APPRAISER: DNN Fault Resilience Analysis Employing Approximation Errors

Mahdi Taheri[1], Mohammad Hasan Ahmadilivani[1], Maksim Jenihhin[1], Masoud Daneshtalab[1,2], and Jaan Raik[1]

[1]Tallinn University of Technology, Tallinn, Estonia
[2]Mälardalen University, Västerås, Sweden
[1]mahdi.taheri@taltech.ee

Abstract—Nowadays, the extensive exploitation of Deep Neural Networks (DNNs) in safety-critical applications raises new reliability concerns. In practice, methods for fault injection by emulation in hardware are efficient and widely used to study the resilience of DNN architectures for mitigating reliability issues already at the early design stages. However, the state-of-the-art methods for fault injection by emulation incur a spectrum of time-, design- and control-complexity problems. To overcome these issues, a novel resiliency assessment method called AP-PRAISER is proposed that applies functional approximation for a non-conventional purpose and employs approximate computing errors for its interest. By adopting this concept in the resiliency assessment domain, APPRAISER provides thousands of times speed-up in the assessment process, while keeping high accuracy of the analysis. In this paper, APPRAISER is validated by comparing it with state-of-the-art approaches for fault injection by emulation in FPGA. By this, the feasibility of the idea is demonstrated, and a new perspective in resiliency evaluation for DNNs is opened.

Index Terms—Deep Neural Networks, approximate computing, fault injection, reliability, resiliency assessment

Fig. 1: DNN fault resiliency assessment methods: (a) Fault injection by emulation in FPGA; (b) APPRAISER approach using errors by AxC units.

I. INTRODUCTION

In recent years, Deep Neural Networks (DNNs) surpassed human-level precision [1] that made them attractive for several safety-critical applications such as autonomous driving [2], [3].

Faults that can be caused by soft errors, aging, etc., are the source of threatening the reliability of DNN inference hardware accelerators. Here, *soft errors*, are of particular concern for researchers in the industry and academia. It is a class of faults caused by ionized particles hitting transistors that can flip a logic value in a memory cell or a logic gate.

In today's applications, network parameters, e.g., weights, occupy most of the inference accelerator's areal footprint, making them natural targets for soft-errors-caused disturbances. Unlike other logic structures, DNNs are known to be relatively resilient to transient faults. However, in practice, such faults still may cause a significant accuracy drop in DNNs because of the large area and memory requirements for the state-of-the-art DNNs accelerators. Although numerous techniques have been proposed recently to evaluate the architectural fault resilience of DNNs, they are still rather costly. Throughout the literature, Fault Injection (FI) is the most commonly used method for resilience evaluation of DNNs.

Fault injection by emulation in hardware, usually in FPGAs, is widely adopted by the industry [4] because of its ability to evaluate real-scale DNN accelerator designs with significantly shorter run times compared to software-based simulation.

However, the state-of-the-art approaches for fault injection by emulation in hardware imply iterative procedures for each injected fault, including numerous extra memory accesses, which make them time-consuming and imply complex implementation. Fig. 1(a) illustrates the execution overheads of the general flow of FI by emulation in hardware. In particular, such an iterative approach is breaking the pipeline and requires a complex FI Controller and an extra FI control interconnect [5]–[8]. Fig. 1 (b) illustrates the proposed approach AP-PRAISER, which allows reducing the fault resiliency assessment overheads. The ability to tolerate the impact of faults on the output accuracy is called *fault resiliency* and, in practice, it is one of the contributors to the final DNN accelerators' reliability [9].

In this paper, our contribution is a novel method of fault resiliency analysis for DNN architectures that applies functional approximation for a non-conventional purpose and harnesses approximate computing errors for its interest. To the best of our knowledge, for the first time, *Approximate Computing*

(AxC) units are adopted to improve the processing time-, design-, and control-complexity for DNN fault resiliency analysis process.

APPRAISER provides a rapid exploration of different options of the network architecture, training, dataset, etc., to study the fault resilience of the DNNs. In particular, it enables efficient analysis of subsequent layers' resilience to faults in the weights of a compromised layer.

