2017 47th Annual IEEE/IFIP International Conference on Dependable Systems and Networks Workshops

Evaluation and Mitigation of Soft-Errors in Neural Network-based Object Detection in Three GPU Architectures

Fernando Fernandes dos Santos, Lucas Draghetti, Lucas Weigel, Luigi Carro, Philippe Navaux, and Paolo Rech
Instituto de Informatica, Universidade Federal do Rio Grande do Sul, Porto Alegre, Rio Grande do Sul, Brasil
email: {ffsantos, lucas.kleindraghetti, lfweigel, carro, navaux, prech}@inf.ufrgs.br

***Abstract*—In this paper, we evaluate the reliability of the You Only Look Once (YOLO) object detection framework. We have exposed to controlled neutron beams GPUs designed with three different architectures (Kepler, Maxwell, and Pascal) running Darknet, a Convolutional Neural Network for automotive applications, detecting objects in both Caltech and Visual Object Classes data sets. By analyzing the neural network corrupted output, we can distinguish between tolerable errors and critical errors, i.e., errors that could impact on real-time system execution.**
Additionally, we propose an Algorithm-Based Fault-Tolerance (ABFT) strategy to apply to the matrix multiplication kernels of neural networks able to detect and correct 50% to 60% of radiation-induced corruptions. We experimentally validate our hardening solution and compare its efficiency and efficacy with the available ECC.

I. Introduction

Artificial Neural Networks (ANNs) are becoming a widely adopted computational approach in many fields, from data mining to pattern recognition, High-Performance Computing (HPC), and data analysis [1], [2]. Lately, ANNs have been employed in self-driving cars, which rely on multiple pattern recognition algorithms to identify and classify objects based on captured frames or signals. Most of these algorithms can be efficiently implemented using ANNs [3], [4], [5].

ANNs have a massively parallel structure and require high computational capabilities. Graphics Processing Units (GPUs) appear then to be very suitable to execute ANN algorithms. Thanks to their low cost, increased energy efficiency, and flexible development platforms, embedded GPUs are adopted to detect obstacles or objects in automotive applications. The NVIDIA *kepler* GPUs, for instance, are used in Tesla self-driving models [6].

There are several sources of errors that could undermine the device reliability, including environmental perturbations, software errors, and process, temperature, or voltage variations [7], [8], [9], [10]. In this paper, we focus on radiation-induced soft errors, which have been shown to be the most critical type of errors in modern electronic devices as they produce a failure rate that is higher than all the other reliability mechanisms combined [10].

Radiation-induced failures are particularly relevant in safety-critical applications, like automotive, for which reliability is mandatory. The reliability characterization of ANNs that implement objects detection is then necessary to ensure automotive systems safety. We specifically consider Darknet, which is an open source convolutional neural network for object classification and detection, written in C and CUDA [3].

We exposed three different platforms to the control neutron beam available at the ChipIR facility at the Rutherford Appleton Laboratories (RAL), Didcot, UK and LANSCE, Los Alamos National Lab. (LANL), Los Alamos. Exposed GPUs were designed with three different NVIDIA architectures: the $Kepler$ K40, the embedded $Maxwell$ Tegra X1, and the new $Pascal$ Titan X. GPUs were tested while executing Darknet on two universally accepted datasets: Caltech and Visual Object Classes (VOC). As both ChipIR and LANSCE neutron spectrum are suitable to mimic the terrestrial one and the selected datasets are generic, the errors observed during our experiments are representative of errors that can occur in realistic applications. Experimental data is then a collection of the possible radiation-induced outcomes for Darknet. We first compare the radiation-induced error probability in the three different architectures. Then, we compare the corrupted output of the three architectures and distinguish between critical errors (i.e. errors that could impact a self-driving vehicle) and acceptable errors by measuring the Precision and Recall of the corrupted output.

Among the considered GPU architectures, only the K40 has the main memory structures protected with a Single Error Correction Double Error Detection (SECDED) ECC. Our experiments confirm that ECC is very effective as it provided a reduction in the number of errors of about $4x$. To protect also devices without ECC, and to improve ANN reliability, we propose an Algorithm-Based Fault Tolerance (ABFT) strategy for neural networks. We take advantage of the efficient ABFT that we have designed for GPUs to detect and correct single and multiple errors in matrix multiplication operations [11]. We tested this ABFT strategy on Kepler and Pascal architectures. The idea of applying the existing ABFT to ANN arises from the observation that about 67% of operations in Darknet are matrix multiplication related. We modify Darknet to call the ABFT-protected kernel at each matrix multiplication call. The proposed ABFT can detect and correct about 60% of radiation-induced errors in Kepler devices and 50% in Pascal devices.

The remainder of the paper is organized as follows. Section II gives a background on Darknet structure and reviews

2325-6664/17 $31.00 © 2017 IEEE
DOI 10.1109/DSN-W.2017.47

functional safety for automotive applications concepts. Section III describes the proposed ABFT for convolutional neural networks, Section IV describes the adopted experimental methodology, Section V presents and discusses experimental results and evaluates ABFT and ECC efficiency, while Section VI concludes the paper.

