UCLA Samueli School of Engineering

Understanding Error Propagation in Deep Learning Neural Network (DNN) Accelerators and Applications

Original Authors:

G. Li, S. Hari, et al.
UBC, NVIDIA

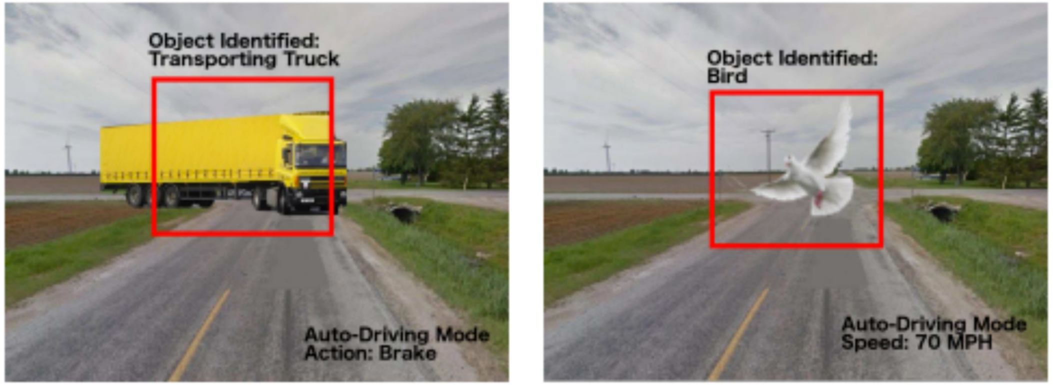
Presented By:

Parangat Mittal
Electrical and Computer Engineering

Problem

Silent Data Corruption in Deep Neural Networks.

- Generally caused by high-energy particles striking electronic devices
- Most common impact is single bit-flips in DRAMs, causing change in values
- Leads to incorrect inference results



DNN Accelerators are deployed to increase throughput and efficiency for real-time inference, but incorrect results defeats the whole purpose.

Traditional Solutions

Triple Modular Redundancy (TMR) for Execution Units.

- Replicate hardware thrice and combine the results to get the final result
- Too costly and large overheads in real-time systems
- Workaround: Replicate vulnerable instructions instead of whole programs
- DNN programs have highly parallel and repetitive structure with typically 5-6 unique instructions

Error Correcting Codes (ECC) for DRAMs.

- Error Correcting Codes detects single-bit errors and makes the recovery of original message/data easier after corruption
- Protecting small buffers through ECC incurs high overheads and increases the cost of real-time systems

Proposed Solutions

Characterize and Quantify Error Propagation across DNNs

- SDCs in different DNNs using different data types
- Sensitive bit positions in different data types
- Error propagation layer by layer in DNN

Mitigate SDCs in DNNs, particularly in DNN Accelerators

- Symptom-based Error Detector (SED), implemented in Software
- Selective Latch Hardening (SLH), implemented in Hardware

Methodology

Environment

- DNN Simulation in *Tiny CNN*
- Fault injection at C lines (mapped to microarchitecture)

Fault Model

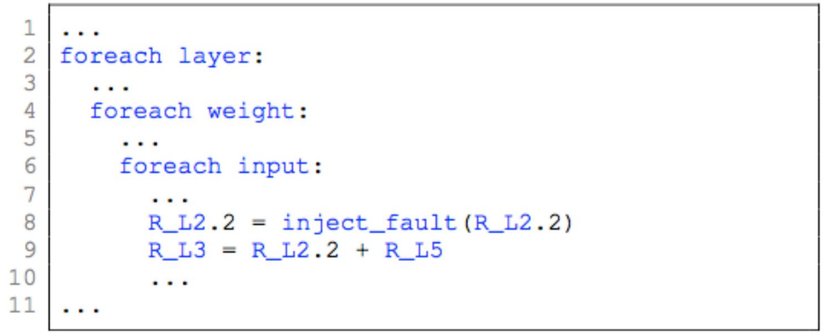
- Single-bit flips in data path and buffers
- No error in combinational / control units