The new method has the following advantages:

- It eliminates the need for designing and deploying an extra complex controller for the fault injection procedure. A simple approximate units enabling circuitry (AxC Activator) is employed instead.
- The inference pipeline process executes a batch of inputs with no need to break this process.
- The resilience assessment process is performed without an extra interconnect for weight sampling.
- The proposed approach is not iterative for each potential fault location, unlike the traditional fault injection. Thus, the analysis complexity is vastly reduced.

The rest of the paper is organized as follows. An analysis of Related Works in Section II is followed by the new methodology presented in Section III. The experimental results, along with their discussion, are presented in section IV. Finally, this work is concluded in Section V.

II. RELATED WORKS

The extensive growth of the memory footprint size in today's practical DNN inference HW accelerators increases the chances of soft errors' occurrences causing prediction failures. Even a minor change in the DNN architecture may cause a notable difference in the DNNs' architectural fault resiliency [9]. Evaluating the resiliency of DNNs with FI by emulation in hardware is a practical method used today by the industry. There are several works emulating fault injection on FPGAs as a hardware platform.

Fiji-FIN [8] is one of such DNNs' resiliency evaluation frameworks. It considers the model's accuracy degradation as a metric to study the impact of soft errors on the network's parameters, such as weights and activation. Unfortunately, it implies severe effort for designing the fault injection campaigns. For each single fault injection, the execution of the inference should be halted for manipulating the DNN parameters, and it has to be resumed thereafter. It means that the classification time for a batch of inputs should be interrupted to apply fault injection between the classification process of two consecutive inputs.

A similar method is also used in [6], [7]. These works also propose injecting transient faults into on-chip memories of the design implemented on the FPGA. In these works, the bit stream file of the FPGA is obtained by a High-Level Synthesis (HLS) tool and imported to the FPGA. While the system is running, the faults are generated and injected by the embedded processor and the reliability is evaluated in comparison with the golden model.

In contrast to the works mentioned above, this paper proposes a novel non-iterative fault resilience analysis by exploiting the approximation errors instead of fault injection

It enables keeping the inference pipeline process to be executed on a batch of inputs unbroken.

III. PROPOSED METHOD APPRAISER

The proposed approach for applying errors of approximate computing units for DNN fault resiliency assessment is outlined in Fig. 1(b). An AxC Activator unit on the Processing System (PS) side enables the AxC units to induce errors. These units are AxC multipliers in the multiply-and-accumulate units (MACs), in the targeted (mimicked to be compromised) layer of the DNN. This activator controls the multiplexers on the Programmable Logic (PL) side to switch between the exact implementation of the units (for the functional mode) and the approximated one (for the resiliency assessment mode). Then, the user runs the inference just once for the validation dataset and stores the results of the layers' outputs.

The flow of APPRAISER method is depicted in Fig. 2.

Step 1 is the initialization that includes the selection of the compromised layer (e.g. one by one in the DNN structure), the validation testset (i.e. DNN inputs), and the assumed application-specific fault rate. In Step 2, suitable AxC units are selected. For example, in this work, we used AxC multipliers from the EvoApproxLib library [10]. Further, a set of ExC units are substituted with the AxC units in the network architecture (Step 3). The DNN inference is executed keeping the pipeline of the network and the *DNN output accuracy drop* is reported. It is used as the main DNN fault resilience analysis metric. The more the accuracy drops with the induced errors, the less fault-resilient the given DNN implementation is.

Fig. 2: APPRAISER Methodology

Fig. 3: APPRAISER assessment flow: Compromised layer in the presence of faults in weights vs. layers under resiliency test

In a traditional application of AxC, the approximation of hardware components is based on their inexact implementa-

tion that creates a functionally tolerable mismatch with the specification while providing gains in compute-efficiency. In practice, there is an error induced by approximation that can also be employed to mimic the error caused by a fault in the inputs of a logic circuit that is propagated to the output. Such approximation-induced errors affect their corresponding outputs, which are also connected to several other neurons in subsequent layers (as their activation) (Fig. 3).

