Characterizing Deep Neural Networks Neutrons-Induced Error Model

Fernando Fernandes dos Santos, Angeliki Kritikakou, Olivier Sentieys, Paolo Rech

Characterizing Deep Neural Networks Neutrons-Induced Error Model

Fernando Fernandes dos Santos, Angeliki Kritikakou, Olivier Sentieys, and Paolo Rech

***Abstract*—We characterize the fault models for Deep Neural Networks (DNNs) in GPUs exposed to neutron. We observe tolerable and critical errors, and show that ECC is not effective in reducing critical errors.**

I. Introduction

Deep neural Networks (DNNs) are an efficient tool to accelerate several applications, such as object detection, image segmentation, classification, and prediction [1]–[3]. As a result, DNNs have been deployed in safety-critical and mission-critical domains, such as robotics, aeronautics and space exploration, smart healthcare, and autonomous driving. To reduce DNNs cost in terms of processing time and energy consumption, dedicated architectures with specialized hardware or reduced precision, e.g., half-precision floating-point (16 bits) or even short integer (8 bits), have been proposed, with satisfactory classification, segmentation, and detection accuracy. GPUs even feature Tensor Cores, i.e., a dedicated hardware to perform $4x4$ matrix multiplications in one cycle [4] while, for more specific domains, ASIC accelerators are employed for DNN, achieving significant performance and low power consumption [5].

When DNNs are deployed in safety-critical domains, real-time execution and reliability need to be paramount. Thus, to leverage the DNNs benefits towards safety-critical systems, a DNN-based system must be compliant with standards, such as ISO26262 and ISO/PAS 21448. GPU reliability can be undermined by several sources, including environmental perturbations, software errors, and process/temperature/voltage variations [6], [7]. Radiation-induced soft errors are particularly critical, as they have been found to dominate error rates in commercial devices [8]. DNNs are susceptible to transient fault induced by radiation [9]–[12]. Particularly, GPUs have a high fault rate because of the high amount of available resources [13]–[15] and the possibility to have multiple output elements corrupted, undermining DNN reliability [9], [16].

The main goal of this work is to obtain a realistic DNNs' transient error model extracted from neutron beam experiments and perform a realistic GPU reliability evaluation. We report findings from beam testing campaigns that assess GPU reliability as a DNN accelerator and identify the causes of *critical* errors (i.e., errors that impact detection) and *tolerable* errors. The testing campaigns consider two different GPU architectures, i.e., Kepler (Tesla K40) and Volta (Tesla V100), and several configuration. Specifically, (1) we evaluate the reliability of the three DNNs, (2) we measure the efficacy of the ECC on protecting the DNNs against errors that impact the classification/detection, (3) we characterize the criticality of the errors that can modify the GEMM output when used as the core of the DNNs operations, (4) we consider different floating-point data and operation precisions. The obtained results show that a single particle can spread through the GPU microarchitecture, affecting several parallel threads and output elements, significantly impacting DNN reliability. For DNNs, Single Error Correction Double Error Detection (SECDED) ECC can be useful but not always effective. Although several errors are masked by ECC and thus, do not propagate through the GPU, ECC can reduce neither the number of critical nor multiple errors.

Fernando Fernandes dos Santos is with Universidade Federal do Rio Grande do Sul (UFRGS), Porto Alegre, Brazil. e-mail: ffsantos@inf.ufrgs.br, and with Univ Rennes, INRIA, Rennes, France, e-mail: fernando.fernandes-dos-santos@inria.fr. Angeliki Kritikakou and Olivier Sentieys are with Univ Rennes, INRIA, Rennes, France. Paolo Rech is with the Department of Industrial Engineering of the University of Trento, Italy.

This project has received funding from the European Union's Horizon 2020 research and innovation programme under the Marie Sklodowska-Curie (MSCA) grant agreement No 886202 and No 899546, and from The Coordenaç ão de Aperfeiçoamento de Pessoal de Nível Superior - Brazil (CAPES) - Finance Code 001. Neutron beam time was provided by ChipIR (DOI: 10.5286/ISIS.E.RB2000161).

II. Evaluation Methodology

This section describes the characterized GPU devices and DNNs, the metrics adopted for DNN reliability evaluation, and how the beam experiments are performed.

