FKeras: A Fault Tolerance Library for Keras

Olivia Weng[1*], Andres Meza[1], Javier M. Duarte[2], Nhan Tran[3], and Ryan Kastner[1]

[1] *Computer Science and Engineering Department, UC San Diego, 9500 Gilman Dr, La Jolla, CA 92093*
[2] *Physics Department, UC San Diego, 9500 Gilman Dr, La Jolla, CA 92093*
[3] *Fermi National Accelerator Laboratory, PO Box 500, Batavia, IL 60510*

[*]*oweng@ucsd.edu*

Abstract: We present FKeras, an open-source tool that uses Hessian information to quickly find which parameters in a neural network are sensitive to radiation faults, reducing the usual 200% resource overhead needed to protect them. © 2023 The Author(s)

1. Motivation

Many scientific applications require neural networks (NNs) to operate correctly in safety-critical or high radiation environments, including automated driving, space, and high energy physics. For example, physicists at the CERN Large Hadron Collider (LHC) deploy an autoencoder in a high radiation environment to filter their experimental data, which is collected at a high data rate (∼40 TB/s). The autoencoder must operate efficiently, within 200 ns, in a resource-constrained setting to process all the data correctly amid high radiation. Developing the autoencoder's hardware is challenging because it must be both efficient and robust.

However, efficiency and robustness are often in conflict with each other. Robust hardware methods like triple modular redundancy (TMR) protect against faults by increasing resources by 200%, reducing efficiency [Bertoa et al.(2023)]. To address these opposing demands, we must understand the fault tolerance inherent to NNs.

NNs have many redundant parameters, suggesting we do not need to introduce a blanket redundancy in the hardware—the common practice—when it is already present in the software. Prior work identifies where this redundancy exists by naively simulating all possible faults via a fault injection campaign, which is computationally intensive when we consider the thousands to millions of parameters that NNs have [Mahmoud et al.(2020)].

In this paper, we develop a more targeted fault injection campaign, which injects faults into the most sensitive parameters first to identify them faster. We present FKeras, an open-source tool that uses Hessian information to guide the fault injection campaign. Once we identify which parts of the NN are sensitive to radiation faults, we need only protect them, reducing the resources spent on robust hardware.

2. Experiments

To understand if the Hessian is a good guide for identifying the sensitive parameters, we perform some preliminary experiments on the autoencoder. The autoencoder consists of a two-layer encoder and a three-layer decoder. Solely the encoder, which consists of a convolutional layer followed by a linear layer, is vulnerable to faults because it must sit in the LHC near the sensors that collect particle collision energy readings. Its job is to compress the energy readings at the particle collision rate, i.e., 40 MHz. To do so, LHC physicists and engineers have designed an ASIC that operates at 40 MHz, executing a quantized encoder in a single cycle [Guglielmo et al.(2021)]. This means that only the 5-bit fixed point parameters are vulnerable to faults because the activations are not stored in memory for longer than a cycle. As such, the engineers have made the encoder ASIC radiation-hard by applying TMR to the parameters. Our goals are to show that

(1) the encoder has relatively few sensitive parameters (because NNs feature high redundancy), and
(2) the Hessian is useful for identifying the few sensitive parameters.

In our experiments, we consider a fault model where a single bit in a parameter is flipped at a time, matching the radiation conditions in the LHC.

To address our first goal, we perform a baseline fault injection campaign, in which we flip a single bit at a time and measure the performance of the encoder on 32 inputs. The performance is measured using Earth mover's distance (EMD) [Rubner et al.(2000)]. The EMD is a measure of the distance between two probability distributions. In our case, the input and output vectors are the encoder's input energy readings and the decoder's outputs, respectively. The lower the EMD the better.

Most of the encoder's parameter bits are not sensitive to faults, as seen in Figure 1. Note that the first 180 weights correspond to the encoder's convolutional layer, and the remaining weights correspond to the encoder's linear layer. The bit index is the index of the bit that was flipped, where bit 0 is the sign bit, bit 1 is the integer bit, and bits 2–4 are the fractional bits. As expected, the sign bit and integer bit create the largest changes in the magnitude of the parameters, so those bit flips lead to the higher EMDs. Overall, only 3% of the bits exceed the baseline EMD and need to be protected, reducing TMR overhead from 200% to a mere 6%.

