Master Thesis

Stochastic Computing in Neural Networks to Defend Against Adversarial Attacks

Manish Mahesh Dalvi

[redacted]

Master Arbeit Nr : [redacted]

09/04/2020 – 09/10/2020

Examiner: Prof. Dr. rer. nat. habil. Ilia Polian
Head and Chair of Hardware Oriented Computer Science (HOCOS)
Institute of Computer Architecture and Computer Engineering

Supervisor: Florian Neugebauer (M.Sc)

Universität Stuttgart
HOCOS

Content

- Goal
- Background of NN and CNN
- Neural Network Threats
- Stochastic Computing and benefits
- Stochastic Computing Neural Network
- Adversarial Attack - Deep Fool Attack
- Dataset and Architecture
- Results
- Analysis
    - Optimizer Analysis
    - Structural Analysis of Images
- Conclusion and Future Work

Goal

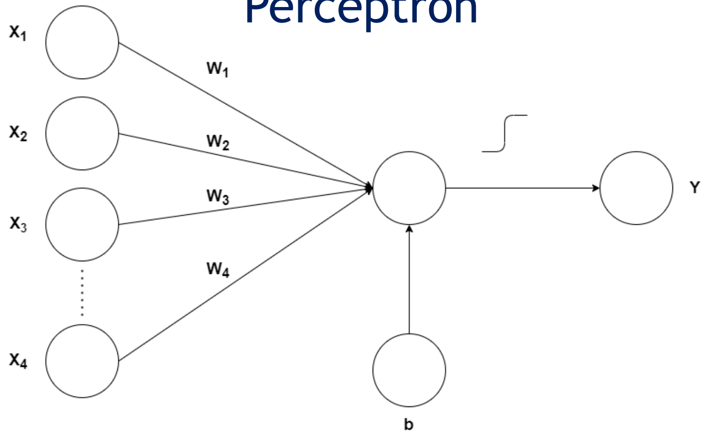
- Evaluate the Robustness of Stochastic Convolutional Neural Network (SCNN) to defend against Adversarial attack for several datasets.

- Variety of datasets – MNIST, Fashion MNIST, CIFAR-10
- Implementation of different SCNN architectures
- Deep Fool Attack
- Comparison between SCNN and Conventional CNN

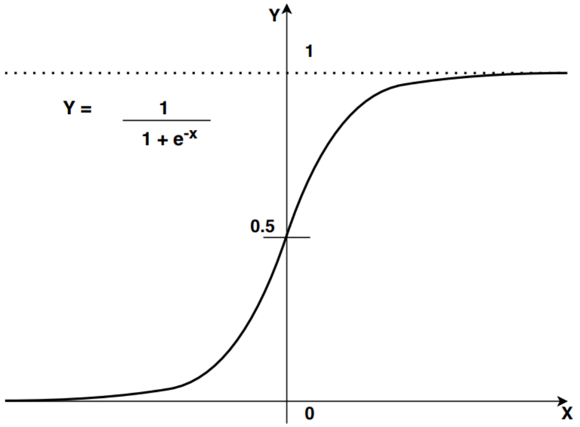
Neural Networks

- Inspired by Human Brain
- First NN – Perceptron
- Logistic Regression
- Neural Network Development and Uses
  - FCN
  - CNN
  - RNN

Perceptron

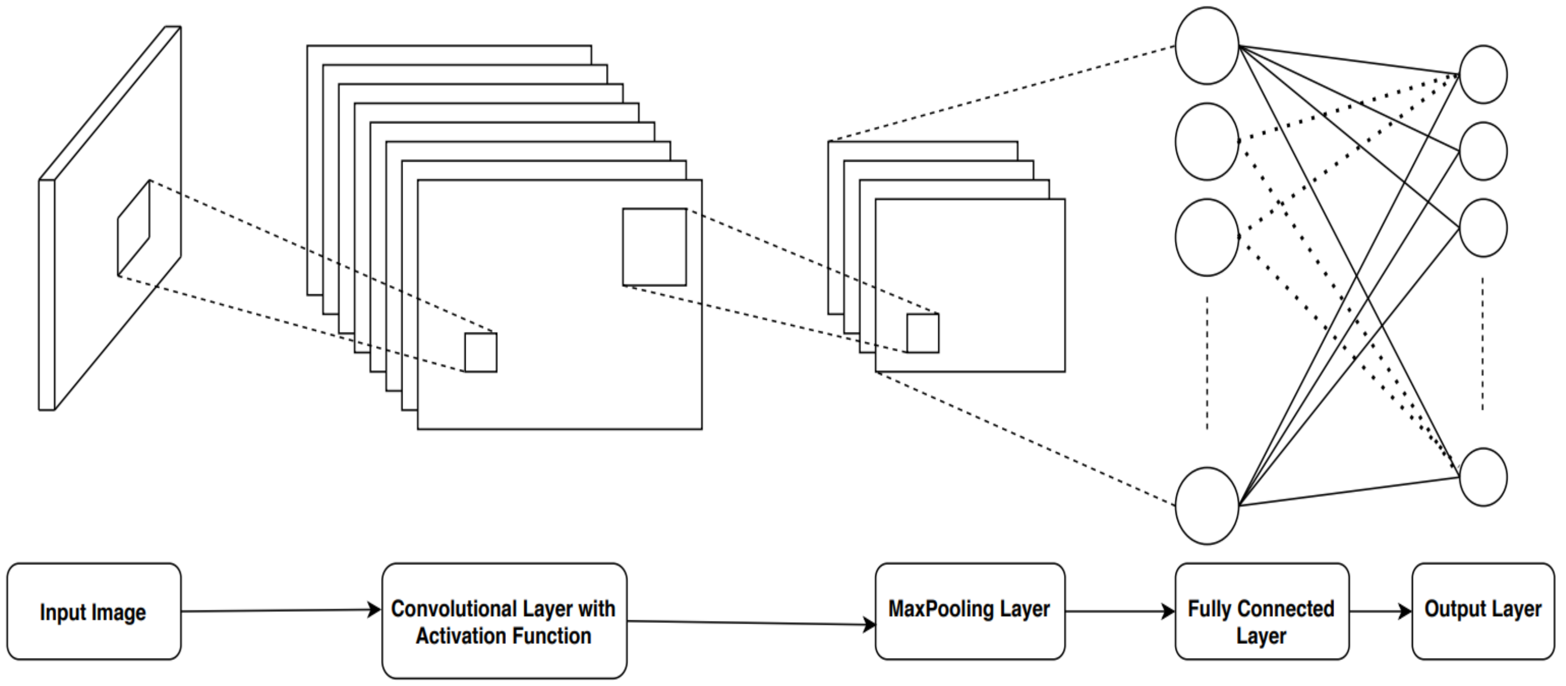


Logistic Curve

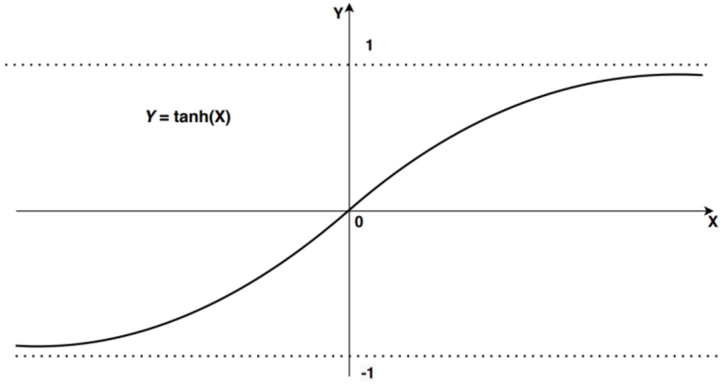


Convolutional Neural Networks

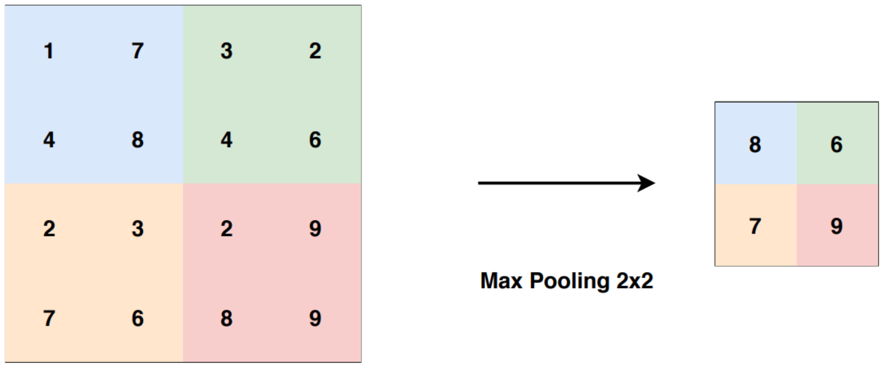
- Inspired by Visual Cortex
- Layered Feature Extraction
- Various layers of CNN
    - Input Image
    - Kernel Matrix
    - Activation Function
    - Max Pooling
    - Fully Connected Layer



