IOWA STATE UNIVERSITY
College of Engineering

Dependable Computing and Networking Laboratory

Impact of Structural Faults on Neural Network Performance

Krishna Teja Chitty-Venkata and Arun Somani
Iowa State University, Ames, IA, USA
{krishnat, arun}@iastate.edu

ASAP 2019 - The 30th IEEE International Conference on Application-specific Systems, Archiectures and Processors
July 15-17,2019, Cornell Tech, New York

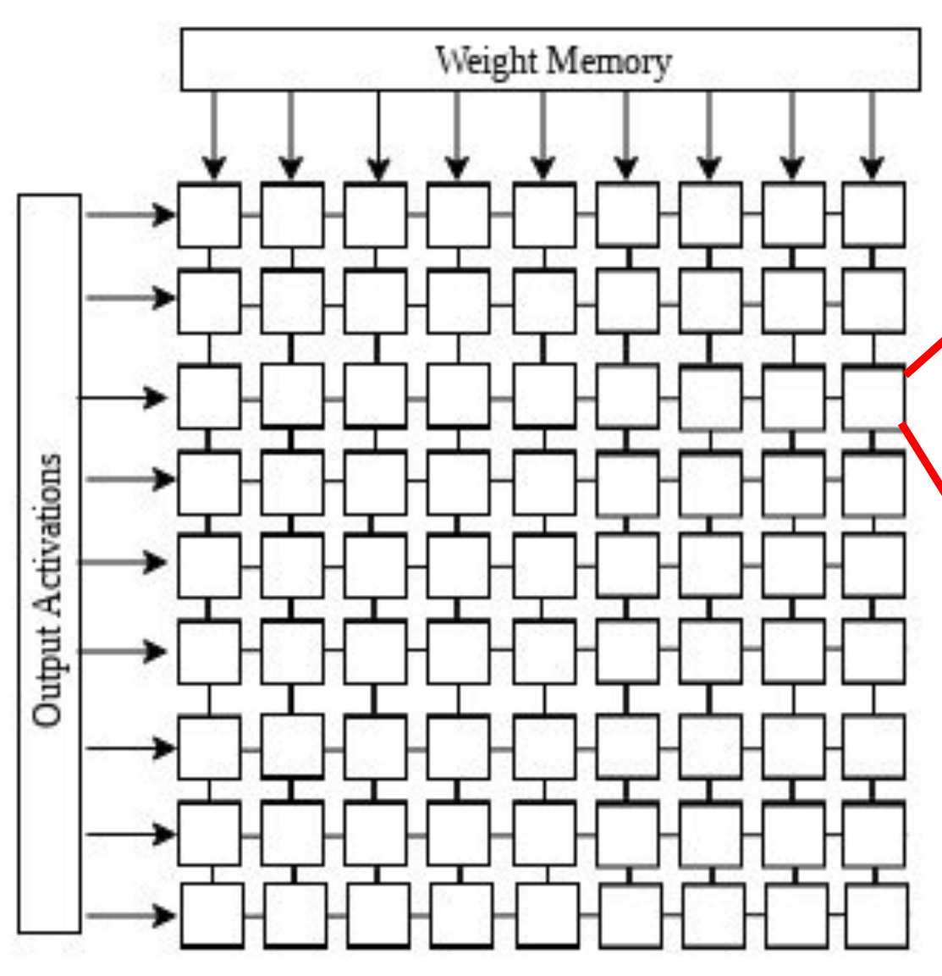
Introduction

Hardware Acceleration of DNN:

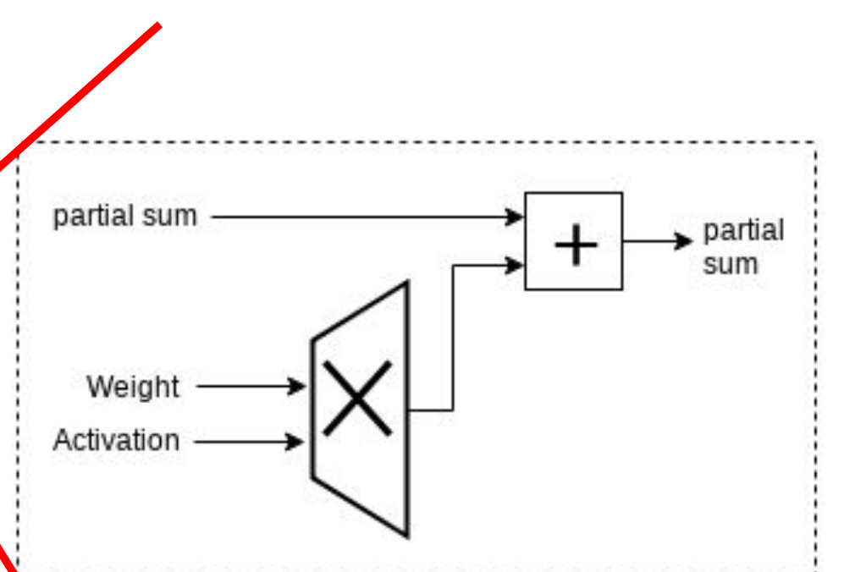
- Specialized hardware architectures, such as a Tensor Processing Unit (TPU) [1], play important role to deliver higher performance in response to increasing size of DNN.
- Systolic Array is the heart of TPU, which is made of series of MAC Units.

Faults in hardware :

- All systems are prone to faults leading to errors
- Faults arise due to various reasons: improper design, environmental factors, manufacturing defects etc
- This work addresses the fault tolerance aspect of DNN accelerators



(Systolic Array)