Evaluation Metric

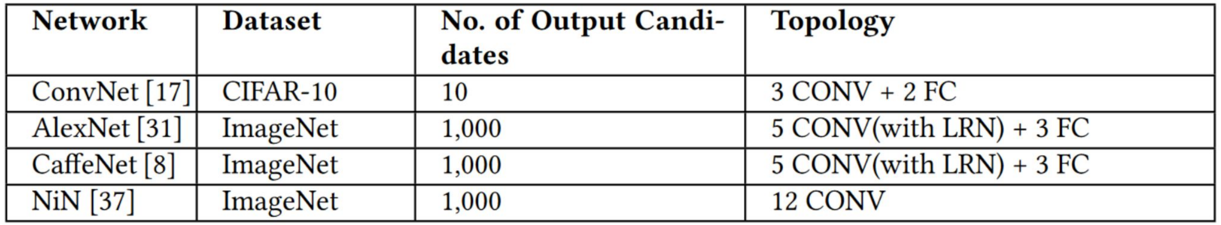
- Mismatch with fault-free results
- SDC : Silent Data Corruption
- SDC-1 : Top ranked element predicted

Sample Space

- Four popular pre-trained DNNs



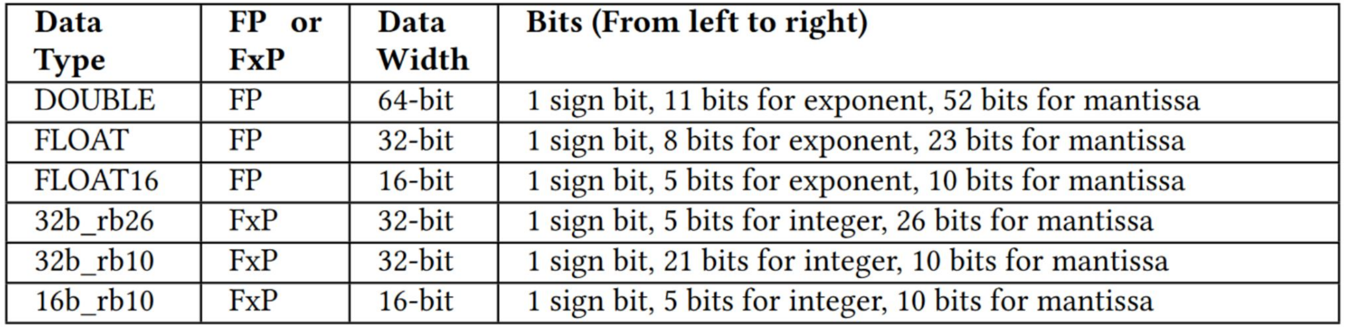
```
...
foreach layer:
  ...
  foreach weight:
    ...
    foreach input:
      ...
      R_L2.2 = inject_fault(R_L2.2)
      R_L3 = R_L2.2 + R_L5
      ...
...
```



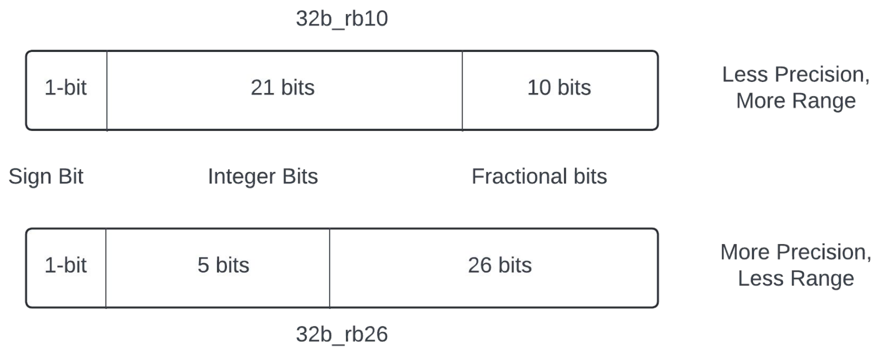
Network	Dataset	No. of Output Candi-dates	Topology
ConvNet [17]	CIFAR-10	10	3 CONV + 2 FC
AlexNet [31]	ImageNet	1,000	5 CONV(with LRN) + 3 FC
CaffeNet [8]	ImageNet	1,000	5 CONV(with LRN) + 3 FC
NiN [37]	ImageNet	1,000	12 CONV