The characteristics of the approximation-induced errors can be assessed by several metrics, normalized error, number of flipped bits, and their impact on the neural network classification accuracy drop. In this study, we rely on the following simple set of metrics:

1) *normalized error*: calculated as the average error on the output of each layer by subtracting the neurons' outputs of that layer from the golden output and dividing all the error values to the maximum value;
2) *network accuracy and recall drop*: calculated by executing the network under different circumstances (faulty vs approximated) over the test set;
3) *bitflips in subsequent layers*: calculated by comparing all bits in the next layers' outputs with the golden model and counting the bits that do not match as flipped bits.

The main objective of APPRAISER is the study of the resiliency of DNN architecture layers to faults that might occur in the weights of a compromised layer. By using this method, the user can rapidly explore the options of network architecture, training, dataset, etc., in terms of fault resiliency analysis.

Unlike some other frameworks (e.g., FijiFin [8], APPRAISER does not support assessing the reliability of the network to faults in the activations and DNN neurons and currently is only aimed at resiliency to faults in stored DNN weights. Other limitations are a lower diagnostic capability and implicit correspondence to traditional fault injection based metrics (e.g. in standards).

IV. EXPERIMENTAL RESULTS

A. Evaluation Methodology

The flow to evaluate the proposed method is illustrated in Fig. 4. Here, Steps 1 and 2 repeat the APPRAISER method execution (Fig. 2).

The list of candidate approximate multipliers from the EvoApproxLib library [10] was narrowed down with several relevant metrics adopted from EvoApproxLib with the main focus on two established features (Variance of Error Distance (Var-ED) and Root Mean Square (RMS-ED)) presented in [11]. These two metrics are the most critical approximation-induced errors' features for the performance of an AxC unit in DNNs. Based on these metrics, Mult8s_1KX2 (further referred to as *Mult1*) and Mult8s_1KRC (*Mult2*) multipliers are selected for the experiment.

For the reference part, the fault resiliency evaluation is repeated on the original network instrumented for a state-of-the-art FI method [8] (Step 4). Two fault models are considered in this study:

Fig. 4: APPRAISER evaluation flow

- Injection of a *single bitflip* at a random location in all weight bits of the compromised layer for every input in the DNN validation test set,
- Injection of *double bitflips* in weights of the compromised layer for every input in the DNN validation test set.

For each fault model, the experiment is repeated for 1000 random faults per image are considered to reach the 95% FI confidence level according to the statistical fault injection approach [12] and in the end, the average accuracy of all repetitions is reported. Finally, the DNN accuracy drops as a result of applying approximation and fault injection along with normalized error, and the number of flipped bits are compared (Step 5).

B. Experimental Setup

To evaluate the feasibility of the proposed method, a simple Convolutional Neural Network (CNN) with two convolutional layers, two max-pooling, and one Fully-Connected (FC) layer was implemented and trained. The simulations were performed on an Intel® Core™ i7-6800K CPU @ 3.40GHz × 12, and the proposed method was implemented with Python 3. The hardware synthesis and implementation results are produced by the Xilinx Vivado HLS tool on a Xilinx Spartan-7 FPGA (xc7s100-fgga676-1) at 100 MHz operational frequency.

The CNN under study is trained on a dataset of 2000 images of animals (cats and dogs) and humans for binary classification. The accuracy of the network over the test set (including 450 images of animals and humans) is 93.34%. Bit truncation quantization is applied in network parameters during training and data precision is reduced to 8-bit.

C. Evaluation Results

The similarity of the fault resiliency analysis results by fault injection emulation and using the APPRAISER method is analyzed using the metrics identified in Section III.

