Microprocessors and Microsystems 106 (2024) 105023

Contents lists available at ScienceDirect

Microprocessors and Microsystems

journal homepage: www.elsevier.com/locate/micpro

Flip-and-Patch: A fault-tolerant technique for on-chip memories of CNN accelerators at low supply voltage

Yamilka Toca-Díaz [a], Reynier Hernández Palacios [b], Rubén Gran Tejero [a], Alejandro Valero [a,*]

[a] Department of Computer Science and Systems Engineering, Universidad de Zaragoza, Spain
[b] Vicerrectoría de Investigaciones, Universidad de Camagüey, Cuba

ARTICLE INFO

Keywords:
Deep learning
Energy efficiency
Network accuracy
Permanent faults

ABSTRACT

Aggressively reducing the supply voltage (V_{dd}) below the safe threshold voltage (V_{min}) can effectively lead to significant energy savings in digital circuits. However, operating at such low supply voltages poses challenges due to a high occurrence of permanent faults resulting from manufacturing process variations in current technology nodes.

This work addresses the impact of permanent faults on the accuracy of a Convolutional Neural Network (CNN) inference accelerator using on-chip activation memories supplied at low V_{dd} below V_{min}. Based on a characterization study of fault patterns, this paper proposes two low-cost microarchitectural techniques, namely Flip-and-Patch, which maintain the original accuracy of CNN applications even in the presence of a high number of faults caused by operating at $V_{dd} < V_{min}$. Unlike existing techniques, Flip-and-Patch remains transparent to the programmer and does not rely on application characteristics, making it easily applicable to real CNN accelerators.

Experimental results show that Flip-and-Patch ensures the original CNN accuracy with a minimal impact on system performance (less than 0.05% for every application), while achieving average energy savings of 10.5% and 46.6% in activation memories compared to a conventional accelerator operating at safe and nominal supply voltages, respectively. Compared to the state-of-the-art ThUnderVolt technique, which dynamically adjusts the supply voltage at run time and discarding any energy overhead for such an approach, the average energy savings are by 3.2%.

1. Introduction

Artificial Intelligence (AI) has emerged as a groundbreaking technology capable of analyzing vasts amounts of data, learn patterns, and make accurate predictions across numerous industrial sectors. To unlock the full potential of AI, specialized hardware accelerators play a crucial role. These accelerators, like GPUs or TPUs, speed up the execution of AI workloads, enabling faster processing and more efficient resource utilization than traditional CPU systems. However, as AI workloads grow in complexity and size, their computational and memory demands increase, resulting in a higher power consumption of AI devices. Addressing power consumption in AI accelerators is essential for energy efficiency, environmental sustainability, and responsible AI deployment.

The energy efficiency of current computing systems is compromised by conservative operation guardbands due to variations in the manufacturing process of current CMOS technology nodes. An example is the supply voltage (V_{dd}) of the transistor. To ensure a safer system operation against sudden V_{dd} droops, the supply voltage is conservatively set above the limit of the safe voltage (V_{min}) imposed by the worst-case transistor. However, significant V_{dd} droops are infrequent events [1]. Moreover, overscaling V_{dd} results in energy wasting, since static and dynamic energy scale linearly and quadratically with V_{dd}, respectively.

Many AI accelerators, such as those employed in the inference process of deep Convolutional Neural Networks (CNNs), integrate large and energy-hungry on-chip memories to store application parameters. Such memory structures commonly employ 6-transistor SRAM bitcells susceptible to the aforementioned process variations.

To reduce energy consumption, a viable solution adopted in various microprocessor components, including on-chip memories, involves relaxing the voltage guardband by lowering V_{dd} toward V_{min} while maintaining a fixed frequency, a technique commonly known as Dynamic Voltage Scaling (DVS) [1,2]. However, aggressively underscaling V_{dd} beyond V_{min} is challenging due to the high number of permanent faults appearing in vulnerable bitcells, causing them to remain

[*] Corresponding author.
E-mail address: alvabre@unizar.es (A. Valero).

https://doi.org/10.1016/j.micpro.2024.105023
Received 4 September 2023; Received in revised form 31 January 2024; Accepted 2 February 2024
Available online 10 February 2024
0141-9331/© 2024 The Author(s). Published by Elsevier B.V. This is an open access article under the CC BY license (http://creativecommons.org/licenses/by/4.0/).

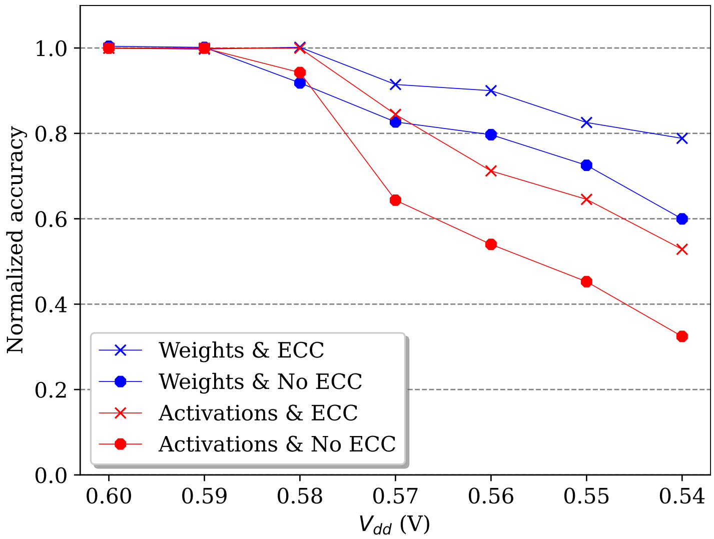


Fig. 1. Normalized accuracy for different supply voltages for weight and activation memories protected or unprotected with ECC codes with respect to the original accuracy (fault-free operation mode at 0.6 V). For a given type of memory, the other memory is free of faults to isolate the respective effects.

stuck at a specific logic value [3]. Unfortunately, conventional Error-Correcting Codes (ECC) offer limited coverage to permanent faults, requiring larger storage capacities, complex encoders/decoders, and higher energy consumption to ensure a reliable operation [4–6].

