Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 182 (2021) 89–94

Procedia Computer Science

www.elsevier.com/locate/procedia

17th International Learning & Technology Conference 2020 (17th L&T Conference)

The impact of the soft errors in convolutional neural network on GPUs: Alexnet as case study

Khalid Adam[a1], Izzeldin I. Mohd[a], Younis M. Younis[b]

[a]*University Malaysia Pahang, College of Engineering, 26300, Pahang, Malaysia*
[b] *College of IoT Engineering Hohai, 310000 Changzhou, China*

Abstract

Convolutional Neural Networks (CNNs) have been increasingly deployed in many applications, including safety critical system such as healthcare and autonomous vehicles. Meanwhile, the vulnerability of CNN model to soft errors (e.g., caused by radiation induced) rapidly increases, thus reliability is crucial especially in real-time system. There are many traditional techniques for improve the reliability of the system, e.g., Triple Modular Redundancy, but these techniques incur high overheads, which makes them hard to deploy. In this paper, we experimentally evaluate the vulnerable parts of Alexnet mode (e.g., fault injector). Results show that FADD and LD are the top vulnerable instructions against soft errors for Alexnet model, both instructions generate at least 84% of injected faults as SDC errors. Thus, these the only parts of the Alexnet model that need to be hardened instead of using fully duplication solutions.

Keywords: Reliability, Fault Injection, convolutional neural networks, soft errors, GPUs;

1. Introduction

Convolutional Neural Networks (CNNs) are become widely use in computational approach in many fields such as object detection [1] and image classification [2]. CNNs have high massive parallelism structure and require high computational capabilities such as Graphics Processing Units (GPUs). GPUs very suitable to accelerate the CNN-models [3], for example, NVIDIA GPUs used in autonomous robotic surgery [4]. Consequently, the reliability investigation becomes imperative, particularly in safety-critical application, as any performance failure may lead to catastrophic consequences such as medical fields [5]. While the CNN model has been increasingly used in safety critical application such as diagnosis [6] and surgical [7]. However, still the reliability of these models one of the big challenging [8][9][10]. Mainly, reliability disturbance in model come from the soft errors [11]. These errors are able to impact the behaviours of the GPU dramatically [12], thus, may corrupt the data values or logic operations and can cause errors. For instance, application crash or system hang (i.e., Detected Unrecoverable Error, DUE), Silent Data Corruptions (SDCs), nevertheless maybe can cause no observable error and masked [13]. These errors may lead to serious adverse problems, including injurie or death [14]. There are several works on CNNs model reliability issues have been published. Overall, [9] [12] they proposed techniques to mitigation the soft error in GPUs and fault tolerance algorithms. Unfortunately, these techniques have overheads [15] [8] [3]. Convolutional Neural Networks (CNNs) are become widely use in computational approach in many fields such as object detection [1] and image classification [2]. CNNs have high massive parallelism structure and require high computational capabilities such as Graphics Processing Units (GPUs). GPUs very suitable to accelerate the CNN-models [3], for example, NVIDIA GPUs used in autonomous

[*] Corresponding author. Tel.: +601125405294
E-mail address: khalidwsn15@gmail.com

10.1016/j.procs.2021.02.012

robotic surgery [4]. Consequently, the reliability investigation becomes imperative, particularly in safety-critical application, as any performance failure may lead to catastrophic consequences such as medical fields [5]. While the CNN model has been increasingly used in safety critical application such as diagnosis [6] and surgical [7]. However, still the reliability of these models one of the big challenging [8][9][10]. One of the main sources to disturb the reliability in the CNN model come from the soft-errors [11]. These soft-errors impact the behaviors of the DNN accelerator (GPU) dramatically then propagate to application-level (CNN models) [12]. Thus it may produce corrupt in the data values or logic operations and cause errors such as SDCs (data silent data corruption), DUEs (detected unrecoverable errors). In addition, it can be masked (cause no observable errors). [13]. These errors may lead to serious adverse problems, including injurie or death [14]. There are several works on CNNs model reliability issues have been published. Overall, [9] [12] they proposed techniques to mitigation the soft error in GPUs and fault tolerance algorithms. Unfortunately, these techniques have overheads [15] [8] [3].