Fig. 5 illustrates *normalized error* distribution in the output of the second convolutional layer (Conv2), in the presence of random double faults in the first convolution layer (dashed grey) vs errors induced by approximate multipliers (Mult1 solid orange, Mult2 solid blue) enabled in the first convolution layer respectively. Fig. 6 reports the result of applying FI and APPRAISER on the same convolutional layer and its impact on the second pooling layer of the network. These results

Fig. 5: Normalized output error of Conv2: Applying AxC and fault injection on the Conv1

Fig. 6: Normalized output error of Pool2: Applying AxC and fault injection on the Conv1

TABLE I: Bitflips and Accuracy/Recall drop induced by APPRAISER vs the reference fault injection method

| Affected/Measured Layers | Bitflips in subsequent layers | | | | | |
| | Injection of a single fault | | | Injection of a double fault | | |
	Fault Injection (reference) [%]	Approximation with MULT1 [%]	Approximation with MULT2 [%]	Fault Injection (reference)[%]	Approximation with MULT1 [%]	Approximation with MULT2 [%]
Conv1/Conv1	10.00	9.97	9.98	9.99	10.00	9.99
Conv1/Pool1	9.03	9.03	9.03	9.06	9.06	9.05
Conv1/Conv2	16.73	16.72	16.74	16.74	16.74	16.74
Conv1/Pool2	16.40	16.45	06.50	16.55	16.50	16.45
Conv1/FC	9.25	9.25	8.50	9.30	9.30	9.30
Conv2/Conv2	16.71	16.72	16.71	16.76	16.74	16.74
Conv2/Pool2	16.40	16.45	16.41	16.50	16.50	16.50
Conv2/FC	10.10	8.50	7.80	10.10	9.30	8.30
Affected Layer	DNN Accuracy/Recall drop					
Conv1	2.3/4.7	2.7/8.0	2.2/6.7	4.7/14.0	5.8/17.4	4.2/12.7
Conv2	1.8/6.0	1.6/5.0	2.7/8.0	9.1/26.4	9.1/26.4	8.9/26.7

TABLE II: Overheads of APPRAISER vs the reference fault injection method (Conv1 layer)

Network	Area LUT utilization	Analysis Control Circuitry	Interconnects	DNN execution time in FPGA
Base CNN	12%	N/A	Data Exchange Interconnect	131ms
Fault Resilience Assessment				
CNN instrumented with FI	**23%**	Complex FI Controller	(Data Exchange + FI) Interconnect	632,000ms
CNN instrumented with APPRAISER	~29%	**Simple AxC Activator**	**Data Exchange Interconnect**	**131ms**

demonstrate the similarity of the trends in error propagation by the proposed and the reference methods.

Table I reports fault resiliency assessment by the proposed and the reference methods using the *bitflips in subsequent layers* and the *DNN accuracy and recall drop* metrics. These results also demonstrate the strong similarity of the trends in error propagation by the proposed and the reference methods.

Table II demonstrates that although APPRAISER is more resource hungry, it is vastly faster than the reference fault injection by emulation method. It should be noted that the extra resources required by APPRAISER or FI are used only for the fault resiliency analysis phase and cleaned out from the final inference accelerator. In this example, the original CNN occupies 12% of the FPGA resources (LUTs). The CNN instrumented with APPRAISER occupies 29% of the FPGA resources and provides the accuracy/recall drop measurement for fault resiliency assessment in 131 ms, i.e. the same time as the original network execution time. On the other hand,

the CNN instrumented with FI utilizes 23% of the FPGA resources and performs the measurement in 632,000 ms, i.e. thousands of times (specifically, 4,824 times in this example) slower than the proposed method. This gain is composed of three components: a) processing of a single image in the CNN instrumented with APPRAISER is 0.29 ms vs 1.40 ms in the CNN instrumented with FI; b) APPRAISER pipelines the processing through the layers while FI has to break the pipeline; c) FI needs numerous iterations for each image to inject the faults (single, double or multiple) at different locations, one combination at a time, while APPRAISER uses only one iteration for each image.

Therefore, the time difference becomes even more drastic when comparing these methods for deeper networks (determining the number of layers in the inference execution pipeline) or DNNs with a larger memory for storing weights (determining the number of potential fault locations).