To illustrate this phenomenon, Fig. 1 depicts the averaged top-1 accuracy for a number of widely used CNN benchmarks as the V_{dd} of on-chip memories of an accelerator reduces below V_{min} (0.6 V).[1] Results are normalized to the original accuracy achieved at 0.6 V without faults and distinguish between weight and activation memories. In turn, memories are protected or unprotected with Single-Error Correction Double-Error Detection (SECDED) ECC at a granularity of 16-bit words. Notably, the inclusion of ECC yields substantial accuracy improvements for both types of memories. However, even with ECC protection, weight and activation memories impose an accuracy degradation exceeding 20% and 40%, respectively, when V_{dd} scales down to 0.54 V.

Notice too that activations are more vulnerable to faults than weights. Unlike weights, which are fixed parameters, activations are dynamically updated through repeated accumulation of partial sums, usually requiring a larger value range than weights [7]. Assuming separate application-specific fixed-point quantizations for weights and activations, the latter usually require more integer bits [8,9], potentially resulting in greater magnitude deviations under faults. This paper focuses on activation memories and leaves weight memories for future work.

In the context of CNN accelerators, researchers have proposed diverse mechanisms to address the impact on accuracy of a large number of permanent faults as a result of $V_{dd} < V_{min}$. Some of these approaches include custom retraining of neural networks under faults [10], enhancing the placement algorithm during the FPGA compilation process to bypass faulty cells [3], or dynamically adjusting V_{dd} for individual network layers at run time [11].

The above works either rely on the programmer or necessitate offline profiling of the target CNN application to adapt the mechanism. Unlike those approaches, in a previous work [12], we proposed a couple of microarchitectural mechanisms that do not impose any burden to the programmer nor depend on application characteristics, making them highly appealing to real CNN accelerators. By considering the impact of faulty activation bitcells on CNN application accuracy, our approach restores the original accuracy under V_{dd} as low as 0.54 V. This is achieved thanks to modifying the representation of activations with a few number of faults, and ensuring a fault-free backup storage for activations with a high number of faults.

This paper extends our previous work [12] according to the following four main contributions: (i) the management of the proposed backup storage is enhanced to support new CNN workloads under evaluation, (ii) the fault characterization study is extended to different supply voltage levels, (iii) a new performance metric based on the softmax probability array is defined and measured for a more comprehensive evaluation of our approach, and (iv) the proposed technique is quantitatively compared against the state-of-the-art ThUnderVolt technique. We refer to our approach as Flip-and-Patch.

Experimental results show that Flip-and-Patch effectively reduces the average energy consumption of activation memories by 10.5% when compared to a conventional CNN accelerator operating at safe V_{min} supply voltage. In comparison to the state-of-the-art ThUnderVolt technique [11], and excluding the non-trivial energy overhead of this approach, the average energy savings are by 3.2%. These numbers are obtained with a minimal impact on system performance (less than 0.05% for every application) and preserving the original CNN accuracy.

The remainder of this paper is organized as follows. Section 2 provides a background for this work. Section 3 introduces a fault characterization study that lays the foundations of our proposal. Section 4 presents the proposed microarchitectural techniques to counteract permanent faults. Section 5 refers to the experimental evaluation. Section 6 discusses related work, and finally, Section 7 concludes this paper.

2. Background

This section summarizes the CNN accelerator architecture and the reliability model used in this work to evaluate the proposed approach.

2.1. Baseline CNN accelerator architecture

Our modeled baseline CNN architecture is based on state-of-the-art accelerator models from both the academia and the industry, like DaDianNao [21] and Google's TPU [22], to speed up the inference process in CNNs. Fig. 2 plots the hardware organization consisting of a 16 × 16 Processing Element (PE) array, on-chip memory storage to reduce costly off-chip memory accesses, dispatchers for every memory, and a control unit. On-chip intermediate storage includes a pair of 2 MiB activation memories and a 2 MiB weight memory. The computational and storage resources are sized according to the domain of

[1] See Section 3 for further details about the experimental environment.

Table 1
Main characteristics of the studied CNN benchmarks.

Benchmark	Depth	Average layer size	Activation representation (integer, fraction)	Accuracy
AlexNet [13]	5 × Conv, 3 × FC, 3 × MaxPooling	153 KiB	4 bits, 4 bits	0.89
DenseNet [14]	120 × Conv, 1 × FC, 5 × Pooling	254 KiB	3 bits, 5 bits	0.92
Inception [15]	149 × Conv, 2 × FC, 7 × MaxPooling, 11 × Pooling	44 KiB	8 bits, 6 bits	0.79
MobileNet [16]	28 × Conv, 1 × Pooling	340 KiB	4 bits, 9 bits	0.88
ResNet [15]	53 × Conv, 1 × FC, 1 × MaxPooling, 1 × Pooling	178 KiB	4 bits, 4 bits	0.81
SqueezeNet [17]	26 × Conv, 4 × Pooling	431 KiB	6 bits, 4 bits	0.93
VGG16 [18]	13 × Conv, 3 × FC, 5 × MaxPooling	1.32 MiB	3 bits, 8 bits	0.81
VGG19 [18]	16 × Conv, 3 × FC, 5 × MaxPooling	606 KiB	8 bits, 2 bits	0.94
Xception [19]	6 × Conv, 1 × FC, 4 × MaxPooling, 1 × Pooling	289 KiB	7 bits, 6 bits	0.90
ZFNet [20]	5 × Conv, 3 × FC, 3 × MaxPooling	324 KiB	4 bits, 6 bits	0.83

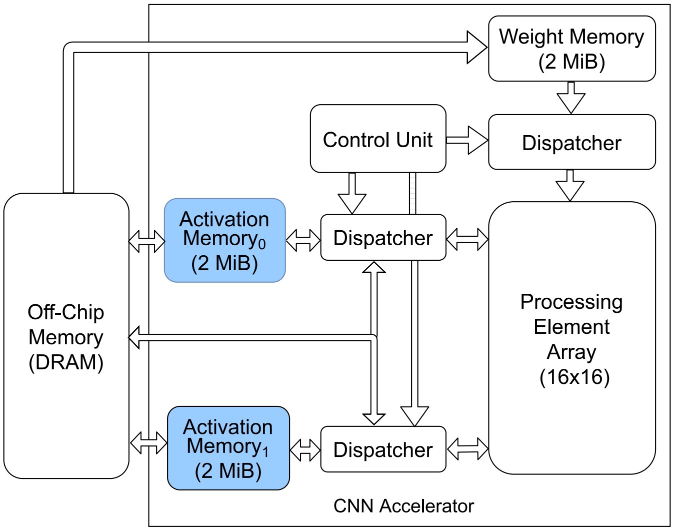


Fig. 2. Overview of the baseline CNN accelerator.

embedded systems [23], although our proposal could be easily adapted for larger accelerators.

