivial lipschitz bound for a particular feed forward neural network refers to the product of the spectral norms of the weight matrix of each individual layer. In the case of CNNs, the ‘unrolled filter’ can be used as a representative matrix whose spectral norm can give us the trivial bound.[2]

Tight Bound

The tight bound for any particular neural network can be defined as the smallest value of the Lipschitz constant L such that it satisfies the following equation:-

$$||f(x) - f(y)|| < L||x - y|| \tag{1}$$

Empirical Bound

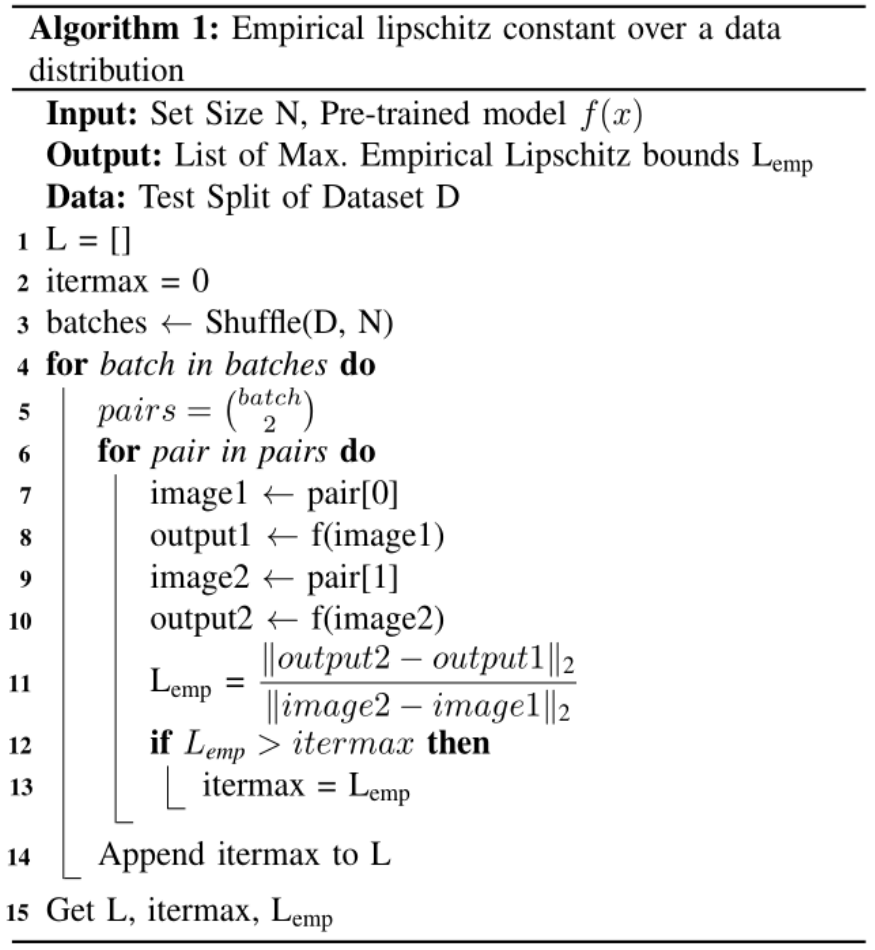
The empirical Lipschitz bound or the true Lipschitz bound is the obtained value from a particular data distribution such that it satisfies equation 1. This reflects the true robustness of a particular neural network.

ICCCNT 2022

Methodology

Proposed Algorithm

Our proposed algorithm is shown alongside. This can be used to find out the Empirical/True Lipschitz constant for any fully connected neural network and hence an idea od the robustness of the particular system. Note that it also depends on the particular dataset being used.