Devices: We consider Kepler (Tesla K40) and Volta (Tesla V100) NVIDIA GPUs. The tested NVIDIA K40 (**Kepler**) is built with the Kepler ISA and fabricated in a $28nm$ TSMC standard CMOS technology [17]. Tesla V100 (**Volta**) are designed with the Volta micro-architecture and built with TSMC FinFET $12nm$ [4]. Volta GPUs feature hardware acceleration for three IEEE754 float point precisions: double (FP64), float (FP32), and half (FP16). We also evaluate the reliability of Volta GPUs' *tensor cores*, i.e., specific hardware that performs $4x4$ Matrix Multiplication, for GEMM kernels with FP16 precision. Tensor cores can also perform matrix multiplication for FP32 precision, however, the data will be cast to FP16 in the low-level operation. Both GPUs architectures have available Single Error Correction Double Error Detection (SECDED) Error Correcting Code (ECC) to protect the register file, shared memory, and caches. Our evaluation considers errors occurring only in the GPU core, not in the main memory. For both devices, we chose a beam spot sufficiently small (2cm of diameter) not to hit the onboard DDR when ECC is disabled.

DNNs: This work considers three modern frameworks: i.) You Only Look Once (YOLO v1 and v3) [18], [19], ii.)

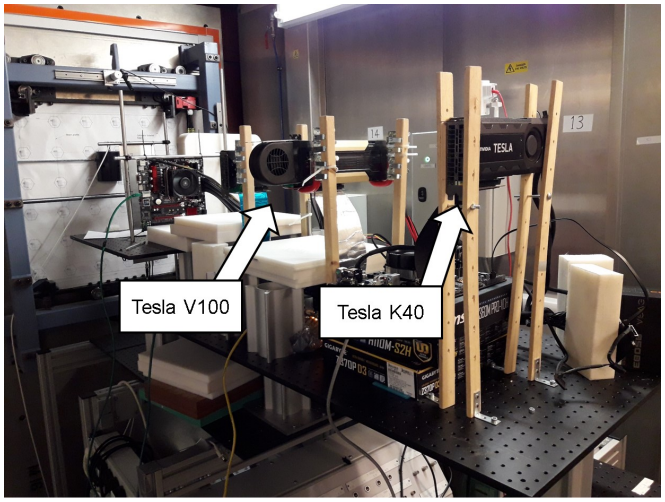

Fig. 1: Beam experiments setup

a 2) a Faster Region-based Convolution Neural Network (Faster R-CNN) [2], and iii.) a Residual Network (Resnet) [1]. **YOLO** is based on *Darknet*, which is an open-source CNN used for object classification and detection written in C and CUDA [18]. **Faster R-CNN** is written in C++ and Python, based on Caffe's [20] deep learning framework. **ResNet** is a CNN for classification based on the Torch deep learning framework [21]. ResNet only performs object classification, while YOLO and Faster R-CNN also provide object detection.

Reliability metrics: A transient fault may lead to one of the following outcomes: (1) No effect on the program output (i.e., the fault is masked, or the corrupted data is not used). (2) A Silent Data Corruption (SDC) (i.e., an incorrect program output). (3) A Detected Unrecoverable Error (DUE) (i.e., a program crash or device reboot). Not all errors in the output of the DNNs will impact the classification or object detection. We classify the observed SDCs between Tolerable SDCs (i.e., that do not impact classification/detection) and Critical SDCs (i.e., that impact classification/detection). When radiation modifies the classification or detection, we check if all the objects are detected and classified correctly. That is, we measure if the error created a false positive (i.e., add unexisting objects), or the error makes the DNN miss detect objects.

Beam Experiment Setup: Our experiments were performed at the ChipIR facility of the Rutherford Appleton Laboratory, UK, and at the LANSCE facility at Los Alamos National Laboratory, US. Figure 1 shows the setup mounted in the ChipIR facility. Both facilities deliver a beam of neutrons with a spectrum of energies similar to the atmospheric neutron one [22]. The available neutron flux was about $3.5 \times 10^6 n/(cm^2/s)$, $\sim$8 orders of magnitude higher than the terrestrial flux (13 $neutrons/(cm^2 \cdot h)$ at sea level [23]). The Failure In Time (FIT) rate is calculated by dividing the number of observed errors by the received particles fluence ($neutrons/cm^2$).