II. Background

A. Neural Network for Automotive Applications

Nowadays, Deep Neural Networks (DNNs) are one of the most efficient ways to perform object detection and classification. These networks achieve great performances in visual recognition challenges, like image classification [12], segmentation [13] and object detection [3], [14], showing significant improvement over other technologies such as Word-Channels and SpatialPooling [15]. Much research has been done specifically regarding using DNNs to perform pedestrian detection in real time and showed promising results [16], [17], [15].

In this paper, we explicitly consider *Darknet*, which is an open source Convolutional Neural Network (CNN) used to implement the YOLO object detection system [3]. YOLO is a Deep Convolutional Neural framework capable of detecting objects in real time, analyzing up to 45 frames per second. Darknet became famous because of its performances in the PASCAL VOC 2007 Challenge, as it was the only real time system and had a high accuracy compared to the other systems [18].

Darknet, as most CNNs, performs three operations to object detection, which are: (1) Layer convolution. (2) Max pooling layers. (3) Classification through a fully connected layer. Convolutional layers work as feature extraction on Darknet, sliding a convolutional kernel to the input image. On Darknet, there are 24 Convolutional layers, and those layers produce different feature maps from the input (i.e. the output of a convolutional kernel in the image). Darknet feature maps must be processed by Max pooling layers, which dimensionally reduces the previously obtained feature map. Max pooling divides the feature map into blocks and reduces each block into one value.

After rearranging the input data and the convolutional kernels, it is possible to perform convolutional operations with General Matrix to Matrix Multiplication (GEMM) on CNNs. On GPUs, GEMM functions are well known for their performance. However, according to Oliveira *et al.* [19], GPU GEMM is particularly prone to experience radiation-induced errors. In Section IV-A we provide the implementation details of Darknet, and of the data sets we used for our evaluation.

B. Radiation-Induced Effects on GPUs and Resilience Support

The impact of galactic cosmic rays with the upper level of the terrestrial atmosphere generates a cascade of particles. A significant number of high energy neutrons (i.e., neutrons with an energy higher than 10MeV) is produced by the primary and secondary impacts of the cosmic ray. A flux of about 13 $n/((cm^2) \times h)$ reaches the ground [20]. The interaction of a neutron may perturb the transistor state, generating bit-flips in memory as well as a current spike in logic circuits that, if latched, leads to an error [21], [22].

Neutron-induced errors may be particularly critical in GPUs architecture [23], [24], [25], [19], [26]. This is because GPUs were originally designed for entertainment and video or image editing applications, for which reliability is not a concern [27]. The corruption of GPU's hardware scheduler or shared memory, used to make parallel executions more efficient, is likely to affect the computation of several parallel processes, reducing the system reliability [28]. Additionally, to feed a high number of parallel processes, GPUs require large caches and register files, which have been demonstrated to be the cause of the majority of errors in modern computing devices [29], [30].

Only the major storage structures of GPUs for HPC applications are protected with Single Error Correction Double Error Detection (SECDED) ECC. ECC has been demonstrated to be effective in reducing the error rate of modern GPUs [26]. In Section III we evaluate how ECC impacts Darknet reliability. Unfortunately, not all GPUs have memory structures protected with ECC. There are several possibilities to mitigate radiation-induced errors in software, like duplication or triplication [31]. Here, we propose an adaptation of an already developed ABFT for matrix multiplication to improve ANN reliability. This hardening philosophy comprises the analysis of an algorithm's basic structure to find an efficient and intelligent way to harden it. ABFT strategies are usually based on the encoding of input data and algorithm redesign to work with this data. At the end of the algorithm, the output is checked taking advantage of the encoding. When an error is detected, it can be corrected if the ABFT strategy has enough information to do so, or a re-computation is triggered. In the following Section, we provide the details of the ABFT for ANN and experimentally evaluate its efficacy.

III. ABFT for Neural Networks

As described in Section II-A, Darknet, and CNNs, in general, make an intensive use of GEMM (i.e. 67% of overall processing time), and GEMM is a sensible kernel in GPUs. Most radiation-induced errors in GEMM have been demonstrated to impact multiple locations in the corrupted output [11]. Multiple errors on the first layers of a neural network could significantly reduce the reliability of the system, as the errors could propagate to the following layers.

The basic idea of our ABFT strategy for neural networks is to harden the matrix multiplication kernels of Darknet using the ABFT strategy for GPUs we have proposed in [11]. Our strategy is then an extension of Huang and Abraham idea [32], which is based on Freivalds' result checking approach [33]. Input matrices A and B are coded before computation, adding column and row checksum vectors (A_c and B_r in Figure 1) by summing all the elements in the correspondent column or row.

Instead of adding an extra column and row, we have decided to reduce of one the number of columns and rows of the matrices to be multiplied in Darknet and substitute them with

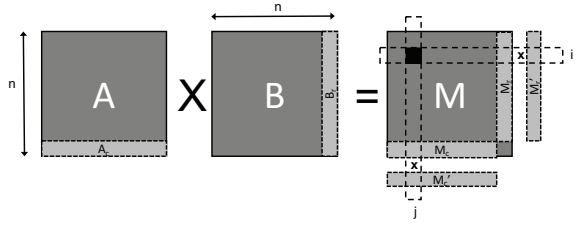


Fig. 1. Algorithm-Based Fault Tolerance for matrix multiplication [32]

the checksums. The reasons for this choice are: (1) $sgemm$ kernels are finely tuned to use the GPU efficiently. Adding an extra row and column would significantly increase Darknet execution time and (2) reducing the number of rows and columns will affect Darknet detection capabilities. However, the detection accuracy will still be acceptable.