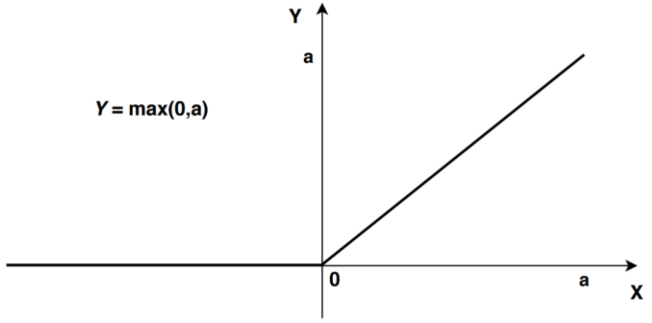
Layers of CNN



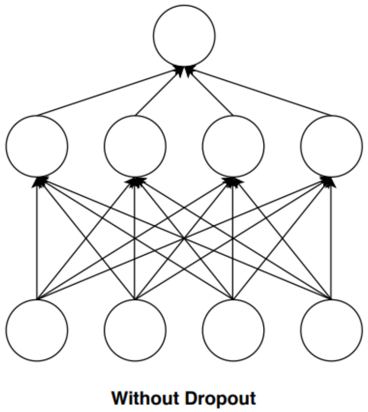
Tanh



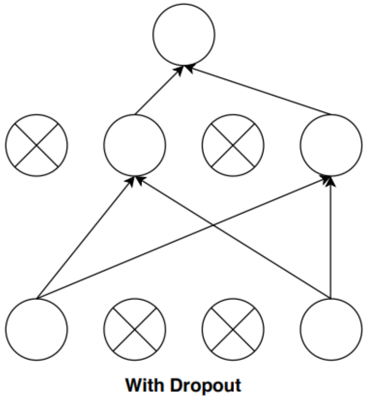
Max Pooling



ReLU



FCN



Dropout

CNN - Disadvantages and Threats

- High number of Computation
- Depth – Overfitting
- Adversarial Attacks
- Cannot handle small pertubations
- Solution ?

Stochastic Computation

- What is Stochastic Computation ?
- Benefits of SC – Low Power, Small Area
- Disadvantages of SC – Approximate computation, SN length, Addition
- Arithmetic Operations in Stochastic Domain

Unipolar

P = 0.75 -> 75% of 1s and 25% of 0s
1110, 1011, 0111, 1101, 1110

P = number of 1s/ length of SN

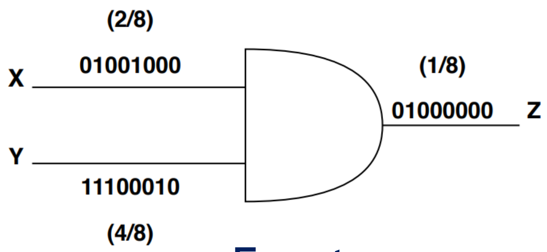
Bipolar

P = 0.75
1111 1110, 1011 1111 …

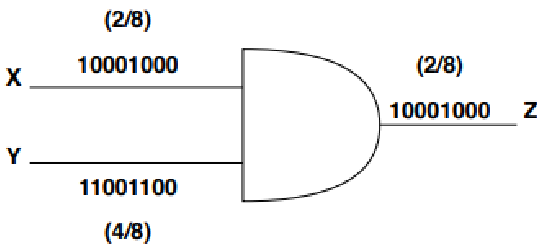
P = (number of 1s – number of 0s)/ length of SN

Stochastic Computation

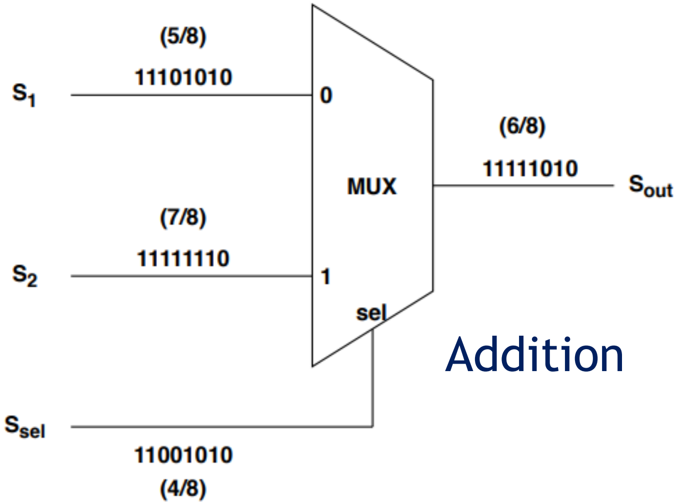
Multiplication



Exact



Approximate



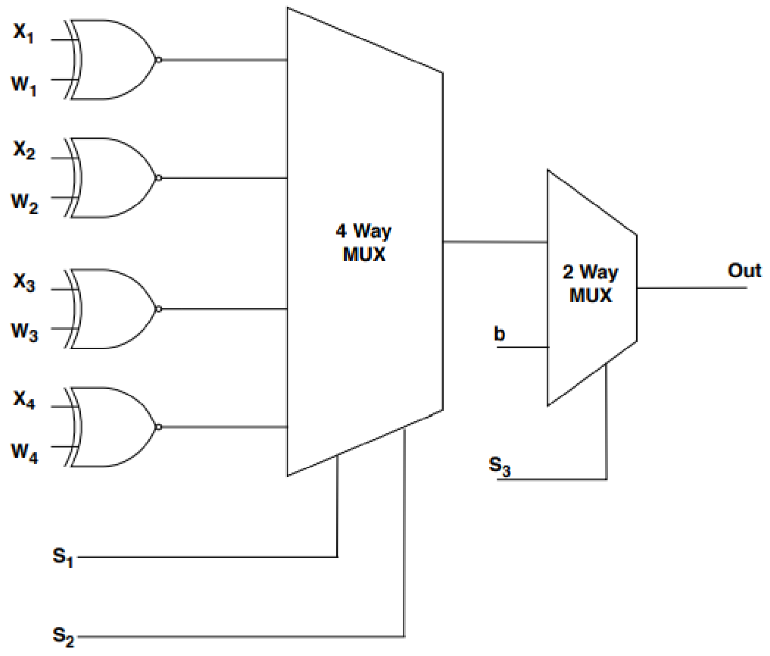
Addition

$$\sum_{i=1}^{n} S_1(i)S_2(i) = \frac{\sum_{i=1}^{n} S_1(i) * \sum_{i=1}^{n} S_2(i)}{n}$$

Correlation

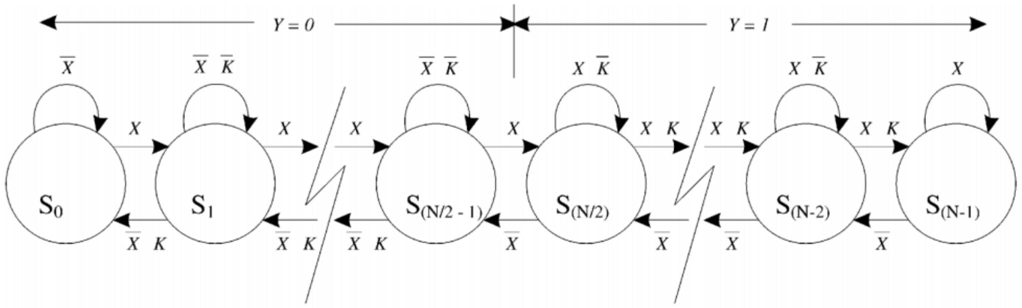
Stochastic Computation in NN

- Why SC in NN ?
- Combining SC and NN
- Convolution Operation in SC
- Activation Function in SC



Convolution in SC

Stanh

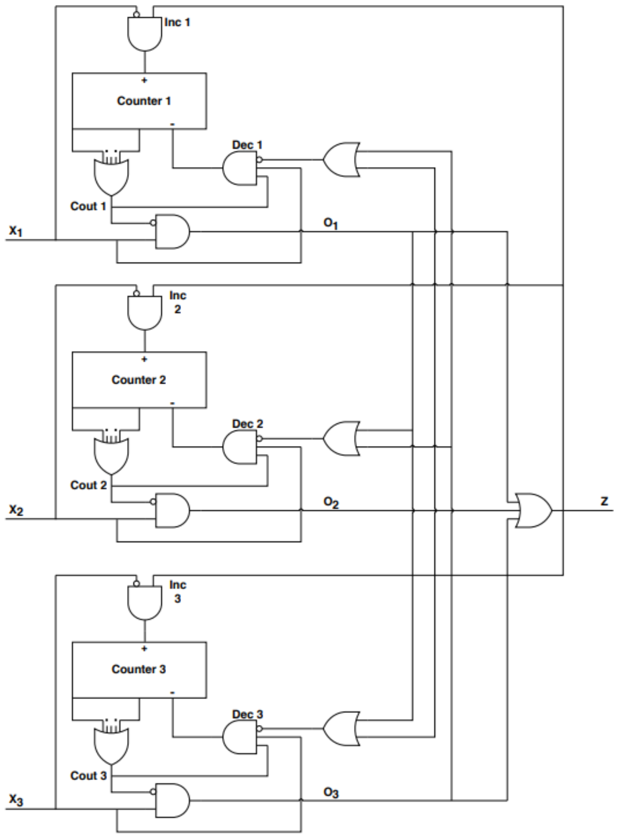


Universität Stuttgart HOCOS

Stochastic Computation in NN

- Max pooling in NN using NMAX function

Clock Cycle	1	2	3	4	5	6	7	8
X_1	1	1	0	1	0	1	1	1
X_1	1	0	0	1	0	0	1	0
X_1	0	0	1	0	0	0	0	1
Counter 1	0	0	0	0	0	0	0	0
Counter 2	0	1	1	1	1	2	2	3
Counter 3	1	2	1	2	2	3	4	4
Z	1	1	0	1	0	1	1	1