Data Types

Data Types used in this study: Float, Float16, 32b_rb26, 32b_rb10, 16b_rb10

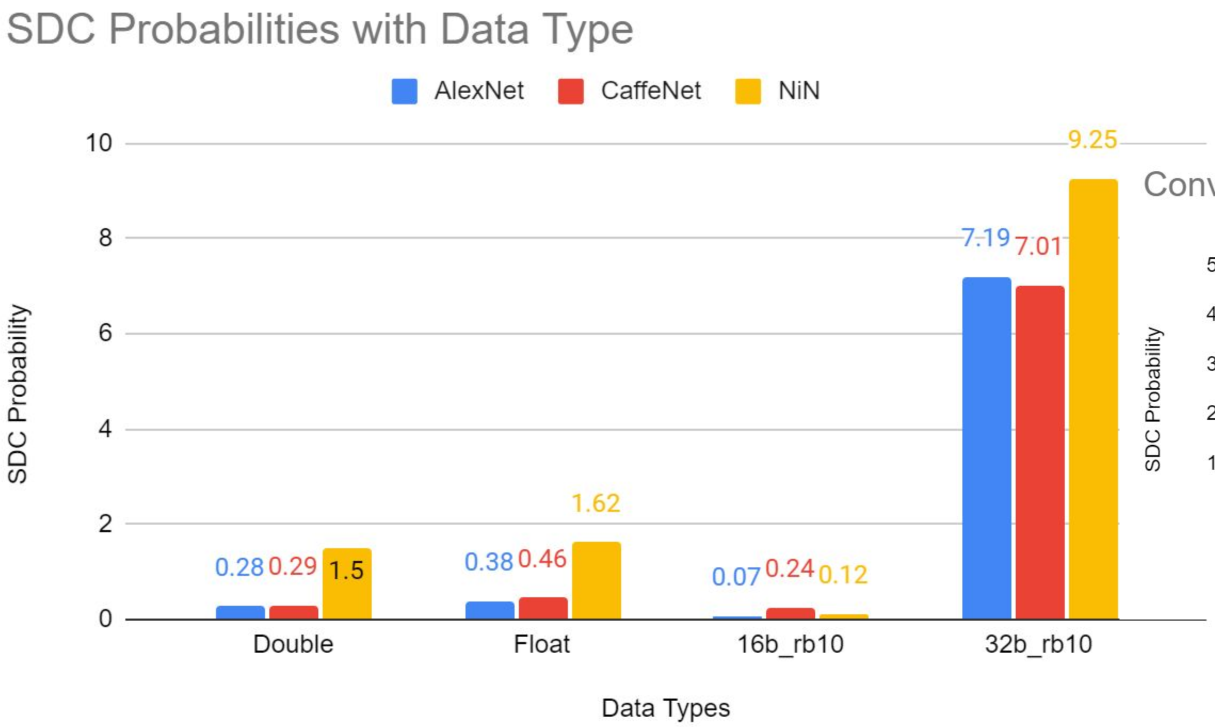


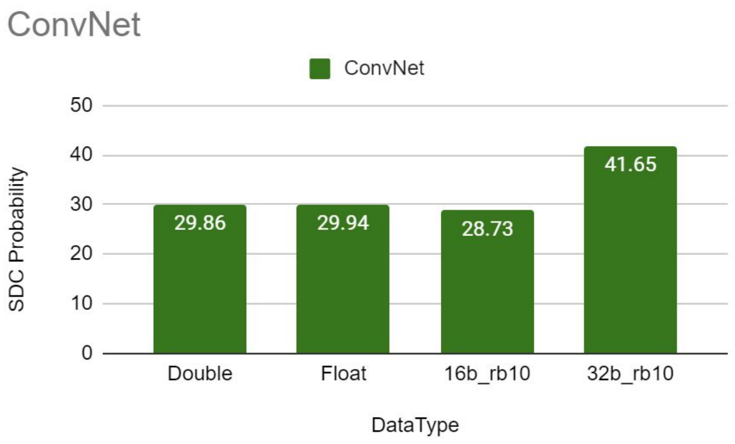
Data Type	FP or FxP	Data Width	Bits (From left to right)
DOUBLE	FP	64-bit	1 sign bit, 11 bits for exponent, 52 bits for mantissa
FLOAT	FP	32-bit	1 sign bit, 8 bits for exponent, 23 bits for mantissa
FLOAT16	FP	16-bit	1 sign bit, 5 bits for exponent, 10 bits for mantissa
32b_rb26	FxP	32-bit	1 sign bit, 5 bits for integer, 26 bits for mantissa
32b_rb10	FxP	32-bit	1 sign bit, 21 bits for integer, 10 bits for mantissa
16b_rb10	FxP	16-bit	1 sign bit, 5 bits for integer, 10 bits for mantissa