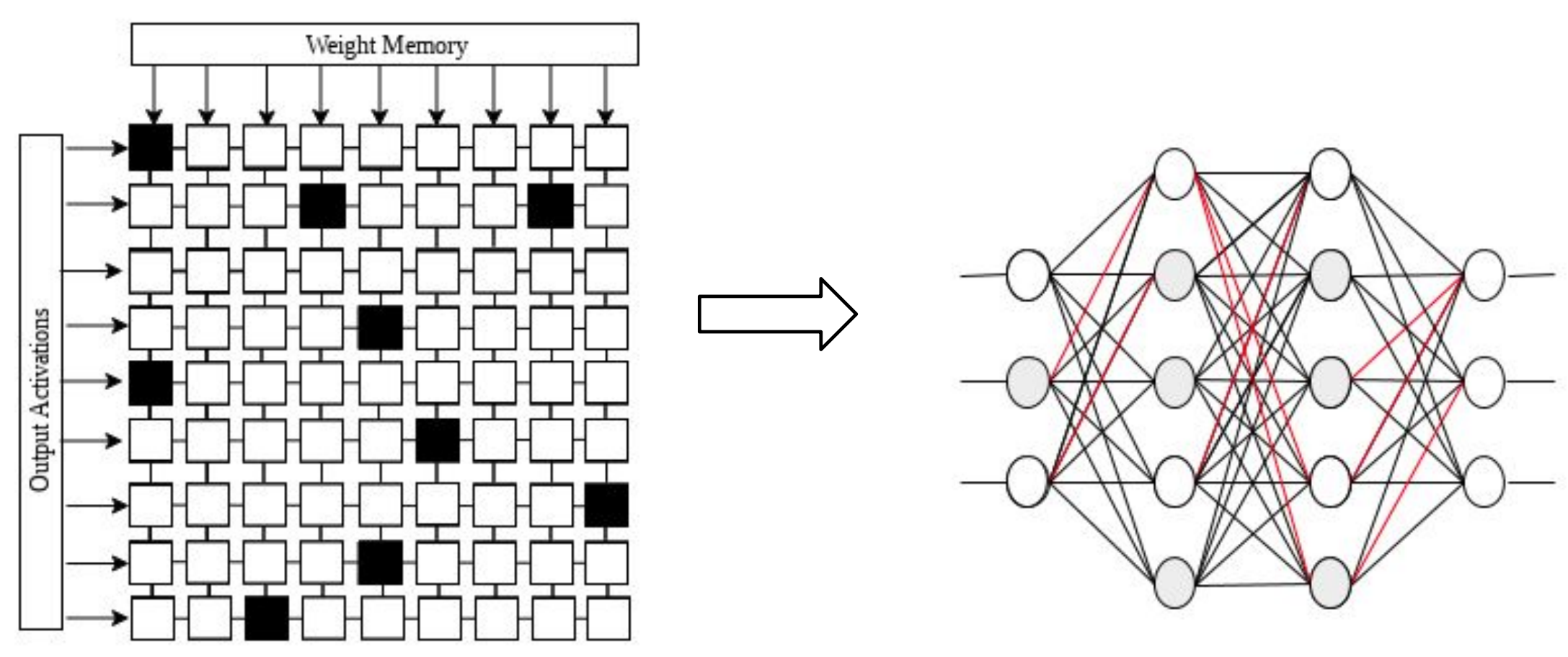


(Multiplication and Accumulation Unit)

Previous Work

Fault Aware Pruning + Retraining:

- Force prune and keep the weights in a DNN to zero which are to be processed at faulty locations and retrain the remaining weights [2]
- Fault Pattern: Random MAC unit failure
- Fails in the case of Column Faults (Our work)



Algorithm

Calculate indices of weight matrix which map on faulty locations
↓
Prune weights at faulty locations
↓
Retrain the remaining weights

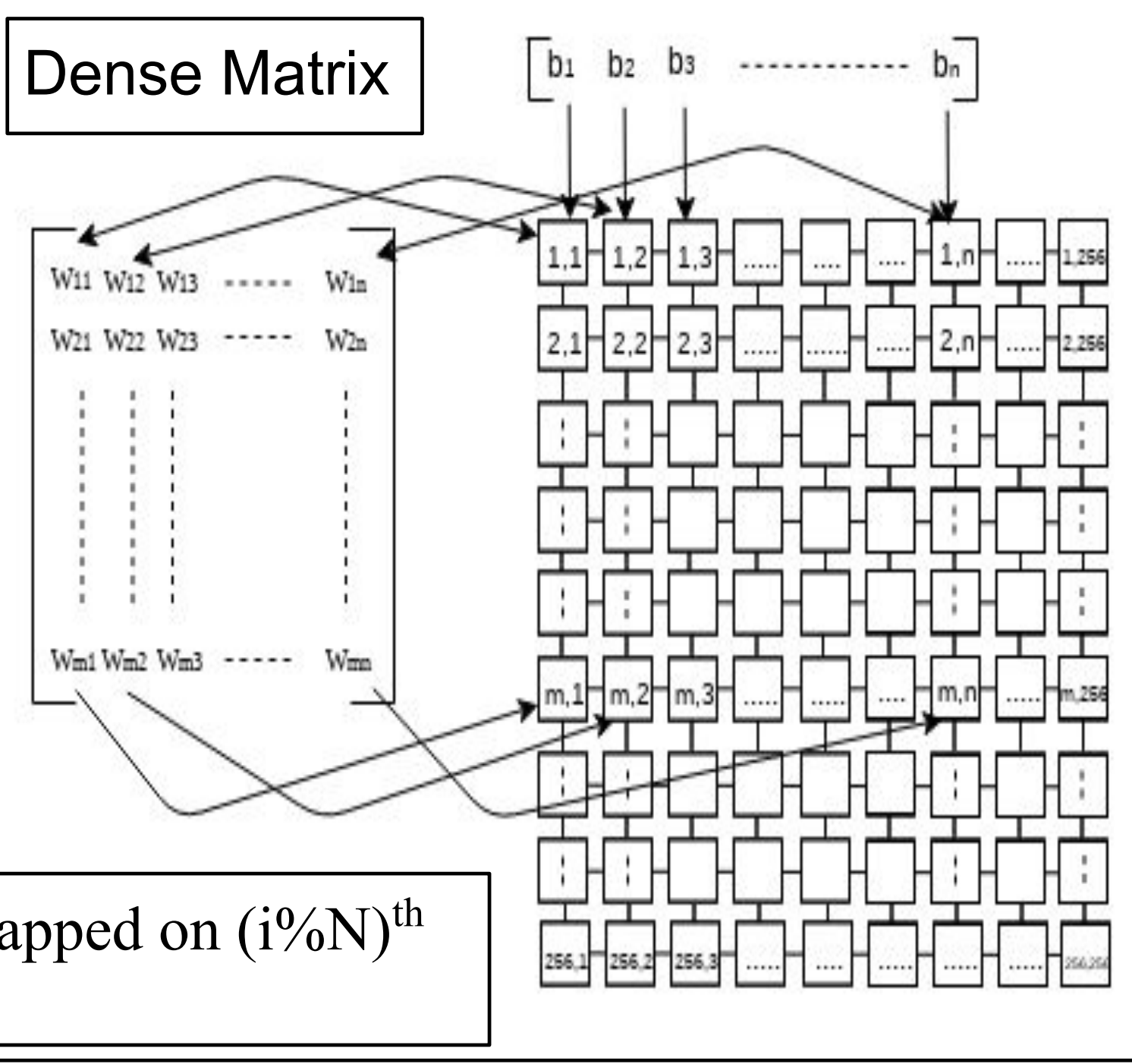
Mapping Policy

Physical Locations on array:

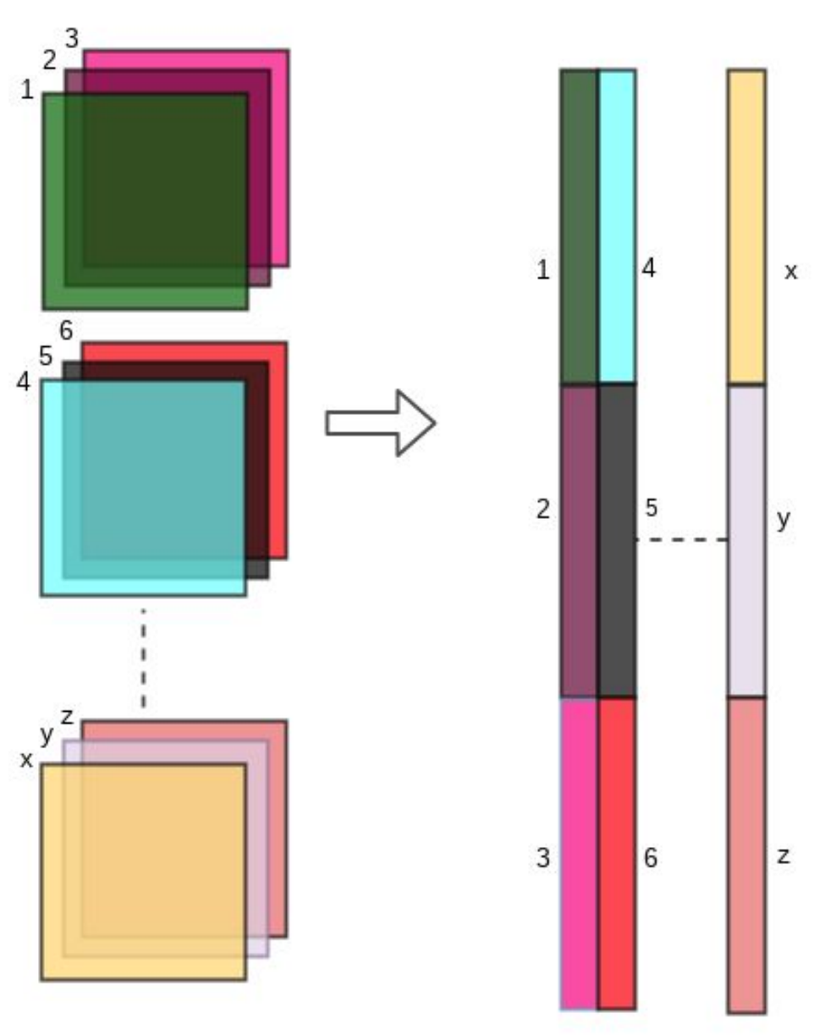
Physical_x() = i%N
Physical_y() = j%N

(i,j) are matrix indices
N = size of array

Dense Matrix



Convolutional Matrices



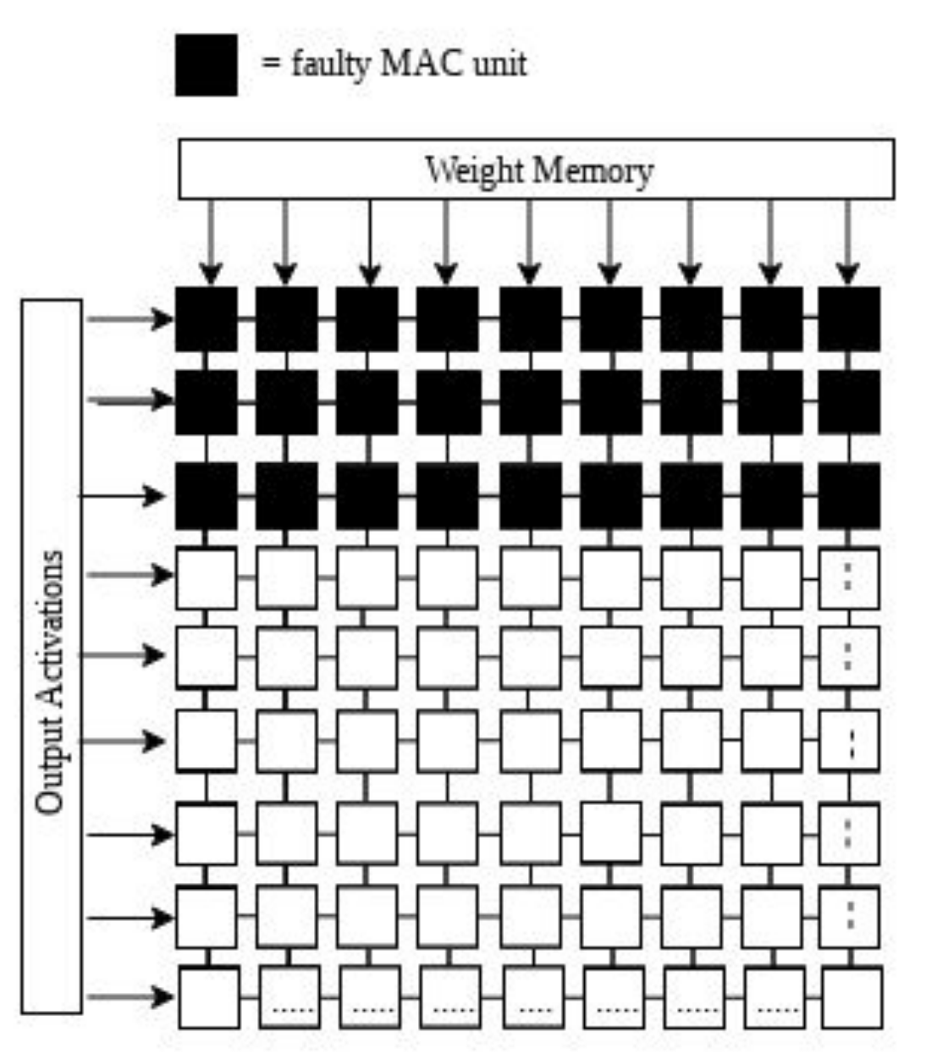
i^{th} Neuron (i^{th} filter) is mapped on $(i\%N)^{th}$ column of systolic array