Since the terrestrial neutron flux is low, it is improbable to see more than a single corruption during program execution in a realistic application. We have carefully designed the experiments to maintain this property (observed error rates were lower than 1 error per 1,000 executions). Each DNN code was tested for at least 24 effective hours, not including the setup, result check, initialization, and recovery from the DUE time.

We added setup software and hardware watchdogs to monitor the experiments. The software watchdog controls the application under test, and if it stops responding in a predefined time interval, the kernel is killed and relaunched. This watchdog detects kernel crashes or software hangs, i.e., application crashes or control flow errors that prevent the GPU from completing assigned tasks (e.g., an infinite loop). The hardware watchdog is an Ethernet-controlled switch that performs a host computer's power cycle if the host computer itself does not acknowledge any ping requests in a predefined time interval. The hardware watchdog is necessary to detect when the operating system hangs.

III. DNNs Reliability and Error Model

This section presents the results from radiation experiments for the evaluated Deep Neural Networks (DNNs) with ECC ON and OFF. Note that YOLOv3 could not be tested on ECC OFF due to beam time limitations.

A. Deep Neural Networks FIT rate

Figure 2a shows the normalized SDC and DUE FIT rates of YOLOv1, Faster R-CNN, and ResNet on Kepler, and YOLOv3 on Volta GPUs. The reported results are normalized to not reveal business-sensitive information. Even if the data is normalized, it allows a direct comparison among configurations. The reported values are relative to YOLOv1 ECC ON SDC FIT for Kepler and YOLOv3 ECC ON SDC FIT for Volta. Experimental data is presented with a 95% confidence interval.

DUE: DUEs are more probable than SDCs for all tested configurations. DNN kernels have a high level of reuse that requires several device-host (CPU-GPU) synchronizations. A transient fault during those synchronizations could potentially result in a GPU DUE. As a result, while a significant portion of SDCs could be masked, DUEs could still undermine the device's reliability. For the same reason, Faster R-CNN and Resnet, which require a much larger number of synchronizations, show up to $5\times$ higher DUE rate than YOLO.

ECC ON DUE rate increases up to 30% for Faster R-CNN, Resnet, and YOLOv1. ECC is able to correct one-bit flip in the memories, and when a double-bit flip is detected, it throws a system exception. As DNNs use a large memory to perform classification/detection, multiple errors on memories are expected to happen, resulting in a higher DUE rate when ECC is ON. Equivalently, on Volta GPU, the DUE rate grows as the data representation increases. Comparing FP16 and FP32 versions of YOLOv3, the size of memory grows $2x$. As the FP32 version of YOLOv3 performs more memory transfer than the FP16 version, the DUE is expected to be higher for FP32. Additionally, with the ECC ON on Volta, and when caches and register usage increase to store the FP32 precision, the double bit flips in the memories will be more frequent.

SDC: Figure 2a shows that the SDC rates are related to framework complexity and accuracy. Compared to YOLO, the complex structures used on Faster R-CNN and Resnet increase

