POLYTECHNIQUE MONTRÉAL
UNIVERSITÉ D'INGÉNIERIE

Designing Robust DNN Models That Exploit Energy-Reliability Tradeoffs

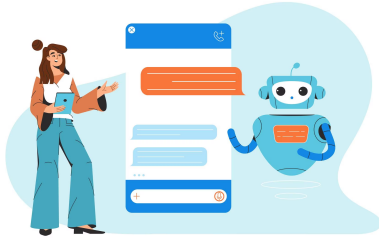
François Leduc-Primeau
École Polytechnique de Montréal

Accelerate AI Workshop 2023 – Thursday May 4, 2023

Energy consumption of AI in perspective

- Large language models
  - 4 Wh / query $\times$ 3 queries/min
    = 720 W

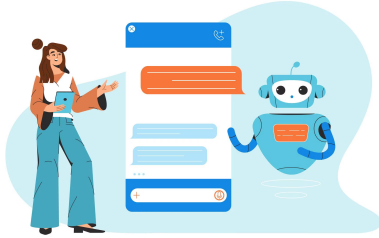
[A. S. Luccioni et al. Estimating the Carbon Footprint of BLOOM, a 176B Parameter Language Model, 2022]



Energy consumption of AI in perspective

- Large language models
  - 4 Wh / query $\times$ 3 queries/min
    = 720 W

[A. S. Luccioni et al. Estimating the Carbon Footprint of BLOOM, a 176B Parameter Language Model, 2022]

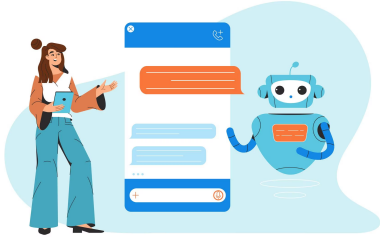


- Self-driving cars
  - NVIDIA “Robotaxi” platform:
    2$\times$ Orin SOCs + 2$\times$ Ampere GPUs
    = 800 W (TDP)

Energy consumption of AI in perspective

- Large language models
  - 4 Wh / query $\times$ 3 queries/min
    = 720 W

[A. S. Luccioni et al. Estimating the Carbon Footprint of BLOOM, a 176B Parameter Language Model, 2022]



- Self-driving cars
  - NVIDIA “Robotaxi” platform:
    2$\times$ Orin SOCs + 2$\times$ Ampere GPUs
    = 800 W (TDP)

- Human brain: $\approx$ 20 W

Digital IC Context

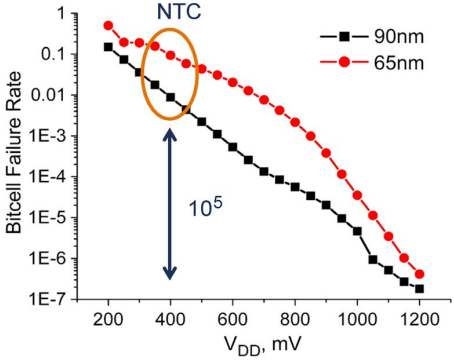
Issues:

- Energy efficiency of CMOS is not improving
- Data movement consumes most of the energy

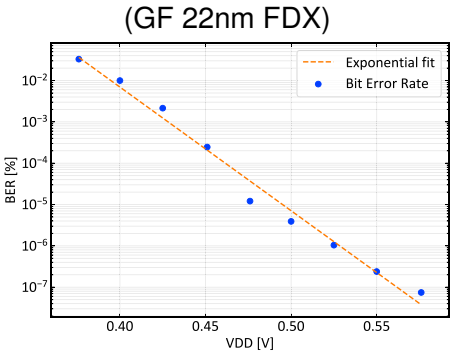
(Partial) solutions:

- Reduce data movement
- Increase on-chip memory
- Use specialized hardware
- In-memory computing

SRAM Energy *vs* Reliability



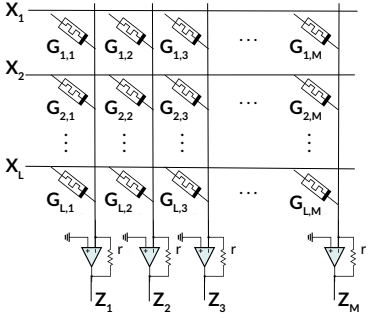
[R. G. Dreslinski et al., 2010]



[Di Mauro et al., TCAS-I, 2020]

Computing in Memory with Memristor Crossbars

Memristor crossbar :

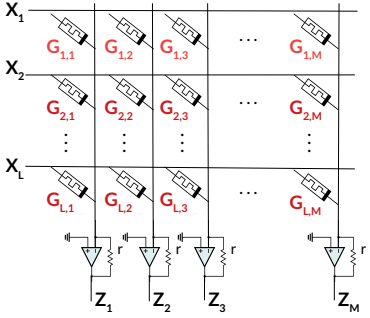


$Z_j = r \sum_{i=1}^{L} G_{i,j} X_i$

- $G_{i,j}$ is the conductance value
- X_i is the input voltage
- Z_j is the output voltage

Computing in Memory with Memristor Crossbars

Memristor crossbar :

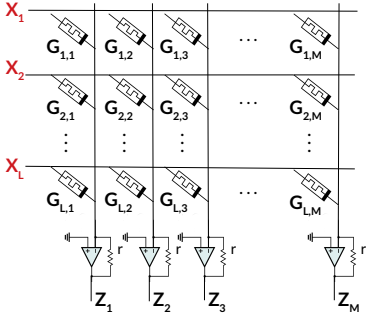


$Z_j = r \sum_{i=1}^{L} G_{i,j} X_i$

- $G_{i,j}$ is the conductance value
- X_i is the input voltage
- Z_j is the output voltage

Computing in Memory with Memristor Crossbars

Memristor crossbar :

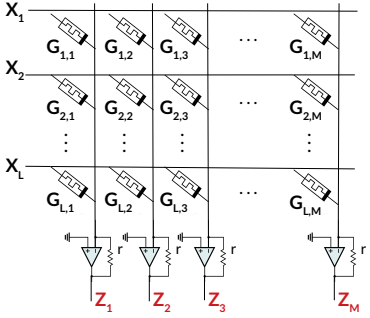


$$Z_j = r \sum_{i=1}^{L} G_{i,j} X_i$$

- $G_{i,j}$ is the conductance value
- X_i is the input voltage
- Z_j is the output voltage

Computing in Memory with Memristor Crossbars

Memristor crossbar :

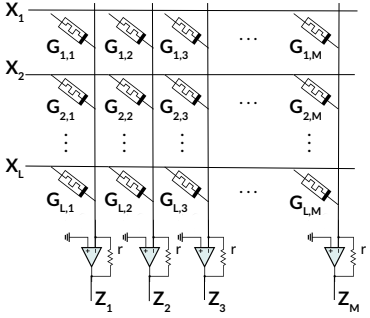


$Z_j = r \sum_{i=1}^{L} G_{i,j} X_i$

- $G_{i,j}$ is the conductance value
- X_i is the input voltage
- Z_j is the output voltage

Computing in Memory with Memristor Crossbars

Memristor crossbar :



$Z_j = r \sum_{i=1}^{L} G_{i,j} X_i$

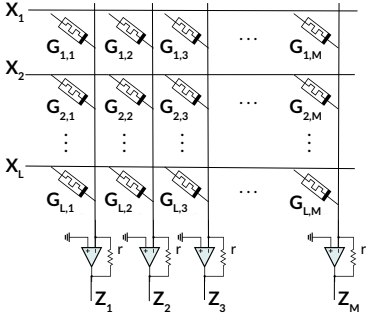
- $G_{i,j}$ is the conductance value
- X_i is the input voltage
- Z_j is the output voltage

