SMART: Selective MAC zero-optimzation for neural network reliability under radiation

Anuj Justus Rajappa, Philippe Reiter, Tarso Kraemer Sarzi Sartori, Luiz Henrique Laurini, Hassen Fourati, Siegfried Mercelis, Peter Hellinckx, Rodrigo Possamai Bastos

SMART: Selective MAC zero-optimzation for neural network reliability under radiation

Anuj Justus Rajappa[a,1,2], Philippe Reiter[a,1,3], Tarso Kraemer Sarzi Sartori[b,d,1], Luiz Henrique Laurini[b,1], Hassen Fourati[d,1], Siegfried Mercelis[a,1], Peter Hellinckx[c,1], Rodrigo Possamai Bastos[b,1]

[a] *IDLab, University of Antwerp - imec, Sint-Pietersvliet 7, 2000, Antwerp, Belgium*
[b] *Univ. Grenoble Alpes, CNRS, Grenoble INP, TIMA, 38000, Grenoble, France*
[c] *University of Antwerp, Groenenborgerlaan 171, 2020, Antwerp, Belgium*
[d] *Univ. Grenoble Alpes, Inria, CNRS, Grenoble INP, GIPSA-Lab, 38000, Grenoble, France*

Abstract

Neural networks running on low-power edge devices can help in achieving ubiquitous computing with limited infrastructure. When such edge devices are deployed in conventional and extreme environments without the necessary shields, they must be fault tolerant for reliable operation. As a pilot study, we focus on embedding fault tolerance into neural networks by proposing a novel selective multiply-accumulate zero-optimization technique based on whether the value of an input provided to a neuron of a neural network is zero. If the value is zero, then the corresponding multiply-accumulate operation is bypassed. We subjected the implementation of our optimization technique to radiation test campaigns using ~14 MeV neutrons, and found the proposed optimization technique to improve the fault tolerance of the tested neural network by a factor of 1.78 times.

Keywords: sparsity, radiation, MAC, multiply-accumulate, optimization

1. Introduction

Machine learning algorithms for making decisions at the edge [10] and reducing the data transferred between edge devices can reduce the strain on networks and cloud infrastructure [22]. Thus, when targeting ubiquitous computing [9], machine learning algorithms can allow increasing the quantity of the raw data processed and edge devices deployed even when limited by cloud infrastructure. Edge devices are typically placed close to the data source [22], which could expose them to cosmic rays, hazardous radiation levels, extreme temperatures, unreliable power supplies, etc. [6] at ground level [17], space, nuclear facilities and other hard to reach environments [19]. This exposure can cause transient errors [21], that typically manifest as single bit-flips in the edge devices with potential to cause system failure [4]. Hence, these edge systems must be fault tolerant for reliable operation, which is usually achieved using a combination of hardware [21]and software techniques [7].

We hypothesized that the fault tolerance of a neural network (NN) can be increased by reducing the number of data transfers and overall execution time. The latter can be achieved by replacing longer executing Multiply ACcumulate (MAC) operations with shorter executing zero comparators. While the former can be achieved by reducing the number of arithmetic floating point operations (FLOPs).

The number of FLOPs was reduced by leveraging the sparsity (ratio between number of non-significant values and total number of values) of the runtime input values [3] through all the layers of a NN. If an input value to be multiplied with a weight is zero, then the corresponding MAC operation, which consists of FLOPs, is bypassed. This optimization is termed Selective Multiply-Accumulate zeRo-opTimization (SMART). A process flow diagram for SMART is shown in Figure 1.

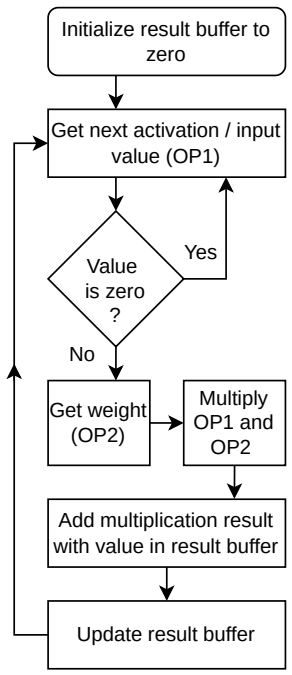

Figure 1: Abstract of SMART implementation.