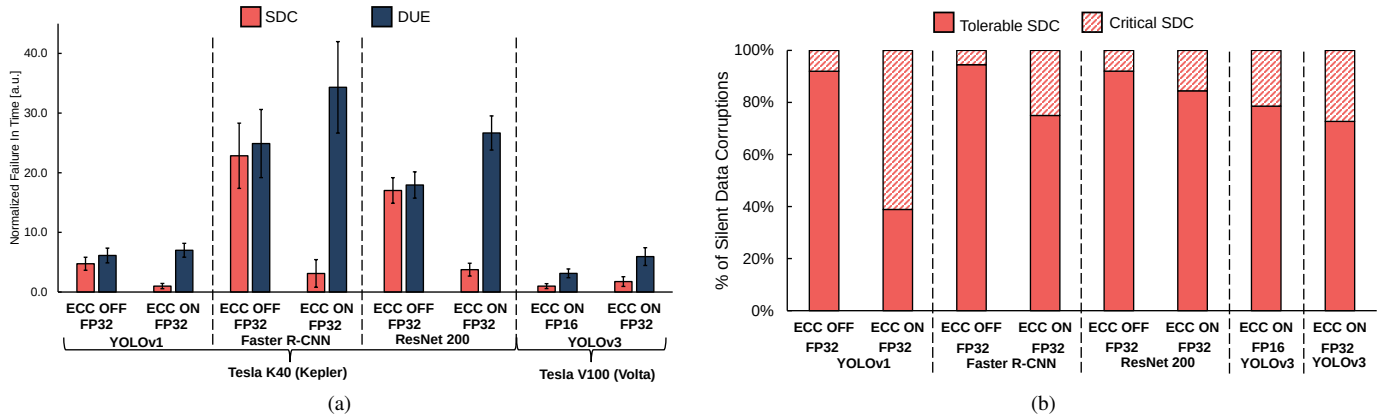

Fig. 2: Figure 2a shows the normalized Failure In Time for YOLOv1, Faster R-CNN, ResNet, and YOLOv3. Figure 2b shows the distribution of the Tolerable vs Critical SDCs for the tested DNNs.

their SDC rate by more than $10x$. Figure 2a shows that Resnet has a higher FIT rate than YOLO (i.e., v1 and v3), though the rate is similar when compared to Faster R-CNN. Although Faster R-CNN and ResNet have a high detection/classification accuracy, it is insufficient to compensate for the higher error rate associated with a complex framework used for both DNNs.

The SDC rate for YOLOv1, Faster R-CNN, and Resnet on the ECC ON is 21%, 13.6%, and 22% the SDC rate seen for ECC OFF, respectively. ECC is not as effective on these workloads as other codes, mainly because neural networks are intrinsically resilient to data errors. ECC can reduce the GEMM SDC rate by about one order of magnitude [24] (details Section III-B). However, ECC is less efficient in protecting a DNN, as some of the SDCs, that ECC masks would not have affected the DNN execution. On Volta GPU, YOLOv3 FP16 has a lower FIT than YOLOv3 FP32. The amount of per-core resources required to perform the operations depends on the chosen data precision. Namely, fewer resources will be used for the lower float precisions.

Figure 2b shows the percentage of the Tolerable SDCs vs. the Critical SDCs for all configurations tested. For object detection, the percentage of critical SDCs is much lower for Faster R-CNN and YOLOv3 than for YOLOv1. For YOLOv1, the percentage of critical SDCs is 8% for ECC OFF and 61% for the Kepler with ECC ON. For YOLOv3 with ECC ON, the percentage of critical SDCs is 21% and 27% for FP16 and FP32, respectively. For Faster R-CNN, the critical SDCs are 5% for ECC OFF and 25% for the Kepler with ECC ON. The differences in the critical errors percentages between the DNNs come from the detection mechanism. YOLOv1 is the simplest DNN tested in our setup (only 31 layers). An error is expected to impact much more the detection of simpler DNNs than the more accurate ones.

ECC can reduce the error rate, but the proportion of critical errors is not reduced, which is a symptom of the poor resiliency provided by ECC. Based on the GPU-Qin analysis in [25], it is known that ECC does not mask all the faults as the error in computing elements could propagate to the output. ECC reduces the absolute number of SDCs but has the side effect of increasing the portion of multiple errors (Details Section III-B), which are more likely to propagate through DNN layers and affect detection. On Volta, the percentages of critical errors are similar comparing the two data types when ECC is ON. It is worth noting that critical SDCs are less dependent on data type as the two DNNs have similar detection accuracy regarding the data precision.

B. Complex error models for DNNs

To completely characterize the critical errors on DNNs and how they can be generated at the low level, we also evaluated how the errors affect the General Matrix Multiplication algorithm (GEMM) used as the core of DNNs execution on modern GPUs. In fact, 70% to 86% of the operations for the evaluated DNNs are GEMM related operations.

Figure 3a shows the SDC and DUE normalized FIT rates for GEMM. Kepler values are relative to the SDC rate with ECC enabled, while Volta values are relative to the SDC rate of FP16 without Tensor cores. We selected matrix sizes that saturate the resources of each device (2048×2048 for the Kepler and 16384×16384 for Tesla V100). As device resources are saturated, data in Figure 3a can be used to compare the GEMM reliability of GEMM across architectures. Smaller matrices will lower the FIT rate given the number of unused resources.

From Figure 3a, it is clear that GEMM reliability depends on the device and the precision. Following previous studies [24], the DUE rate is lower than the SDC rate for ECC OFF on Kepler. ECC triggers an application DUE when there is a double-bit error. When ECC is OFF, the double bit flips may lead to an SDC at the output of GEMM.