Advantages:

- High density
- Non volatile
- Operates at low voltage
- Computation within memory

Computing in Memory with Memristor Crossbars

Memristor crossbar :



$Z_j = r \sum_{i=1}^{L} G_{i,j} X_i$

- $G_{i,j}$ is the conductance value
- X_i is the input voltage
- Z_j is the output voltage

Advantages:

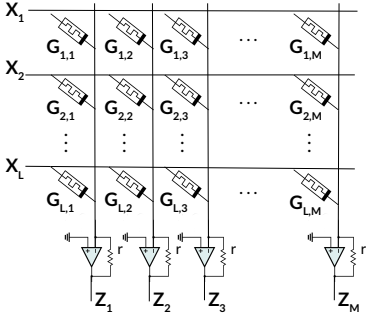
- High density
- Non volatile
- Operates at low voltage
- Computation within memory

Challenges:

- Writing can be slow
- Cannot have a negative resistance
- Device-to-device and cycle-to-cycle variations

Computing in Memory with Memristor Crossbars

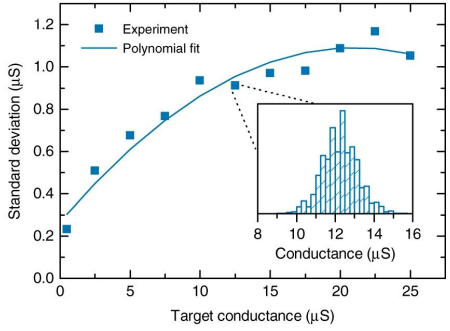
Memristor crossbar :



$$Z_j = r \sum_{i=1}^{L} G_{i,j} X_i$$

- $G_{i,j}$ is the conductance value
- X_i is the input voltage
- Z_j is the output voltage

The conductance values are noisy



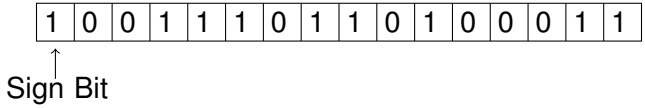
[Source: Joshi et al. Accurate deep neural network inference using computational phase-change memory. Nature Communications, 2020.]

Outline

Outline

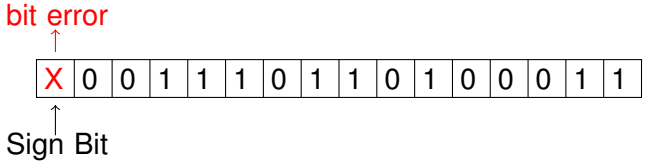
System Model

- Energy-reliability tradeoff: $p(\eta) = e^{-a\eta}$
  (η normalized energy, p bit flip probability)
- "Bit-masking" faults:



System Model

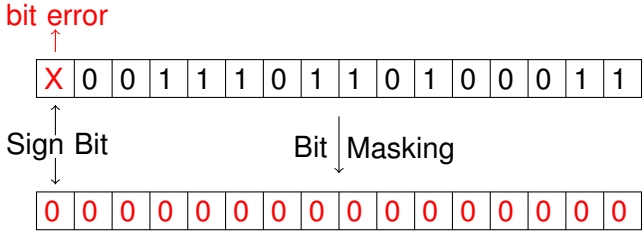
- Energy-reliability tradeoff: $p(\eta) = e^{-a\eta}$
  (η normalized energy, p bit flip probability)
- "Bit-masking" faults:



fault is detected on the sign bit of a value

System Model

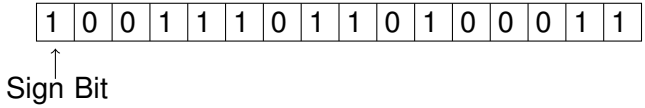
- Energy-reliability tradeoff: $p(\eta) = e^{-a\eta}$
  (η normalized energy, p bit flip probability)
- "Bit-masking" faults:



value replaced by zero

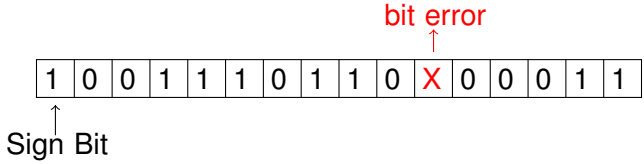
System Model

- Energy-reliability tradeoff: $p(\eta) = e^{-a\eta}$
  (η normalized energy, p bit flip probability)
- "Bit-masking" faults:



System Model

- Energy-reliability tradeoff: $p(\eta) = e^{-a\eta}$
  (η normalized energy, p bit flip probability)
- "Bit-masking" faults:



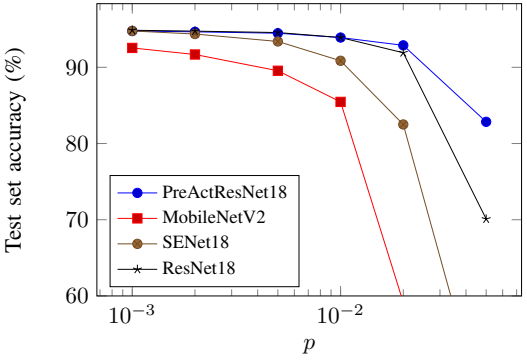
fault is detected on any other bit of a value

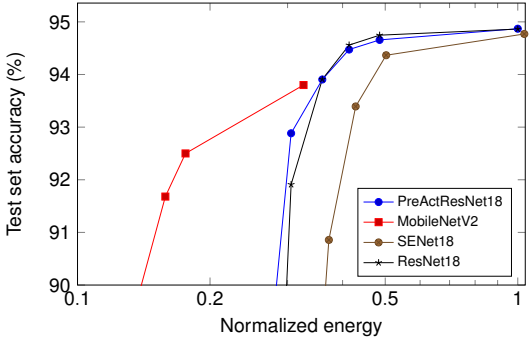
System Model

- Energy-reliability tradeoff: $p(\eta) = e^{-a\eta}$
  (η normalized energy, p bit flip probability)
- "Bit-masking" faults:

Affected bit value is replaced with the sign bit

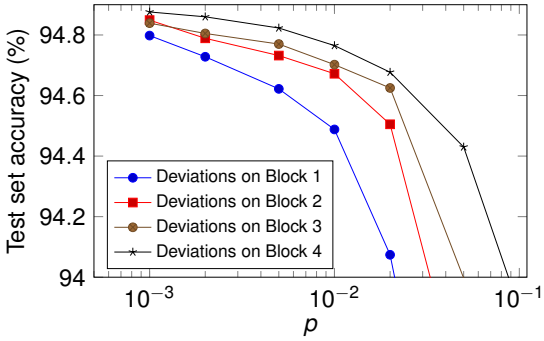
DNN architecture comparison (CIFAR-10)





G. B. Hacene, et al. Training modern deep neural networks for memory-fault robustness. In IEEE ISCAS, 2019.

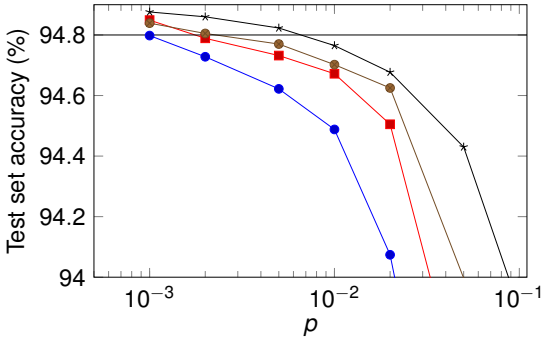
Sensitivity of each block (group of layers)



