Reduce: A Framework for Reducing the Overheads of Fault-Aware Retraining

Muhammad Abdullah Hanif, Muhammad Shafique
Division of Engineering, New York University Abu Dhabi (NYUAD), Abu Dhabi, United Arab Emirates
mh6117@nyu.edu, muhammad.shafique@nyu.edu

***Abstract*—Fault-aware retraining has emerged as a prominent technique for mitigating permanent faults in Deep Neural Network (DNN) hardware accelerators. However, retraining leads to huge overheads, specifically when used for fine-tuning large DNNs designed for solving complex problems. Moreover, as each fabricated chip can have a distinct fault pattern, fault-aware retraining is required to be performed for each chip individually considering its unique fault map, which further aggravates the problem. To reduce the overall retraining cost, in this work, we introduce the concept of resilience-driven retraining amount selection. To realize this concept, we propose a novel framework, *Reduce*, that, at first, computes the resilience of the given DNN to faults at different fault rates and with different amounts of retraining. Then, based on the resilience, it computes the amount of retraining required for each chip considering its unique fault map. We demonstrate the effectiveness of our methodology for a systolic array-based DNN accelerator experiencing permanent faults in the computational array.**

I. Introduction

Deep Neural Networks (DNNs) have emerged as a promising set of models for solving complex problems [1]. They are now state of the art for many AI applications, e.g., image classification, object detection and language translation [1] [2]. However, these DNNs have high computational complexity [2]. To meet stringent performance and efficiency constraints of real-world applications, specialized DNN hardware accelerators, such as Eyeriss [3] and TPU [4], are used. These accelerators are usually built using nano-scale CMOS technology and face various reliability issues.

One of the foremost reliability concerns with nano-scale CMOS devices is permanent faults induced due to imperfections in the manufacturing process. Prior works, such as [5], have shown that even a small fraction of these faults can drastically reduce the accuracy of DNNs. Hence, these faults render some of the fabricated chips useless, which negatively impacts the manufacturing yield. To address permanent faults in DNN hardware accelerators, various fault-mitigation techniques have been proposed. For example, Kim et al. [6] propose to bypass faulty Processing Elements (PEs) and view a faulty array as a smaller fault-free array. Such techniques improves the yield, but at the cost of performance loss. Techniques like [7] propose to add redundancy such that each redundant PE in the architecture is dedicated for a limited region of the computing array. These techniques also significantly impact the performance of the system, as they employ redundancy for fault mitigation. Apart from redundancy-based techniques, *Fault-Aware Pruning (FAP)* [5] is proposed which exploits intrinsic resilience of DNNs to pruning (zeroed weights/computations) to mitigate the effects of permanent faults in the computational array of a systolic array-based DNN accelerator. *Fault-Aware Mapping (FAM)* [8] further improves the effectiveness of *FAP* by permuting the DNN weights such that less significant weights are mapped to the bypassed (faulty/zeroed) PEs. The main shortcoming of these methods is that they offer fault-mitigation at the cost of accuracy loss. To offer fault mitigation without significant accuracy loss, *Fault-Aware Pruning + Training (FAP+T)* is proposed in [5]. *Fault-Aware Training (FAT)* is also exploited in [9] to mitigate permanent faults in DNN accelerators. The above works clearly show that *FAT* leads to the best accuracy results under hardware faults, and it achieves this with minimal impact on system's performance. Even though *FAT* offers the best accuracy, it has serious limitations. *Its core drawback is that it incurs huge (re)training overheads, specifically in cases where a DNN has to be tuned for numerous faulty chips having distinct fault patterns. Towards this, we aim at addressing the following challenging question: how to reduce the (re)training overheads of FAT when a given DNN has to be tuned for numerous chips having different fault maps.*

A. Our Novel Contributions

To address the above-mentioned research question, in this work, we present a novel framework, *Reduce*. The framework mainly estimates the resilience of the given DNN to faults and defines the amount of retraining required for each individual faulty chip based on its fault characteristics and the resilience characteristics of the given DNN.