The result of the multiplication of the expanded matrices is a fully-checksum matrix M, where the n_{th} row and the n_{th} column contain the column (M_c) and row (M_r) checksum vectors of M, respectively [33]. When the multiplication is finished, M column and row checksum vectors are re-calculated summing the first $n-1$ columns and $n-1$ rows of M, resulting on M'_c and M'_r, respectively. Output verification is done by comparing the checksum vectors from the multiplication and the newly computed ones.

If a mismatch is detected between $M_r[i]$ and $M'_r[i]$, it means that at least one error is present in the i_{th} row of M, and respectively for columns. If $M[i,j]$ is identified as the only error in M, it can be corrected quickly using the row (or column) checksum vectors following Equation 1 [32].

$$M_{correct}[i,j] = M[i,j] - (M'_r[i] - M_r[i]) \qquad (1)$$

On a GPU, the operations required to compute the checksums and detect errors can be made in $O(n)$, while error correction takes constant time [11]. The technique proposed by Huang and Abraham is only capable of correcting single output errors [32], which we have experimentally demonstrated to correspond to less than 43% of the cases [11]. Thanks to experimental radiation data, the ABFT strategy was extended to correct multiple errors in the same row or column of M in constant time and randomly distributed errors in $O(e_r \times e_c)$, where e_r and e_c are the number of mismatches between M row and column checksums, respectively (details in [11]).

The proposed ABFT strategy has been tested on both the K40 and the Titan X, and results are presented in Section V-C.

IV. Reliability Evaluation Methodology

In this section, we briefly describe Darknet, the selected data sets, the devices used in this study and our radiation sensitivity data collection methodology.

A. Darknet Neural Network and Data Set

YOLO [3] performs its detection by dividing the input image into a $S \times S$ grid and then calculating the class probability for each grid cell, as well as the bounding boxes and confidence. Hence, it can detect and classify objects in real time. The network uses 24 convolutional layers, which are then followed by two fully connected layers. There are also four maxpool layers in specific stages of the network. The activation function used is Leaky.

The datasets we use are the PASCAL VOC 2012 Classification/Detection Dataset [18] and the Caltech Pedestrian Dataset [34]. PASCAL VOC is chosen because it is a renowned dataset and because it was used to evaluate Darknet detection capabilities [3]. It consists of more than ten thousand images in which the system should detect and classify objects of 20 predetermined classes, determining their bounding boxes and classes. Caltech was then chosen because of its relevance in today's research in the automotive area.

B. Devices Under Test

In this study, we consider NVIDIA GPUs from three different product families: K40, Tegra X1, and Titan X.

K40 GPUs are designed with the $Kepler$ architecture in a 28nm standard CMOS technology. This model has 2880 CUDA cores, and there are a total of 15 Streaming Multiprocessors (SMs) and 192 CUDA cores within each SM. The register files, shared memory, L1 and L2 caches are SECDED protected, while read-only data cache is parity protected.

The **Tegra X1** is an embedded SOC designed with a $Maxwell$ based GPU and an ARM quad core. It is built in a 20nm standard CMOS technology. The TX1 GPU has 256 CUDA cores, which are divided in two SMs.

Titan X is designed with the novel $Pascal$ architecture, with a 16nm FinFET technology. It has 3584 CUDA cores split into 24 SMs. Titan X was explicitly designed to be more efficient in executing neural networks operations.

It is worth noting that the three different transistor layouts will impact the radiation-induced error rate. Additionally, the architectural differences will affect the way low-level faults will propagate until the output. One of the scopes of our experimental study is to evaluate both the cited aspects.

C. Neutron Beam Test Experimental Set Up

Radiation experiments were performed at the ChipIR facility at the Rutherford Appleton Laboratory (RAL) in Didcot, UK and the Los Alamos National Laboratory (LANL) Los Alamos Neutron Science Center (LANSCE) Irradiation of Chips and Electronics House. ChipIR and LANSCE provide a white neutron source that emulates the energy spectrum of the atmospheric neutron flux between 10 and 750 MeV.

The available neutron flux was between 10^6 and $10^7 n/(cm^2 \times s)$ for energies above 10 MeV. The experimental flux is 5-6 orders of magnitude higher than the terrestrial one (which is $13n/(cm^2 \times h)$ at sea level [20]). This means that in one hour of experiments we can collect data that corresponds to more than 100 years of utilization of GPUs in the natural environment. We have carefully designed the experiments to ensure that the probability of more than one neutron generating a failure in a single code execution remains practically

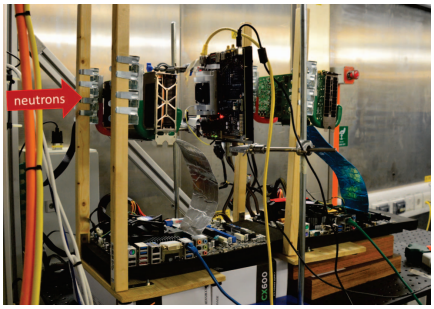


Fig. 2. Radiation test setup inside the ChipIR facility, RAL, Didcot, UK.