The PE array is a systolic array processor with PEs interconnected through a 2D mesh. Each PE independently computes 16-bit fixed-point dot-products through partial sums with an input from one activation memory, acting as input memory, and a weight from the weight memory. The dataflow in the PE array corresponds to the output stationary approach described in SCALE-Sim [24].

Like EIE accelerator [25], activation memories swap their roles after the computation of every network layer. In this way, a given activation memory stores even layers and the counterpart memory stores odd layers. On the other hand, the weight memory caches weights to be issued in the proper order by the dispatcher to the PE array.

Similarly to previous CNN accelerator models [7,26,27], network parameters occupy 16 bits and are represented in fixed-point arithmetic, adjusting the number of integer and fraction bits to the requirements of each CNN application at run time (see Section 3).

The relatively small size of activation memories implies to spill to off-chip memory those activations of layers exceeding 2 MiB. These memories are arranged as scratchpad memories and designed to provide 16 activations (32 bytes per cycle) to the parallel processing in the PE array. Finally, dispatchers are driven by the control unit, which exploits control information of the currently computed layer.

2.2. Reliability model

Our reliability model for permanent faults is based on the publicly available bit-level model of a real hardware platform, where Salami et al. test the on-chip memory reliability of a VC707 Xilinx FPGA working under different low-power operation modes, underscaling V_{dd} from 0.6 V (V_{min}) to 0.54 V [3]. Setting V_{dd} below 0.54 V is not possible since the tested FPGA stops operating. The FPGA includes 4 MiB on-chip memory storage, which corresponds to the activation storage of our CNN accelerator. Refer to the next section for a detailed characterization of activation fault patterns.

Remark that this work focuses on permanent faults as a consequence of underscaling V_{dd} below V_{min}. These faults manifest during the entire period of time in which $V_{dd} < V_{min}$ and are detected during post-fabrication testing before deploying the device in the field [28,29]. The memory test does not depend on the applications to be run in the field. In particular, all the memory bits are tested to check if they correctly store '0' and '1' logic values at specific V_{dd} levels. A bitcell is considered faulty if the read value does not match the last written value.

Dealing with other types of faults as a consequence of voltage noise, aging effects, or particle strikes, which are unpredictable and appear at specific execution cycles are out of the scope of this paper. A significant advantage of our proposal over prior work is that, once the faulty bitcells are established at a specific V_{dd} level, the proposed approach operates exclusively at the microarchitecture level without any impact on the regular operation of a CNN application.

Finally, similar to previous academic work [30] and commercial devices [31], our baseline CNN accelerator has dedicated voltage domains for logic and arrays, which allows reducing V_{dd} below V_{min} in activation memories while keeping the rest of the hardware components at V_{dd} above V_{min} to avoid faults.

3. Characterization study

We have chosen a number of widely used CNN benchmarks with different data representation, computational, and memory storage requirements. Table 1 summarizes the main characteristics of these CNNs. The depth in number of layers largely varies between the smallest (AlexNet and ZFNet) and largest (Inception) neural network, as well as the average layer size from tens to thousands of KiB. The next column refers to the required number of bits for fixed-point representation, distinguishing between integer and fraction parts, to avoid accuracy losses with respect to the top-1 accuracy with 32-bit floating-point (IEEE-754) representation. As observed, benchmarks require between 8 and 14 bits to represent activations, plus an additional bit for the sign.[2] For simplicity, we assume that fraction bits are extended to represent up to 16-bit words.[3] All the CNNs run a colorectal cancer histology dataset for image classification purposes [32]. All the presented results in this work are averaged for the inference of the entire dataset consisting of 750 different test images. The rightmost column of the table shows the accuracy of each CNN, ranging from 0.79 (Inception) to 0.94 (VGG19).

[2] Unlike activations, whose integer bits range from 3 to 8, weights only require from 0 to 4 integer bits, making them inherently more resilient to faults (see Fig. 1).

[3] Exploiting bit over-provisioning as a backup for faulty bits deserves further exploration and is out of the scope of this work.

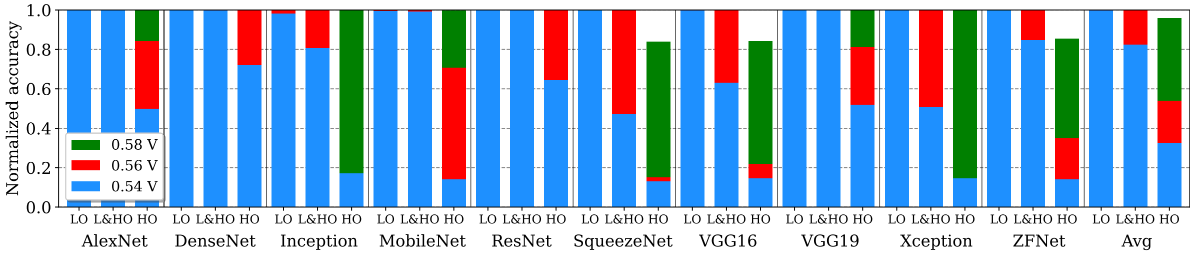


Fig. 3. Normalized accuracy for different types of faulty activations: Low-Order (LO), Low- & High-Order (L&HO), and High-Order (HO), varying the supply voltage with respect to the original (fault-free) accuracy.

More details about the experimental environment can be found in Section 5.1.

The impact of faults on the accuracy of a CNN application mainly depends on the position of the affected activation bits. More precisely, for a fixed-point data type, the numerical deviation of a faulty activation scales exponentially with the significance of the affected bit. We have defined three types of faulty activations according to the faulty bit locations in 16-bit activation words:

- Low-Order (LO). These activations only contain faults in the least significant byte.
- High-Order (HO). These activations only exhibit faults in the most significant byte.
- Low- & High-Order (L&HO). Activations with this pattern show faults in both least and most significant bytes.