II. *REDUCE*: Proposed Framework for Reducing the Overheads of Fault-Aware Retraining

Fig. 1 shows our proposed *Reduce* framework, which receives a pre-trained DNN, a dataset, a user-defined accuracy constraint, and fault maps of the faulty chips as input and defines a retraining policy to efficiently generate fault-aware DNNs for the given faulty chips. The framework first computes the resilience of the given DNN to faults using fault-injection experiments at different fault rates and with different levels of retraining (Step ❶). This resilience is then used in Step ❷ to select the amount of fault-aware retraining for each individual faulty chip based on its unique fault characteristics. The selection is performed in such a way that each output DNN offers accuracy close to the user-defined accuracy constraint without incurring unnecessary overheads. In the final step, Step ❸, *FAT* is performed and the generated DNNs are then distributed to their corresponding faulty chips.

978-3-9819263-7-8/DATE23/© 2023 EDAA

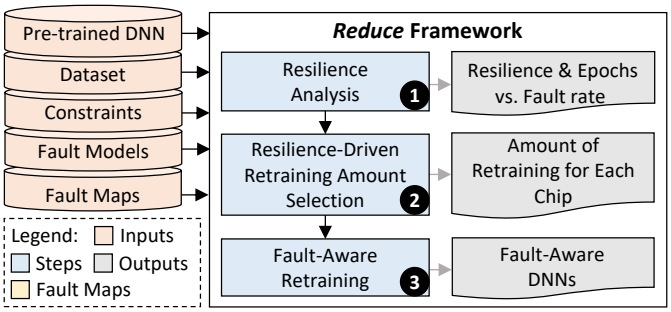

Fig. 1: Overview of the proposed *Reduce* framework

III. Results and Discussion

A. Experimental Setup

To evaluate the effectiveness of the proposed technique, we consider the case of mitigating permanent faults in the computational array of a DNN accelerator. We consider the modified DNN accelerator design presented in [5] with FAP support. We assume the size of the systolic array to be 256×256. For evaluation, we built our entire framework using PyTorch. Similar to [5], we consider a random fault injection model for generating fault maps.

B. Resilience Trends for VGG11 trained on Cifar10 Dataset

Fig. 2a shows the impact of different levels of FAT on the accuracy of the VGG11 at different fault rates while Fig. 2b shows the amount of FAT required at each fault rate to achieve a particular accuracy level. For each data point in Fig. 2b, we repeated the experiment five times and reported the minimum and maximum number of epochs along with the mean. The error bars in the figure show that the use of mean values can lead to undertraining. Therefore, we propose to use the maximum reported values for estimating the amount of retraining for each faulty chip, as it leads to higher confidence that the generated model meets the accuracy constraint.

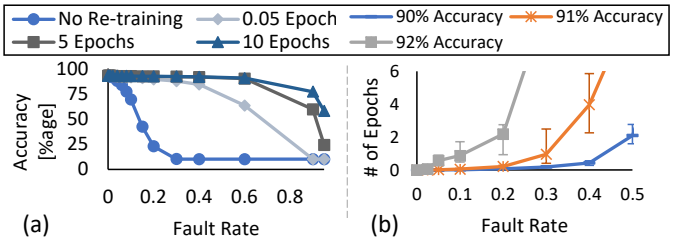

Fig. 2: Resilience trend of VGG11 trained on Cifar10 dataset.

C. Comparison with State of the Art

To highlight the effectiveness of the proposed technique, we compared our *Reduce* framework with fixed-policy retraining method proposed in [5]. Fig. 3a and b shows the results of the proposed methodology when employed for retraining VGG11 (trained on Cifar10) for 100 faulty chips. Fig. 3c, 3d and 3e correspond to the cases where the DNN is trained for each faulty chip individually for a pre-specified number of epochs. The figures show that as the amount of retraining is increased the number of samples that meet the accuracy constraint increases. The results of Fig. 3a - 3e are summarized in Fig. 3f. The figures clearly show that the proposed *Reduce* framework produces better (more robust) models with lesser training compared to the fixed-policy techniques.