Universität Stuttgart ITI HOCOS

Adversarial Attacks

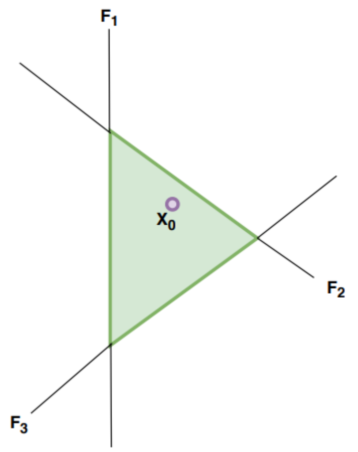
- Various types of Attack
  - Black Box
  - White Box
  - Targeted
  - Non-Targeted

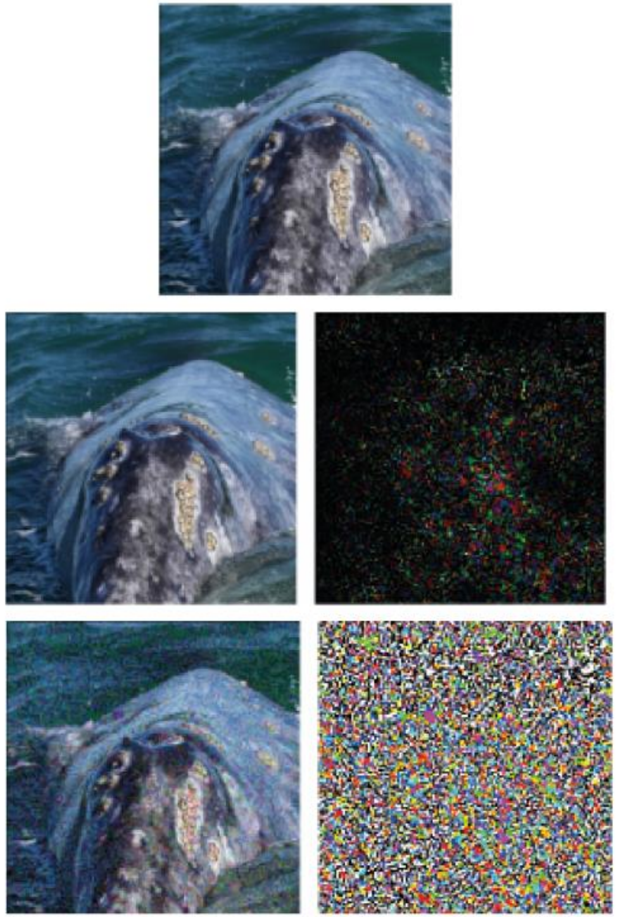
- Deep Fool
- FGSM
- One Pixel Attack..

Focus on Deep fool attack due to smaller level of perturbations and higher success rate of attack

Deep Fool Attack

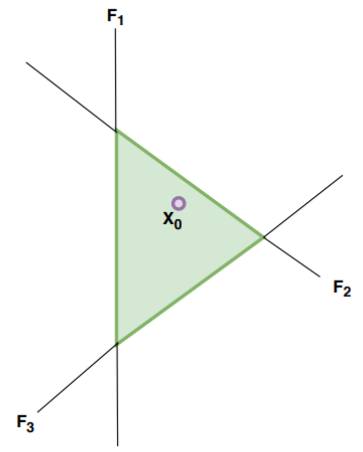
- Detection of Nearest Hyperplane
- Calculation of perturbations
- Image + Perturbation = Modified Image





Deep Fool Attack

1. Initialize input to input image
2. Loop for all labels other than original label
3. Modify weights for each class using affine function
4. Calculate distance of current point to all hyperplanes with label different than original
5. Fetch nearest hyperplane
6. Update perturbation value to the selected hyperplane which is distance in the direction of selected hyperplane
7. Add perturbations to image
8. Repeat from Step 3 until Original Label is not equal to Perturbed label



Deep Fool Attack Important Equations

- Calculate Closest hyperplane to point x_0

$$\hat{l}(\boldsymbol{x}_0) = \underset{k \neq \hat{k}(\boldsymbol{x}_0)}{\arg\min} \frac{\left| f_k(\boldsymbol{x}_0) - f_{\hat{k}(\boldsymbol{x}_0)}(\boldsymbol{x}_0) \right|}{\| \boldsymbol{w}_k - \boldsymbol{w}_{\hat{k}(\boldsymbol{x}_0)} \|_2}.$$

- Compute Minimum perturbation required to shift x_0 to hyperplane

$$\boldsymbol{r}_*(\boldsymbol{x}_0) = \frac{\left| f_{\hat{l}(\boldsymbol{x}_0)}(\boldsymbol{x}_0) - f_{\hat{k}(\boldsymbol{x}_0)}(\boldsymbol{x}_0) \right|}{\| \boldsymbol{w}_{\hat{l}(\boldsymbol{x}_0)} - \boldsymbol{w}_{\hat{k}(\boldsymbol{x}_0)} \|_2^2} (\boldsymbol{w}_{\hat{l}(\boldsymbol{x}_0)} - \boldsymbol{w}_{\hat{k}(\boldsymbol{x}_0)}).$$

- Compute perturbed Image by adding previous image + obtained perturbation

$$\boldsymbol{x}_{i+1} \leftarrow \boldsymbol{x}_i + \boldsymbol{r}_i$$

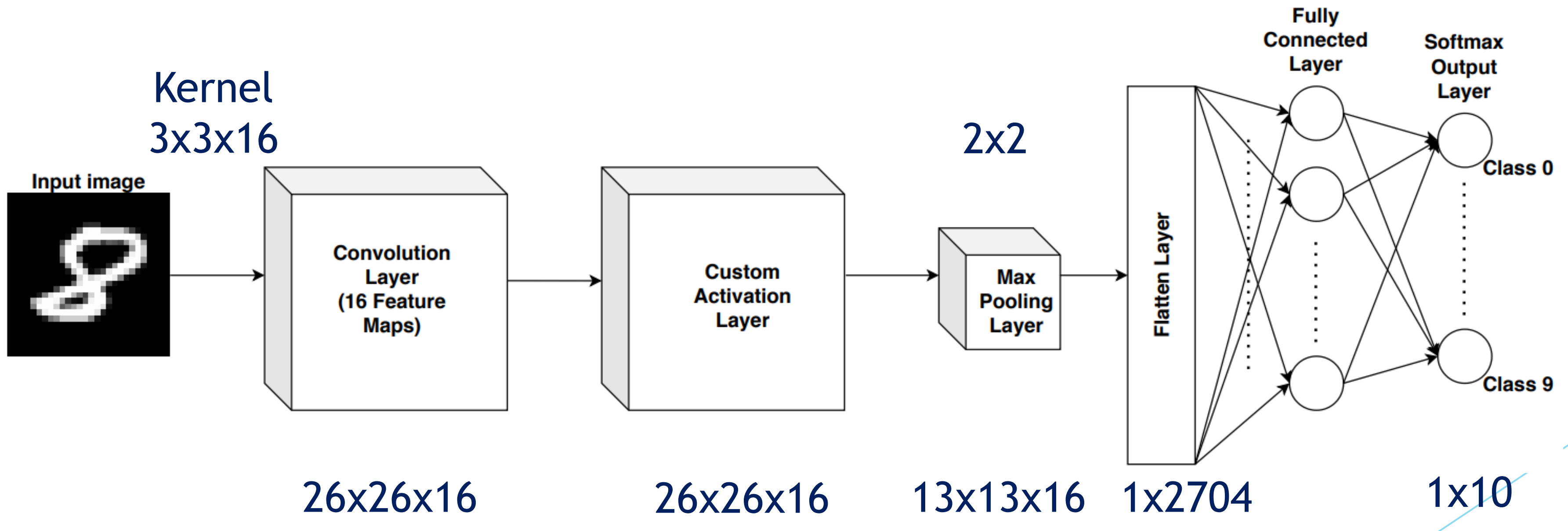
Dataset and Architecture

- MNIST, Fashion MNIST and CIFAR-10

- Complexity of Dataset
- Architecture Overview
  - Depth
  - Number of Feature Maps
  - Dropouts
  - Activation Layer (Tanh, ReLU)