This paper aims at defining the error propagation in CNNs model to mitigate error propagation in GPUs. To achieve this, we are proposed analysis methodology to inject the model to identify the vulnerability instructions to soft errors. As the first aim, we proposed technique to mitigate soft errors, and we implement it on CNN model Alexnet. The main contributions of this work are: (1) a methodology to evaluate the probability of fault in specific parts of the source code that likely to errors at the output; (2) an extensive analysis of Alexnet characteristics under SASSIFI fault-injection. This paper organized as follows. Section 2 background and previous works. Then in section 3 we analyze and discuss the results. Finally, in section 4 concludes the paper.

2. Background and Previous Works

In this section, we provide a brief background on the convolutional networks an Alexnet. Then, some previous works on the CNNs reliability.

2.1. Convolutional Neural Networks

Recently, CNNs have been used as powerful tool have achieved great success in many issues of machine learning and computer vision [16]. In addition, CNNs a wide range structure type using in Deep Learning architecture [17]. The convolutional operation is key component in CNNs. Therefore, CNNs consists of three different types of layer: convolution layers, subsampling layers, and fully-connected layers as shown in Fig 1 in feed-forward structure. and each of these layers have task in the network. For example, convolutional layers are responsible to extract the features using two key concepts of receptive local fields and mutual weights. Thus, detected the characteristics in exact location is less critical, compare to the relative position to other characteristics is significant. Therefore, a local average and subsampling is performed by the subsequent subsampling layer. A detailed description behind these layers of the CNNs architectural concepts is given in [18].

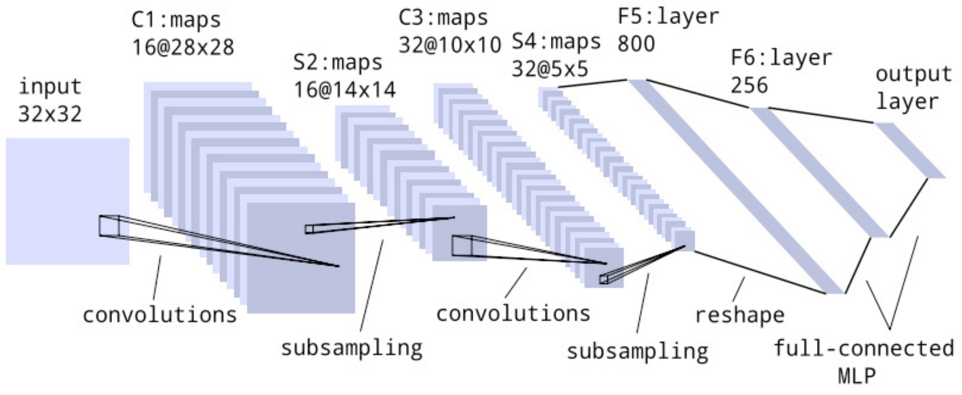

Fig. 1: A simple CNN architecture [19]

2.2. Alexnet

Alexnet, a CNN model developed by [20], is the winner of the ILSVRC-2012 ImageNet competition and has achieved a 15.3 % top five test error rate. The model consists of eight layers of learning, five layers of convolution, and three fully connected layers. This model has been widely used in numerous applications, including healthcare [21], which is a pure safety critical application, for automatic features extraction and classification. The main structure is as of the model shown in figure 2; first convolutional layer filters an input picture of (224x224x3) with 96 kernels of size (11x11x3) with a 4-pixel phase. The second convolutional layer takes as its input the output of the first convolutional layer and filters it with 256 kernels of size (5x5x48). The third convolution layer has 384 size (3x3x256) kernels connected to the second convolutional layer's (normalized, pooled) outputs. The fourth convolution layer has 384 kernels of size (3x3x192), and the fifth convolution layer has 256 kernels of size (3x3x192). In the fully connected layers, there are 4096 neurons. Thus, our goal is to analyze and evaluate the vulnerability and resilience of instructions, to distinguish between reliable and vulnerable instructions to soft errors.