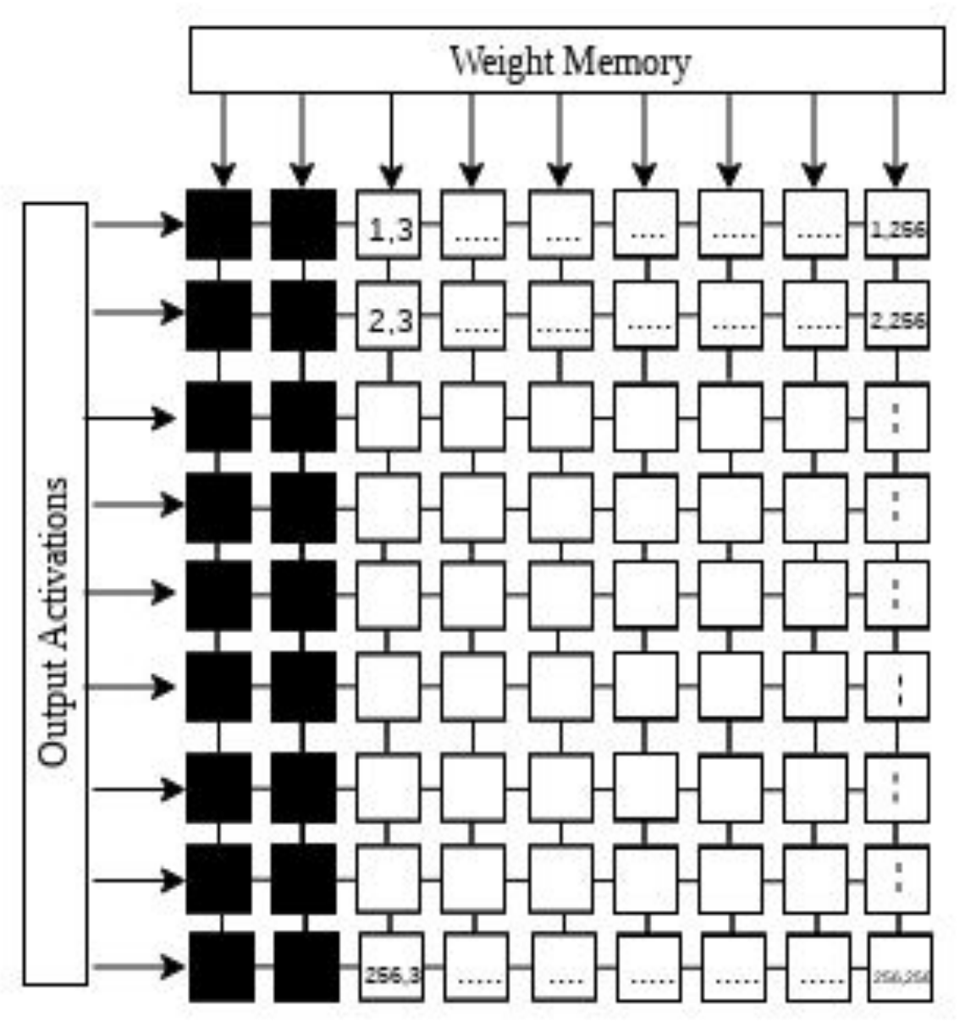
- Convolutional matrices are linearized before mapping onto the systolic array

Fault Model/Pattern

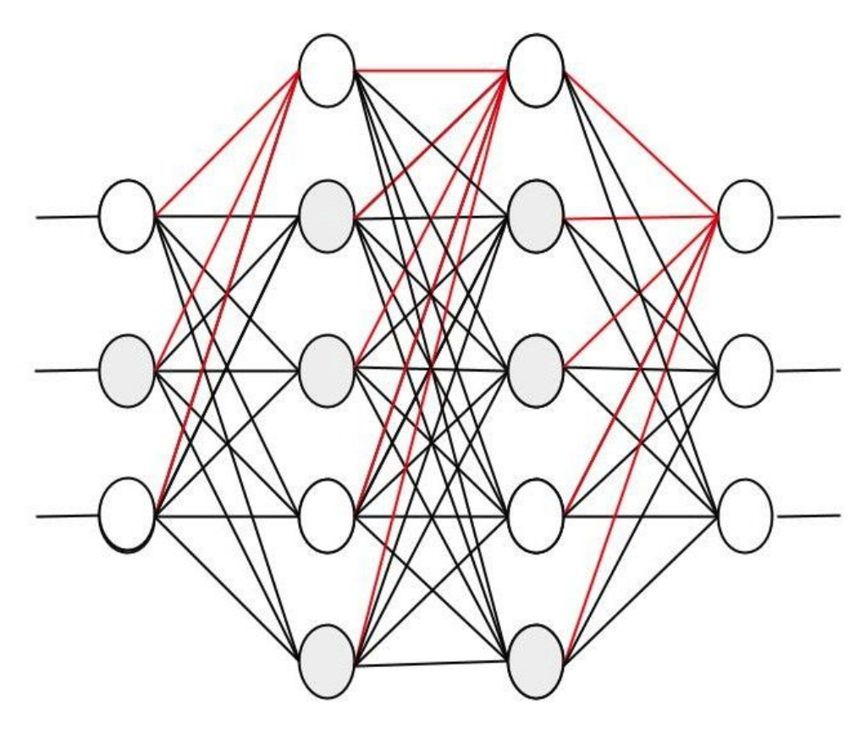
- Fault model: A diagnosed faculty MAC unit behaves as if stuck-at-zero.
- Any other faults (Eg. bit flips) can be converted into stuck-at-zero by using a multiplexer around every MAC
- Faults appear to be "Structured" (along row or column)