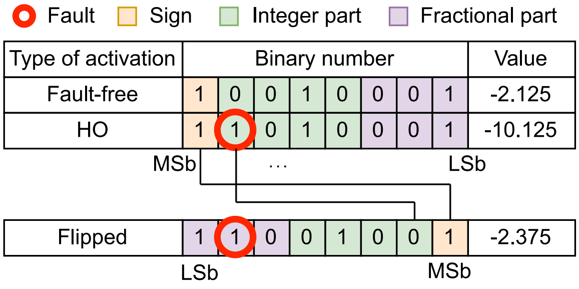
The number of faulty activations grows with the V_{dd} reduction. In particular, according to the reliability model of a real hardware platform evaluated in [3], see Section 2.2, the percentage of faulty activations is 0.005%, 0.107%, and 0.904%, for 0.58 V, 0.56 V, and 0.54 V V_{dd} values, respectively. In turn, for the faultiest operation mode with V_{dd} = 0.54 V, LO, HO, and L&HO faulty activations represent 0.45%, 0.45%, and 0.004% of all the activations, respectively.

Fig. 3 shows the impact of the different types of faulty activations on the accuracy of the studied CNN applications. Note that, for a given CNN and type of faulty activation, the normalized accuracy with respect to a reliable operation mode ($V_{dd} \geq V_{min}$ = 0.6 V) is shown as stacked bars for the chosen V_{dd} levels.

LO activations do not compromise the accuracy of any CNN regardless of the V_{dd} value. That is, the faultiest operation mode reaches the original accuracy obtained with a reliable operation mode. This is mainly due to the deviation of the resulting magnitude in LO activations is rather low. In fact, according to Table 1, the least significant byte only stores fraction bits for most of the studied benchmarks.

Inception is the only benchmark with a minor accuracy loss for LO activations at 0.54 V. This is due to the combination of three main effects. First, Inception requires up to 8 bits to represent the integer part of activations, leading to larger magnitude deviations in LO activations. Second, this benchmark has the lowest average layer size, in the sense that it does not include as many redundant parameters as other CNNs, which may mitigate the impact of faults. Third, Inception is the deepest CNN. Despite the presence of pooling layers and Rectified Linear Units (ReLU), which may reduce the impact of faults, deeper CNNs may magnify and broadcast faulty activation values.

On the other hand, HO activations largely affect the accuracy of all the neural networks. MobileNet, SqueezeNet, VGG16, Xception, and ZFNet perform a random guessing at V_{dd} = 0.54 V. That is, they suffer the greatest possible accuracy loss (accuracy scales down to one eighth according to the eight classes of the dataset). Scaling V_{dd} up to 0.56 V does not help much. In fact, it makes no difference for Xception. Moreover, in SqueezeNet, VGG16, and ZFNet, 0.58 V are still not enough to reach the original accuracy. For HO activations, the normalized accuracy ranges from 13.4% (SqueezeNet) to 72.2% (DenseNet), with an average of 32.7%, for V_{dd} = 0.54 V.



Type of activation	Binary number								Value
Fault-free	1	0	0	1	0	0	0	1	-2.125
HO	1	1	0	1	0	0	0	1	-10.125
Flipped	1	1	0	0	1	0	0	1	-2.375

Fig. 4. Example of applicability of the flipping technique to an 8-bit HO activation. Labels *MSb* and *LSb* stand for more and less significant bit, respectively. A stuck-at '1' fault is located in a high-order bit of the activation.

Interestingly, despite the percentage of L&HO activations being as low as 0.004% for V_{dd} = 0.54 V, they significantly hurt the accuracy of CNNs like SqueezeNet, VGG16, and Xception down to 47.1%, 63.2%, and 50.7%, respectively. This is mainly due to these benchmarks require a high memory storage demand (see Table 1), meaning that they are more exposed to this type of faults in the activation memory. In addition, these applications require a significant number of integer bits to represent activations, increasing the probability of large numerical deviations in L&HO activations.

Based on the above insights, we propose to turn HO activations into LO activations, and provide an alternative fault-free memory storage for L&HO activations. On the other hand, LO activations remain untouched since they do not affect the accuracy of any CNN. Unless otherwise stated, the remainder of this work assumes the faultiest operation mode with V_{dd} = 0.54 V during the entire inference process of CNN benchmarks.

4. Proposed approach: Flip-and-Patch

Flip-and-Patch is based on the observation that activations with faults in the most significant byte (HO activations) largely degrade the accuracy of CNN applications. In addition, a small number of activations with faults in both most and less significant bytes (L&HO activations) also compromise the accuracy of some CNNs. The proposed approach consists of a couple of techniques. First, we introduce a word flipping mechanism to deal with HO activations. Then, we propose a patching approach to deal with L&HO activations. Finally, the overhead of both techniques is quantified.

4.1. Flipping technique

The aim of this technique is to minimize the weight of faulty bits in HO activations. Assuming a little-endian data representation, N-bit

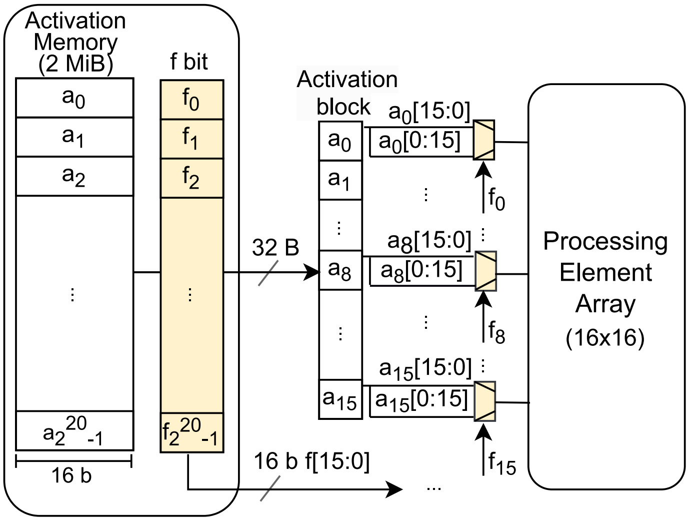


Fig. 5. Proposed flipping technique consisting of 16 2:1 multiplexers of 16-bits width in the read port of an activation memory. The read block comprises activations from a_0 to a_{15}. Added circuitry is highlighted in yellow. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

HO activations only contain faults in bit positions from $N/2$ to $N-1$. After flipping an activation, a bit occupying the ith position occupies the $(N-1-i)$-th position. For instance, for $N = 16$, bit 12 becomes bit 3 and vice versa. The flipping technique ensures that HO activations turn into LO activations, where bit faults are only located in low-order bits from 0 to $N/2-1$. The flipping action significantly reduces the impact of a fault on the magnitude of the activation.