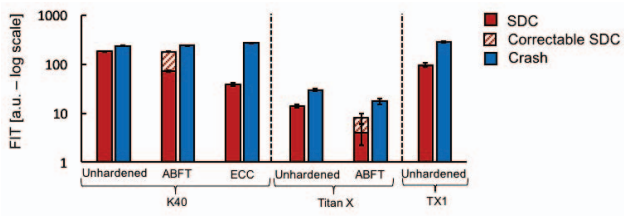


Fig. 3. SDC and Crash normalized FIT for all tested architectures and configurations.

negligible. The observed error rates were lower than 10^{-3} errors/execution. Since a much lower neutron flux may hit a GPU in a realistic environment, it is highly likely not to have more than one corruption during a single execution. We can, therefore, scale the experimental data in the natural radioactive environment without introducing artificial behaviors.

A host computer (or the embedded ARM processor in the TX1) initializes the test, sending a pre-selected input to the GPU, and gathers the computation results. When results are available, they are compared with a pre-computed golden output. When a mismatch is detected, the execution is marked as affected by a *Silent Data Corruption (SDC)* and data is stored for post-processing. A software and a hardware watchdog were included in the setup to detect *application crash* and *system hang*. Fig. 2 shows part of our experimental setup at ChipIR. We have irradiated a total of 3 K40s, 1 Titan X, and 6 TX1s.

V. Reliability Evaluation of Neural Networks

In this section, we present and compare our experimental results on three different GPU architectures. We evaluate both the radiation-induced error rate and Precision-Recall of the corrupted outputs, to distinguish between tolerable and critical errors.

A. Darknet Radiation Sensitivity

Our beam experiment setup allows us to evaluate the realistic radiation-induced error rate of Darknet, and to compare the measured sensitivities of the considered architectures. Additionally, we will try to draw general conclusions on the reliability of the tested neural network.

TABLE I
Normalized FIT

	SDC [a.u.]	Crashes [a.u.]
K40 Unhardened	183.25	236.70
K40 ABFT	179.42	239.95
K40 ECC	38.61	270.30
Titan X Unhardened	14.04	29.83
Titan X ABFT	8.00	17.33
Tegra X1	96.17	288.51

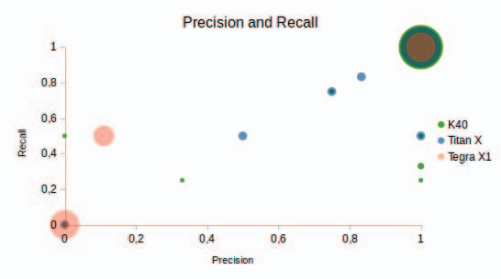
To evaluate the application sensitivity, we calculate the cross section dividing the number of observed errors by the received neutron fluence ($neutrons/cm^2$). As such, cross section comes with the unit of an area (cm^2). By multiplying the cross section with the expected neutron flux at which the device will operate, we obtain a realistic evaluation of the error rate, expressed in Failure In Time (FIT), i.e. errors per 10^9 hours of operation.

In Table I we report the normalized FIT for all the tested configurations. K40 is the only tested GPU which has ECC-protected memories. For the K40 we compare three configurations: Unhardened (i.e., without ECC or any software reliability), ABFT (without ECC but with software error correction), and with ECC enabled. The Titan X was tested using Unhardened mode and ABFT (there is no ECC implemented in the Titan X). The Tegra X1 was tested only on Unhardened configuration due to limited beam time. Data was normalized, so to protect our industrial partners while still allowing a direct comparison among the configurations. Figure 3 shows Darknet normalized FIT (please mind that the y-axis is in log scale) for the three different architectures. Experimental data is presented with Poisson's 95% confidence intervals, and results are shown separately for SDC and Crashes.

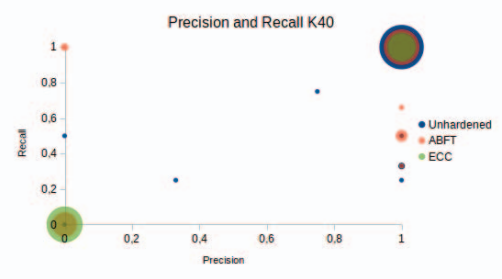
B. Comparison among different architectures

From Figure 3 it is clear that the radiation sensitivity of Darknet strongly depends on the architecture. The error rate of the Unhardened version of Darknet, in fact, changes in about one order of magnitude across the considered architectures. Titan X is the most reliable architecture, as its FIT rates are about $10x$ lower than K40's. The main reasons for Pascal devices to be more reliable are: (1) 3-D transistors have shown an $10x$ reduced per bit sensitivity to neutron compared to standard planar devices [35]. The raw resources corruption probability for the Titan X (built using FinFET technology) is then expected to be lower than that for the K40 (built in standard CMOS technology). (2) Pascal architecture has been explicitly designed and optimized for neural network execution. Titan X is then likely to use its resources in a more efficient (and, eventually, reliable) way compared to the K40. Tegra X1 has an error rate which is about half the one of the K40. This is mainly caused by the much smaller area of the Tegra X1 and by the different transistor layout (20nm for the TX1 and 28nm for the K40).