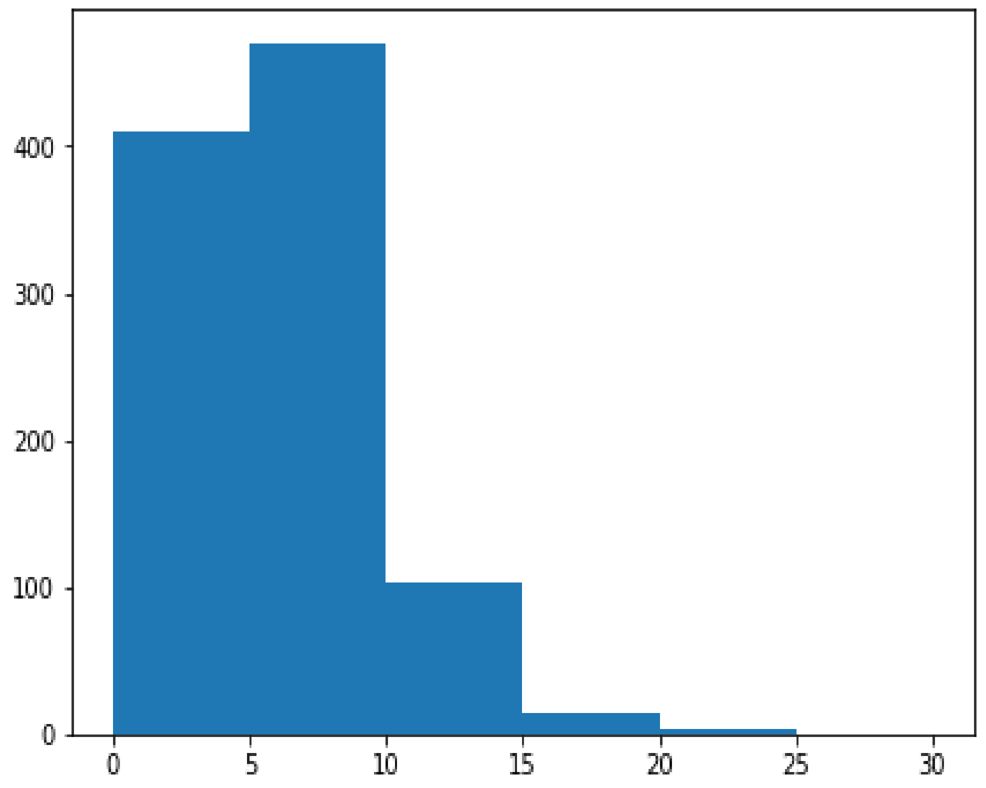


Algorithm 1: Empirical lipschitz constant over a data distribution

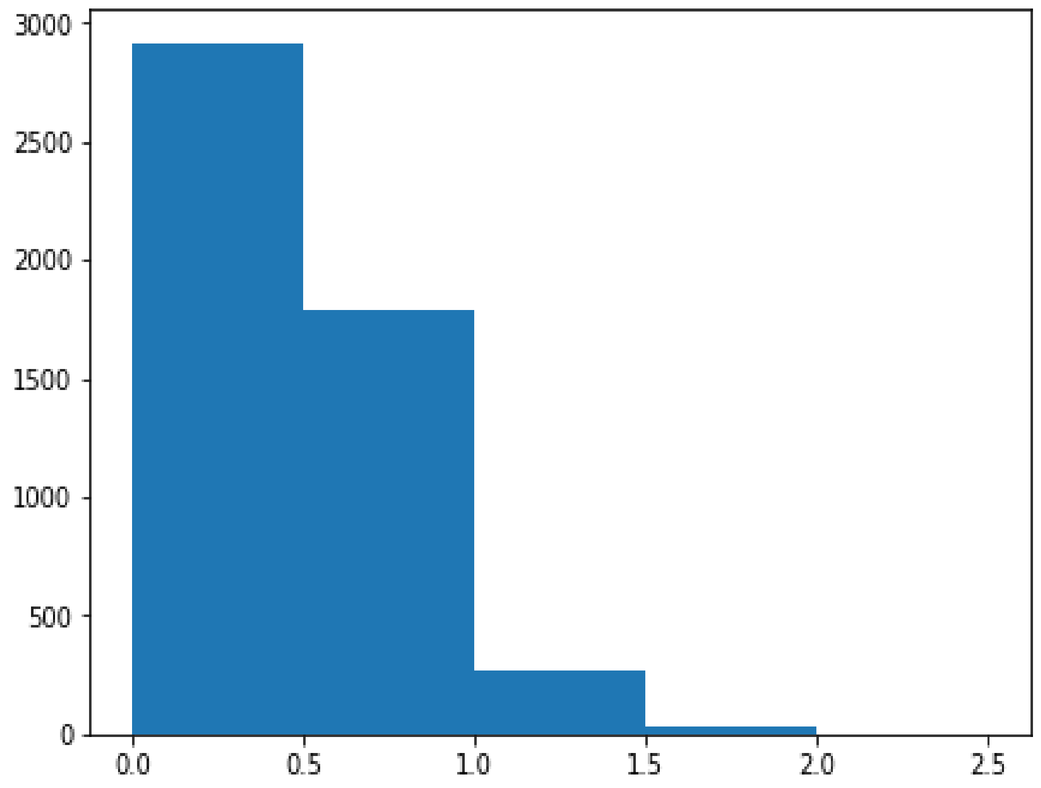
```
Input: Set Size N, Pre-trained model $f(x)$
Output: List of Max. Empirical Lipschitz bounds $L_{emp}$
Data: Test Split of Dataset D
L = []
itermax = 0
batches ← Shuffle(D, N)
for batch in batches do
    $pairs = \binom{batch}{2}$
    for pair in pairs do
        image1 ← pair[0]
        output1 ← f(image1)
        image2 ← pair[1]
        output2 ← f(image2)
        $L_{emp} = \frac{\|output2 - output1\|_2}{\|image2 - image1\|_2}$
        if $L_{emp} > itermax$ then
            itermax = $L_{emp}$
    Append itermax to L
Get L, itermax, $L_{emp}$
```

Methodology

- **Empirical Analysis** Using the Lipschitz formula in eqn (1), we conduct am empirical analysis to find out the distribution of Lipschitz constants over the test split of the MNIST and CIFAR-10 datasets respectively. These histograms are extremely important since they show us the order of magnitude in difference between the bounds obtained algorithimically and the actual bound.



FCN Histogram



CNN Histogram

Experimental Setup

- **Implementation Details** Here Fig 1 and Fig 2 denotes the CNN and FCN architectures which was used for our experiements - this was then converted to a fully connected network using the technqiues as explained earlier. Alll experiemnts were ran on a NVIDIA GTX 980 GPU with a batch size of 128, a learning rate of 1e-3 and for 4 epochs. We achieved a 96.9% test accuracy on the MNIST dataset after conversion of the CNN.