[1] E-mail addresses: { anuj.justusrajappa, philippe.reiter, siegfried.mercelis, peter.hellinckx } @uantwerpen.be, { tarso.kraemer-sarzi-sartori, luiz-henrique.laurini, rodrigo.bastos } @univ-grenoble-alpes.fr, hassen.fourati@grenoble-inp.fr

[2] https://orcid.org/0000-0001-8167-9171

[3] https://orcid.org/0000-0002-2548-7172

The rationale behind the MAC bypass is that zero multiplied by any real number is zero and zero is also the additive identity for real numbers. Hence, when an addition or multiplication is carried out between two operands, the results can be directly deduced from the operands if at least one operand is zero, without using an adder or multiplier [15, 18] .

The number of zero comparators replacing FLOPs is proportional to the input sparsity. SMART can be implemented through software changes. We consider SMART to be novel as we could not identify a similar technique for NN fault tolerance among the current state-of-the-art techniques. The closest we could find was the exploration of the relationship between static sparsity of weights and fault resiliency of NNs [20]. While SMART can be achieved in hardware [15, 18], it would require specialised processor architectures, unlike our proposed software-based approach that can be executed on commercial off-the-shelf processors.

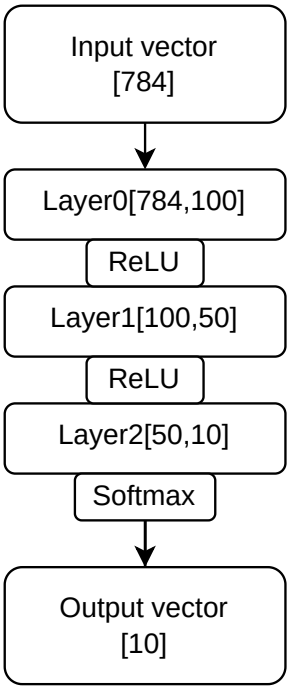

Figure 2: Architecture of the NN.

Nonetheless, several other optimizations for improving the fault tolerance of NNs have been proposed. One proposal uses the Feature-map and Inference Level Resilience (FILR) [14] technique for statically protecting vulnerable parts of a Convolutional Neural Network (CNN) by duplicating the corresponding logical operations and rerunning vulnerable inferences by analyzing their output. Another uses model compression techniques such as binary quantization for improving the fault tolerance of a Deep Neural Network (DNN) [20]. Ranger [4] is another technique used to improve fault tolerance of a DNN by correcting transient faults without re-computation. Others have evaluated the effects of neutron radiation and simulated fault injections on machine learning algorithms like Support Vector Machines (SVMs) [5, 23] and CNNs [24], and assessed the fault tolerance of these algorithms. Studies linking the reliability of CNNs on FPGAs to their paramters and metrics, such as model accuracy, degree of parallelism, quantization and reduced data precision, [12, 13] has also been conducted. The effect of instruction set architecture on the reliability of CNNs has also been studied [2] on an ARM platform with simulated fault injections. However, the study uses the Common Microcontroller Software Interface Standard-NN (CMSIS-NN) [11] library for CNN execution with low-precision fixed-point representation and does not consider runtime input sparsity.

The following sections describe the NNs subjected to the radiation test campaigns; the effects of SMART and temporal Triple Module Redundancy (TMR) techniques on the NNs; the test setup and methodology; preliminary analysis of the radiation test results; and, concluding observations and future work.

2. Case Study Algorithms

The architecture of the NN used during the radiation test campaign is shown in Figure 2. This NN was designed, trained and evaluated using the TensorFlow [1] Python library, and 60000 training images and 10000 testing images from the Modified National Institute of Standards and Technology (MNIST) database. This NN is also known as MNIST digit classifier as the NN is used to classify the images representing digits from 0 to 9. The input sparsity to the different layers of the NN generated at runtime is shown in Table 1, which was computed using all 10000 test images from MNIST.

Table 1: Sparsity of input values to different layers in the NN

	Layer 0	**Layer 1**	**Layer 2**
Sparsity	80.7%	66.6%	47.3%

The parameters of the trained network are fed to a custom implementation of the NN algorithm in C language, using a custom framework to create four different versions of the NN. These are: (1) a version of the NN without any of the proposed optimization (**simple**), (2) NN with SMART (**SMART**), (3) an NN with TMR (**TMR**), and (4) an NN with TMR and SMART (**TMR+SMART**). In a TMR version of the NN, the corresponding non-TMR NN is executed thrice and a majority vote is applied to the output.