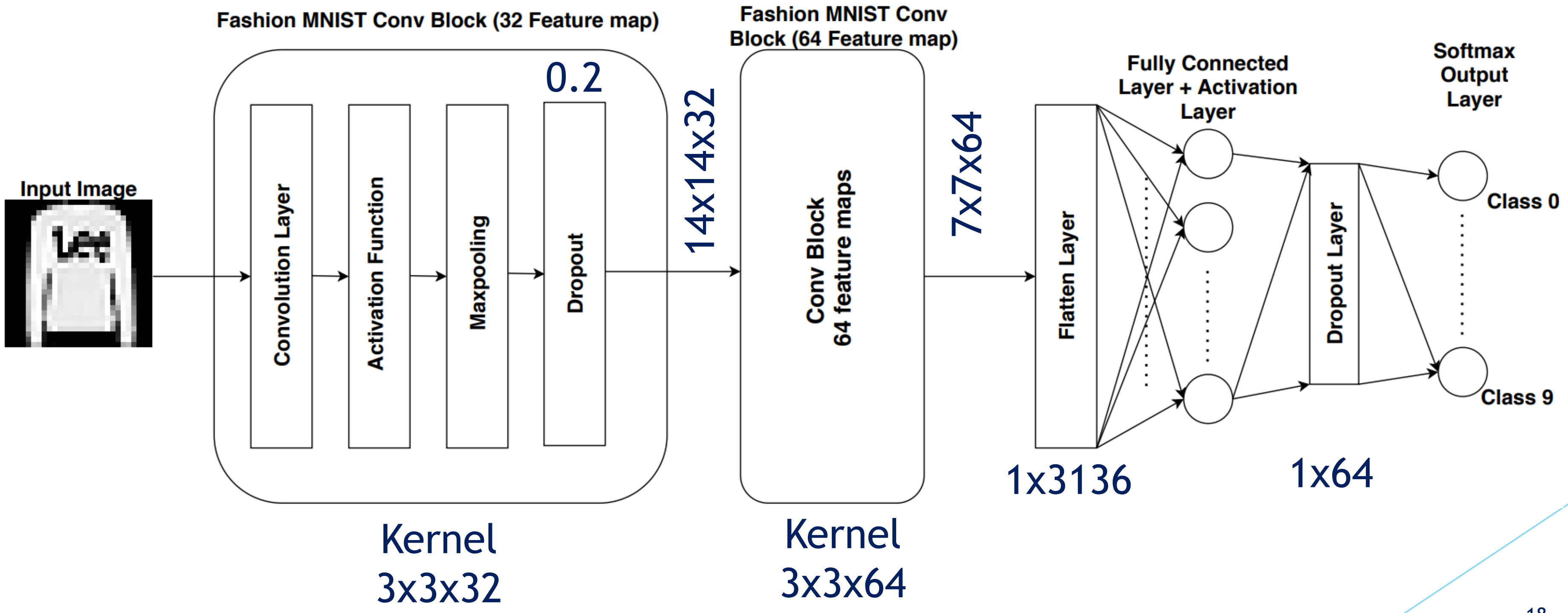
MNIST CNN Architecture

- Test Accuracy – 93.8%
- Single Layer
- 16 Feature maps
- Stanh activation
- Nmax Max Pooling layer



Fashion MNIST CNN Architecture

- Two Layer
- Accuracy – Tanh – Test – 94.02%

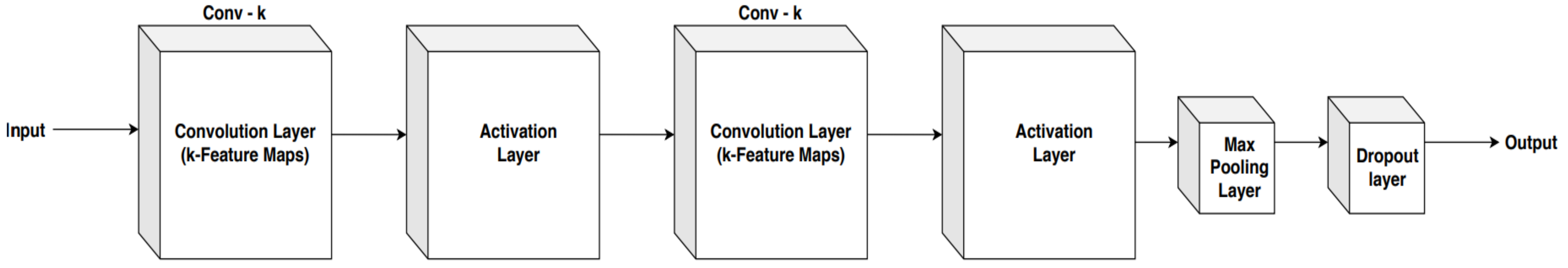


CIFAR-10 CNN Architecture

- 8 Layer
- Depth
- Convolution Blocks
- Accuracy – Tanh – Test - 81.3%

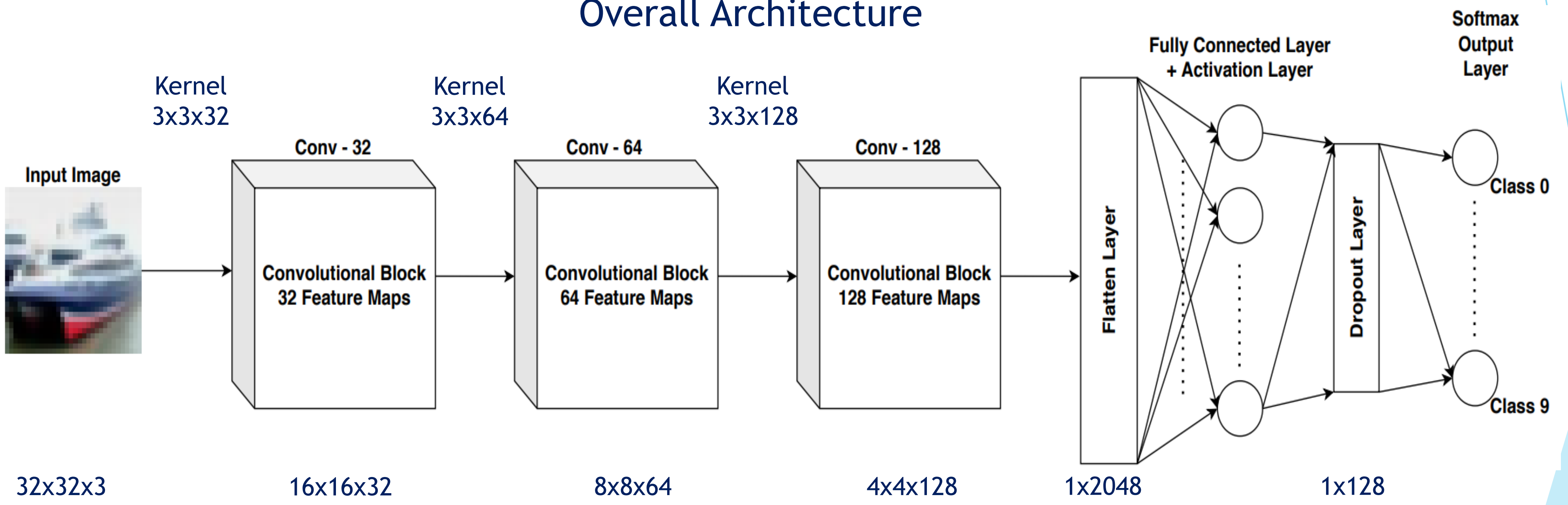
  ReLU – Test - 85.93%

Single Convolutional Block



CIFAR-10 CNN Architecture

Overall Architecture



CIFAR-10 CNN Training

- Select the layer
- Train network with Keras/Tensorflow
- Scale weights between -1 to 1 for selected layer using call back
- Export weights after training
- Usually 200 epochs, 64 batch size for CIFAR-10

- Build Stochastic layer using Numpy
- Convert Binary number to stochastic number before the selected stochastic layer
- Convert back to binary number after the stochastic layer
- Compute accuracy

Evaluation Criteria

- Accuracy = (number of correctly classified images/total number of images) * 100