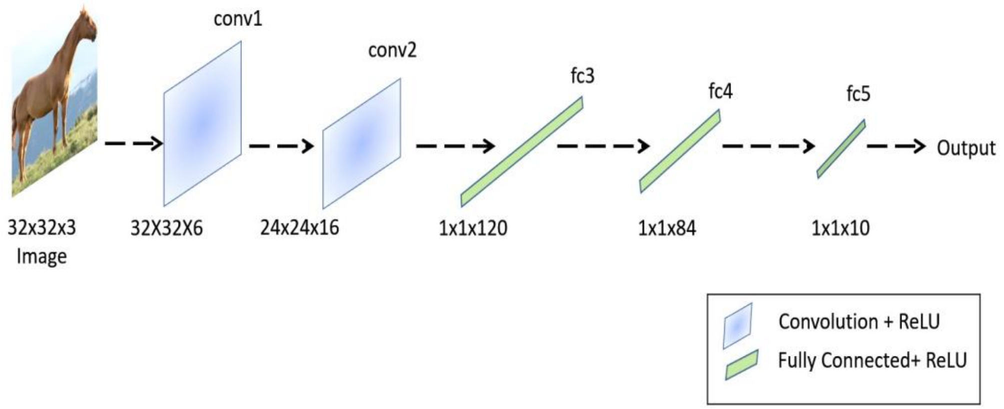


Fig 1: Architecture of the CNN

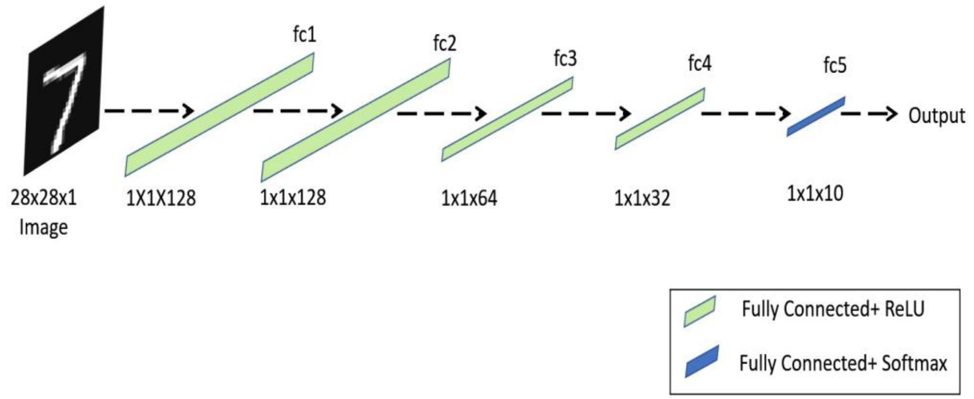


Fig 2: Architecture of the Fully Connected Network

I
C
C
C
N
T
2
0
2
2

Results

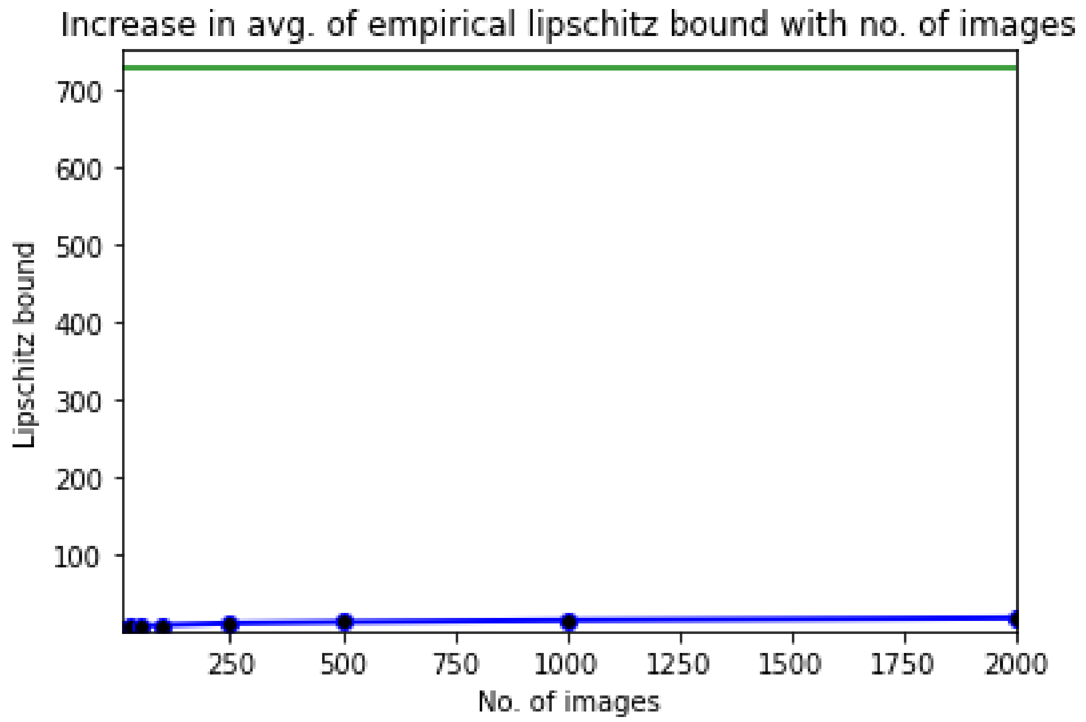