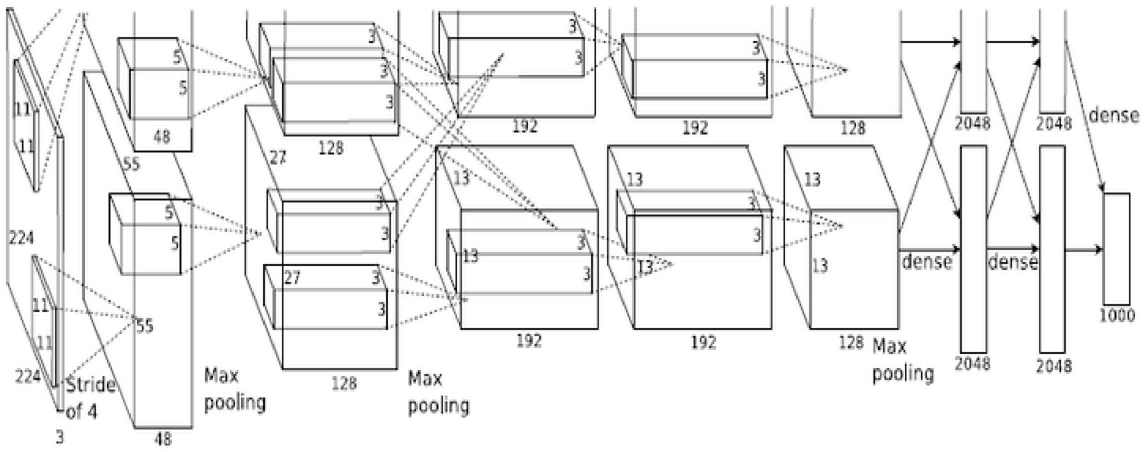

Fig. 2: Architecture of Alexnet: Convolution, max-pooling, LRN and fully connected (FC) layer

2.3. Previous Works

Previous studies [12][15][8] the accelerators have been used to evaluating and measure the reliability of applications and devices. In addition, different abstraction levels are also generally used for estimation the resilience of applications and hardware, by using software fault injection techniques. While new architectures such as DNN accelerators (e.g., GPUs) came into the market for high-performance computing, intense research was conducted to understand and enhance their reliability. Previous studies indicated early assessment of reliability of parallel devices [10] [3]. Also reported is the reliability evaluation of GPUs running high-performance computing codes and safety critical applications through radiation-induced experiments [22] or error injection [23]. Within modern architectures, preliminary studies find certain inherent weaknesses within reliability [14][24]. The accuracy of kernels or multiple parallel threads can be affected by a single particle-induced failure in the parallel computer shared memory or hardware scheduler, which leads to corruption in output values.[10]. Additionally, a corrupted single thread can feed thousands of future parallel threads with faulty data, resulting in output errors [11]. Consequently, the vast majority of transient errors affects output components. Therefore, to best of our knowledge, this is first study that characterizes the reliability of Alexnet on GPUs. This paper, analysis of the classification (Alexnet), from perspectives of instruction's vulnerability.

3. Experimental Results and Discussion

3.1. Settings

We used PC with the following configuration in this experiment; GeForce GTX 750 Ti form NVIDIA, core i7, 4 GB main memory and SASSIFI “fault injector”. We inject errors at the GPU to test the Program Vulnerability Factor (PVF), which is the likelihood that the performance of a program will be propagated by a single fault that modifies the outcome of instruction. Thus, to measure PVF of the program we inject errors with IOA and IOV modes. Instruction Output Address (IOA) to study the probability that the address instruction has errors, and Instruction

Output Value (IOV) to study the probability that the value executed instruction has errors. Then, we injected 1000 injections at IOA and IOV and we are interested in faults that are not masked and make their way to the application (e.g., SDC).