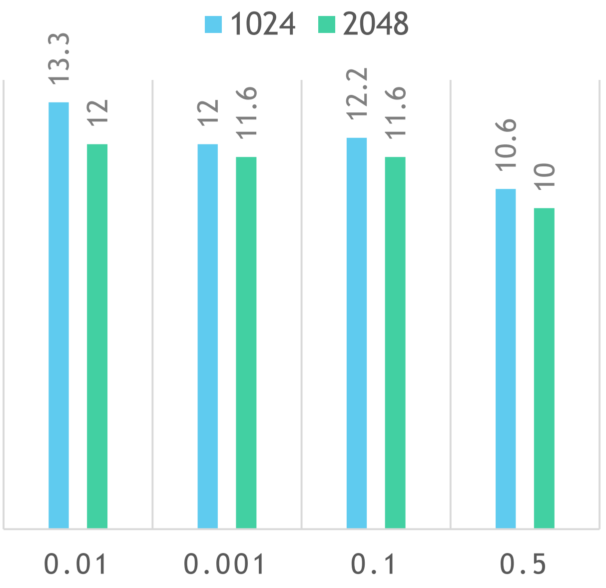
- Success rate of the attack =

  (accuracy before attack − accuracy after attack)/accuracy before attack

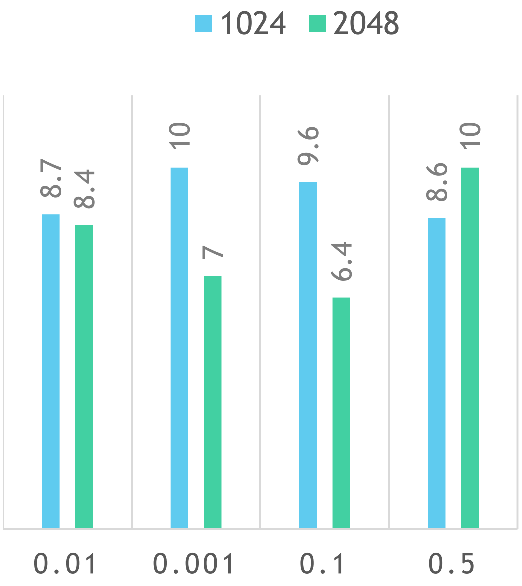
Results

- Stochastic ReLU with MNIST provides best accuracy over tanh activation

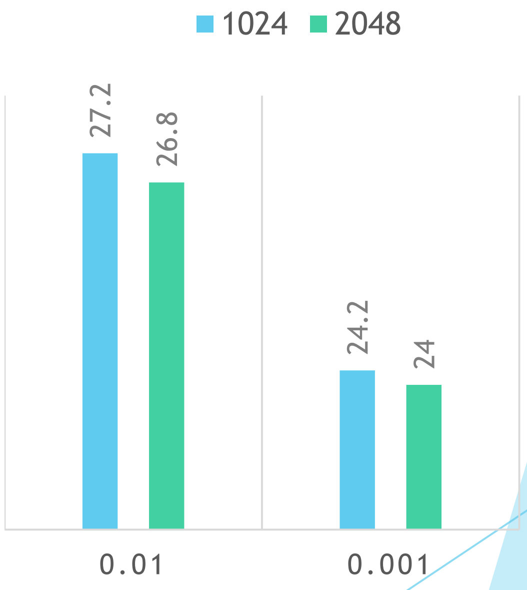
MNIST – Complete layer Stochastic



Tanh Activation



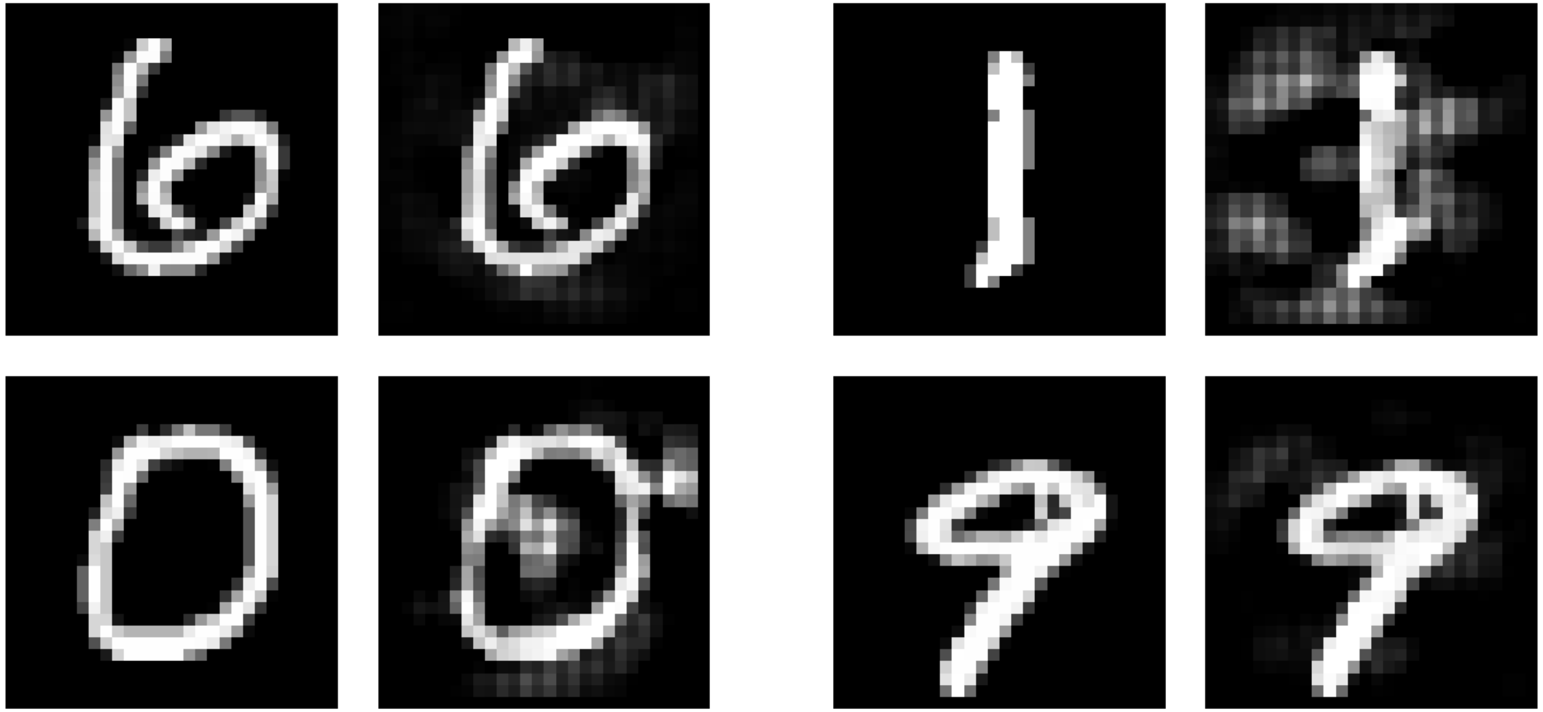
Btanh Activation



ReLU Activation

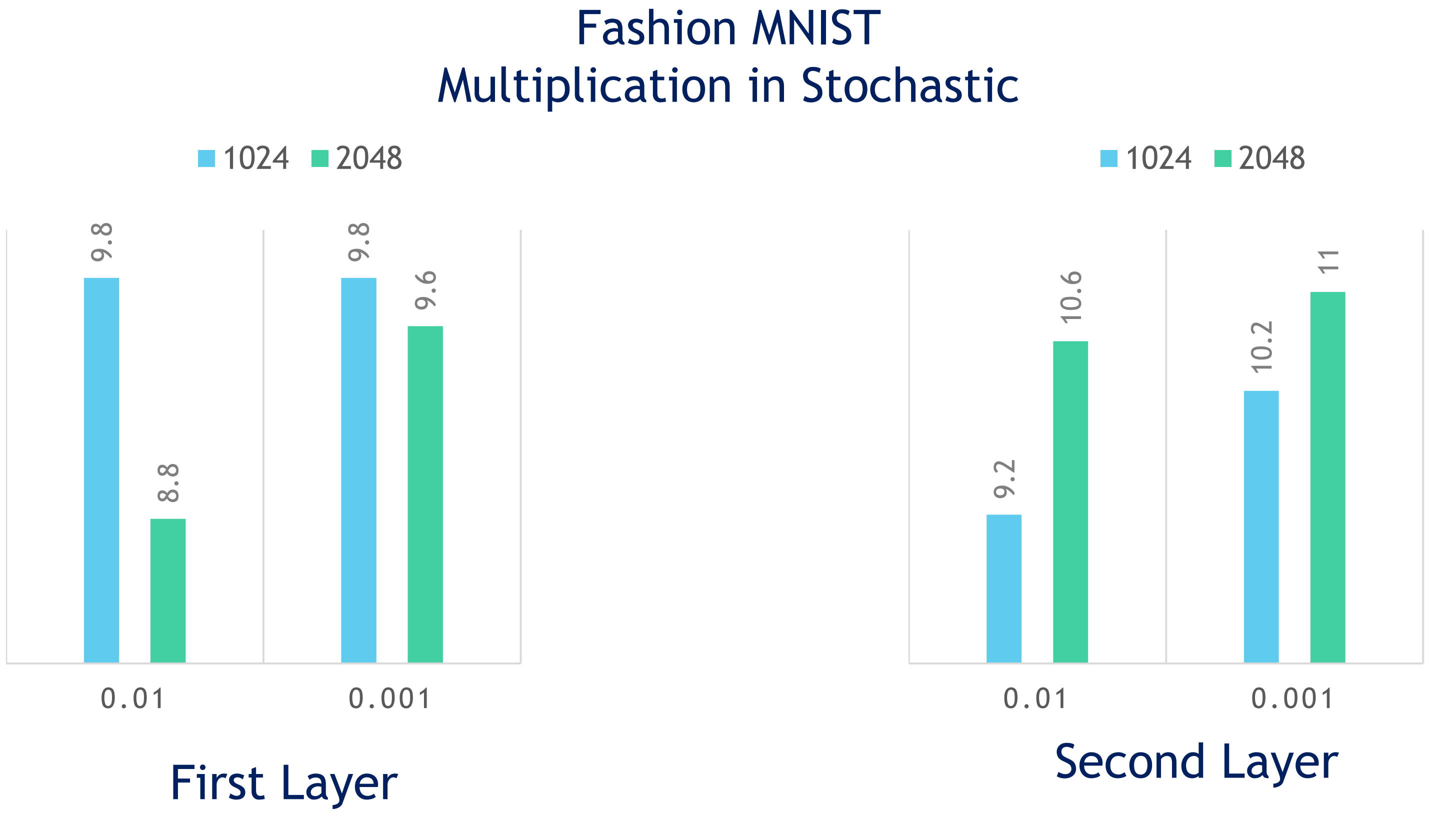
Results

MNIST perturbed images



Results

- Fashion MNIST provides marginal resistance to Deep fool attack



Universität Stuttgart ITI HOCOS

Results

- Randomness due Addition + Multiplication in stochastic reduces more errors compared to only multiplication in Stochastic.
- Second layer performs better over the first layer.