Characterization

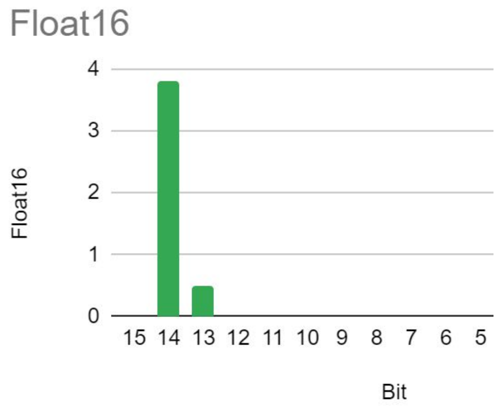
SDCs in Different DNNs Using Different Data Types

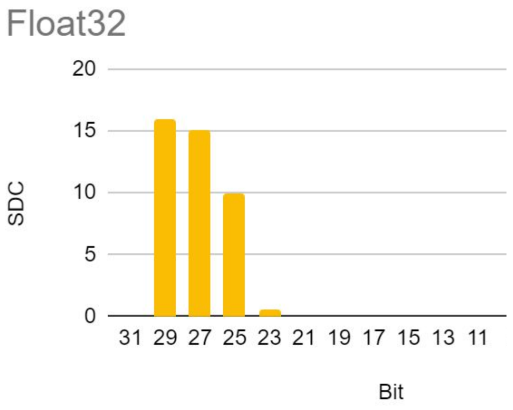


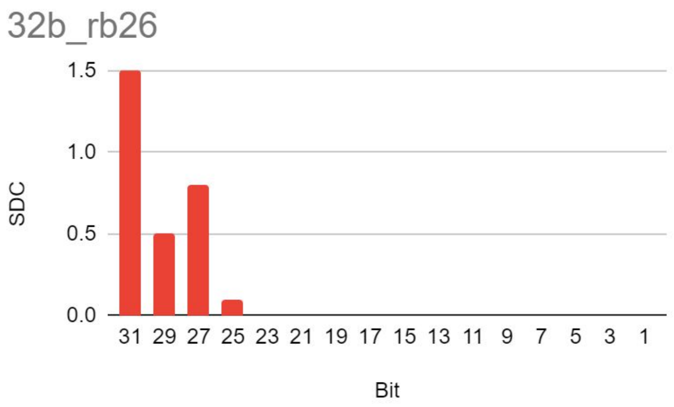


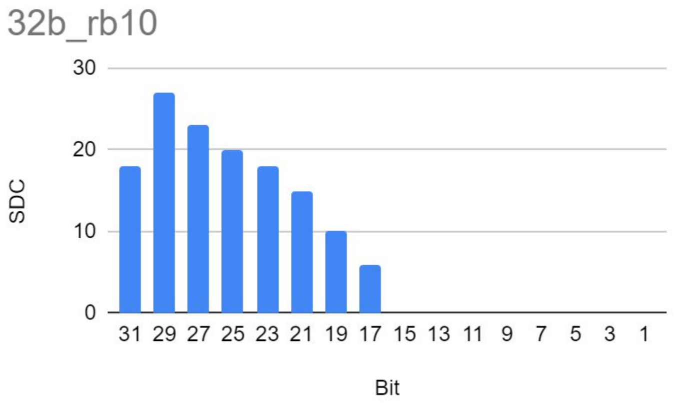
Characterization

Sensitive Bit Positions in Different Data Types







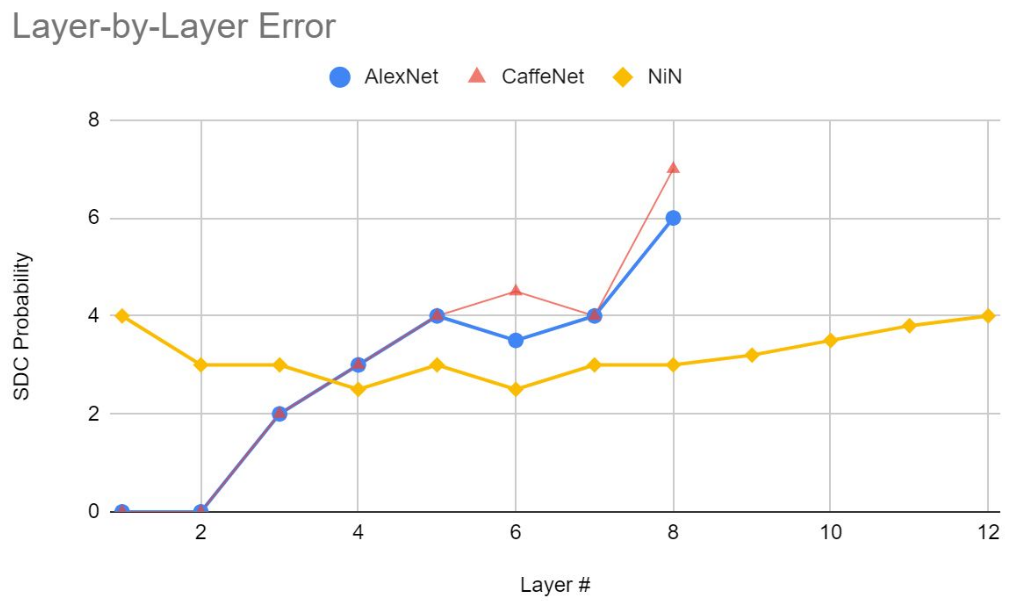


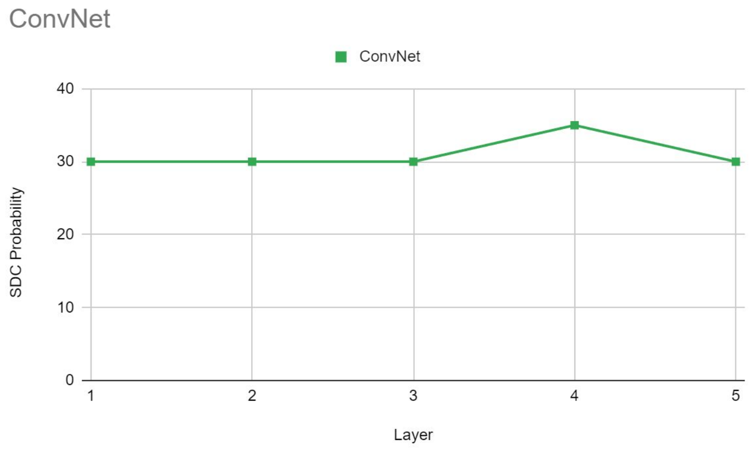
FP: Only certain exponent bits are vulnerable to SDCs

FxP: Higher Order Bits are vulnerable to SDCs; More value range exposes more bits to vulnerability

Characterization

Error Propagation Layer by Layer in DNN





- Faults occurring in earlier layers have higher chances of being propagated to deeper layers.
- First two have lesser SDC probabilities since these have Normalization layers .

Mitigation

Data Type Choice.

- Data type should provide just enough range and precision required
- ***But***, Accelerators use one-size-fits-all approach and use data types to cover wide range
- Detect possible errors and stop them -> ***Symptom-based Error Detector (SED)***

- With correct data type, out-of-range value SDCs are mitigated
- SDCs still hidden in normal value ranges.
- Make sensitive bits more resistant to errors -> ***Selective Latch Hardening (SLH)***

Normalization Layers.

- Add more normalization layers to normalize faulty values to adjacent fault-free values.
- LRN (Local Response Normalization) Layers make the network more resilient.

Mitigation

Symptom-Based Error Detector. (SED)

Use specific properties of system-under-fault to detect anomalies

Symptom: Large activation values likely cause an SDC