Figure 1

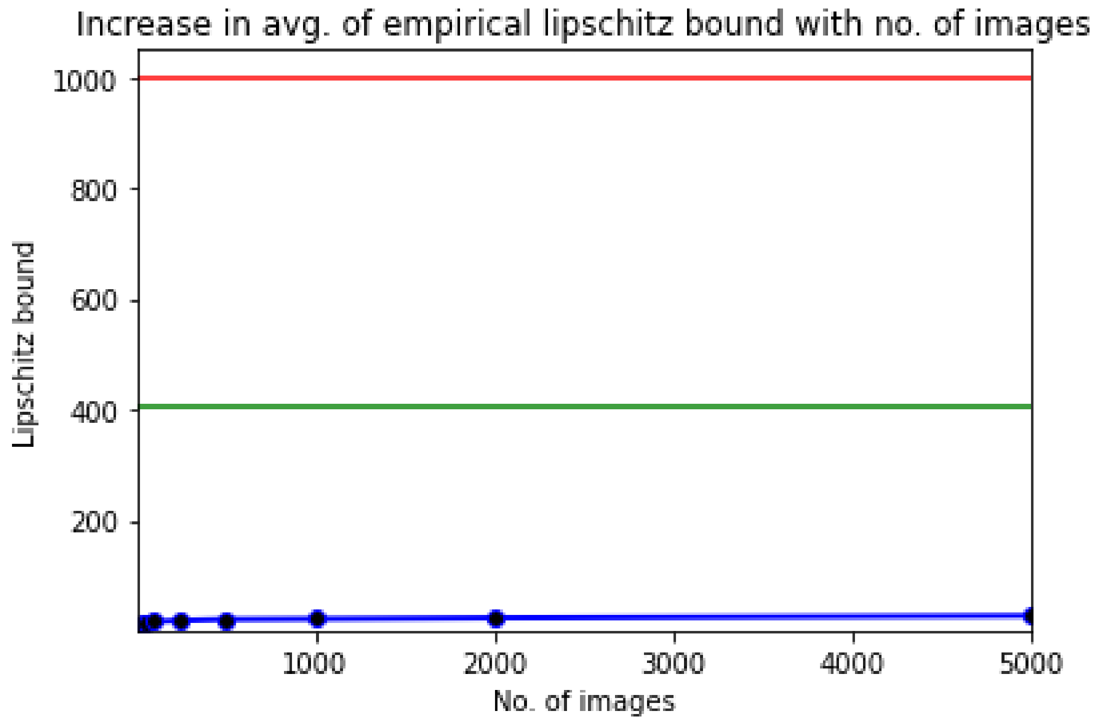


Figure 2

Obtained results based on Algorithm 1. The red line denotes the trivial lipschitz bound, the green line denotes the tight bound and the blue line denotes the actual lipschitz constant.