From Figure 3a we can notice that, independently of precision, the use of tensor cores increases DUE FIT. The GEMM with tensor cores is optimized to use the tensor cores to the maximum performance by performing warp level synchronizations. Our data suggest that the corruption of internal resources, necessary to perform specific warp level synchronizations of the tensor core, is particularly prone to

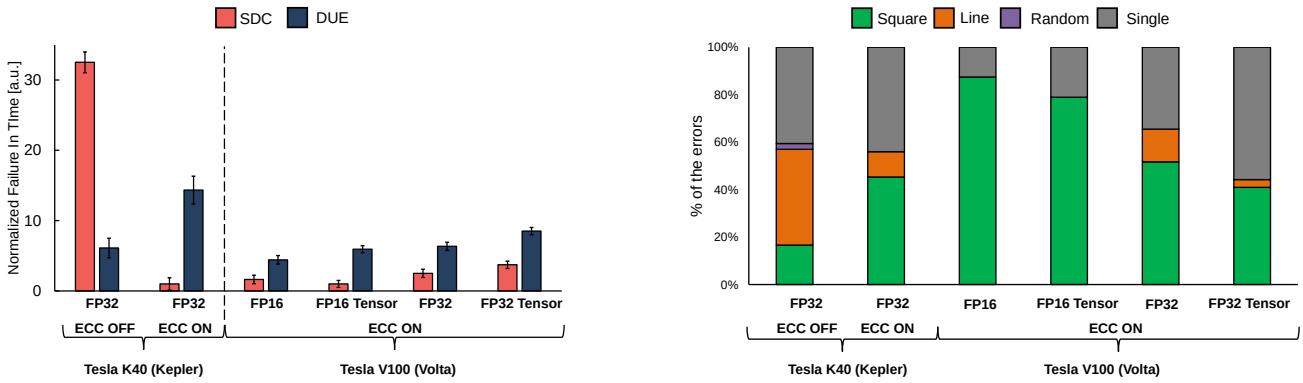

Fig. 3: Figure 3a shows normalized Failure In Time for the General Matrix Multiplication (GEMM) on Kepler and Volta. Figure 3b GEMM Error Geometry.

generate a DUE. When used to improve the performance of the DNNs, tensor cores may also increase the DNNs' DUE rate even more.

Figure 3a shows that tensor cores GEMM has a lower FIT rate when executed in FP16 precision for Volta GPU. This attests that the tensor core circuit is slightly more reliable than the combination of ADD, MUL, and the loop control variables needed to implement matrix multiplication in software. When GEMM is executed in FP32 precision, the tensor core SDC FIT rate increases significantly, being 20% higher than the pure GEMM one. In fact, as stated in Section II, the tensor cores on Volta architecture execute physical operations only in FP16 precision. FP32 precision inputs require a hardware casting to FP16 precision, increasing the GPU's occupation and, thus, the SDCs (and DUEs) FIT rate. When executed on Volta without the tensor cores, the GEMM FIT rate increases as the precision increases. The increase from FP16 to FP32 is about $2\times$. As FP32 occupies $2\times$ more GPU resources, it is expected that the FIT rate will also have a factor near to $2\times$.

We also investigate the output of the GEMM to classify the errors based on their coordinates. That is, Figure 3b shows the percentage of corrupted executions that the output affected Single, Square (four or more elements distributed in a rectangle), Line (i.e., multiple corrupted elements on the same row/column of the output matrix), and Randomly distributed errors. In most cases, GPU corruption affects more than a single output element. It is worth noting that multiple corrupted elements are not caused by multiple impinging particles. Instead, the impact of a single particle is *spread* during computation to across multiple output elements.

Square errors are potentially the most severe for DNNs. Square errors in GEMM are caused by faults that impact the scheduling or the execution of multiple threads in a Streaming Multiprocessor (SM). GEMM is a highly optimized version of matrix multiplication since it can divide the input matrices into chunks that fit nicely in the cache of a SM, reducing memory latency. If a fault can cause a thread to be incorrectly assigned or scheduled to a SM, or if some threads fail to synchronize, the whole SM output matrix portion is likely to be corrupted, leading to a Square error. Errors in memory elements protected by ECC (e.g., registers and caches) have been reported to manifest into either single corrupted element or line errors [26]. When ECC is turned ON, single and line errors (which are less critical for CNNs) are corrected, but all the other errors (including rectangular errors) are not corrected. As a result, the percentage of rectangular errors increases when ECC is enabled.