Block	Nb param.
B1	$1.5 \cdot 10^5$
B2	$5.2 \cdot 10^5$
B3	$21 \cdot 10^5$
B4	$84 \cdot 10^5$

G. B. Hacene, et al. Training modern deep neural networks for memory-fault robustness. In IEEE ISCAS, 2019.

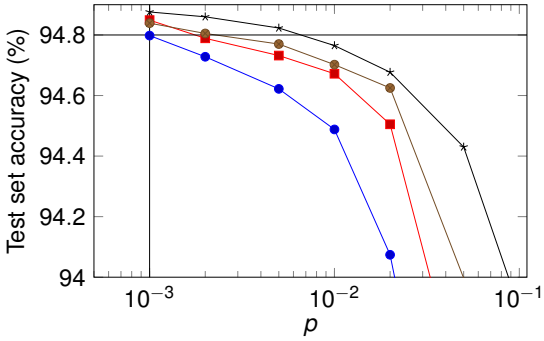
Sensitivity of each block (group of layers)



Block	Nb param.
B1	$1.5 \cdot 10^5$
B2	$5.2 \cdot 10^5$
B3	$21 \cdot 10^5$
B4	$84 \cdot 10^5$

G. B. Hacene, et al. Training modern deep neural networks for memory-fault robustness. In IEEE ISCAS, 2019.

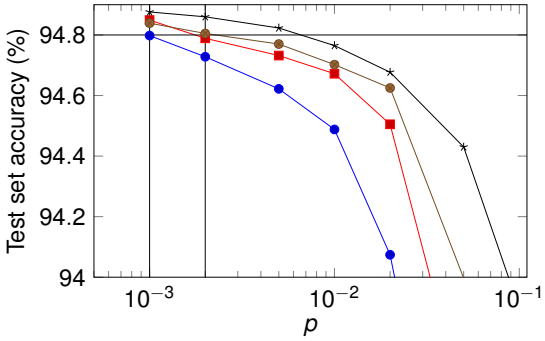
Sensitivity of each block (group of layers)



Block	Nb param.
B1	$1.5 \cdot 10^5$
B2	$5.2 \cdot 10^5$
B3	$21 \cdot 10^5$
B4	$84 \cdot 10^5$

G. B. Hacene, et al. Training modern deep neural networks for memory-fault robustness. In IEEE ISCAS, 2019.

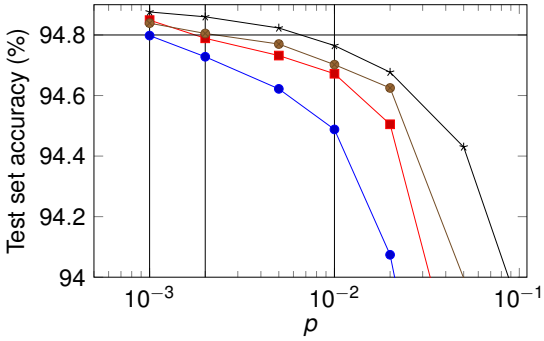
Sensitivity of each block (group of layers)



Block	Nb param.
B1	$1.5 \cdot 10^5$
B2	$5.2 \cdot 10^5$
B3	$21 \cdot 10^5$
B4	$84 \cdot 10^5$

G. B. Hacene, et al. Training modern deep neural networks for memory-fault robustness. In IEEE ISCAS, 2019.

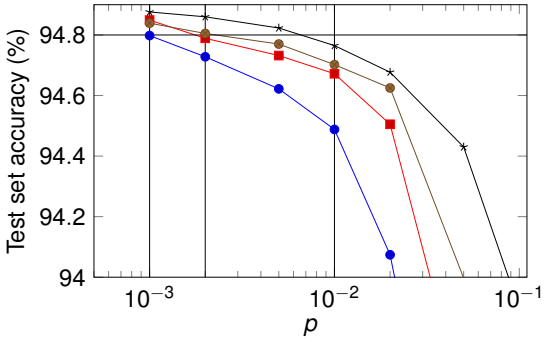
Sensitivity of each block (group of layers)



Block	Nb param.
B1	$1.5 \cdot 10^5$
B2	$5.2 \cdot 10^5$
B3	$21 \cdot 10^5$
B4	$84 \cdot 10^5$

G. B. Hacene, et al. Training modern deep neural networks for memory-fault robustness. In IEEE ISCAS, 2019.

Sensitivity of each block (group of layers)

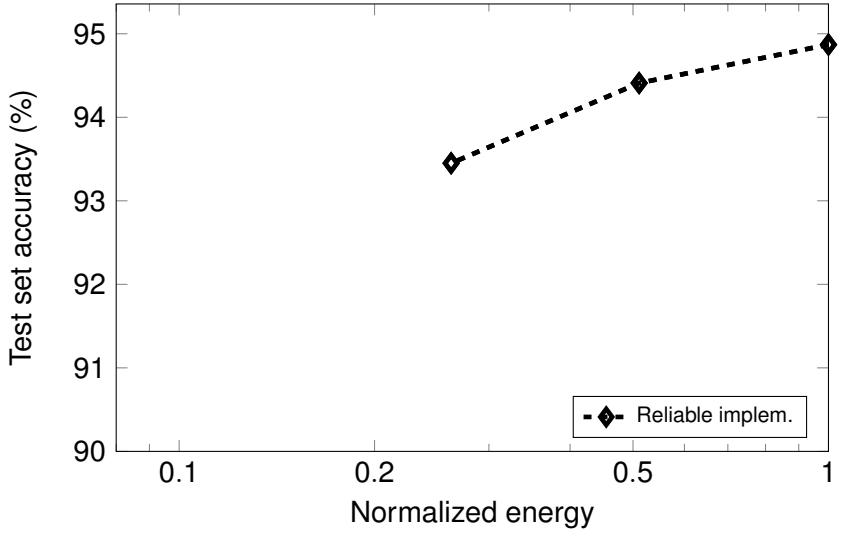


Block	Nb param.
B1	$1.5 \cdot 10^5$
B2	$5.2 \cdot 10^5$
B3	$21 \cdot 10^5$
B4	$84 \cdot 10^5$

$$p_{B_4} = 5p_{B_3} = 5p_{B_2} = 10p_{B_1}$$

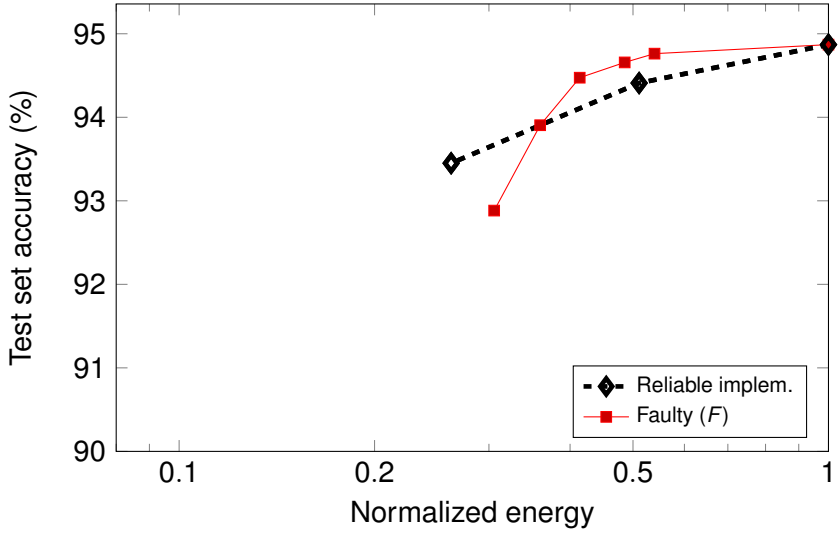
G. B. Hacene, et al. Training modern deep neural networks for memory-fault robustness. In IEEE ISCAS, 2019.