The crash rate is higher than the SDC one for all the tested GPUs. Crashes are $1.2x$ more likely to happen than SDCs



(a) Comparison among architectures



(b) Precision and Recall for K40

Fig. 4. Experimentally obtained values for Precision and Recall for all tested platforms. Values are calculated comparing corrupted outputs with the radiation-free output. 4a All Precision and Recall values for the tested devices: Tegra X1, K40 and Titan X. 4b Precision and Recall values comparing three different configurations: Unhardened, ABFT, and ECC for K40.

on the K40, $2.1x$ on the Titan X and $3x$ on TX1. Neural networks have already been demonstrated to be intrinsically resilient to SDCs [36], [37], [38]. Even if one or more neurons are compromised, a neural network could potentially provide a correct output. On object detection operations, all GPU kernels have a high level of reuse, which means that several device-host synchronizations are required. A radiation-induced error on those synchronizations could potentially lead the GPU to crash or the system to hang. As a result, while a significant portion of SDCs could be masked in the neural network, crashes could still undermine the device reliability. Finally, the higher crash rate for the TX1 is not surprising as the embedded host processor (the ARM) was irradiated while the host CPU for Titan X and K40 were carefully placed out of the beam. Corruption in the host processor or the operating system is likely to generate a crash or a hang and will hardly lead to an SDC [39].

Even if crashes are more frequent than SDCs, they could be considered less critical, since crashes could be, at least, detected [40], [41], [42]. However, crash detection in a real-time system must guarantee that deadlines are met. That is, the system must be able to recover before causing any harm to the environment [43], [44], [45].

C. ABFT and ECC efficiency evaluation

Figure 3 also shows the error rate of the K40 when ECC is enabled and of both the K40 and Titan X protected with the ABFT strategy described in Section III. ECC has already been demonstrated to be effective in reducing of about one order of magnitude the SDC error rate of GPUs while increasing of about 50% their crash rate [26]. For the K40, ECC reduces the SDC rate of Darknet of more than $4x$. ECC is not as effective as for other codes probably because neural networks are intrinsically resilient to data errors. Crashes are increased by about 20% when ECC is enabled. Interestingly, even with ECC-protected memories, the K40 is less reliable than the unhardened Titan X, which is a highly promising result for the Pascal architecture.

Our ABFT procedure is simple and, as such, does not significantly affect the Darknet SDC or crash error rate. However, the ABFT is capable of detecting and correcting a significant portion of SDCs. About 60% of SDC are correctable for the K40 and 50% for the Titan X. It is worth noting that this result is achieved protecting only a portion of Darknet (i.e., $sgemm$ kernels). While ECC is shown to be more efficient than ABFT, we believe that our strategy is a promising solution to harden neural networks in software. In the future, we plan to extend our ABFT and analyze errors propagation through architectural-level fault injection to detect the cause of the undetected SDCs.

D. Error Analysis

In the image processing community, Precision and Recall are widely used to compare the BBs (Bounding Boxes, i.e. detected objects) found by the algorithm with the ground truth, to assess the quality of a given classifier [46]. Here we use Precision and Recall to evaluate how radiation affects Darknet execution comparing the experimental output with the radiation-free one (not with the ground truth).

Precision and Recall are given by:

$$Precision = \frac{TP}{TP + FP} \tag{2}$$

$$Recall = \frac{TP}{TP + FN} \tag{3}$$

where TP is the number of *True Positives* (pedestrians that were detected), FP is the number of *False Positives* (outcomes of the classifier that do not correspond to a pedestrian), and FN is the number of *False Negatives* (a pedestrian that was not detected by the classifier). To qualify experimental results, we consider a BB n in the radiation-corrupted output as TP if, for any BB m of the radiation-free output the following condition is verified:

$$Jaccard\_similarity(n, m) > T_J \tag{4}$$

where T_J is the acceptance threshold. Otherwise, we consider n as an FP. If for a given BB m of the radiation-free output there is no BB n on the corrupted output which satisfies this condition, an FN is detected.

In the image processing community, T_J is an arbitrary threshold, with $0 \leq T_J \leq 1$. Values of T_J close to 1 impose the classifier to be extremely precise, i.e. BBs detected have to match the ground truth exactly, while values of T_J closer to 0 relax detection precision. In the reliability evaluation context, any corruption that preserves Equation 4 is not to be considered significant, as BBs are sufficiently similar to the radiation-free output. When Equation 4 is not verified, it means that radiation induces an FP or FN (i.e., additional or missed pedestrian, respectively). As such, $T_J = 1$ imposes any radiation-induced error to be marked as critical while values of T_J close to 0 allow most corruptions to be acceptable. We choose $T_J = 0.5$ to compare Darknet radiation corrupted output with the radiation-free output, following the T_J trade-off discussion presented in [46], which is valid independently from the source of detection imprecision (intrinsic algorithm detection imprecision or radiation-induced corruption).

When Equation 4 is not preserved for a given BB on the output, we mark it as corrupted and measure its Precision and Recall relative to the radiation-free output. The Recall rate provides the fraction of existing pedestrians that were detected by the classifier, even in the event of a radiation-induced error. Hence, $R = 100\%$ means that all pedestrians were successfully detected, even if Equation 4 is not preserved. On the other hand, Precision measures the fraction of the produced detection by the classifier that relates to a pedestrian, so that $P = 100\%$ means that all output generated by the classifier correspond to pedestrians.