Fashion MNIST
Complete Layer in Stochastic

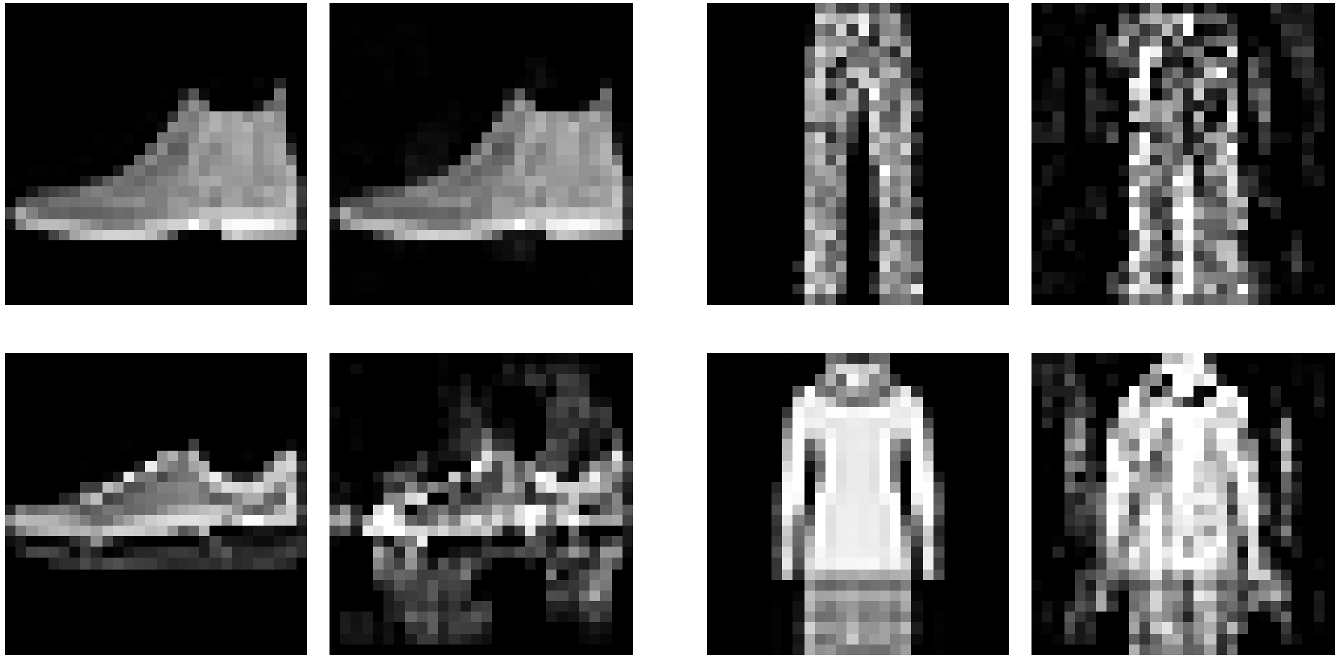
First Layer

Second Layer

Universität Stuttgart ITI HOCOS

Results

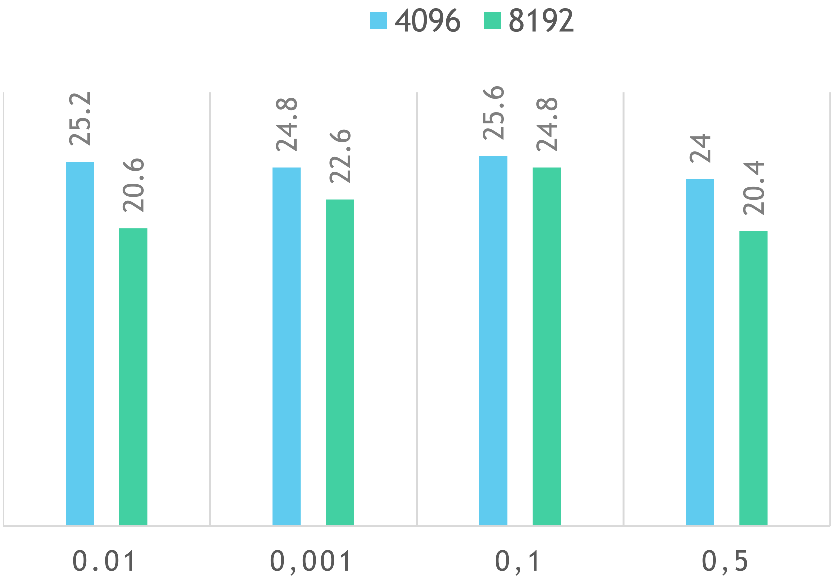
Deep Fool Attack on Fashion MNIST dataset at 0.1 overshooting value



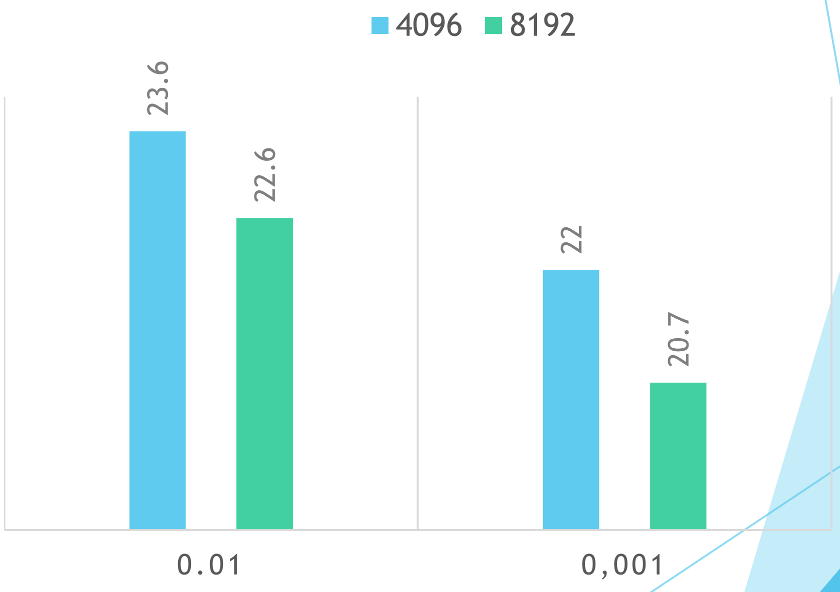
Results

- In CIFAR-10 architecture, First layer provides best results
- Mainly due to reducing errors in initial stages
- Errors are magnified in subsequent layers hence lower results

CIFAR-10
Multiplication in Stochastic



First Layer Convolution

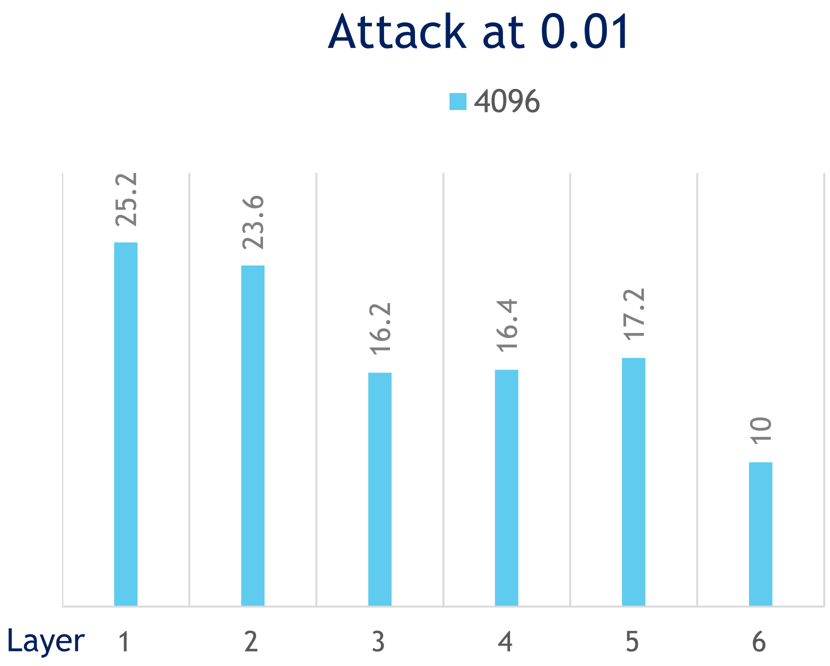


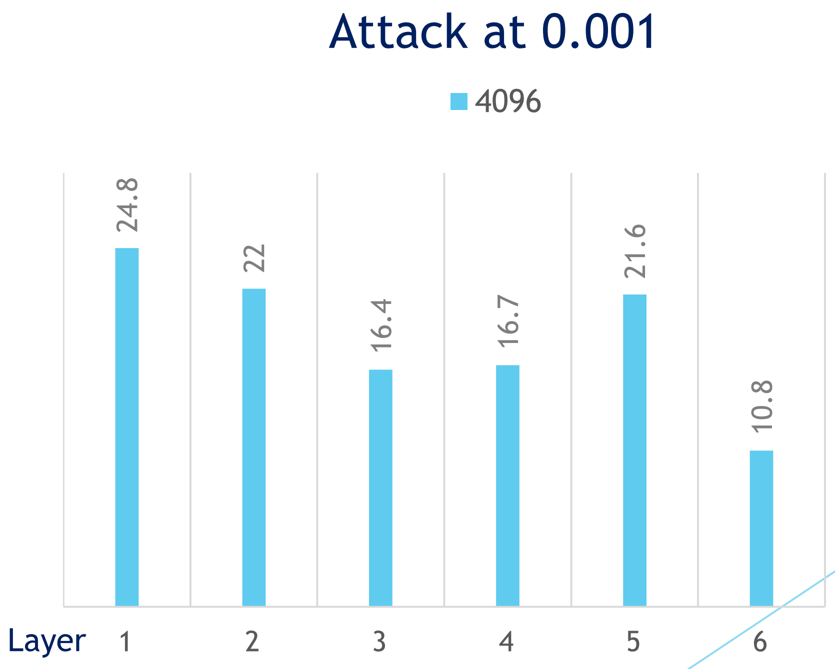
Second Layer Convolution

Results

- Steady drop in accuracy over layers
- Perturbations spread across 3 channels compared to single channel in MNIST
- Error spread due arithmetic operations in subsequent layers