Fig. 4 illustrates an example of applicability of the flipping technique to an 8-bit HO activation with a stuck-at '1' fault in the second bit from left to right. The fault transforms the original value −2.125 into −10.125. After flipping the activation, the value is −2.375, largely reducing the magnitude deviation.

To differentiate between HO (flipped) activations and the remaining (non-flipped) activations, the proposed design includes an f control bit per activation. This bit is set, for different V_{dd} levels, during post-fabrication testing before deploying the device in the field, similarly to as done in faulty bitmaps used by error detection/correction techniques to differentiate between faulty and reliable contents [28,29]. Remark that permanent faults appearing when the device is already in the field due to aging phenomena could be covered using a more complex self-testing mechanism and periodically writable control bits at run time without further changes in the proposed approach.[4]

Fig. 5 shows how the read port of an activation memory is enhanced to perform the flip operation in selected activations. The port includes 16 × 2:1 multiplexers of 16-bit width, according to the size of read blocks in number of activations. These multiplexers are driven by the f_i bit of activations, reversing back the bit order of a flipped activation (a_i[0:15]) when necessary ($f_i = 1$). Note that the write port (not shown in the figure) requires the same number of multiplexers to store activations in the proper bit order.

Fig. 6 confirms the effectiveness of the flipping technique, where the cumulative distribution of number of permanent faults for every bit position of activations is shown. In a conventional design, bit faults accumulate linearly along the 16-bit width. On the other hand, the proposed approach removes all the faults in high-order bits (apart from those of L&HO activations), in exchange of faults accumulating faster in low-order bits compared to the conventional design.

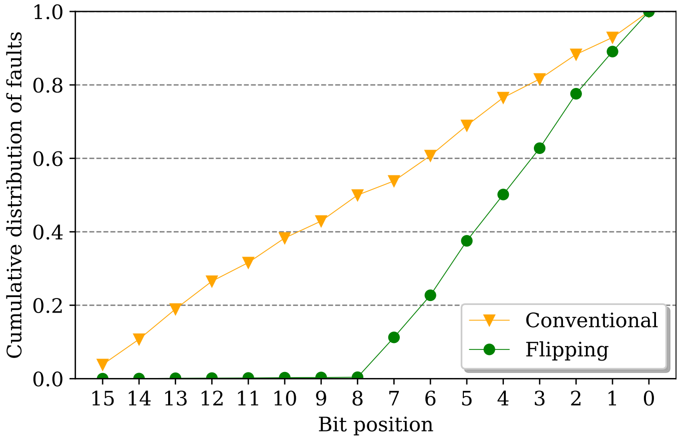
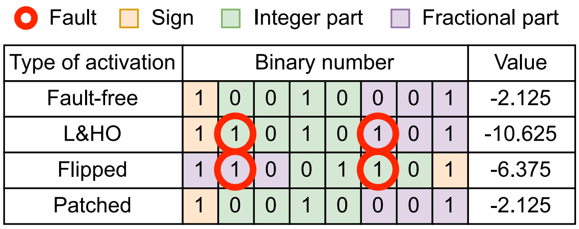


Fig. 6. Cumulative distribution of number of faults for every bit position.


Fault
Sign
Integer part
Fractional part

Type of activation	Binary number								Value
Fault-free	1	0	0	1	0	0	0	1	-2.125
L&HO	1	1	0	1	0	1	0	1	-10.625
Flipped	1	1	0	0	1	1	0	1	-6.375
Patched	1	0	0	1	0	0	0	1	-2.125

Fig. 7. Example of applicability of the patching technique to an 8-bit L&HO activation. Stuck-at '1' faults are located in both low-order and high-order bits of the activation.

4.2. Patching technique

The flipping technique does not remove the impact of L&HO activations on the CNN accuracy. The second technique consists of a tiny cache, referred to as *patching cache*, that stores the original (fault-free) value of such activations.

Fig. 7 plots an example where the effectiveness of the flipping technique is compromised. Stuck-at '1' faults in both low-order and high-order bits result in an L&HO activation where the original value −2.125 turns into −10.625. After flipping the L&HO activation, the

[4] This design enhancement is beyond the scope of this paper. The reader is referred to orthogonal work on specific aging-aware techniques for CNN accelerators [33–35].

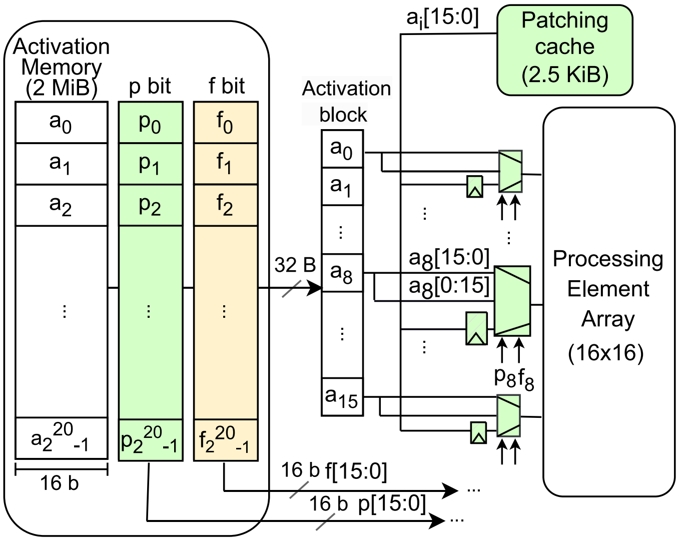


Fig. 8. Proposed Flip-and-Patch technique in the read port of an activation memory. The read block comprises activations from a_0 to a_{15}. Required components of the patching approach are highlighted in green. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

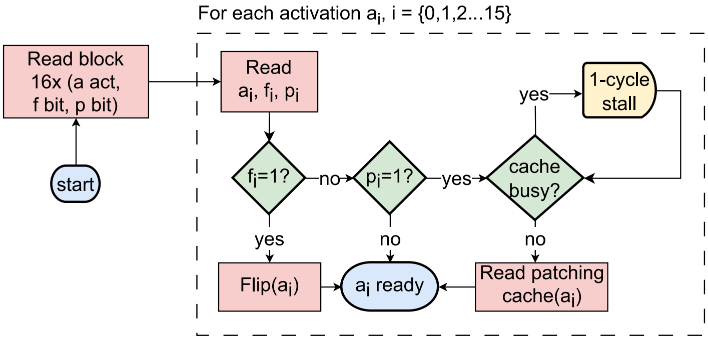


Fig. 9. Flowchart of a read operation using the Flip-and-Patch technique.

resulting magnitude deviation is reduced to −6.375, but it is much larger than the value −2.375 obtained in the previous example (see Fig. 4). In this case, it is imperative to retrieve the original value from the patching cache.