Strategies to improve robustness (CIFAR-10)



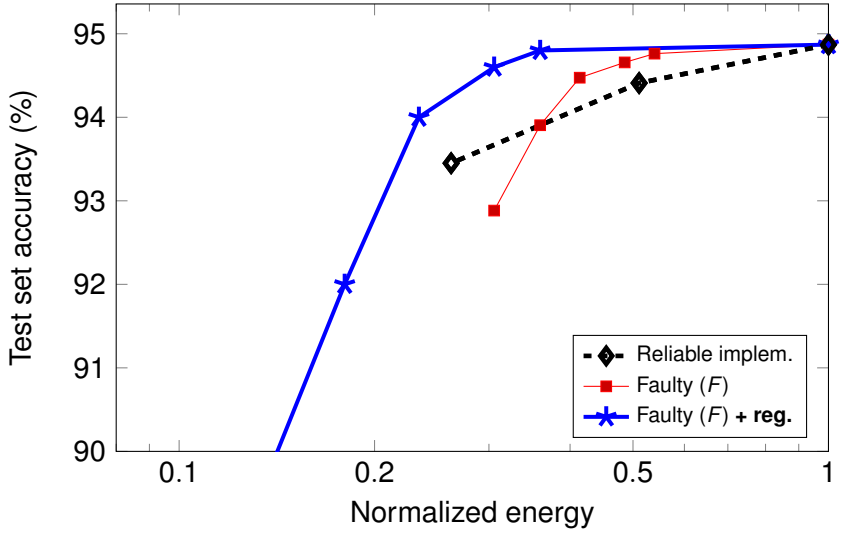
G. B. Hacene, et al. Training modern deep neural networks for memory-fault robustness. In IEEE ISCAS, 2019.

Strategies to improve robustness (CIFAR-10)



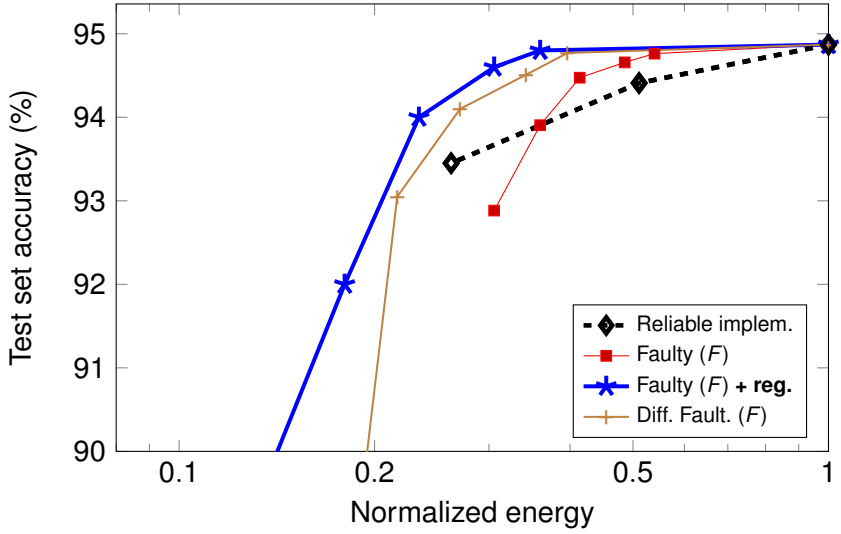
G. B. Hacene, et al. Training modern deep neural networks for memory-fault robustness. In IEEE ISCAS, 2019.

Strategies to improve robustness (CIFAR-10)



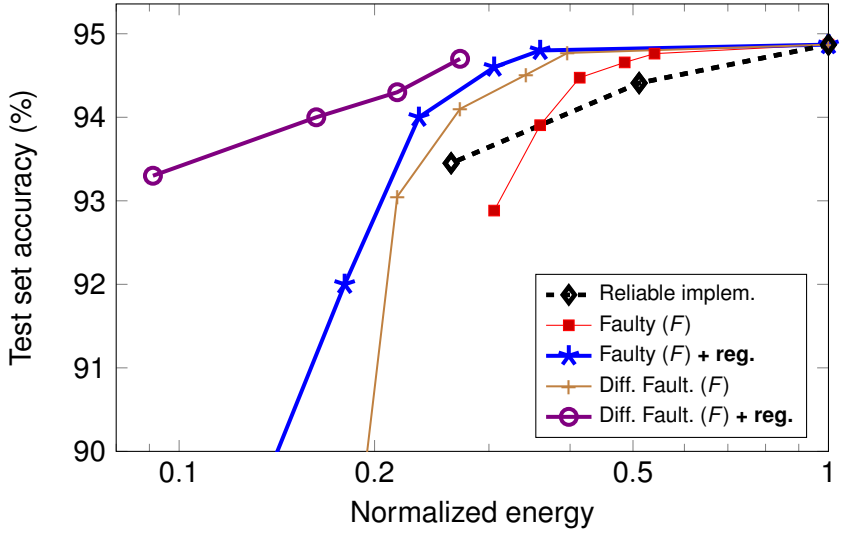
G. B. Hacene, et al. Training modern deep neural networks for memory-fault robustness. In IEEE ISCAS, 2019.

Strategies to improve robustness (CIFAR-10)



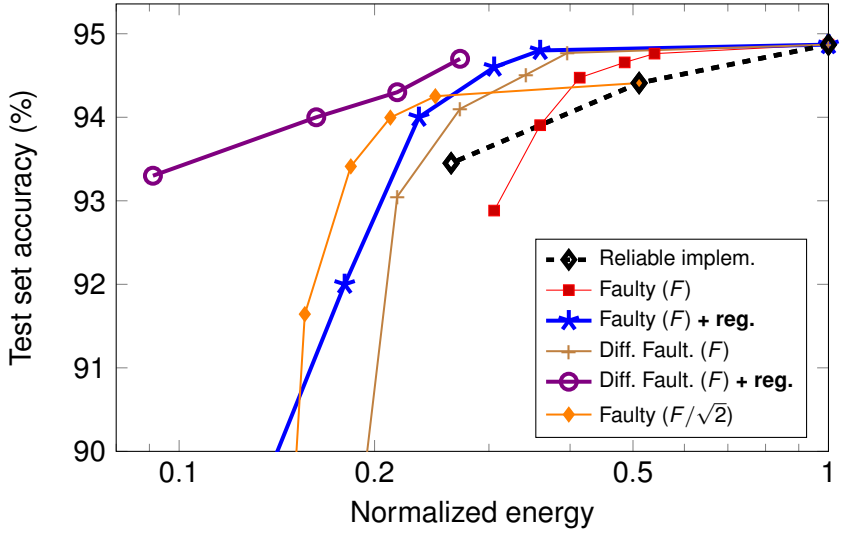
G. B. Hacene, et al. Training modern deep neural networks for memory-fault robustness. In IEEE ISCAS, 2019.