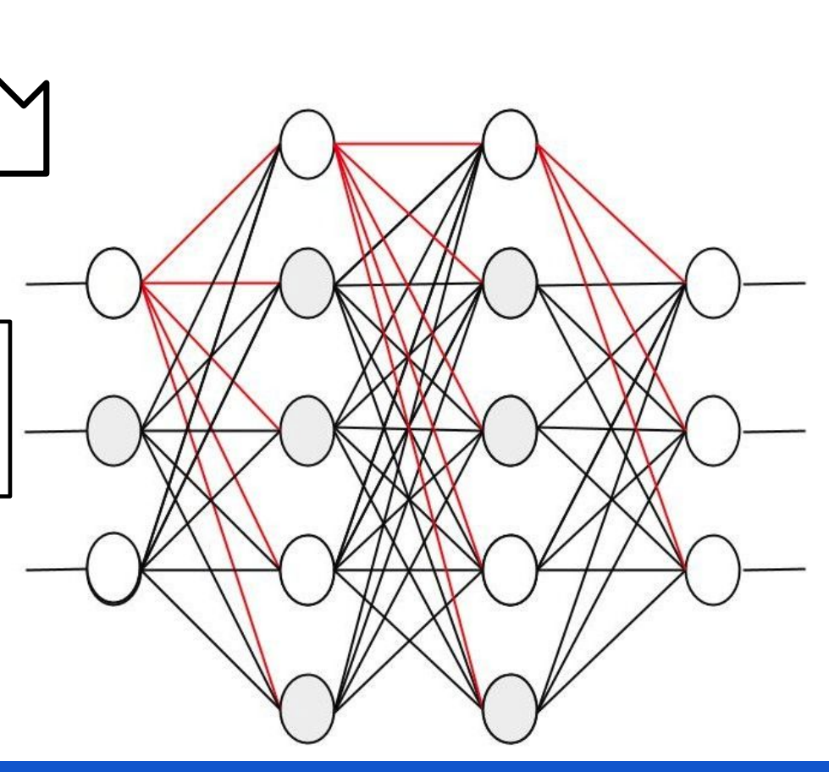
Row Faults — Column Faults





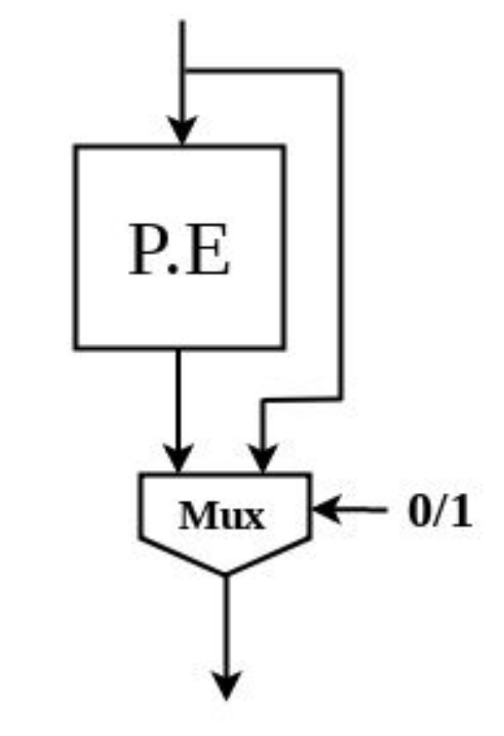
Output Neuron is affected





Input Neuron is affected

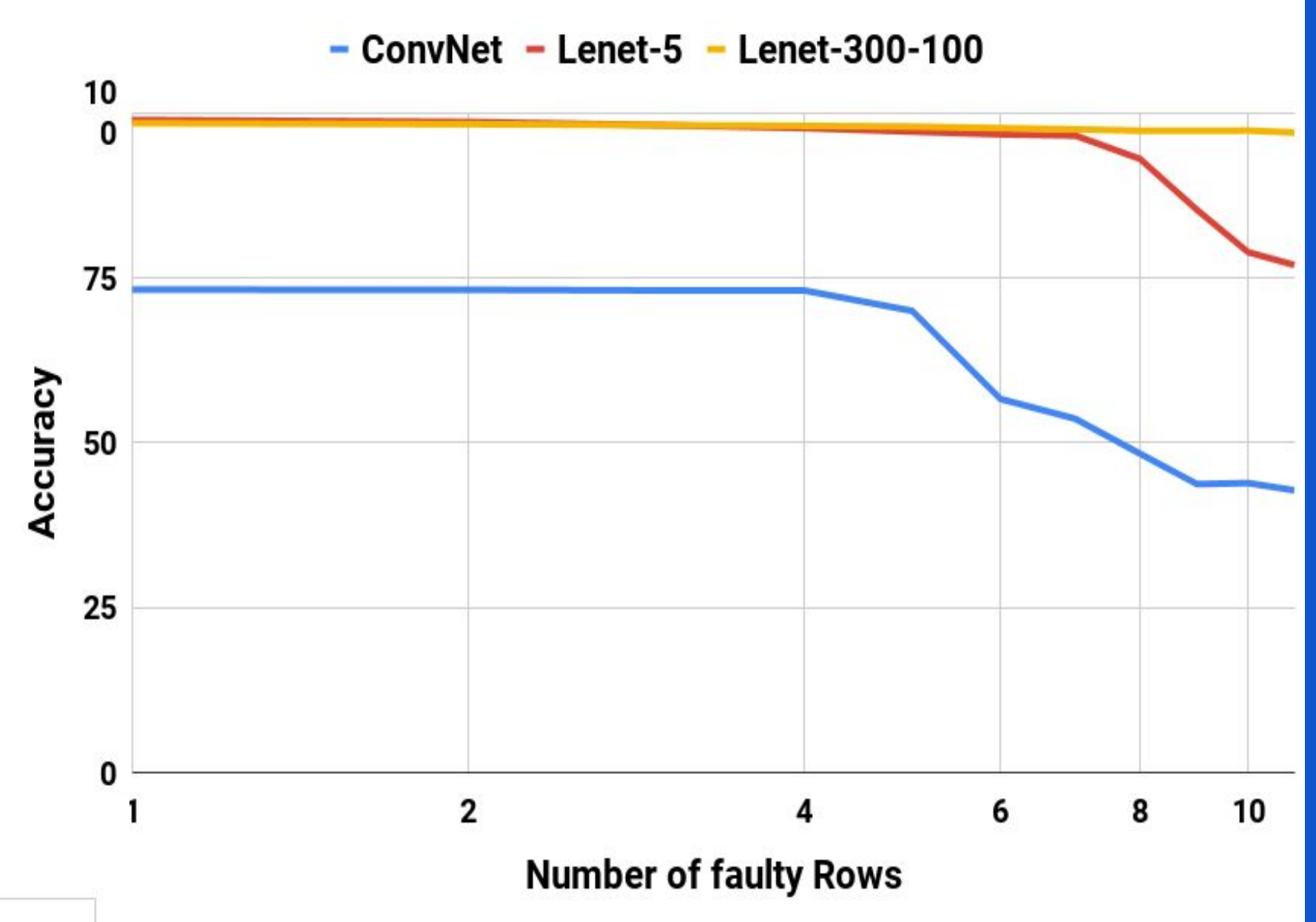
Cause of Faults:
Row and Column faults might be caused due to the failure of data bus which propagates weights and activations into the array