CIFAR-10
Multiplication in Stochastic



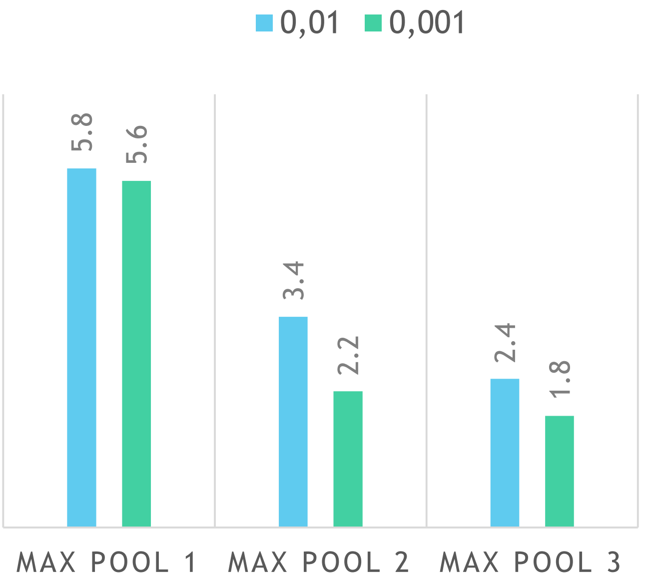


Universität Stuttgart ITI HOCOS

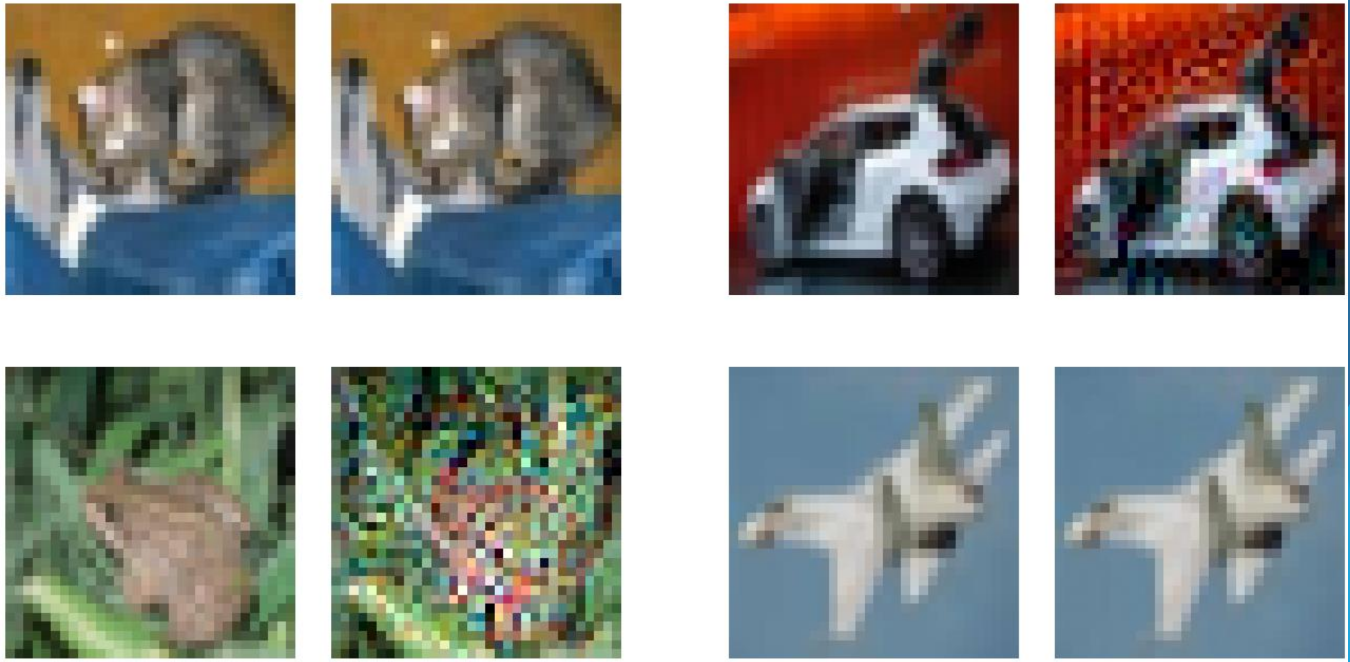
Results

- Stochastic Max pooling alone provides marginal resistance to attacks
- Accuracy decrease over deeper layers

Nmax function at each layer in Stochastic



CIFAR-10 Perturbed Images

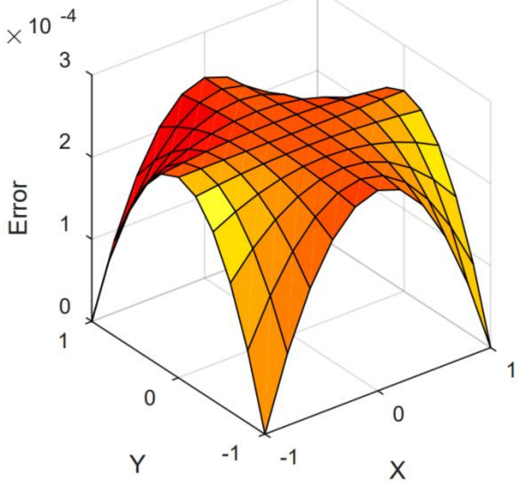


Universität Stuttgart ITI HOCOS

Analysis

- Why Adadelta as Activation function ?
- Relation between Structural Index and SCNN correct classification

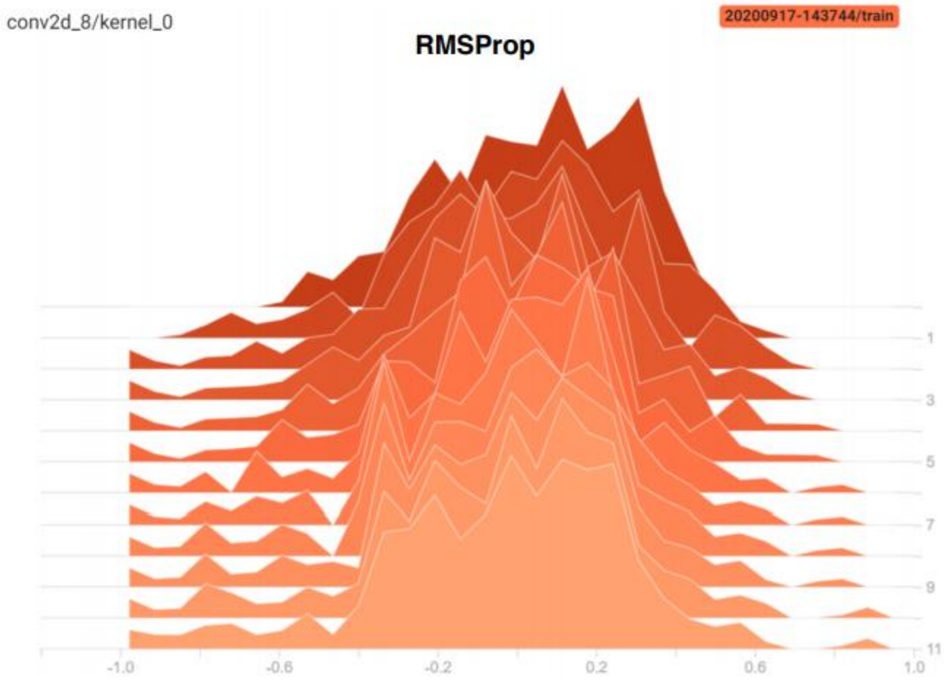
Optimizer	CNN	SCNN (SN length 2048)
RMSprop	96.2%	57.6%
Adadelta	94.8%	87.4%
AdaGrad	95.4%	58.4%
Adam	97%	79.8%