Strategies to improve robustness (CIFAR-10)



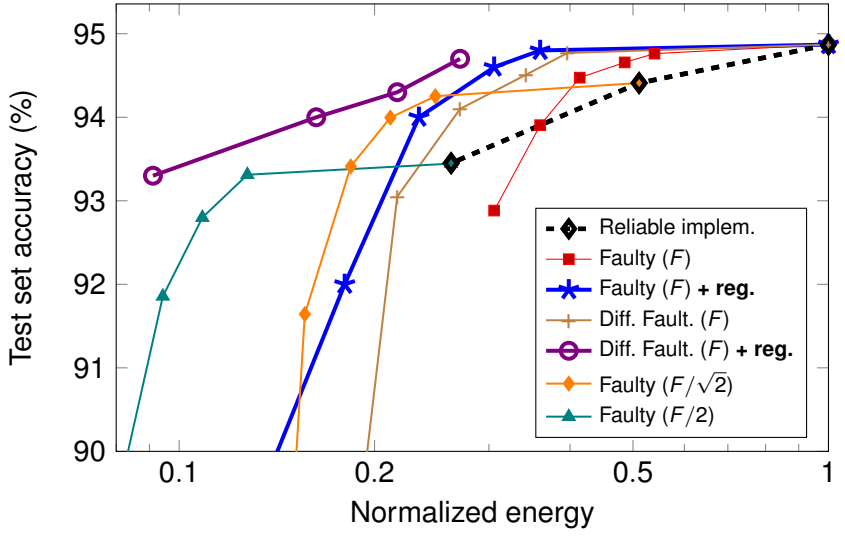
G. B. Hacene, et al. Training modern deep neural networks for memory-fault robustness. In IEEE ISCAS, 2019.

Strategies to improve robustness (CIFAR-10)



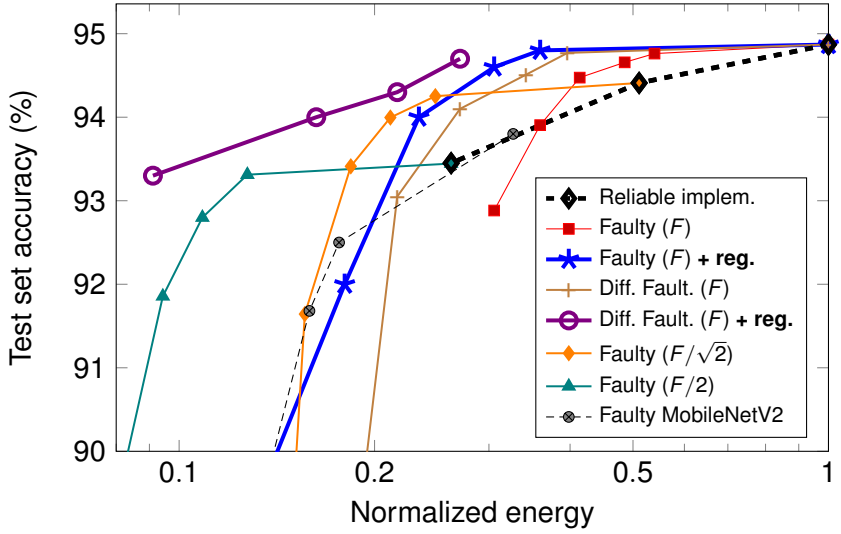
G. B. Hacene, et al. Training modern deep neural networks for memory-fault robustness. In IEEE ISCAS, 2019.

Strategies to improve robustness (CIFAR-10)



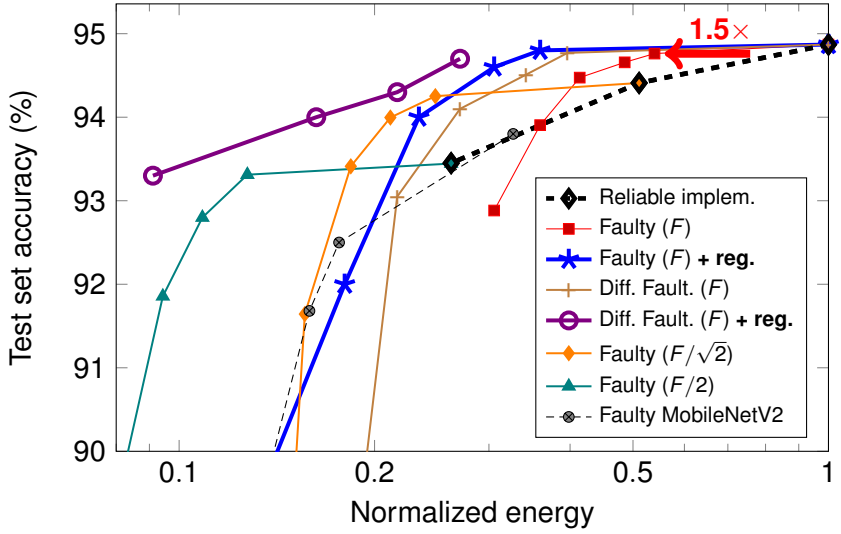
G. B. Hacene, et al. Training modern deep neural networks for memory-fault robustness. In IEEE ISCAS, 2019.

Strategies to improve robustness (CIFAR-10)



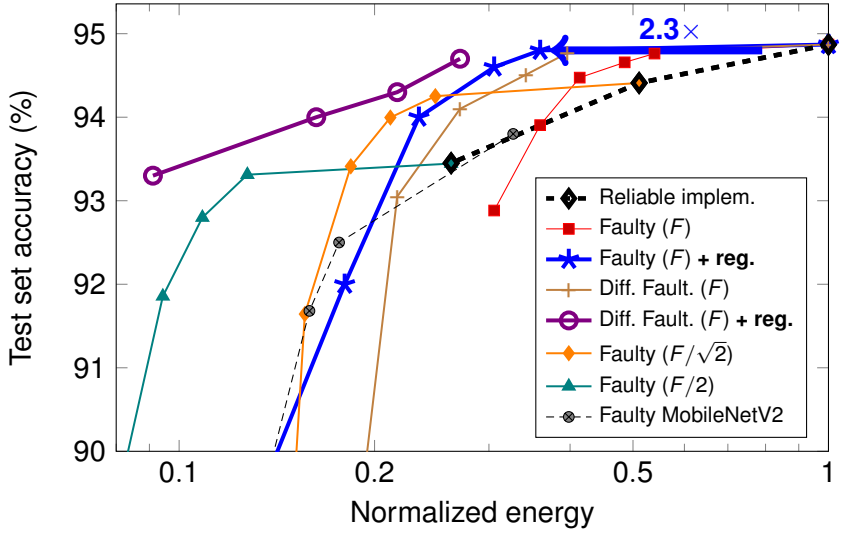
G. B. Hacene, et al. Training modern deep neural networks for memory-fault robustness. In IEEE ISCAS, 2019.