Implementation:

1. Learning Phase - Derive fault-free value ranges for each layers, with an additional 10% cushion to prevent false alarms
2. Deployment Phase - At the end of each layer, the data is stored in global buffer as input for next layer. SED checks for erogenous values asynchronously in the buffer to avoid runtime overhead.

Mitigation

Selective Latch Hardening. (SLH)

- Adding redundant circuitry to sequential latches to make them less error-sensitive
- Comes with area and power overheads and varying levels of protection

Implementation:

- To counter overheads, harden only the most sensitive error-prone bits
- For optimal protection, use a combination of latch hardening techniques

Strike Suppression	Tailor the design and layout of a latch to increase the critical charge that is required to cause an error
Redundant Node	Uses two interleaved nodes to store data and rely on internal feedback to restore the correct data in the presence of an error
Triplicated Module	Same bit is stored in three locations and a majority voter is used to correct a bit flip

Discussion

- Fault injection proposed in this research has been done at code-level in a simulated environment. Their accuracy in predicting real-world behavior is limited by the assumptions and simplifications made.

- The trends and methods suggested in this research are based on a specific kind of networks. With more and more complex DNNs, scaling these methods would pose the same challenges as the traditional methods because of high overheads in terms of area and computation.

Q&A

UCLA Samueli School of Engineering