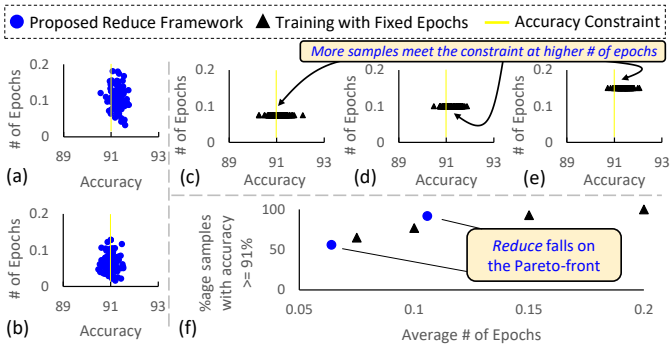

Fig. 3: Comparison with state of the art. (a) Results of the *Reduce* framework using the maximum values from the resilience analysis for training amount estimation. (b) Results of *Reduce* using mean values from the resilience analysis. (c), (d) and (e) correspond to cases where VGG11 is trained for each chip using fixed number of epochs. (f) A summary of the results in (a)-(e). All the results are generated assuming 91% as the accuracy constraint.

IV. Conclusion

In this paper, we proposed *Reduce*, a methodology for reducing the overheads of fault-aware retraining when used for tuning a given DNN for multiple faulty chips. We mainly addressed how to compute the amount of retraining required for tuning the given DNN for a specific faulty chip such that the given DNN meets the user-defined accuracy constraint without incurring high retraining overheads. The results showed that the proposed technique could significantly reduce the retraining cost compared to state-of-the-art methods.

Acknowledgement

This work has been supported in part by the Center for Artificial Intelligence and Robotics (CAIR), funded by Tamkeen under the NYUAD Research Institute Award CG010, and the Center for Cyber Security (CCS), funded by Tamkeen under the NYUAD Research Institute Award G1104.

References

[1] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *nature*, vol. 521, no. 7553, p. 436, 2015.

[2] V. Sze, Y.-H. Chen, T.-J. Yang, and J. S. Emer, "Efficient processing of deep neural networks: A tutorial and survey," *Proceedings of the IEEE*, vol. 105, no. 12, pp. 2295–2329, 2017.

[3] Y.-H. Chen, T.-J. Yang, J. Emer, and V. Sze, "Eyeriss v2: A flexible accelerator for emerging deep neural networks on mobile devices," *IEEE Journal on Emerging and Selected Topics in Circuits and Systems*, vol. 9, no. 2, pp. 292–308, 2019.

[4] N. P. Jouppi, C. Young, N. Patil, D. Patterson, G. Agrawal, R. Bajwa, S. Bates, S. Bhatia, N. Boden, A. Borchers *et al.*, "In-datacenter performance analysis of a tensor processing unit," in *2017 ACM/IEEE 44th Annual International Symposium on Computer Architecture (ISCA)*. IEEE, 2017, pp. 1–12.

[5] J. J. Zhang, T. Gu, K. Basu, and S. Garg, "Analyzing and mitigating the impact of permanent faults on a systolic array based neural network accelerator," in *2018 IEEE 36th VLSI Test Symposium (VTS)*. IEEE, 2018, pp. 1–6.

[6] J. H. Kim and S. M. Reddy, "On the design of fault-tolerant two-dimensional systolic arrays for yield enhancement," *IEEE Transactions on Computers*, vol. 38, no. 4, pp. 515–525, 1989.

[7] I. Takanami and M. Fukushi, "A built-in circuit for self-repairing mesh-connected processor arrays with spares on diagonal," in *2017 IEEE 22nd Pacific Rim International Symposium on Dependable Computing (PRDC)*, 2017, pp. 110–117.

[8] M. A. Hanif and M. Shafique, "SalvageDNN: salvaging deep neural network accelerators with permanent faults through saliency-driven fault-aware mapping," *Philosophical Transactions of the Royal Society A*, vol. 378, no. 2164, 2020.

[9] J. J. Zhang, K. Basu, and S. Garg, "Fault-tolerant systolic array based accelerators for deep neural network execution," *IEEE D&T*, vol. 36, no. 5, pp. 44–53, 2019.