Strategies to improve robustness (CIFAR-10)



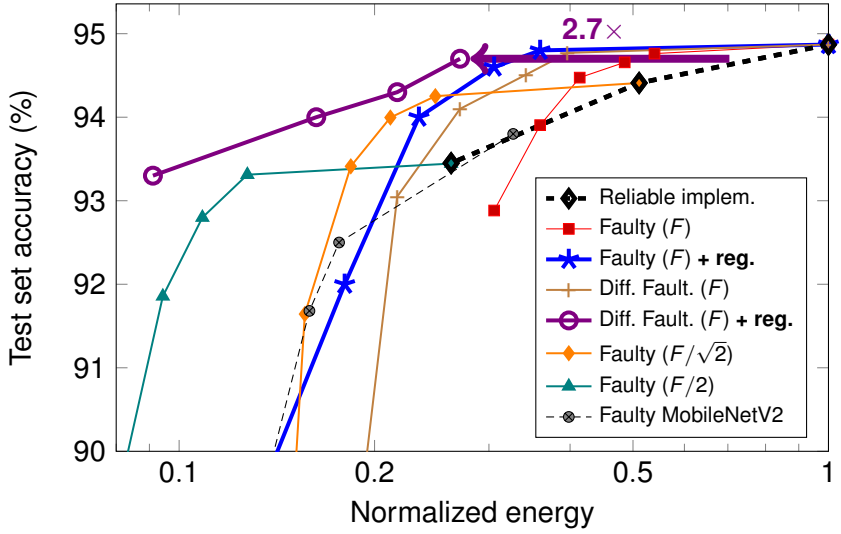
G. B. Hacene, et al. Training modern deep neural networks for memory-fault robustness. In IEEE ISCAS, 2019.

Strategies to improve robustness (CIFAR-10)



G. B. Hacene, et al. Training modern deep neural networks for memory-fault robustness. In IEEE ISCAS, 2019.

Strategies to improve robustness (CIFAR-10)



G. B. Hacene, et al. Training modern deep neural networks for memory-fault robustness. In IEEE ISCAS, 2019.

Outline

Sharpness-aware training

- Optimize for the loss value and the loss sharpness: “sharpness-aware minimization (SAM)” [Foret et al. ICLR 2021]

Sharpness-aware training

- Optimize for the loss value and the loss sharpness: "sharpness-aware minimization (SAM)" [Foret et al. ICLR 2021]

- Minimize the loss of a worst-case perturbation ϵ within a neighborhood region of radius ρ of a weight w:

$$L_{\text{SAM}}(w) = \min_{w} \max_{\|\epsilon\|_2 \leq \rho} L(w + \epsilon)$$

Sharpness-aware training

- Optimize for the loss value and the loss sharpness: "sharpness-aware minimization (SAM)" [Foret et al. ICLR 2021]

- Minimize the loss of a worst-case perturbation ϵ within a neighborhood region of radius ρ of a weight w:

$$L_{\text{SAM}}(w) = \min_{w} \max_{\|\epsilon\|_2 \leq \rho} L(w + \epsilon)$$

- Adaptive sharpness-aware minimization (ASAM) builds on SAM: [Kwon et al. ICML 2021]
  - Sharpness is scale-invariant
  - Allows larger ρ

$$L_{\text{ASAM}}(w) = \min_{w} \max_{\|\epsilon/|w|\|_2 \leq \rho} L(w + \epsilon)$$

SAMSON

- SAMSON = Sharpness-aware minimization scaled by outlier normalization

Gonçalo Mordido, Sébastien Henwood, Sarath Chandar, François Leduc-Primeau. SAMSON: Sharpness-Aware Minimization Scaled by Outlier Normalization for Improving DNN Generalization and Robustness, arXiv 2211.11561, 2023.

SAMSON

- SAMSON = Sharpness-aware minimization scaled by outlier normalization
- Targeting both generalization and model robustness

Gonçalo Mordido, Sébastien Henwood, Sarath Chandar, François Leduc-Primeau. SAMSON: Sharpness-Aware Minimization Scaled by Outlier Normalization for Improving DNN Generalization and Robustness, arXiv 2211.11561, 2023.

SAMSON

- SAMSON = Sharpness-aware minimization scaled by outlier normalization
- Targeting both generalization and model robustness
- No noise simulation and no knowledge of the expected noise is required.

Gonçalo Mordido, Sébastien Henwood, Sarath Chandar, François Leduc-Primeau. SAMSON: Sharpness-Aware Minimization Scaled by Outlier Normalization for Improving DNN Generalization and Robustness, arXiv 2211.11561, 2023.

SAMSON

- SAMSON = Sharpness-aware minimization scaled by outlier normalization
- Targeting both generalization and model robustness
- No noise simulation and no knowledge of the expected noise is required.

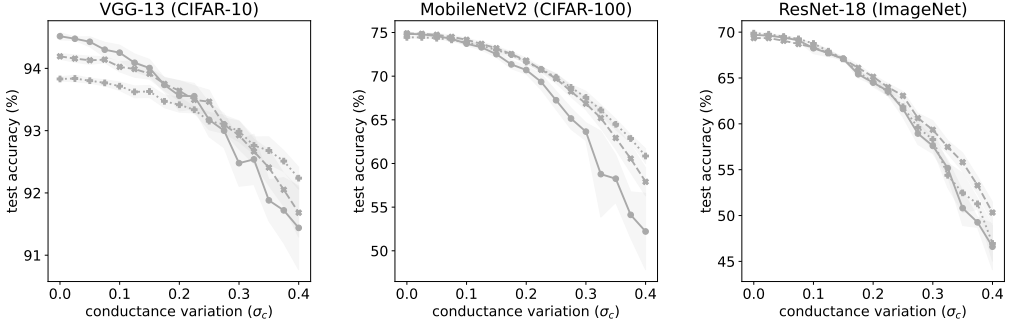
Loss function

$$L_{\text{SAMSON}}(w) = \min_{w} \max_{\|\epsilon\|_{\boldsymbol{w}}\|_p/|w|\|_2 \leq \rho} L(w + \epsilon),$$

Gonçalo Mordido, Sébastien Henwood, Sarath Chandar, François Leduc-Primeau. SAMSON: Sharpness-Aware Minimization Scaled by Outlier Normalization for Improving DNN Generalization and Robustness, arXiv 2211.11561, 2023.

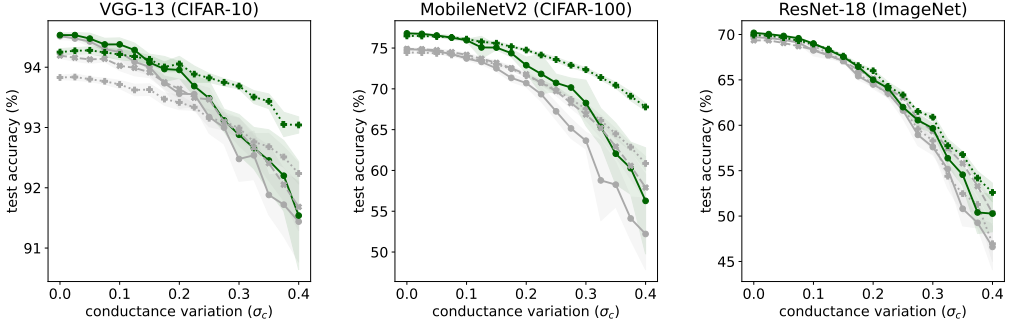
SAMSON Performance Results

- $g_i^\ell = w_i^\ell \times \frac{G_{\max}}{W_{\max}^\ell} \times \delta_i, \qquad \delta_i \sim \mathcal{N}(1, \sigma_c^2)$