In order to understand the effects of SMART on NNs, the above four versions were subjected to radiation test campaigns. The NN(simple) and NN(TMR) were included in the test to provide reference results which can facilitate relativistic comparison with NN(SMART). NN(simple) is a non-optimized version expected to provide low fault tolerance results. NN(TMR) is optimized using an industry standard temporal TMR [16] technique and was expected to provide results for high fault tolerance. While NN(TMR+SMART) also provides results for comparison, this version was primarily intended to observe the effects of a combined TMR and SMART hybrid optimization on NN fault tolerance.

Table 2: Preliminary analysis of results from February and July test campaigns

NN version	Avg. Neutron flux $(10^5 neutrons/cm^2)/s$	Irradiation time (h)	Iterations	Number of errors			Neutron fluence $(10^{10} neutrons/cm^2)$	Cross section $(10^{-10} cm^2)$
				Tolerable	Critical	Total		
Simple	4.272	12.0	1892	11	0	11	1.8530	5.9371
SMART	3.843	43.7	452	29	2	49	14.6963	3.3342
	4.272	56.3	1195	17	1			
TMR	4.272	11.9	1813	5	0	5	1.8297	2.7326
TMR+SMART	3.843	44.5	451	23	0	39	14.7672	2.6410
	4.272	56.0	1181	13	3			

3. Radiation Test Setup

Each of the four versions of the NN algorithm were packaged into separate radiation test programs, shown in Figure 3, to facilitate executing the case study algorithms on the radiation test setup developed by Université Grenoble Alpes (UGA) [8]. The number of iterations of the test program is controlled by the radiation test setup and each iteration corresponds to an execution of the test program. To limit the size of a test program, 250 inputs were randomly selected from the MNIST testing images, and inference results for all of these images are computed in one iteration. To reduce the variables in the experiments, a single input data set was used across all campaigns. Each of the inputs contain a one-dimensional array of size 784 in single-precision floating-point format (FP32), which is obtained by normalizing and flattening the two dimensional array of order 28×28 representing the resolution of an image in the MNIST database. Each input is used to compute 120 inferences within one iteration. This number was chosen to cause the total execution time of one iteration to lie between 10s to 20s, for optimal scheduling of the test programs during the radiation test campaigns. Each inference of an NN generates a one-dimensional array of size 10 in FP32 as output, which represents the probability of the input being an image of a digit from 0 to 9.

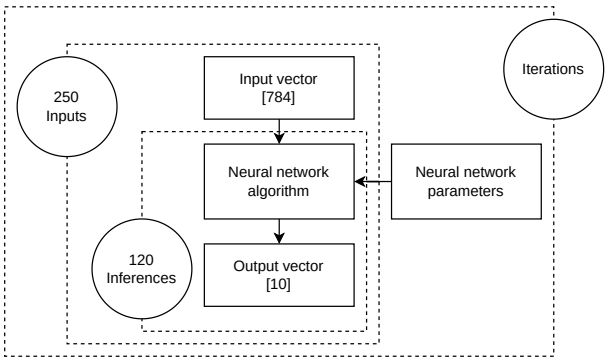

Figure 3: Iterations of radiation test program

4. Experiment and Analysis

The radiation test programs were executed from February 17-18 and July 4-8 of 2022 at Laboratory of Subatomic Physics & Cosmology (LPSC) in Grenoble, France. The radiation test setup utilizes Raspberry Pi 4B with Raspberry Pi OS Lite version 11 and a superscalar quadcore Cortex-A72 (ARM v8) 64-bit SoC. Once the results were obtained from the experiments, the error analysis was done by comparing the results with the golden reference, which was obtained by running the radiation test program under normal operating conditions.

The preliminary analysis of the results from the radiation test campaigns is as shown in Table 2. The first column of this table represents the four versions of the NN algorithm tested under radiation. The second column contains the corresponding average neutron flux to which the various NN versions were exposed. The third column represents the time spent by each NN version executing under various neutron flux levels on the setup's multi-core CPU . The fourth column represents the number of iterations of the radiation test program for each NN version under various neutron flux levels. The fifth column represents the number of errors that occurred during all the corresponding iterations of the NN versions. This column is divided into three sub-columns which represent the following counts.

1. *Tolerable error* is incremented by one if one or multiple errors occurred within an iteration but did not result in any classification mismatch when compared with the golden reference.
2. *Critical error* is incremented by one if one or multiple errors occurred within an iteration and includes classification mismatches when compared with the golden reference.
3. *Total error* is the sum of tolerable and critical error counts for all NN versions. This error count is used for cross section [25] calculation.