3.2. Results

Here we analyze and discuss the results we have obtained using the SASSIFI framework. Specifically, we analyze the vulnerability of various instruction types while the GPU executing AlexNet model. We evaluate the error resilience of individual instructions and their contribution to the overall error resilience of our model. Based on SASSIFI, there are ten instruction groups as follows: (1) Store instructions (STORE), (2) Single-precision floating-point fused multiply-add instructions (FFMA), (3) Load instructions (LD), (4) Integer add and multiply instructions (IADD_IMUL), (5) Instructions that write to condition-code registers (CC), (6) Instructions that write to general-purpose registers (GPR), (7) Instructions that write to predicate registers (PR), (8) Single-precision floating-point add and multiply instructions (FADD_FMUL), (9) Integer fused multiply and add instructions (MAD), and (10) Register (SETP). Fig. 3 shows the results obtained by injecting faults into instruction with IOA mode. It is important to notice that for IOA injections, only two instructions (i.e., GPR and STORE) are used to accomplish the task. By taking a look at Fig. 1, we can observe that most of the faults have been injected into GPR instruction were masked (58%), while still producing a considerable amount of SDC (28.3%) and DUEs (13.7%) errors. STORE instruction group, on the other hand, seems more vulnerable than GPR. Where it masked only (39.9%) of the errors, while produced (46.7%) SDC and (13.4%) DUE errors.

Fig. 4 we show the results obtained by injecting faults into instruction with IOV mode. In contrast to IOA mode, with this injection site, we use all the instruction groups (as listed above) to evaluate the error resilience of the model. Starting with FADD and LD, they appear to be the most vulnerable instructions at all. Where both of them produced at least 84% SDC errors, which is an extremely high percentage and it is unacceptable for safety- compliant systems. Injections with these two instructions masked only 15.8% and 11.8% of the errors, respectively. While they do not produce DUEs at all. These findings make FADD and LD are the top vulnerable instructions against soft errors. There are two reasons as we believe: (1) as CNN models are basically built of many layers (i.e., computationally hungry), they load data (i.e., with LD instructions) as many times as number from layers in the model; (2) most of the CNN architectures including AlexNet are greatly rely on FADD instructions in their operations. This require attention from researchers who work on GPU architecture designing or CNN model building to take the reliability issue into considerations. Especially when the CNN model is intended to be used in some safety-critical environments. IADD, MAD, GPR instructions have moderate error resilience, where each of which produced number of SDC errors on the average of 25% and masked almost the same percentage. Injecting fault into the value (IOV) of the STORE instruction is not as injecting into its address (IOA). At least 60% of the injected faults became SDCs, while masking about 38% but surprisingly there is no DUE has been generated. SETP and FFMA instructions also have approximately the same characteristics as STORE, they produce 61% and 39% SDCs, respectively.

It is worth mentioning that CC and PR instructions are one-bit operations. In other words, they only perform toggle between zero and one, thus, we can only inject single-bit-flip faults into them. From a reliability perspective, every single fault has been injected to CC instruction was masked, therefore, there is no SDC or DUE has been observed. While for PR, it has a similar behavior as CC in terms of DUEs. Nevertheless, it produces a large amount of SDC errors (58%). To conclude it, as most of the instructions generate errors, our findings highly suggest developing a selective-based protecting technique to mitigate soft errors for the top vulnerable instructions.