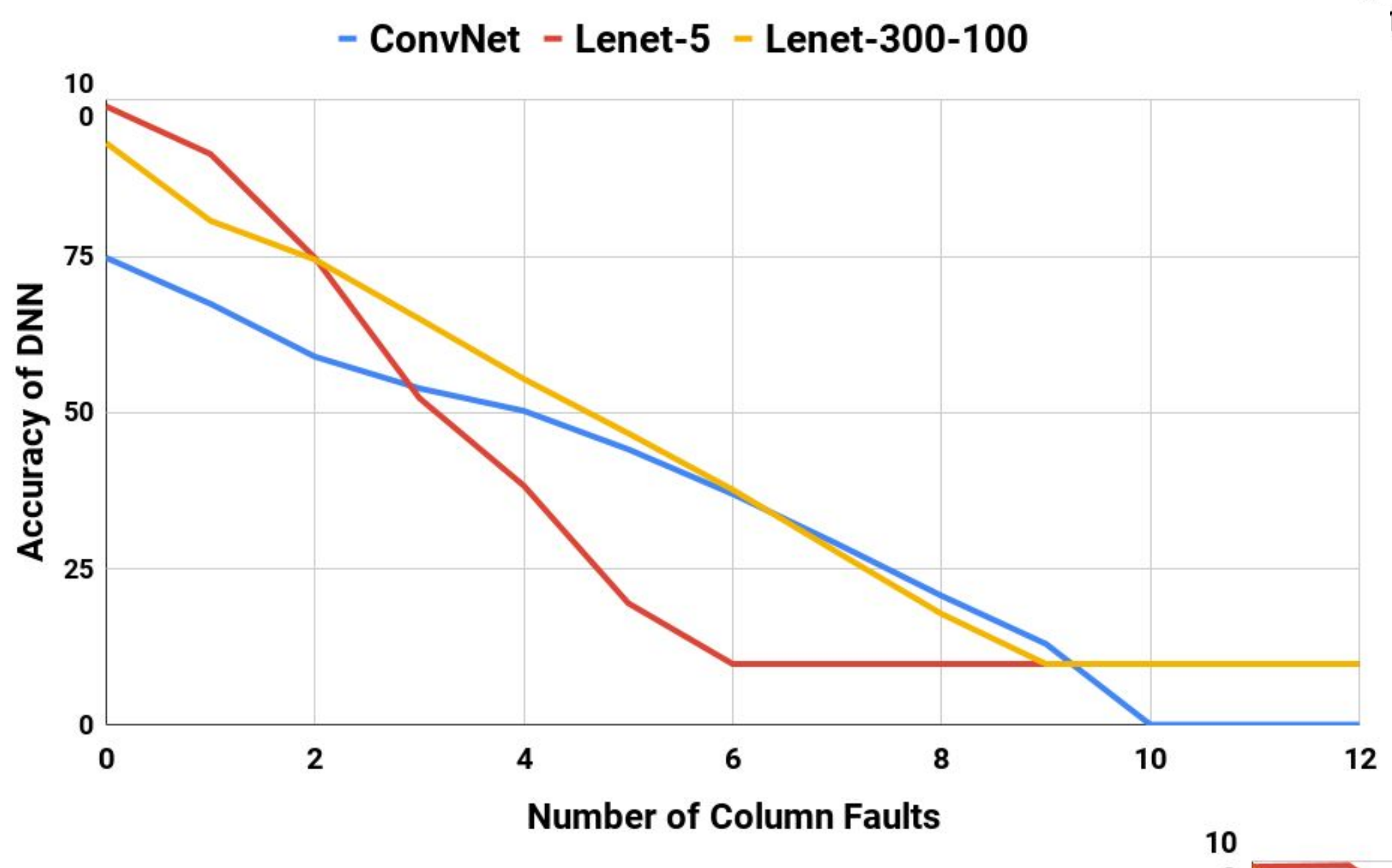


(Multiplexer to bypass the faulty node)