Figures 4b and 4a show the Precision and Recall for all the observed SDCs in the K40, Titan X, and Tegra X1 obtained comparing the radiation-corrupted output with the expected output.

Precision Values are represented on the x-axis, and Recall values are represented on the y-axis. As there may be several corrupted executions with the same values of Precision and Recall, we use bubbles with a diameter directly proportional to the percentage of SDCs with that value of Precision and Recall. Bigger bubbles imply that a higher percentage of errors have that value of Precision and Recall.

Most errors show both Precision and Recall equal to 1 in Figure 4, that is, corrupted executions that result in values of Precision equal to 1 and Recall equal to 1 will be represented on the same bubble. For those executions, radiation corruption is not critical, as the error was insufficient to modify any BB in a way that refutes Equation 4. We recall that T_J in equation 4 is arbitrary. Results presented here are obtained with $T_J = 0.5$. As mentioned earlier, if Precision is lower than 100%, some objects were wrongly detected by the Neural Network, while if Recall is lower than 100%, some objects were missed. Any Precision and Recall values lower than 100% are potentially critical.

Figure 4 shows that Precision and Recall values strongly depend on the architecture. Despite the fact that all architectures produced at least one SDC with precision and recall equal to zero, Precision and Recall patterns are different on all tested devices. From our results, it seems that the K40 and the Titan X follow a similar trend, having most errors with Precision and Recall equal to 100% and then gradually reducing both Precision and Recall equally, until reaching 0%. Then, some errors preserve Precision to 100% and gradually reduce Recall. It is worth noting that having Recall lower than 100% is to be considered more critical than having Precision lower than 100%. While Precision values lower than 100% could potentially lead to unnecessary vehicle stops (as Darknet detects a non-existing object), which may cause inconveniences more often than accidents, Recall values lower than 100% indicate missed objects, which may lead the vehicle not to stop when it should, likely resulting in accidents.

The Tegra X1, unlikely the K40 and the Titan X, has most of its errors being either not critical (both Precision and Recall equal to 100%) or extremely critical (Precision and Recall equal to 0%). This is probably due to the small embedded architecture of the Tegra X1, which requires the same hardware to perform several operations on the same layer. An error is then likely to impact the detection significantly.

Figure 4b shows the differences on the different K40 configurations: unhardened, ECC and ABFT. While the ABFT version may present many extremely critical errors (Precision and Recall equal to 0%), all such errors are detected on the ABFT checksum verification, and thus, can be corrected.

Comparing Figures 4b and 4a it seems that the hardening techniques (either architectural or software) do not change significantly the overall distribution of output errors, just their rates.

It is reasonable to believe that an SDC on the first convolutional layers could directly impact Precision and recall values. The final part of Darknet execution is classification, performed by two fully connected layers. Fully connected neural networks are known to be fault tolerant methods due to its parallel characteristics and the intrinsic extremely low data interdependency [36], [37], [38]. However, on CNNs, the final fully connected layers are fed by the first convolutional layers, which could bring corrupt data produced by an SDC, leading to a wrong classification. We are injecting errors at the architectural level and checking the correctness of the results at each layer to understand error propagation in neural networks better.

VI. Conclusion

In this paper, we propose a first experimental evaluation of Darknet neural network reliability. With metrics that derive from the image processing community, we investigate the behavior of a CNN on GPUs exposed to atmospheric-like neutrons. As most Neural Networks, Darknet is robust against SDCs, while it experiences a nonnegligible crash rate. Our analysis helps to identify those errors that are critical to a real system application, which are a small fraction of the total. On the K40, our ABFT technique could detect the most critical errors caused on GEMM, demonstrating it to be a robust fault tolerance technique.

In the future, we plan to evaluate the fault tolerance of Darknet against other CNNs, like Faster R-CNN. We also aim

to compare Darknet with built-in libraries, like CUDA Deep Neural Network (cuDNN).

VII. Acknowledgement

This work was supported by the EU H2020 Programme and MCTI/RNP-Brazil under the HPC4E Project. Tested K40 boards were donated thanks to Steve Keckler, Timothy Tsai, and Siva Hari from NVIDIA.

References

[1] B. Van Essen, H. Kim, R. Pearce, K. Boakye, and B. Chen, "Lbann: Livermore big artificial neural network hpc toolkit," in *Proceedings of the Workshop on Machine Learning in High-Performance Computing Environments*, ser. MLHPC '15. New York, NY, USA: ACM, 2015, pp. 5:1–5:6. [Online]. Available: http://doi.acm.org/10.1145/2834892.2834897

[2] S. U. Amin, K. Agarwal, and R. Beg, "Genetic neural network based data mining in prediction of heart disease using risk factors," in *2013 IEEE Conference on Information Communication Technologies*, April 2013, pp. 1227–1231.

[3] J. Redmon, S. K. Divvala, R. B. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," *CoRR*, vol. abs/1506.02640, 2015. [Online]. Available: http://arxiv.org/abs/1506.02640

[4] 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

[5] V. E. Neagoe, A. D. Ciotec, and A. P. Brar, "A concurrent neural network approach to pedestrian detection in thermal imagery," in *2012 9th International Conference on Communications (COMM)*, June 2012, pp. 133–136.