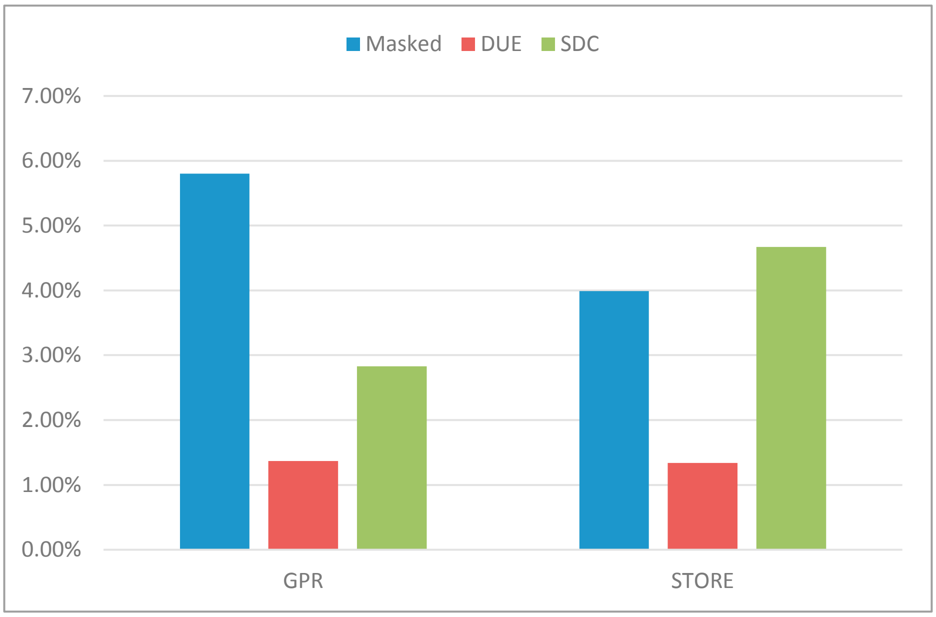

Fig. 3. Instruction output address groups (IOA mode)

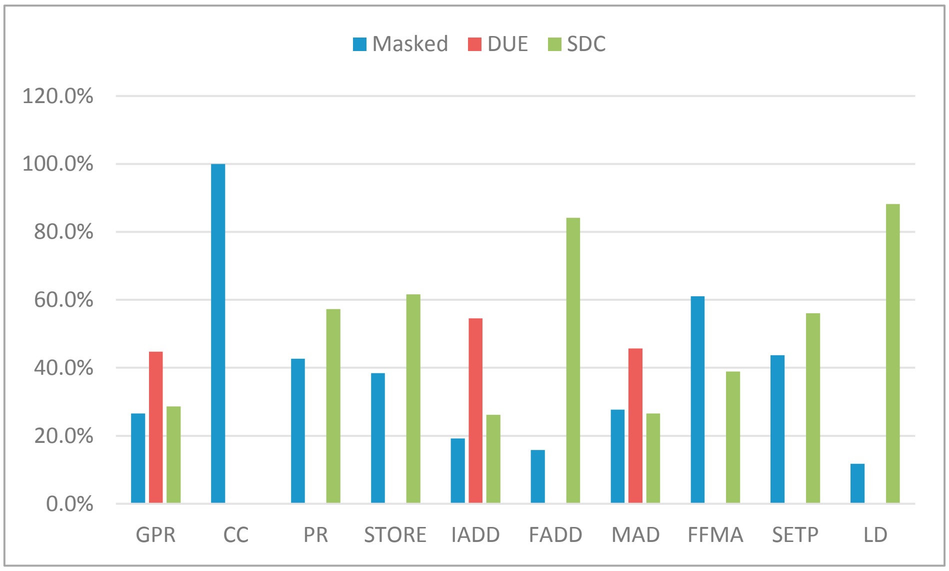

Fig. 4. Instruction output value groups (IOV mode)

4. Conclusions

In this paper, we have analyzed the error resilience of Alexnet, a well-known CNN model, from the perspective of architecture-level instructions. Our analysis showed that Alexnet is more prone to SDC than DUE errors, which are more crucial because they modify model's final output. Accordingly, we found that FADD and LD are the top vulnerable instructions against soft errors for Alexnet model, both instructions generate at least 84% of injected faults as SDC errors. While other instructions including STORE, SETP and FFMA have relatively high vulnerability, and thus, they produce amount of SDC errors that could be addressed. CC instructions, on the other hand, have shown to have very high resilience against soft errors, they do not produce SDC or DUE errors. Therefore, GPU low-level instructions can properly be selected to mitigate SDC or DUE errors for Alexnet model.

References

[1] Ren S, He K, Girshick R, et al. (2017) "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks". *IEEE Transactions on Pattern Analysis and Machine Intelligence*,; **39**: (**6**): 1137–1149.

[2] Vasuki A, Govindaraju S. (2017) "Deep neural networks for image classification". In: *Deep Learning for Image Processing Applications*, pp. 27–49.

[3] Strigl D, Kofler K, Podlipnig S. (2010) "Performance and scalability of GPU-based convolutional neural networks". In: *2010 18th Euromicro Conference on Parallel, Distributed and Network-based Processing*. IEEE, pp. 317–324.

[4] Deshmukh NP, Kang HJ, Billings SD, et al. (2014) "Elastography using multi-stream GPU: an application to online tracked ultrasound elastography, in-vivo and the da Vinci surgical system". *PloS one*,; **9**: (**12**): e115881.