ICCCNT 2022

Results

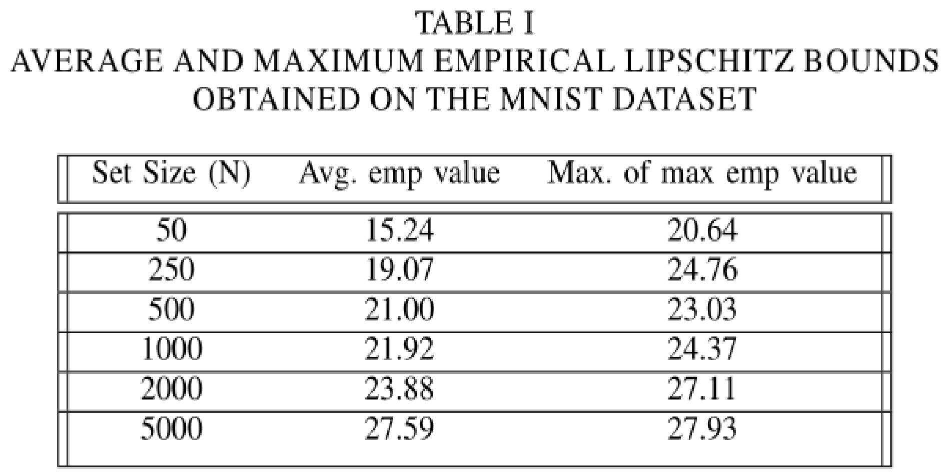


TABLE I
AVERAGE AND MAXIMUM EMPIRICAL LIPSCHITZ BOUNDS OBTAINED ON THE MNIST DATASET

Set Size (N)	Avg. emp value	Max. of max emp value
50	15.24	20.64
250	19.07	24.76
500	21.00	23.03
1000	21.92	24.37
2000	23.88	27.11
5000	27.59	27.93

Table 1

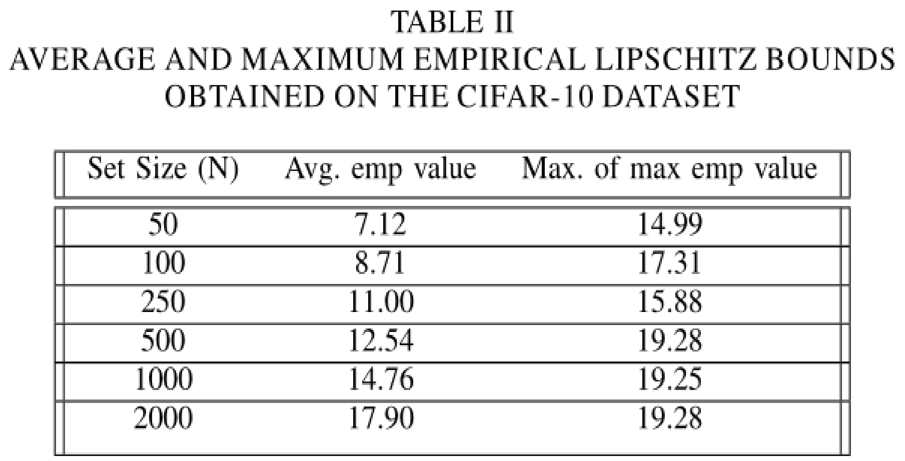


TABLE II
AVERAGE AND MAXIMUM EMPIRICAL LIPSCHITZ BOUNDS OBTAINED ON THE CIFAR-10 DATASET

Set Size (N)	Avg. emp value	Max. of max emp value
50	7.12	14.99
100	8.71	17.31
250	11.00	15.88
500	12.54	19.28
1000	14.76	19.25
2000	17.90	19.28

Table 2

Obtained results based on Algorithm 1. Here, the set size (N) refers to the batch size of images being used.

Future Directions

- Possible next approaches are briefly summarized below:-

1. Utilizing a memory free way of computing Lipschitz constants for CNNs.
2. Improvement of the algorithm to account for non-linearity in all types of networks other than Fully Connected Networks and CNNs. This can be achieved with the help of 1.
3. Move towards data free/online approaches - useful for real time robustness.
4. If possible, find more use cases such as (Reinforcement Learning) in which this algorithm can be applied. For Ex: adverse systems where the data is higly unreliable.

References

[1] Araujo, Alexandre, et al. "On lipschitz regularization of convolutional layers using toeplitz matrix theory." *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 35. No. 8. 2021.

[2] Szegedy, Christian, et al. "Intriguing properties of neural networks." *arXiv preprint arXiv:1312.6199* (2013).

[3] Chellapilla, Kumar, Sidd Puri, and Patrice Simard. "High performance convolutional neural networks for document processing." *Tenth international workshop on frontiers in handwriting recognition*. Suvisoft, 2006.

** Please see our paper for the entire list of references.*

I
C
C
C
N
T
2
0
2
2

Thank You!

For more information, please read our paper: “Lipschitz Bound Analysis of Neural Networks”.

Code is available at: https://github.com/sarosijbose/LBANN