Since the patching cache stores a reliable copy of L&HO activations of the entire activation memory, alias addresses may conflict in the small cache. To avoid conflict misses, the patching memory is organized as a 2.5 KiB 5-way set-associative cache. The capacity is large enough to store all the L&HO activations, even for a reliability model with 4× more faults than the former one (see Section 5.5).

Like f bits for the flipping approach, the patch mechanism requires a p control bit per activation to determine whether a requested activation is to be found in the patching cache or not.

Fig. 8 shows the main components of the proposed approach, distinguishing between flipping and patching components in yellow and green, respectively. The combination of the two techniques requires to replace 2:1 with 4:1 multiplexers, which are driven by the p_i and f_i bits, to select among three possibilities: neither flipping nor patching (00), flipping (01), or patching (10).[5] Therefore, during the post-fabrication testing, both p and f control bits are set according to the type of activation entry: original or LO (00), HO (01), and L&HO (10).

Fig. 9 depicts a flowchart of a read operation in an activation memory enhanced with Flip-and-Patch. First, a 32-byte block consisting of 16 activations is read, plus the f and p control bits of each individual activation (32 bits). Then, every activation value is adjusted according to the control bits as follows. If $f_i = 1$, the corresponding multiplexer forwards to the PE array the entry with the flipped activation ($a_i[0{:}15]$, see Fig. 8). Otherwise, if $p_i = 0$, the multiplexer forwards to the PE array the original activation $a_i[15{:}0]$.

In case of $p_i = 1$, the multiplexer selects the latched path. These latches temporarily store reliable copies of L&HO activations from the patching cache. For design simplicity, the patching cache includes a single 16-bit read port, forcing to read reliable L&HO activations of a same block in consecutive cycles as shown in the flowchart. Nevertheless, the impact on system performance is small since the patching cache is rarely exercised (L&HO activations represent 0.004% of all the activations, see Section 3). Experimental results show that system performance degradation is less than 0.05% for every CNN application. Once the complete activation block is ready, it is forwarded to the PE array.

Finally, the write port maintains 2:1 multiplexers to perform either flipped or non-flipped writes to the activation memory. In addition, 16 latches are required to perform up to 16 sequential stores to the patching cache if necessary.

4.3. Power, energy, area, and timing overhead

The proposed approach requires two control bits (p and f bits) per 16-bit activation word. That is, the storage overhead grows linearly with the size of the activation memory as one eighth of the total effective capacity of the memory. In particular, for a 2 MiB activation

[5] A simultaneous activation of both control bits is not allowed.

Table 2
Leakage power, dynamic energy, and area of activation memories with DVS (0.6 V) and baseline (0.54 V) supply voltages. The overhead of Flip-and-Patch (F+P) is also shown for 0.54 V.

	Activation memory			Patching
	DVS	Base	F+P	cache
Leakage power (mW)	315	269.8	288.5	7.8
Dynamic read energy (pJ)	83.8	64.5	71.3	2.5
Dynamic write energy (pJ)	66.4	48	53.7	2.7
Area (mm^2)	6.207		7.129	0.031

memory, the storage overhead is 128 + 128 KiB. Note that, despite conventional memory designs already include similar control bits to distinguish between reliable and faulty contents [29], we conservatively take into account the energy and area overhead of both bit arrays.

Table 2 summarizes the leakage (static) power, dynamic energy, and area of activation memories under different operation modes. Dynamic Voltage Scaling (DVS) and baseline (*Base*) refer to activation memories supplied at safe 0.6 V (V_{min}) and faulty 0.54 V, respectively, whereas label *F+P* alludes to the proposed Flip-and-Patch technique applied to activation memories powered at 0.54 V. The overhead of F+P consists of the required multiplexers, latches, and control bits. Finally, the rightmost column refers to the overhead of the 2.5 KiB patching cache (0.12% bit storage overhead with respect to the 2 MiB activation memory). Control bits and patching cache are supplied at 0.6 V to avoid faults. All the results were obtained with CACTI-P for a 32-nm technology node and ITRS low-power device type [36]. As observed, the power and energy overhead of the F+P approach maintains the enhanced activation memory between the results for 0.6 V and 0.54 V operation modes, whereas the area overhead is 15.4% of the conventional activation memory.

Finally, F+P includes a pair of control bits and 4:1 multiplexers in the critical path of activation memories. As obtained with CACTI-P, the access time increases from 2.69 ns to 2.75 ns. Therefore, we assume that such a small timing overhead does not compromise the cycle time of the accelerator.

5. Experimental evaluation

This section describes the simulation framework used to obtain experimental results. Then, results are presented and discussed, including the impact on CNN accuracy of voltage underscaling and the energy consumption of our proposed techniques. After that, a sensitivity study to the number of faults is introduced. Finally, the effectiveness of our approach is evaluated under the CIFAR-10 input dataset.

5.1. Simulation environment

Similarly to previous frameworks like Ares [9], our fault-injection framework is built on top of the TensorFlow 2.5.0 library [37], which executes high-level CNN descriptions specified in Python. In particular, our framework snoops the output activation values to be provided as input values to the following network layer and modifies them according to the faulty memory bitmap. In this way, we model activation memory operation as if the activation memory would have been exposed to permanent bitcell faults. The snoop stage also performs flipping and patching operations to the corresponding activation words as discussed in Section 4.

Additionally, our framework has been also extended to model the dataflow of the baseline CNN accelerator architecture introduced in Section 2.1, incorporating the proposed Flip-and-Patch technique in the memory ports. Similarly to other recent works [38–40], our framework accurately calculates the execution time (in processor cycles), assuming an access latency of one and three cycles for the patching cache and activation/weights memories, respectively. These latency penalties are consistent with the timing results provided by CACTI-P at a clock frequency of 1 GHz [36]. The systolic PE array accounts for one-cycle penalty for each partial sum and accumulation in a PE.