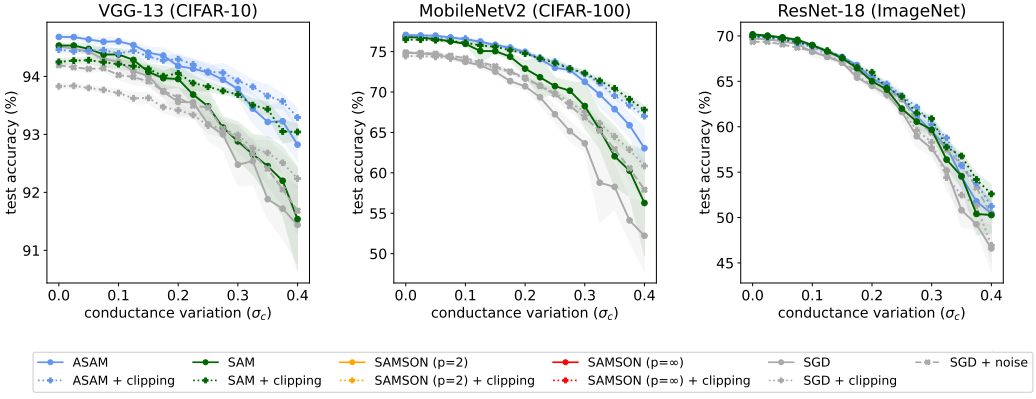
SAMSON Performance Results

- $g_i^\ell = w_i^\ell \times \frac{G_{\max}}{W_{\max}^\ell} \times \delta_i, \qquad \delta_i \sim \mathcal{N}(1, \sigma_c^2)$



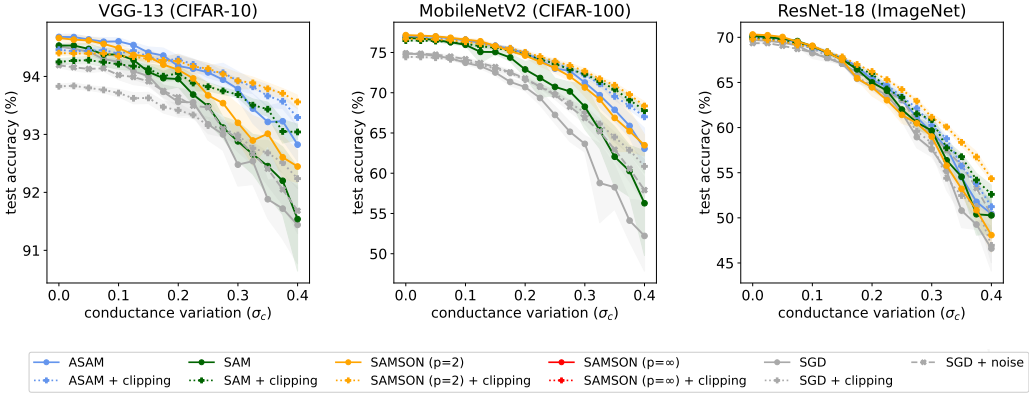
SAMSON Performance Results

- $g_i^\ell = w_i^\ell \times \frac{G_{\max}}{W_{\max}^\ell} \times \delta_i, \qquad \delta_i \sim \mathcal{N}(1, \sigma_c^2)$



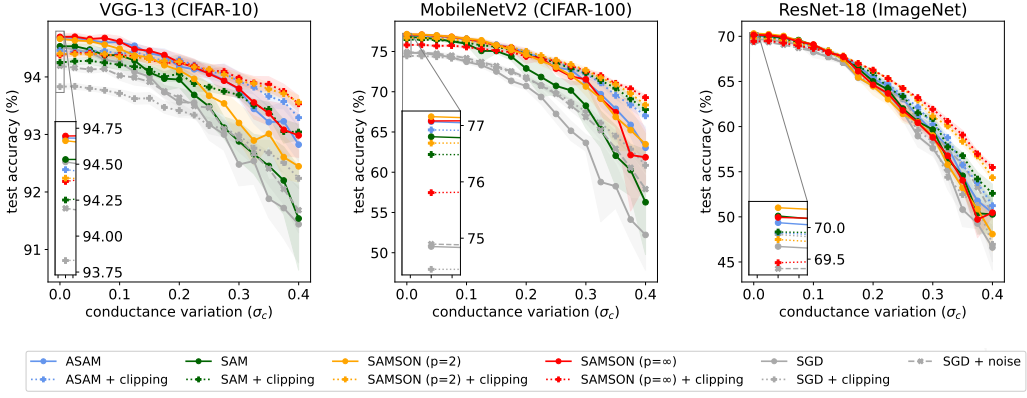
SAMSON Performance Results

- $g_i^\ell = w_i^\ell \times \frac{G_{\max}}{W_{\max}^\ell} \times \delta_i, \qquad \delta_i \sim \mathcal{N}(1, \sigma_c^2)$



SAMSON Performance Results

- $g_i^\ell = w_i^\ell \times \frac{G_{\max}}{W_{\max}^\ell} \times \delta_i, \qquad \delta_i \sim \mathcal{N}(1, \sigma_c^2)$



Outline

Objective of MemSE

1. Develop analytical formulas of the mean squared error (MSE) of a neural network implemented using (noisy) memristors.

J. Kern, S. Henwood, G. Mordido, E. Dupraz, A. Aïssa-El-Bey, Y. Savaria, and F. Leduc-Primeau. MemSE: Fast MSE prediction for noisy memristor-based DNN accelerators. In IEEE AICAS, 2022.

Objective of MemSE

1. Develop analytical formulas of the mean squared error (MSE) of a neural network implemented using (noisy) memristors.

2. Hopefully they are faster to evaluate than Monte-Carlo simulations,

J. Kern, S. Henwood, G. Mordido, E. Dupraz, A. Aïssa-El-Bey, Y. Savaria, and F. Leduc-Primeau. MemSE: Fast MSE prediction for noisy memristor-based DNN accelerators. In IEEE AICAS, 2022.

Objective of MemSE

1. Develop analytical formulas of the mean squared error (MSE) of a neural network implemented using (noisy) memristors.
2. Hopefully they are faster to evaluate than Monte-Carlo simulations,
3. . . . and differentiable.

J. Kern, S. Henwood, G. Mordido, E. Dupraz, A. Aïssa-El-Bey, Y. Savaria, and F. Leduc-Primeau. MemSE: Fast MSE prediction for noisy memristor-based DNN accelerators. In IEEE AICAS, 2022.

Memristor model

Noisy conductance values : $G_{i,j} = g_{i,j} + \delta^{q}_{i,j} + \epsilon^{v}_{i,j}$

Memristor model

Noisy conductance values : $G_{i,j} = g_{i,j} + \delta^{q}_{i,j} + \epsilon^{v}_{i,j}$

- $g_{i,j}$ is the target conductance value
  $0 < g_{i,j} < G_{\max}$

Memristor model

Noisy conductance values : $G_{i,j} = g_{i,j} + \delta^q_{i,j} + \epsilon^v_{i,j}$

- $g_{i,j}$ is the target conductance value
  $0 < g_{i,j} < G_{\max}$
- $\delta^q_{i,j}$ is the quantization error
  Uniform quantization with a resolution of $\Delta = \frac{G_{\max}}{N}$

Memristor model

Noisy conductance values : $G_{i,j} = g_{i,j} + \delta^q_{i,j} + \epsilon^v_{i,j}$

- $g_{i,j}$ is the target conductance value
  $0 < g_{i,j} < G_{\max}$
- $\delta^q_{i,j}$ is the quantization error
  Uniform quantization with a resolution of $\Delta = \frac{G_{\max}}{N}$
- $\epsilon^v_{i,j}$ is the noise due to variability in conductance programming
  $\epsilon^v_{i,j} \sim \mathcal{N}(0, \sigma^2)$

Memristor model

Noisy conductance values : $G_{i,j} = g_{i,j} + \delta^q_{i,j} + \epsilon^v_{i,j}$

- $g_{i,j}$ is the target conductance value
  $0 < g_{i,j} < G_{\max}$
- $\delta^q_{i,j}$ is the quantization error
  Uniform quantization with a resolution of $\Delta = \frac{G_{\max}}{N}$
- $\epsilon^v_{i,j}$ is the noise due to variability in conductance programming
  $\epsilon^v_{i,j} \sim \mathcal{N}(0, \sigma^2)$