The sixth column represents the neutron fluence associated with each of the NN versions. This represents the total number of neutrons that passed through a $1cm^2$ area of the radiation setup while the corresponding NN version was executing. The last column represents the cross section calculated from total error count for each of the NN versions.

5. Observation and Discussion

The last column of Table 2 suggests that SMART has reduced the cross section – i.e., the probability of either a tolerable or critical error occurring for a given neutron radiation –, increasing the overall fault tolerance of NN(SMART) compared to NN(Simple). NN(TMR) has superior fault tolerance

compared to NN(SMART) but at an increased computational cost. Finally, NN(TMR+SMART) outperforms NN(TMR) by a small margin in terms of fault tolerance improvement. However, more radiation tests are required to confirm this advantage as the observed margin is small. Future radiation tests will ensure approximately equal irradiation time for all algorithms under test.

6. Conclusion and Future work

Radiation test campaigns for evaluating the fault tolerance of the four NN versions were conducted. Preliminary analysis of the results from these campaigns suggests that NN(SMART) outperforms NN(Simple) by ∼1.78 times in terms of the probability of a neutron-induced error occurring. However, these NNs are outperformed by their TMR counterparts, as expected. Further analyses will reveal more information on the nature of the errors that occurred, and will associate the errors with the various computing strategies [8] and characteristics of the radiation test setup used during the experiments for all four NN versions. The relationship between input sparsity, different NN output functions [3], different NN architecture and SMART will be explored in future work.

Acknowledgements

This work has been partially supported by: MultiRad (PAI project funded by Région Auvergne-Rhône-Alpes); IRT Nanoelec (ANR-10-AIRT-05) and LabEx PERSYVAL-Lab (ANR-11-LABX-0025-01), both projects funded by the French program PIA; UGA/LPSC/GENESIS platform; Bourse de mobilité Génération IA 2030, funded by French embassy in Belgium; and MOVIQ (Mastering Onboard Vision Intelligence and Quality) project funded by Flanders Innovation & Entrepreneurship (VLAIO) and Flanders Space (VRI).

References

[1] Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G. S., Davis, A., Dean, J., Devin, M., Ghemawat, S., Goodfellow, I., Harp, A., Irving, G., Isard, M., Jia, Y., Jozefowicz, R., Kaiser, L., Kudlur, M., Levenberg, J., Mané, D., Monga, R., Moore, S., Murray, D., Olah, C., Schuster, M., Shlens, J., Steiner, B., Sutskever, I., Talwar, K., Tucker, P., Vanhoucke, V., Vasudevan, V., Viégas, F., Vinyals, O., Warden, P., Wattenberg, M., Wicke, M., Yu, Y., and Zheng, X. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. Software available from tensorflow.org.

[2] Abich, G., Gava, J., Reis, R., and Ost, L. Soft error reliability assessment of neural networks on resource-constrained iot devices. In *2020 27th IEEE International Conference on Electronics, Circuits and Systems (ICECS)* (2020), pp. 1–4.

[3] Apicella, A., Donnarumma, F., Isgrò, F., and Prevete, R. A survey on modern trainable activation functions. *Neural Networks 138* (2021), 14–32.

[4] Chen, Z., Li, G., and Pattabiraman, K. A low-cost fault corrector for deep neural networks through range restriction. In *2021 51st Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN)* (2021), pp. 1–13.

[5] Garay Trindade, M., Benevenuti, F., Letiche, M., Beaucour, J., Kastensmidt, F., and Possamai Bastos, R. Effects of thermal neutron radiation on a hardware-implemented machine learning algorithm. *Microelectronics Reliability 116* (2021), 114022.

[6] Hanif, M. A., Khalid, F., Putra, R. V. W., Rehman, S., and Shafique, M. Robust machine learning systems: Reliability and security for deep neural networks. In *2018 IEEE 24th International Symposium on On-Line Testing And Robust System Design (IOLTS)* (2018), pp. 257–260.

[7] Huang, Q., and Jiang, J. An overview of radiation effects on electronic devices under severe accident conditions in npps, rad-hardened design techniques and simulation tools. *Progress in Nuclear Energy 114* (2019), 105–120.

[8] Kraemer Sarzi Sartori, T., Fourati, H., Letiche, M., and Bastos, R. P. Assessment of radiation effects on attitude estimation processing for autonomous things. *IEEE Transactions on Nuclear Science 69*, 7 (2022), 1610–1617.