Apart from performance results (i.e., CNN accuracy and execution time), our framework also provides statistics of read/write memory accesses required to estimate energy consumption. These results are combined with the energy numbers per memory access obtained with CACTI-P (see Table 2) to estimate overall energy consumption. Refer to Section 3 for a description of CNN benchmarks and input dataset.

5.2. Impact on accuracy

Fig. 10 depicts the normalized accuracy of different fault-tolerant mechanisms in activation memories supplied at 0.54 V with respect to a fault-free operation mode with V_{dd} over V_{min}. *Base* refers to the baseline scheme without any fault protection. *I-A ECC* implements an iso-area SECDED ECC protection with respect to the proposed approach, that is, with an equivalent bit-cost. In this way, the activation memory is protected at a granularity of 8-byte blocks. Label *ECC* refers to a higher fault protection at a granularity of 2-byte activation words. That is, 5 ECC bits are required for a 16-bit activation. Label *Flip* refers to the flipping technique alone, whereas *F+P* applies to both flipping and patching techniques in conjunction.

Accuracy is severely affected for the baseline approach due to the high number of permanent faults, falling down to 32.5% on average. Results are consistent with those discussed in Section 3. I-A ECC experiences a marginal accuracy improvement (if any) with respect to the baseline, whereas ECC at a finer granularity improves the accuracy, reaching the original (fault-free) value in DenseNet and ResNet. However, the average accuracy is almost cut in half with respect to the original value.

On the other hand, turning HO activations into LO activations allows the flipping technique to boost the accuracy at least up to 77.2% (VGG16) for every benchmark. However, the presence of L&HO activations prevents the flipping technique from reaching the original accuracy. The combination of flipping and patching techniques overcomes this limitation, obtaining nearly the original accuracy in all the benchmarks.

Notice too that, counter-intuitively, F+P obtains slightly less accuracy than Flip in MobileNet despite the latter approach incorporates L&HO activations. In line with previous work [41], small variations in CNN parameters (e.g., L&HO activations) may actually improve the accuracy of the model.

5.3. Deviation in softmax probability array

With the aim to provide more insights on the accuracy of the different approaches, this section focuses on the probability array of the softmax function of CNN applications. This function establishes the final output of a CNN, assigning a probability to every class of the dataset. In this probability array, the addition of every term sums up to 1, and the probability value of one class usually dominates over the others, that is the top-1 prediction of the network. Unlike the previous section, this section does not focus exclusively on the top-1 accuracy value but on the probabilities of all the dataset classes.

$$SD = \sum_{i=0}^{N-1} |PPA[i] - OPA[i]| \quad (1)$$

In particular, we define the *softmax deviation* (SD) metric as Eq. (1). SD accumulates the absolute probability differences for N dataset classes between the proposed probability array (PPA) obtained with a fault-tolerant approach and the original probability array (OPA) of the

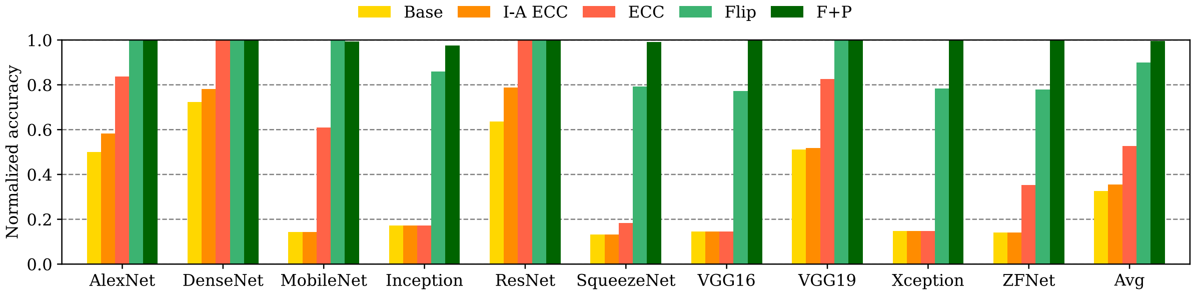


Fig. 10. Normalized accuracy of different approaches at $V_{dd} = 0.54$ V with respect to a conventional fault-free operation mode ($V_{dd} \geq V_{min}$).

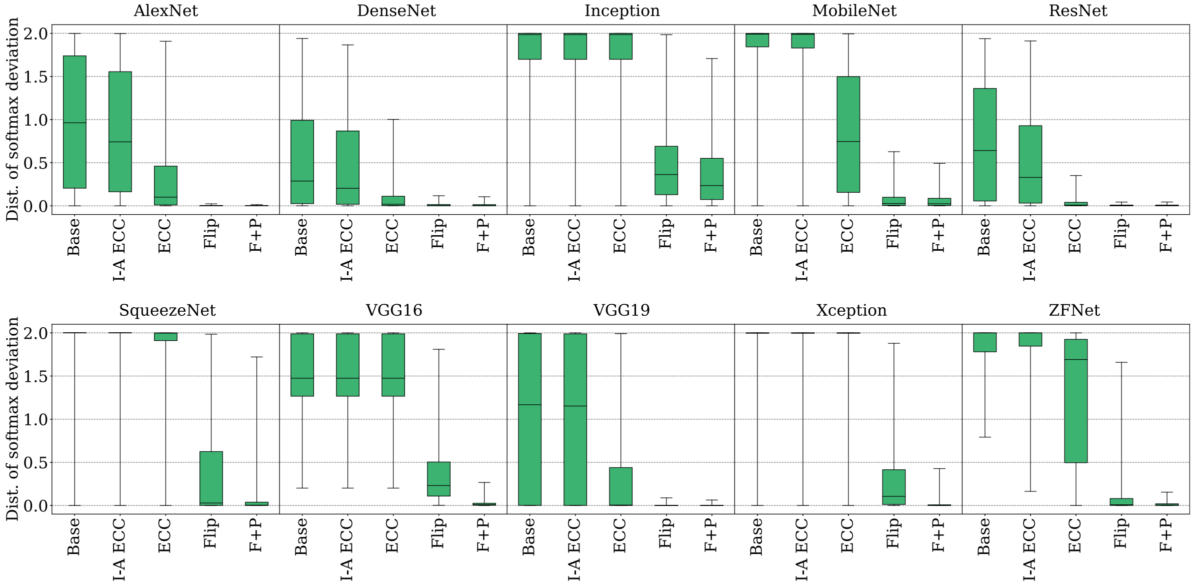


Fig. 11. Distribution of softmax deviation (SD) at $V_{dd} = 0.54$ V with respect to a conventional fault-free operation mode ($V_{dd} \geq V_{min}$).

conventional (fault-free) design. Note that SD ranges from 0 (exactly the same probability arrays) and 2 (both PPA and OPA are completely biased to different classes).