V. CONCLUSION

The state-of-the-art methods for fault injection by emulation incur a spectrum of time-, design- and control-complexity problems. To overcome these issues, a novel resiliency assessment method called APPRAISER is proposed that applies functional approximation for a non-conventional purpose and employs approximate computing errors for its interest. By adopting this concept in the resiliency assessment domain, APPRAISER provides thousands of times speed-up in the assessment process, while keeping high accuracy of the analysis.

In this paper, APPRAISER is validated by comparing it with state-of-the-art approaches for fault injection by emulation in FPGA. By this, the feasibility of the idea is demonstrated, and a new perspective in resiliency evaluation for DNNs is opened.

VI. Acknowledgement

References

[1] D. Silver, J. Schrittwieser, K. Simonyan, I. Antonoglou, A. Huang, A. Guez, T. Hubert, L. Baker, M. Lai, A. Bolton *et al.*, "Mastering the game of go without human knowledge," *nature*, vol. 550, no. 7676, pp. 354–359, 2017.

[2] M. Al-Qizwini, I. Barjasteh, H. Al-Qassab, and H. Radha, "Deep learning algorithm for autonomous driving using googlenet," in *2017 IEEE Intelligent Vehicles Symposium (IV)*. IEEE, 2017, pp. 89–96.

[3] M. Taheri, "Dnn hardware reliability assessment and enhancement," *27th IEEE European Test Symposium (ETS).*, May 2022. [Online]. Available: https://upcommons.upc.edu/handle/2117/369987

[4] Y. Ibrahim, H. Wang, J. Liu, J. Wei, L. Chen, P. Rech, K. Adam, and G. Guo, "Soft errors in dnn accelerators: A comprehensive review," *Microelectronics Reliability*, vol. 115, p. 113969, 2020.

[5] M.-C. Hsueh, T. K. Tsai, and R. K. Iyer, "Fault injection techniques and tools," *Computer*, vol. 30, no. 4, pp. 75–82, 1997.

[6] N. Khoshavi, C. Broyles, and Y. Bi, "Compression or corruption? a study on the effects of transient faults on bnn inference accelerators," in *2020 21st International Symposium on Quality Electronic Design (ISQED)*. IEEE, 2020, pp. 99–104.

[7] N. Khoshavi, A. Roohi, C. Broyles, S. Sargolzaei, Y. Bi, and D. Z. Pan, "Shieldenn: Online accelerated framework for fault-tolerant deep neural network architectures," in *2020 57th ACM/IEEE Design Automation Conference (DAC)*. IEEE, 2020, pp. 1–6.

[8] N. Khoshavi, C. Broyles, Y. Bi, and A. Roohi, "Fiji-fin: A fault injection framework on quantized neural network inference accelerator," in *2020 19th IEEE International Conference on Machine Learning and Applications (ICMLA)*. IEEE, 2020, pp. 1139–1144.

[9] M. Taheri, M. Riazati, M. H. Ahmadilivani, M. Jenihhin, M. Daneshtalab, J. Raik, M. Sjõdin, and B. Lisper, "Deepaxe: A framework for exploration of approximation and reliability trade-offs in dnn accelerators," in *24th International Symposium on Quality Electronic Design*. In press, 2023.

[10] V. Mrazek, R. Hrbacek, Z. Vasicek, and L. Sekanina, "Evoapprox8b: Library of approximate adders and multipliers for circuit design and benchmarking of approximation methods," in *Design, Automation Test in Europe Conference Exhibition (DATE), 2017*, March 2017, pp. 258–261.

[11] M. S. Ansari, V. Mrazek, B. F. Cockburn, L. Sekanina, Z. Vasicek, and J. Han, "Improving the accuracy and hardware efficiency of neural networks using approximate multipliers," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 28, no. 2, pp. 317–328, 2019.

[12] R. Leveugle, A. Calvez, P. Maistri, and P. Vanhauwaert, "Statistical fault injection: Quantified error and confidence," in *2009 Design, Automation & Test in Europe Conference & Exhibition*. IEEE, 2009, pp. 502–506.