[9] Krumm, J. *Ubiquitous Computing Fundamentals*. CRC Press, 2018.

[10] Kukunuri, R., Aglawe, A., Chauhan, J., Bhagtani, K., Patil, R., Walia, S., and Batra, N. Edgenilm: Towards nilm on edge devices. In *Proceedings of the 7th ACM International Conference on Systems for Energy-Efficient Buildings, Cities, and Transportation* (New York, NY, USA, 2020), BuildSys '20, Association for Computing Machinery, p. 90–99.

[11] Lai, L., Suda, N., and Chandra, V. Cmsis-nn: Efficient neural network kernels for arm cortex-m cpus. *arXiv preprint arXiv:1801.06601* (2018).

[12] Libano, F., Rech, P., Neuman, B., Leavitt, J., Wirthlin, M., and Brunhaver, J. How reduced data precision and degree of parallelism impact the reliability of convolutional neural networks on fpgas. *IEEE Transactions on Nuclear Science 68*, 5 (2021), 865–872.

[13] Libano, F., Wilson, B., Wirthlin, M., Rech, P., and Brunhaver, J. Understanding the impact of quantization, accuracy, and radiation on the reliability of convolutional neural networks on fpgas. *IEEE Transactions on Nuclear Science 67*, 7 (2020), 1478–1484.

[14] Mahmoud, A., Sastry Hari, S. K., Fletcher, C. W., Adve, S. V., Sakr, C., Shanbhag, N., Molchanov, P., Sullivan, M. B., Tsai, T., and Keckler, S. W. Optimizing selective protection for cnn resilience. In *2021 IEEE 32nd International Symposium on Software Reliability Engineering (ISSRE)* (2021), pp. 127–138.

[15] Masadeh, M., Hasan, O., and Tahar, S. Input-conscious approximate multiply-accumulate (mac) unit for energy-efficiency. *IEEE Access 7* (2019), 147129–147142.

[16] Morgan, K. S., McMurtrey, D. L., Pratt, B. H., and Wirthlin, M. J. A comparison of tmr with alternative fault-tolerant design techniques for fpgas. *IEEE Transactions on Nuclear Science 54*, 6 (2007), 2065–2072.

[17] O'Gorman, T. The effect of cosmic rays on the soft error rate of a dram at ground level. *IEEE Transactions on Electron Devices 41*, 4 (1994), 553–557.

[18] Parashar, A., Rhu, M., Mukkara, A., Puglielli, A., Venkatesan, R., Khailany, B., Emer, J., Keckler, S. W., and Dally, W. J. Scnn: An accelerator for compressed-sparse convolutional neural networks. *SIGARCH Comput. Archit. News 45*, 2 (jun 2017), 27–40.

[19] Prinzie, J., Simanjuntak, F. M., Leroux, P., and Prodromakis, T. Low-power electronic technologies for harsh radiation environments. *Nature Electronics 4*, 4 (2021), 243–253.

[20] Sabbagh, M., Gongye, C., Fei, Y., and Wang, Y. Evaluating fault resiliency of compressed deep neural networks. In *2019 IEEE International Conference on Embedded Software and Systems (ICESS)* (2019), pp. 1–7.

[21] Sayil, S. *Single Event Soft Error Mechanisms*. Springer International Publishing, Cham, 2016, pp. 31–48.

[22] Shi, W., and Dustdar, S. The promise of edge computing. *Computer 49*, 5 (2016), 78–81.

[23] Trindade, M. G., Coelho, A., Valadares, C., Viera, R. A. C., Rey, S., Cheymol, B., Baylac, M., Velazco, R., and Bastos, R. P. Assessment of a hardware-implemented machine learning technique under neutron irradiation. *IEEE Transactions on Nuclear Science 66*, 7 (2019), 1441–1448.

[24] Wang, H.-B., Wang, Y.-S., Xiao, J.-H., Wang, S.-L., and Liang, T.-J. Impact of single-event upsets on convolutional neural networks in xilinx zynq fpgas. *IEEE Transactions on Nuclear Science 68*, 4 (2021), 394–401.

[25] Wrobel, F., Aguiar, Y., Marques, C., Lerner, G., García Alía, R., Saigné, F., and Boch, J. An analytical approach to calculate soft error rate induced by atmospheric neutrons. *Electronics 12*, 1 (2023).