Figure 3b shows that the geometry of the output errors changes slightly when comparing Kepler and Volta. For FP32 GEMM on both GPUs with ECC ON, the distribution of the errors are mainly of Square and Line errors, showing that the criticality of the GEMM is directly related to how the error is propagated from the low-level fault to the algorithm output.

It is worth noting that the criticality of the errors increases for the GEMM without the tensor compared to the GEMM optimized with the tensor core. The matrix multiplication performed in hardware generated more Single errors than the version only performed in software. This result is following past results [12] where the reliability of DNNs hardware accelerators have been shown to have a less critical error model than the operations performed only on software.

IV. Conclusions

We have discussed the reliability of DNNs on beam experiments and evaluated the criticality of the errors in the output. Although the ECC is a powerful technique to reduce the SDC rate, it cannot reduce the Critical errors proportionally on DNNs. We then characterized the GEMM kernels as the core of DNNs, to investigate the error models that can impact the DNNs reliability.

As radiation experiments demonstrated, the single bit flip in the memories is not the leading cause of critical errors. Additionally, the criticality of the output of the GEMM indicates that a single error in a single element of a GEMM kernel is not realistic as an error model to simulate critical errors on DNNs. For the final paper, we will perform an assembly-level fault injection in a range of DNNs with different error models to investigate the leading cause of the critical errors.

REFERENCES

[1] K. He, X. Zhang, S. Ren, and J. Sun, "Identity mappings in deep residual networks," *CoRR*, vol. abs/1603.05027, 2016. [Online]. Available: http://arxiv.org/abs/1603.05027

[2] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards real-time object detection with region proposal networks," in *Advances in Neural Information Processing Systems (NIPS)*, 2015.

[3] G. Gkioxari, J. Malik, and J. Johnson, "Mesh R-CNN," *CoRR*, vol. abs/1906.02739, 2019. [Online]. Available: http://arxiv.org/abs/1906.02739

[4] NVIDIA. (2017) NVIDIA Tesla V100 GPU architecture. [Online]. Available: https://images.nvidia.com/content/volta-architecture/pdf/volta-architecture-whitepaper.pdf

[5] N. P. Jouppi *et al.*, "In-datacenter performance analysis of a tensor processing unit," in *Proceedings of the 44th Annual International Symposium on Computer Architecture*, ser. ISCA '17. New York, NY, USA: ACM, 2017, pp. 1–12. [Online]. Available: http://doi.acm.org/10.1145/3079856.3080246

[6] J. C. Laprie, "Dependable computing and fault tolerance : Concepts and terminology," in *Fault-Tolerant Computing, 1995, Highlights from Twenty-Five Years., Twenty-Fifth International Symposium on*, Jun 1995, pp. 2–12.

[7] M. Nicolaidis, "Time redundancy based soft-error tolerance to rescue nanometer technologies," in *VLSI Test Symposium, 1999. Proceedings. 17th IEEE*, 1999, pp. 86–94.

[8] R. Baumann, "Soft errors in advanced computer systems," *IEEE Design Test of Computers*, vol. 22, no. 3, pp. 258–266, 2005.

[9] F. F. d. Santos, P. F. Pimenta, C. Lunardi, L. Draghetti, L. Carro, D. Kaeli, and P. Rech, "Analyzing and increasing the reliability of convolutional neural networks on GPUs," *IEEE Transactions on Reliability*, vol. 68, no. 2, pp. 663–677, 2019.

[10] G. Li, S. K. S. Hari, M. Sullivan, T. Tsai, K. Pattabiraman, J. Emer, and S. W. Keckler, "Understanding error propagation in deep learning neural network (dnn) accelerators and applications," in *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*, ser. SC '17. New York, NY, USA: Association for Computing Machinery, 2017. [Online]. Available: https://doi.org/10.1145/3126908.3126964

[11] S. Blower, P. Rech, C. Cazzaniga, M. Kastriotou, and C. D. Frost, "Evaluating and mitigating neutrons effects on cots edgeai accelerators," *IEEE Transactions on Nuclear Science*, vol. 68, no. 8, pp. 1719–1726, 2021.

[12] R. L. Rech, S. Malde, C. Cazzaniga, M. Kastriotou, M. Letiche, C. Frost, and P. Rech, "High energy and thermal neutrons sensitivity of google tensor processing units," *IEEE Transactions on Nuclear Science*, pp. 1–1, 2022.