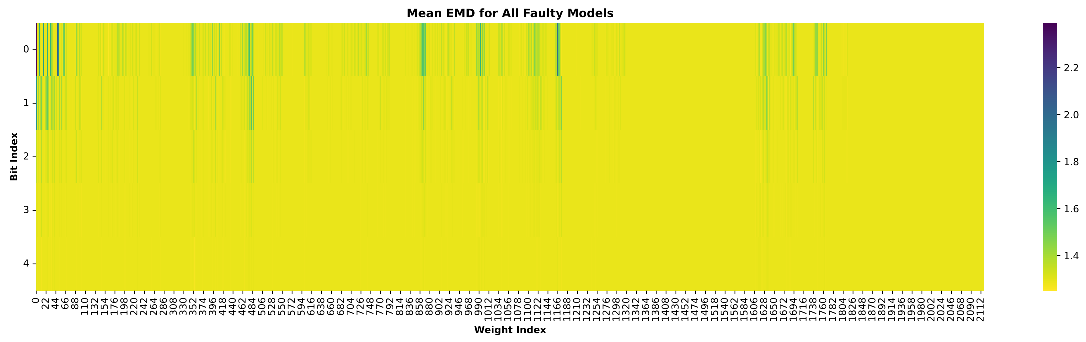

Fig. 1. Normalized average EMD of the faulty autoencoder models.

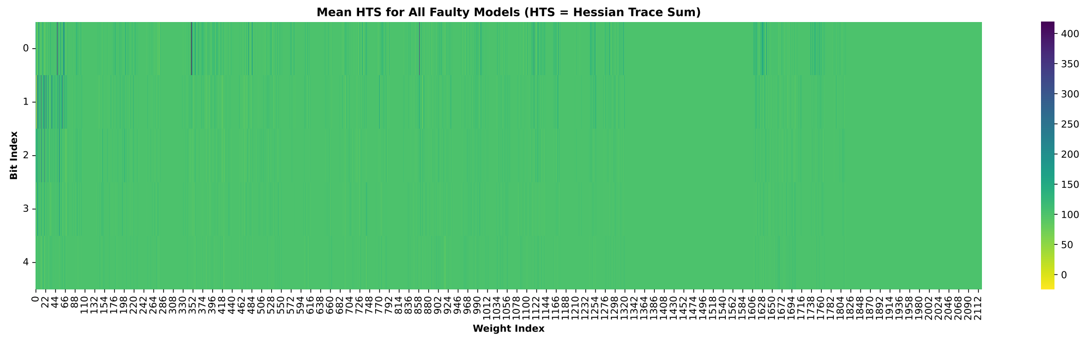

Fig. 2. Sum of the encoder's two Hessian traces, one for each layer, i.e., $\text{Tr}(\text{Conv}) + \text{Tr}(\text{Linear})$

To address our second goal, we also measure the Hessian trace of the encoder with respect to its parameters. We expect higher Hessian values to correlate with the bit flips that result in higher EMDs, indicating that the Hessian is a good guide for identifying the sensitive parameters.

As seen in Figures 1 and 2, the Hessian trace values largely correlate with the EMD values. There are some discrepancies, e.g., with weight index 1 having high EMD but lower Hessian trace. Even more so, the Hessian trace is a bit overly sensitive to fractional bit flips, as seen in the first 20 weights. Nonetheless, our ultimate goal is for the Hessian values to operate on the parameter level, indicating which parameters to inject faults into.

3. Conclusion

We have shown that the Hessian is a good guide for identifying the sensitive parameters in a NN. In our experiments, we find that the Hessian trace is sensitive to the bit flips that result in the largest EMDs, suggesting we can identify the sensitive parameters faster with a targeted Hessian search. Our next steps will be to statically analyze the Hessian eigenvalues and eigenvectors, which we can compute quickly using power iteration, to identify the sensitive parameters and guide a targeted fault injection campaign for speedy evaluation of a NN's reliability.

References

[Bertoa et al.(2023)] Timoteo García Bertoa et al. 2023. Fault Tolerant Neural Network Accelerators with Selective TMR. *IEEE Des. Test* 40, 2 (2023), 67. https://doi.org/10.1109/MDAT.2022.3174181

[Guglielmo et al.(2021)] Giuseppe Di Guglielmo et al. 2021. A reconfigurable neural network ASIC for detector front-end data compression at the HL-LHC. *IEEE Trans. Nucl. Sci.* 68 (1 8 2021), 2179. https://doi.org/10.1109/TNS.2021.3087100 arXiv:2105.01683 [physics.ins-det]

[Mahmoud et al.(2020)] Abdulrahman Mahmoud et al. 2020. HarDNN: Feature map vulnerability evaluation in CNNs. In *1st Workshop on Secure and Resilient Autonomy (SARA) at MLSys 2020.* arXiv:2002.09786 [cs.LG]

[Rubner et al.(2000)] Yossi Rubner, Carlo Tomasi, and Leonidas J. Guibas. 2000. The Earth Mover's Distance as a Metric for Image Retrieval. *Int. J. Comput. Vis.* 40 (2000), 99. https://doi.org/10.1023/A:1026543900054