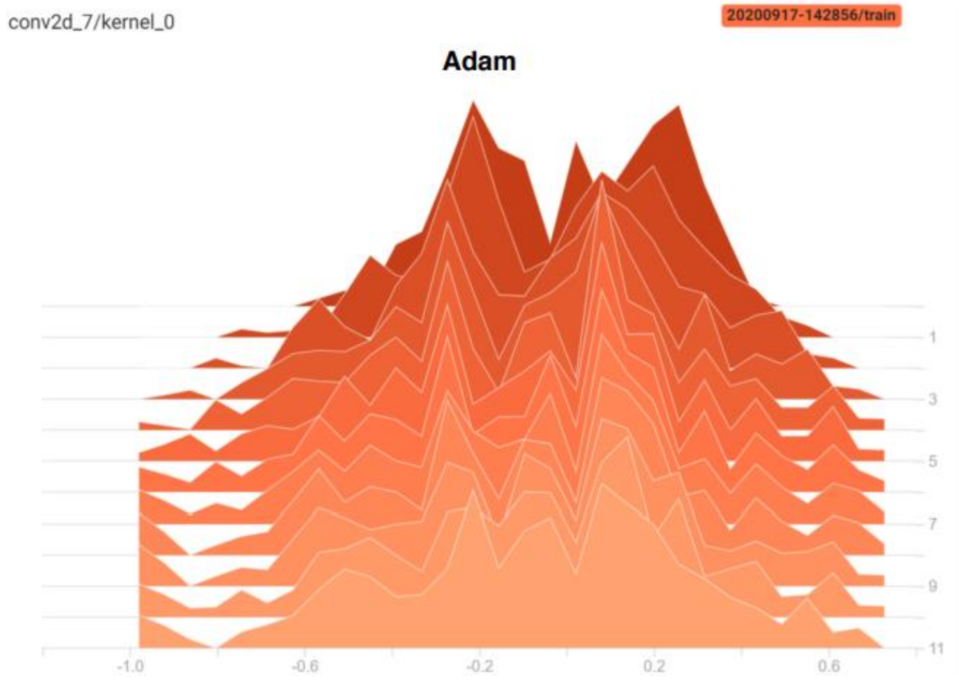


XNOR Error

Optimizer Analysis

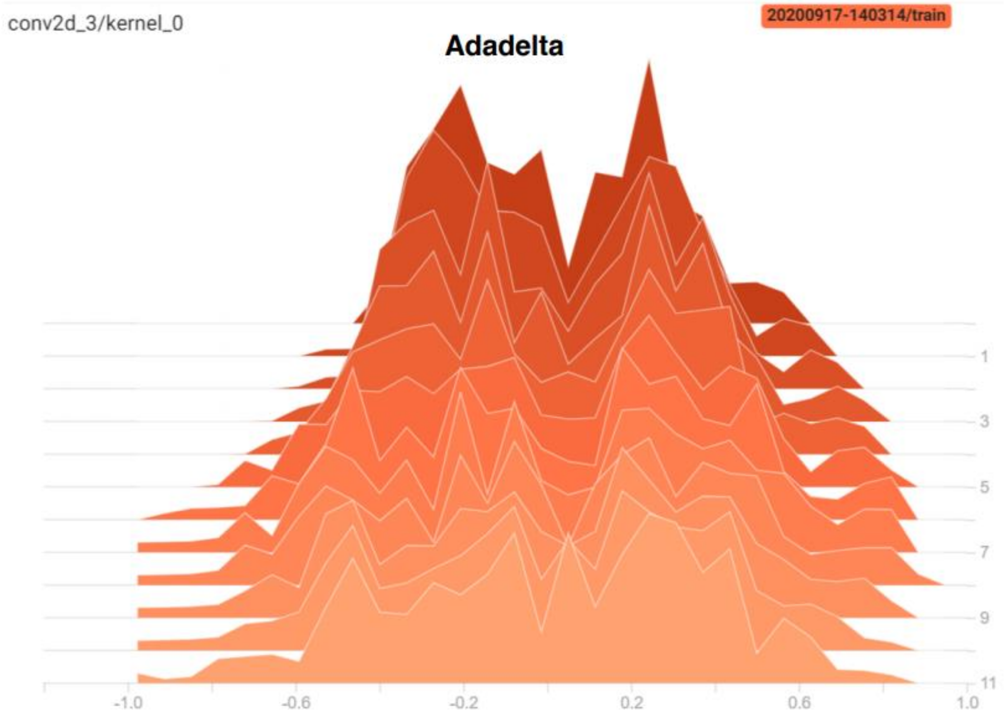
- Weight Distribution of RMSProp and Adam
- RMSProp weight distribution around 0
- Adam more distribution around 0.2 hence better accuracy in stochastic domain as well

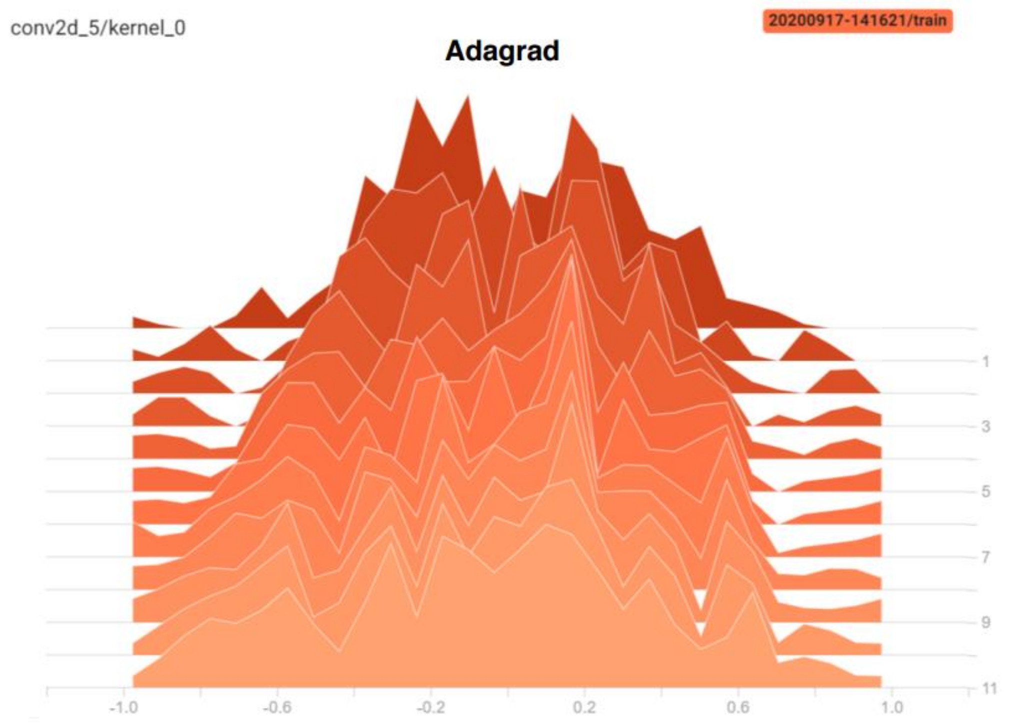




Optimizer Analysis

- Weight Distribution of Adadelta and Adagrad
- Adadelta – distribution around +0.2
- Adagrad – distribution around 0





Structural Analysis of Images

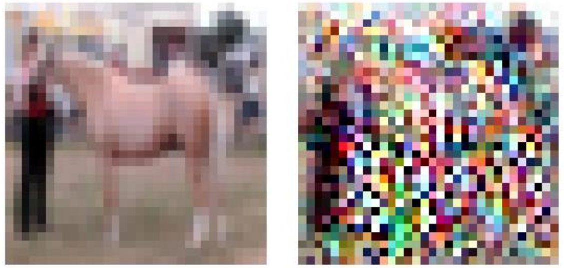
- Structural Similarity Index (SSIM)
- When SSIM=1 Images very similar
- SSIM~0 = Not very similar

$$SSIM(x,y) = \frac{(2\mu_x\mu_y + c_1)(2\sigma_{xy} + c_2)}{(\mu_x^2 + \mu_y^2 + c_1)(\sigma_x^2 + \sigma_x^2 + c_2)}$$

0.9999683

0.9604659

0.93969005



0.25640938

Conclusion and Future Work

- SC good for Initial Layers.
- Level of Perturbation should be low/ non perceivable.
- Choosing Correct Optimizer is very important.
- Stochastic Computing helps against adversarial attacks.
- SCNN adds good amount of robustness to a Conventional NN.

Future Work

- Can be extended to other datasets like ImageNet
- Test for other Architectures and different adversarial attacks
- SCNN Comparison for Multilabel classification vs multiclass classification
- SCNN and adversarial for more number of Images

References

- F. Neugebauer, I. Polian, and J. P. Hayes. On the maximum function in stochastic computing. Proceedings of the 16th ACM International Conference on Computing Frontiers, ser. CF ’19. New York, NY, USA: Association for Computing Machinery, page pp. 59–66, 2019
- B.R. Gaines. Stochastic computing systems. Advances in Information Systems Science, vol. 2:pp. 37 – 172, 1969.
- C. Cortes Y. LeCun and C. Burges. Mnist handwritten digit database. ATT Labs, vol. 2, 2010.
- K. Rasul H. Xiao and R. Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. CoRR, vol. abs/1708.07747, 2017.
- Alex Krizhevsky. Learning multiple layers of features from tiny images. Chapter 3, 2009.
- Pascal Frossard Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi. Deepfool: a simple and accurate method to fool deep neural networks. Ecole Polytechnique F ed erale de Lausanne, 2016.
- P. Barham E. Brevdo Z. Chen C. Citro G. S. Corrado A. Davis J. Dean M. Devin S. Ghemawat I. Goodfellow A. Harp G. Irving M. Isard Y. Jia R. Jozefowicz L. Kaiser M. Kudlur J. Levenberg D. Mane R. Monga S. Moore D. Murray C. Olah M. Schuster J. Shlens B. Steiner I. Sutskever K. Talwar P. Tucker V. Vanhoucke V. Vasudevan F. Viegas O. Vinyals P. Warden M. Wattenberg M. Wicke Y. Yu M. Abadi, A. Agarwal and X. Zheng. Tensorflow: Large-scale machine learning on heterogeneous distributed systems. 2016.
- Jonas Rauber, Wieland Brendel, and Matthias Bethge. Foolbox: A python toolbox to benchmark the robustness of machine learning models. 2017.
- Joonsang .Y Jungwoo .S Jongeun .L Kiyoung .C Kyounghoon .K, Jungki .K. Dynamic energy-accuracy trade-off using stochastic computing in deep neural networks. 53nd ACM/EDAC/IEEE Design Automation Conference (DAC), 2016.

Thank you Note

Thank you to Prof.Dr. Ilia Polian for the Thesis opportunity.

Thank you to M.Sc. Florian Neugebauer for all the motivation, guidance and help provided during the period of the Master Thesis.

Questions ?

University of Stuttgart
Hardware Oriented Computer Science
Prof. Dr. rer. nat. habil. Ilia Polian

Thank you!

Manish Mahesh Dalvi

E-Mail manishdalvim@gmail.com

University of Stuttgart
Hardware Oriented Computer Science
Institut für Technische Informatik
Pfaffenwaldring 47
D-70569 Stuttgart
Germany