[6] Nvidia, "Tegra k1 techinical reference manual," *DP-06905-001-v03p*, 2014.

[7] R. R. Lutz, "Analyzing software requirements errors in safety-critical, embedded systems," in *Requirements Engineering, 1993., Proceedings of IEEE International Symposium on*, Jan 1993, pp. 126–133.

[8] 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–.

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

[10] R. Baumann, "Radiation-induced soft errors in advanced semiconductor technologies," *Device and Materials Reliability, IEEE Transactions on*, vol. 5, no. 3, pp. 305–316, Sept 2005.

[11] P. Rech, C. Aguiar, C. Frost, and L. Carro, "An Efficient and Experimentally Tuned Software-Based Hardening Strategy for Matrix Multiplication on GPUs," *Nuclear Science, IEEE Transactions on*, vol. 60, no. 4, pp. 2797–2804, 2013.

[12] D. Ciregan, U. Meier, and J. Schmidhuber, "Multi-column deep neural networks for image classification," in *2012 IEEE Conference on Computer Vision and Pattern Recognition*, June 2012, pp. 3642–3649.

[13] D. C. Cireşan, A. Giusti, L. M. Gambardella, and J. Schmidhuber, "Deep neural networks segment neuronal membranes in electron microscopy images," in *Proceedings of the 25th International Conference on Neural Information Processing Systems*, ser. NIPS'12. USA: Curran Associates Inc., 2012, pp. 2843–2851. [Online]. Available: http://dl.acm.org/citation.cfm?id=2999325.2999452

[14] C. Szegedy, A. Toshev, and D. Erhan, "Deep neural networks for object detection," in *Advances in Neural Information Processing Systems 26*, C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger, Eds. Curran Associates, Inc., 2013, pp. 2553–2561. [Online]. Available: http://papers.nips.cc/paper/5207-deep-neural-networks-for-object-detection.pdf

[15] A. Angelova, A. Krizhevsky, V. Vanhoucke, A. Ogale, and D. Ferguson, "Real-time pedestrian detection with deep network cascades," in *Proceedings of BMVC 2015*, 2015.

[16] P. Luo, Y. Tian, X. Wang, and X. Tang, "Switchable deep network for pedestrian detection," in *2014 IEEE Conference on Computer Vision and Pattern Recognition*, June 2014, pp. 899–906.

[17] D. Ribeiro, A. Mateus, J. C. Nascimento, and P. Miraldo, "A real-time pedestrian detector using deep learning for human-aware navigation," *CoRR*, vol. abs/1607.04441, 2016. [Online]. Available: http://arxiv.org/abs/1607.04441

[18] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman, "The pascal visual object classes (voc) challenge," *International Journal of Computer Vision*, vol. 88, no. 2, pp. 303–338, 2010. [Online]. Available: http://dx.doi.org/10.1007/s11263-009-0275-4

[19] D. Oliveira, P. Rech, H. Quinn, T. Fairbanks, L. Monroe, S. Michalak, C. Anderson-Cook, P. Navaux, and L. Carro, "Modern gpus radiation sensitivity evaluation and mitigation through duplication with comparison," *Nuclear Science, IEEE Transactions on*, vol. 61, no. 6, pp. 3115–3122, Dec 2014.

[20] JEDEC, "Measurement and Reporting of Alpha Particle and Terrestrial Cosmic Ray-Induced Soft Errors in Semiconductor Devices," JEDEC Standard, Tech. Rep. JESD89A, 2006.

[21] S. Buchner, M. Baze, D. Brown, D. McMorrow, and J. Melinger, "Comparison of error rates in combinational and sequential logic," *Nuclear Science, IEEE Transactions on*, vol. 44, no. 6, pp. 2209–2216, 1997.

[22] N. Mahatme, T. Jagannathan, L. Massengill, B. Bhuva, S.-J. Wen, and R. Wong, "Comparison of Combinational and Sequential Error Rates for a Deep Submicron Process," *Nuclear Science, IEEE Transactions on*, vol. 58, no. 6, pp. 2719–2725, 2011.

[23] N. DeBardeleben, S. Blanchard, L. Monroe, P. Romero, D. Grunau, C. Idler, and C. Wright, "GPU Behavior on a Large HPC Cluster," *6th Workshop on Resiliency in High Performance Computing (Resilience) in Clusters, Clouds, and Grids in conjunction with the 19th International European Conference on Parallel and Distributed Computing (Euro-Par 2013), Aachen, Germany,*, August 26-30 2013.

[24] H. J. Wunderlich, C. Braun, and S. Halder, "Efficacy and efficiency of algorithm-based fault-tolerance on gpus," in *On-Line Testing Symposium (IOLTS), 2013 IEEE 19th International*, July 2013, pp. 240–243.

[25] L. A. B. Gomez, F. Cappello, L. Carro, N. DeBardeleben, B. Fang, S. Gurumurthi, K. Pattabiraman, R. Rech, and M. S. Reorda, "GPGPUs: How to Combine High Computational Power with High Reliability," in *2014 Design Automation and Test in Europe Conference and Exhibition*, Dresden, Germany, 2014.