For a neural network weight $w_{i,j}$, the conductance value corresponds to :

$$g_{i,j} = cw_{i,j}$$

Scaling factor $c = \frac{G_{\max}}{W_{\max}}$

Noisy memristor computations

Conductance can only be positive $\rightarrow$ we split G as $G = G^+ - G^-$

For a standard DNN :
$z_j = \sum_{i=1}^{L} w_{i,j} x_i$

With the memristor-DNN :
$Z_j = \frac{1}{c}(\sum_{i=1}^{L} rG_{i,j}^{+} X_i - \sum_{i=1}^{L} rG_{i,j}^{-} X_i)$

We estimate the performance of the noisy network compared with its standard counterpart through the MSE:

$$\begin{aligned} \mathrm{MSE}[Z_j] &= \mathbb{E}[(Z_j - z_j)^2] \\ &= \mathbb{V}[Z_j] + (\mathbb{E}[Z_j] - z_j)^2 . \end{aligned}$$

Noisy memristor computations

Conductance can only be positive $\rightarrow$ we split G as $G = G^+ - G^-$

For a standard DNN :
$z_j = \sum_{i=1}^{L} w_{i,j} x_i$

With the memristor-DNN :
$Z_j = \frac{1}{c}(\sum_{i=1}^{L} rG_{i,j}^{+} X_i - \sum_{i=1}^{L} rG_{i,j}^{-} X_i)$

We estimate the performance of the noisy network compared with its standard counterpart through the MSE:

$$\begin{aligned} \mathrm{MSE}[Z_j] &= \mathbb{E}[(Z_j - z_j)^2] \\ &= \mathbb{V}[Z_j] + (\mathbb{E}[Z_j] - z_j)^2 . \end{aligned}$$

Noisy memristor computations

Conductance can only be positive $\rightarrow$ we split G as $G = G^+ - G^-$

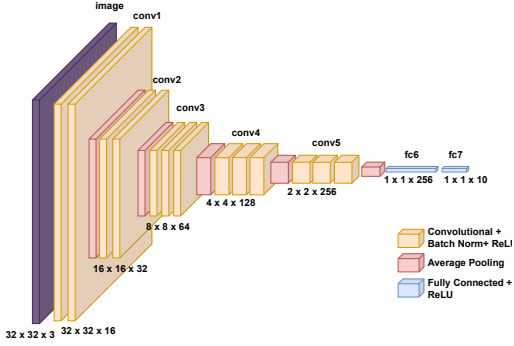
For a standard DNN :
$z_j = \sum_{i=1}^{L} w_{i,j} x_i$

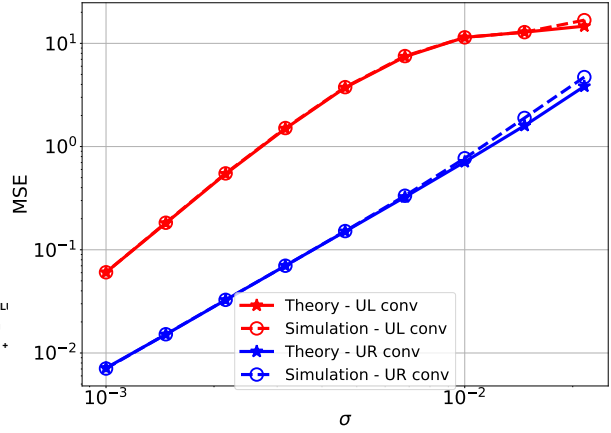
With the memristor-DNN :
$Z_j = \frac{1}{c}(\sum_{i=1}^{L} rG_{i,j}^{+} X_i - \sum_{i=1}^{L} rG_{i,j}^{-} X_i)$

We estimate the performance of the noisy network compared with its standard counterpart through the MSE:

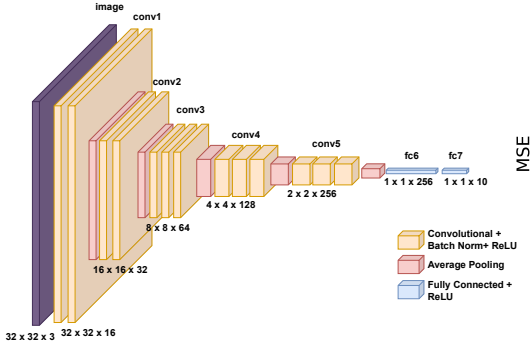
$$\begin{aligned} \mathrm{MSE}[Z_j] &= \mathbb{E}[(Z_j - z_j)^2] \\ &= \mathbb{V}[Z_j] + (\mathbb{E}[Z_j] - z_j)^2 . \end{aligned}$$

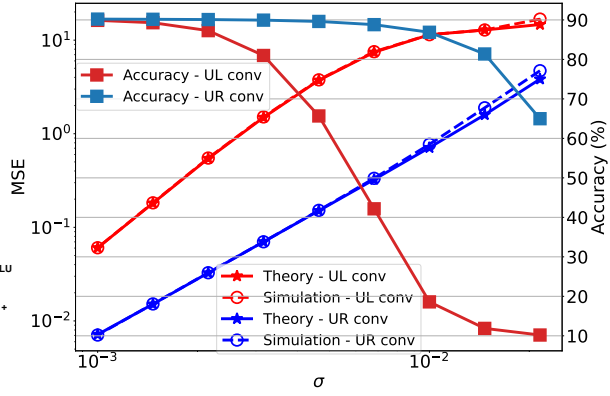
Experiments on CIFAR-10



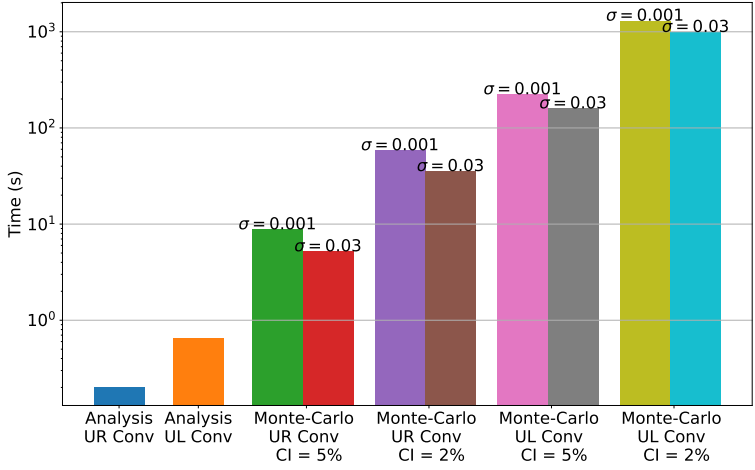


Experiments on CIFAR-10





MemSE Runtime



Conclusion

- Crucial to develop robust DNN models to target energy efficient hardware and be robust to attacks.

Conclusion

- Crucial to develop robust DNN models to target energy efficient hardware and be robust to attacks.
- Robustness is affected by DNN architecture and is not the same for every layer.

Conclusion

- Crucial to develop robust DNN models to target energy efficient hardware and be robust to attacks.
- Robustness is affected by DNN architecture and is not the same for every layer.
- Training for robustness yields significant improvements.

Conclusion

- Crucial to develop robust DNN models to target energy efficient hardware and be robust to attacks.
- Robustness is affected by DNN architecture and is not the same for every layer.
- Training for robustness yields significant improvements.
- Fast analytical characterization is possible.