[5] Pourbabaee B, Roshtkhari MJ, Khorasani K. (2018) "Deep Convolutional Neural Networks and Learning ECG Features for Screening Paroxysmal Atrial Fibrillation Patients". *IEEE Transactions on Systems, Man, and Cybernetics: Systems*,; **48**: (**12**): 2095–2104.

[6] Miotto R, Wang F, Wang S, et al. (2017) "Deep learning for healthcare: Review, opportunities and challenges". *Briefings in Bioinformatics*,; **19**: (**6**): 1236–1246.

[7] Choi B, Jo K, Choi S, et al. (2017) "Surgical-tools detection based on Convolutional Neural Network in laparoscopic robot-assisted surgery". In: *2017 39th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*. Ieee, pp. 1756–1759.

[8] Li G, Hari SKS, Sullivan M, et al. (2017) "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*. ACM, p. 8.

[9] Azizimazreah A, Gu Y, Gu X, et al. (2018) "Tolerating Soft Errors in Deep Learning Accelerators with Reliable On-Chip Memory Designs". *2018 IEEE International Conference on Networking, Architecture and Storage, NAS 2018 - Proceedings*,; 1–10.

[10] Li G, Pattabiraman K. (2016) "Understanding Error Propagation in GPGPU Applications". (**November**).

[11] T D, T L. (2019) "Error Correction For Soft Errors". *Journal of Electrical & Electronic Systems*,; **07**: (**04**). Epub ahead of print 2019. DOI: 10.4172/2332-0796.1000276.

[12] de Oliveira DAG, Pilla LL, Santini T, et al. (2015) "Evaluation and mitigation of radiation-induced soft errors in graphics processing units". *IEEE Transactions on Computers*,; **65**: (**3**): 791–804.

[13] Rech P, Aguiar C, Ferreira R, et al. (2012) "Neutron-induced soft errors in graphic processing units". In: *2012 IEEE Radiation Effects Data Workshop*. IEEE, pp. 1–6.

[14] Ferrarese A, Pozzi G, Borghi F, et al. (2016) "Malfunctions of robotic system in surgery: Role and responsibility of surgeon in legal point of view". *Open Medicine*,; **11**: (**1**): 286–291.

[15] Salami B, Unsal O, Cristal A. "On the Resilience of RTL NN Accelerators : Fault Characterization and Mitigation".

[16] Kim Y. (2014) "Convolutional neural networks for sentence classification". *arXiv preprint arXiv:14085882*,.

[17] Sze V, Member S, Chen Y, et al. "Efficient Processing of Deep Neural Networks : A Tutorial and Survey". 1–32.

[18] Rawat W, Wang Z. (2017) "Deep convolutional neural networks for image classification: A comprehensive review". *Neural computation*,; **29**: (**9**): 2352–2449.

[19] Yang Y, Luo H, Xu H, et al. (2016) "Towards Real-Time Traffic Sign Detection and Classification". *IEEE Transactions on Intelligent Transportation Systems*,; **17**: (**7**): 2022–2031.

[20] Krizhevsky A, Sutskever I, Hinton GE. (2012) "Imagenet classification with deep convolutional neural networks". In: *Advances in neural information processing systems*, pp. 1097–1105.

[21] Zhang J, Xie Y, Wu Q, et al. (2019) "Medical image classification using synergic deep learning ☆". **54**: 10–19.

[22] Asghari SA, Kaynak O, Taheri H. (2014) "An Investigation into Soft Error Detection Efficiency at Operating System Level". **2014**. Epub ahead of print 2014. DOI: 10.1155/2014/506105.

[23] Hari SKS, Tsai T, Stephenson M, et al. (2017) "SASSIFI: An architecture-level fault injection tool for GPU application resilience evaluation". *ISPASS 2017 - IEEE International Symposium on Performance Analysis of Systems and Software*,; (**1**): 249–258.

[24] Oliveira DAG, Member S, Pilla LL, et al. (2015) "Evaluation and Mitigation of Radiation-Induced Soft Errors in Graphics Processing Units". **9340**: (**c**): 1–14.