[26] D. A. G. 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, March 2016.

[27] M. Breuer, S. Gupta, and T. M. Mak, "Defect and error tolerance in the presence of massive numbers of defects," *Design Test of Computers, IEEE*, vol. 21, no. 3, pp. 216–227, May 2004.

[28] 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.

[29] N. Seifert, X. Zhu, and L. W. Massengill, "Impact of scaling on soft-error rates in commercial microprocessors," *Nuclear Science, IEEE Transactions on*, vol. 49, no. 6, pp. 3100–3106, 2002.

[30] J. Tan, N. Goswami, T. Li, and X. Fu, "Analyzing soft-error vulnerability on GPGPU microarchitecture," in *Workload Characterization (IISWC), 2011 IEEE International Symposium on*, Nov 2011, pp. 226–235.

[31] S. Mitra, "System-Level Single-Event Effects," IEEE Nuclear and Space Radiation Effects Conference, NSREC 2012 Short Course, 2010.

[32] K.-H. Huang and J. Abraham, "Algorithm-based fault tolerance for matrix operations," *Computers, IEEE Transactions on*, vol. C-33, no. 6, pp. 518–528, June 1984.

[33] R. Freivalds, "Fast probabilistic algorithms," in *Mathematical Foundations of Computer Science 1979*, ser. Lecture Notes in Computer Science, J. Bev, Ed. Springer Berlin Heidelberg, 1979, vol. 74, pp. 57–69. [Online]. Available: http://dx.doi.org/10.1007/3-540-09526-8_5

[34] P. Dollar, C. Wojek, B. Schiele, and P. Perona, "Pedestrian detection: A benchmark," in *2009 IEEE Conference on Computer Vision and Pattern Recognition*, June 2009, pp. 304–311.

[35] J. Noh, V. Correas, S. Lee, J. Jeon, I. Nofal, J. Cerba, H. Belhaddad, D. Alexandrescu, Y. Lee, and S. Kwon, "Study of neutron soft error rate (ser) sensitivity: Investigation of upset mechanisms by comparative simulation of finfet and planar mosfet srams," *Nuclear Science, IEEE Transactions on*, vol. 62, no. 4, pp. 1642–1649, Aug 2015.

[36] C. H. Sequin and R. D. Clay, "Fault tolerance in artificial neural networks," in *1990 IJCNN International Joint Conference on Neural Networks*, June 1990, pp. 703–708 vol.1.

[37] P. W. Protzel, D. L. Palumbo, and M. K. Arras, "Performance and fault-tolerance of neural networks for optimization," *IEEE Transactions on Neural Networks*, vol. 4, no. 4, pp. 600–614, Jul 1993.

[38] E. B. Tchernev, R. G. Mulvaney, and D. S. Phatak, "Investigating the fault tolerance of neural networks," *Neural Comput.*, vol. 17, no. 7, pp. 1646–1664, Jul. 2005. [Online]. Available: http://dx.doi.org/10.1162/0899766053723096

[39] T. Santini, L. Carro, F. R. Wagner, and P. Rech, "Reliability analysis of operating systems and software stack for embedded systems," *IEEE Transactions on Nuclear Science*, vol. 63, no. 4, pp. 2225–2232, Aug 2016.

[40] M.-L. Li, P. Ramachandran, S. K. Sahoo, S. V. Adve, V. S. Adve, and Y. Zhou, "Understanding the propagation of hard errors to software and implications for resilient system design," *SIGOPS Oper. Syst. Rev.*, vol. 42, no. 2, pp. 265–276, Mar. 2008. [Online]. Available: http://doi.acm.org/10.1145/1353535.1346315

[41] N. Nakka, G. P. Saggese, Z. Kalbarczyk, and R. K. Iyer, *An Architectural Framework for Detecting Process Hangs/Crashes*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2005, pp. 103–121. [Online]. Available: http://dx.doi.org/10.1007/11408901_8

[42] K. Pattabiraman, G. P. Saggese, D. Chen, Z. Kalbarczyk, and R. K. Iyer, "Dynamic derivation of application-specific error detectors and their implementation in hardware," in *2006 Sixth European Dependable Computing Conference*, Oct 2006, pp. 97–108.

[43] G. Candea and A. Fox, "Recursive restartability: Turning the reboot sledgehammer into a scalpel," in *Proceedings of the Eighth Workshop on Hot Topics in Operating Systems*, ser. HOTOS '01. Washington, DC, USA: IEEE Computer Society, 2001, pp. 125–. [Online]. Available: http://dl.acm.org/citation.cfm?id=874075.876408

[44] K. Lee and L. Sha, "Process resurrection: a fast recovery mechanism for real-time embedded systems," in *11th IEEE Real Time and Embedded Technology and Applications Symposium*, March 2005, pp. 292–301.

[45] C.-H. Wu, T.-W. Kuo, and L.-P. Chang, "The design of efficient initialization and crash recovery for log-based file systems over flash memory," *Trans. Storage*, vol. 2, no. 4, pp. 449–467, Nov. 2006. [Online]. Available: http://doi.acm.org/10.1145/1210596.1210600

[46] T. Fawcett, "An introduction to roc analysis," *Pattern recognition letters*, vol. 27, no. 8, pp. 861–874, 2006.