Impact of Faults

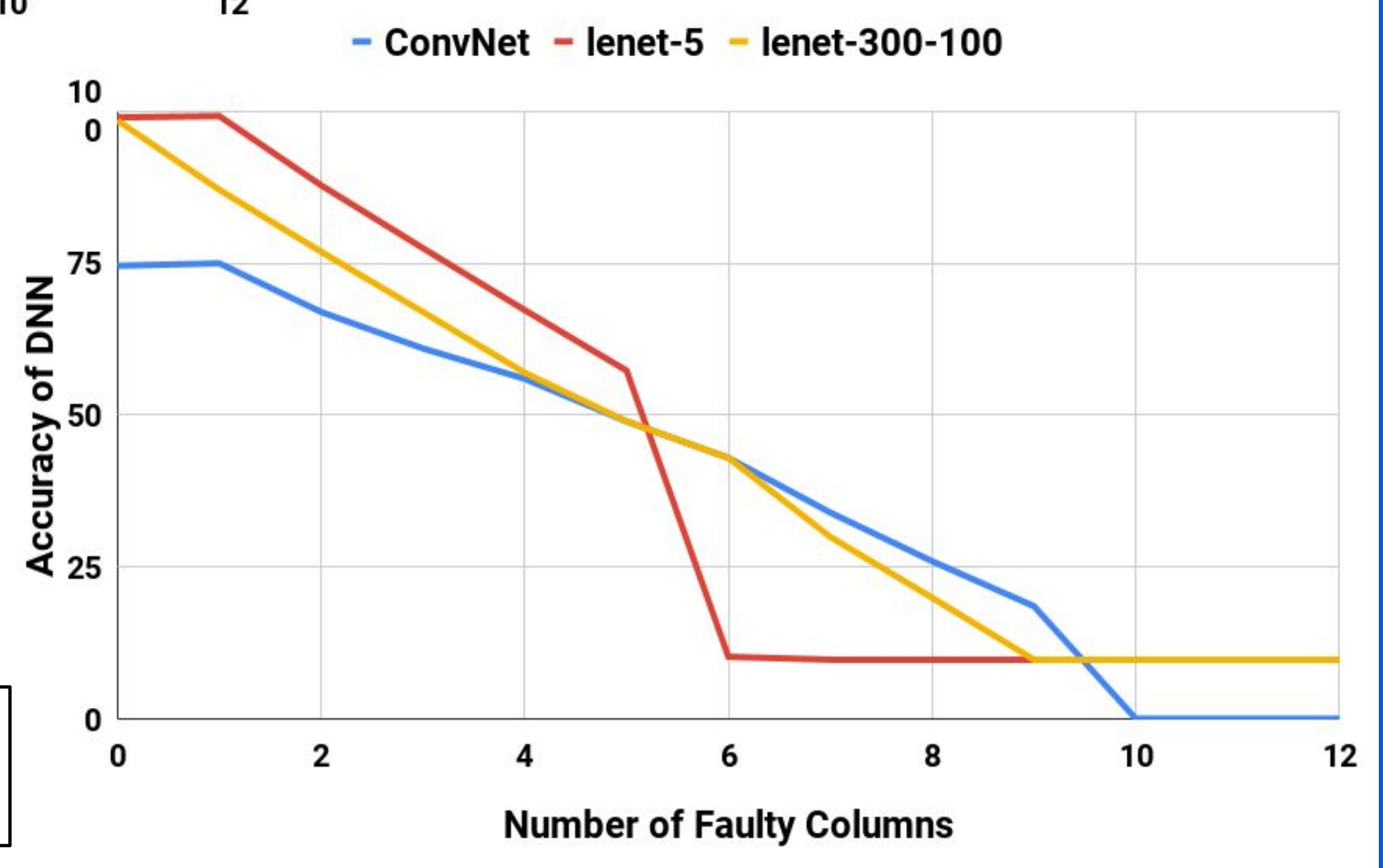
- DNNs are resistant to row faults till an extent
- With n faulty rows, M%n input neurons are completely pruned from the network
- The results of previous work [2] can be applied to restore the accuracy in the case of row faults, but requires additional retraining of the entire network





- Column faults have very high impact on network accuaracy (if the column is used)
- With every column fault, the class associated with it is completely eliminated leading to significant accuracy drop

- The figure on right shows the effect of retraining the network with pruning the weights on faulty nodes
- The results clearly shows that earlier work [2] fails in this scenario
- i^{th} Output Neuron (i^{th} class) is removed from network if $(i\%N)^{th}$ column is faulty