[13] N. DeBardeleben, S. Blanchard, L. Monroe, P. Romero, D. Grunau, C. Idler, and C. Wright, "GPU behavior on a large HPC cluster," in *Euro-Par 2013: Parallel Processing Workshops*, D. an Mey, M. Alexander, P. Bientinesi, M. Cannataro, C. Clauss, A. Costan, G. Kecskemeti, C. Morin, L. Ricci, J. Sahuquillo, M. Schulz, V. Scarano, S. L. Scott, and J. Weidendorfer, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2014, pp. 680–689.

[14] D. Tiwari, S. Gupta, J. Rogers, D. Maxwell, P. Rech, S. Vazhkudai, D. Oliveira, D. Londo, N. DeBardeleben, P. Navaux, L. Carro, and A. Bland, "Understanding GPU errors on large-scale HPC systems and the implications for system design and operation," in *2015 IEEE 21st International Symposium on High Performance Computer Architecture (HPCA)*, 2015, pp. 331–342.

[15] M. B. Sullivan, N. Saxena, M. O'Connor, D. Lee, P. Racunas, S. Hukerikar, T. Tsai, S. K. S. Hari, and S. W. Keckler, "Characterizing and mitigating soft errors in gpu dram," in *MICRO-54: 54th Annual IEEE/ACM International Symposium on Microarchitecture*, ser. MICRO '21. New York, NY, USA: Association for Computing Machinery, 2021, p. 641–653. [Online]. Available: https://doi.org/10.1145/3466752.3480111

[16] Y. Ibrahim, H. Wang, M. Bai, Z. Liu, J. Wang, Z. Yang, and Z. Chen, "Soft error resilience of deep residual networks for object recognition," *IEEE Access*, vol. 8, pp. 19 490–19 503, 2020.

[17] NVIDIA. (2012) Kepler GK110/210 whitepaper. [Online]. Available: https://www.nvidia.com/content/dam/en-zz/Solutions/Data-Center/tesla-product-literature/NVIDIA-Kepler-GK110-GK210-Architecture-Whitepaper.pdf

[18] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 779–788.

[19] J. Redmon and A. Farhadi, "YOLOv3: An incremental improvement," *CoRR*, vol. abs/1804.02767, 2018. [Online]. Available: http://arxiv.org/abs/1804.02767

[20] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell, "Caffe: Convolutional architecture for fast feature embedding," in *Proceedings of the 22Nd ACM International Conference on Multimedia*, ser. MM '14. New York, NY, USA: ACM, 2014, pp. 675–678. [Online]. Available: http://doi.acm.org/10.1145/2647868.2654889

[21] R. Collobert, K. Kavukcuoglu, and C. Farabet, "Torch7: A matlab-like environment for machine learning," 2011.

[22] C. Cazzaniga and C. D. Frost, "Progress of the scientific commissioning of a fast neutron beamline for chip irradiation," vol. 1021. 22nd Meeting of the International Collaboration on Advanced Neutron Sources (ICANS XXII), may 2017, pp. 159–164. [Online]. Available: https://doi.org/10.1088/1742-6596/1021/1/012037

[23] C. Slayman, *JEDEC Standards on Measurement and Reporting of Alpha Particle and Terrestrial Cosmic Ray Induced Soft Errors*. Boston, MA: Springer US, 2011, pp. 55–76. [Online]. Available: https://doi.org/10.1007/978-1-4419-6993-4_3

[24] D. A. G. Gonçalves de Oliveira, L. L. Pilla, T. Santini, and P. Rech, "Evaluation and Mitigation of Radiation-Induced Soft Errors in Graphics Processing Units," *IEEE Transactions on Computers*, vol. 65, no. 3, pp. 791–804, 2016.

[25] G. Li, K. Pattabiraman, C. Y. Cher, and P. Bose, "Understanding error propagation in gpgpu applications," in *SC16: Int. Conference for High Performance Computing, Networking, Storage and Analysis*, Nov 2016, pp. 240–251.

[26] P. Rech, T. Fairbanks, H. Quinn, and L. Carro, "Threads distribution effects on graphics processing units neutron sensitivity," *Nuclear Science, IEEE Transactions on*, vol. 60, no. 6, pp. 4220–4225, Dec 2013.