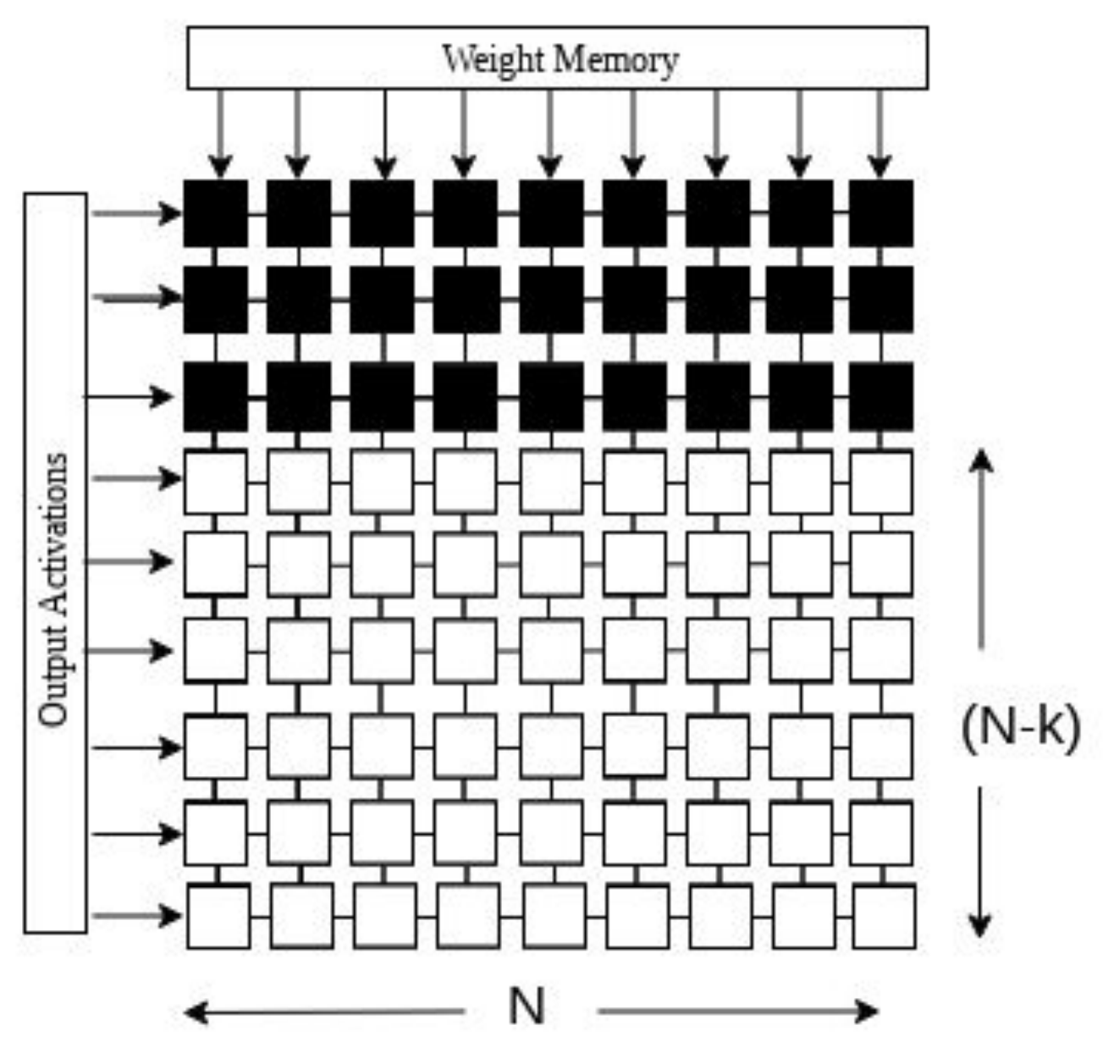


Experiments are done on Lenet-300-100, Lenet-5 (MNIST) and ConvNet (Cifar10) networks

Mitigation Strategies - Row Faults

Low Row Faults

- Low row faults near input have limited impact on the DNN accuracy
- Weights can be mapped at the same locations (i%N, j%N)
- No additional retraining or computational overhead are needed



High Row Faults - Array Reduction

- For these faults, it is recommended to reduce the size of array along the row, i.e. reduce N x N array to N-k x N
- Updated Mapping : (i%(N-k)+k, j%N)
- Faulty rows are completely avoided

Array Reduction along row

Overhead:	
Lenet-300-100	8.6%
Lenet-5	12%
ConvNet	37%

Mitigation Strategies - Column Faults

Low Column Faults - Matrix Transpose

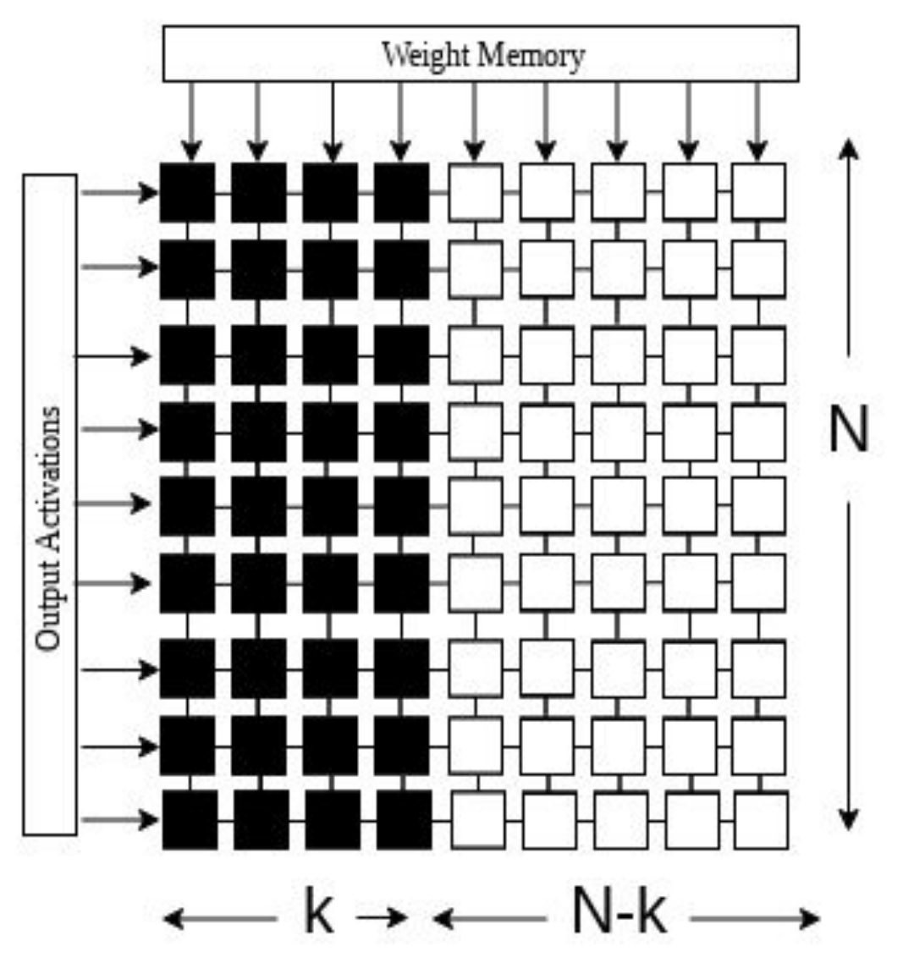
- We know that low "row faults" have smaller impact than low "column faults"
- Suggest to transpose the matrix to convert rows into columns and vice versa
- Updated Mapping : (j%N, i%N)
- All MAC units can be used even in the case of faults
- No retraining is required

$$\begin{bmatrix} 3 & 2 & 1 & 4 & 7 \\ 4 & 2 & 1 & 2 & 9 \\ 9 & 8 & 5 & 3 & 6 \\ 7 & 1 & 0 & 9 & 3 \\ 0 & 4 & 2 & 8 & 7 \end{bmatrix}^T \Rightarrow \begin{bmatrix} 3 & 4 & 9 & 7 & 0 \\ 2 & 2 & 8 & 1 & 4 \\ 1 & 1 & 5 & 0 & 2 \\ 4 & 2 & 3 & 9 & 8 \\ 7 & 9 & 6 & 3 & 7 \end{bmatrix}$$

Matrix Transpose

High Column Faults - Array Reduction

- Reduction along Column
- Reduce N x N array to N x (N-k)
- Updated Mapping : (i%N, j%(N-k)+k)
- Overhead: Same as row faults



Array Reduction along column

References

[1] P.Jouppi et al.,"In-datacenter performance analysis of a tensor processing unit,"in Computer Architecture (ISCA), 2017 ACM/IEEE 44th Annual International Symposium on IEEE, 2017, pp. 1-12
[2] J.J Zhang, T.Gu,K.Basu and S.Garg, "Analyzing and mitigating the impact of permanent faults on a systolic array based neural network accelerator," in VLSI Test Symposium (VTS), 2018 IEEE 36th .IEEE, 2018, pp. 1–6.