Fig. 11 shows the SD distributions as box-and-whisker plots for the entire input dataset. The top and bottom whiskers represent the maximum and minimum softmax deviations, respectively, whereas the top and bottom box edges specify the 75th and 25th percentiles of the distribution. Finally, the line within the box refers to the median value.

The baseline and IA-ECC approaches show wide SD distributions. In fact, for most applications, including CNNs with moderate (e.g., AlexNet or DenseNet) and high (e.g., MobileNet or Inception) accuracy loss, there are input images where the obtained probability array is exactly the same as the original array, or completely biased to different classes. This confirms that permanent faults impact differently on the softmax classifier depending on the input image.

SD distributions help appreciate accuracy differences between ECC and the proposed techniques. In particular, whereas all these approaches show a 100% top-1 accuracy in DenseNet and ResNet (Fig. 10), SD distributions are wider for ECC, that is, this approach has larger softmax probability deviations with respect to the proposed techniques. In turn, slight SD differences can be seen between Flip and F+P for these applications.

Finally, note that Inception and SqueezeNet are the only CNNs where F+P obtains a relatively large top whisker. In this case, for a few input images, Flip-and-Patch obtains different softmax probability arrays than the fault-free approach, but it still preserves nearly the average original top-1 accuracy (Fig. 10).

5.4. Energy consumption

The end-objective of this work is reducing energy consumption of activation memories by aggressively lowering the supply voltage beyond the capability of DVS solutions, while preserving the application accuracy. This section quantifies the energy savings of the proposed Flip-and-Patch technique. In addition, energy consumption of the state-of-the-art ThUnderVolt technique is also shown. The next subsection briefly describes this approach. Then, energy results are discussed.

5.4.1. ThUnderVolt approach

ThUnderVolt is a framework designed to enable aggressive voltage underscaling in CNN accelerators [11]. In particular, ThUnderVolt detects and recovers from timing errors affecting the PE array. This approach is based on the observation that the timing error rate significantly varies across layers of a given CNN. In this way, contrary to our approach operating at a fixed supply voltage level (V_{dd} = 0.54 V) for the entire inference process, ThUnderVolt proposes a dynamic per-layer voltage underscaling scheme to mitigate the impact on accuracy of such errors, adjusting V_{dd} from 0.6 V to 0.54 V at run time.

Despite the primary focus of ThUnderVolt is on reducing timing errors in logic circuitry, we adapt the proposed voltage-underscaling fitting algorithm to the activation memories of the accelerator, reducing the impact of permanent faults. More precisely, we carefully select the most aggressive supply voltage for each layer, ensuring that the original accuracy is preserved. Such a fine-grain supply voltage setup per layer requires a large profiling effort that not only depends on several faulty bitmaps (one for each V_{dd} value) but also on each CNN application.

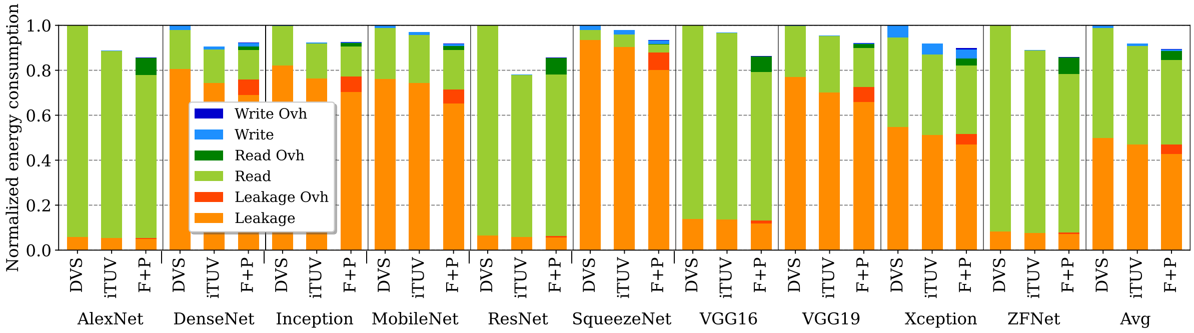


Fig. 12. Normalized energy consumption of an activation memory powered at 0.6 V with DVS, at a variable supply voltage with ideal ThUnderVolt (iTUV), and at 0.54 V with the proposed Flip-and-Patch (F+P) technique.

In this work, we compare Flip-and-Patch against an ideal ThUnderVolt approach that not only assumes a detailed profiling for every CNN benchmark but also discards the non-trivial timing and energy overheads associated with per-layer V_{dd} transitions [42]. These factors collectively establish ThUnderVolt as an ideal approach for reducing V_{dd} below V_{min}.

5.4.2. Results

Fig. 12 plots the normalized energy consumption of an enhanced activation memory supplied at faulty 0.54 V with the proposed F+P technique compared to a conventional activation memory powered at safe 0.6 V using DVS. The ideal ThUnderVolt (iTUV) approach varies V_{dd} from 0.6 V to 0.54 V depending on the requirement of each layer. Energy is classified as either leakage or dynamic consumption. In turn, dynamic energy is split into read and write operations in the activation memory. For each type of energy, label *Ovh* refers to the overhead of the proposed F+P approach as discussed in Section 4.3.

The contribution of leakage expenses over the total energy consumption varies among CNNs and mainly depends on the memory access pattern. That is, applications like SqueezeNet, with a high reuse of activations within the PE array, perform less accesses to the activation memory and consequently mitigate the contribution of dynamic expenses. DenseNet, Inception, and VGG19 are the only benchmarks where the leakage contribution of F+P, including the leakage overhead, noticeably surpasses such a type of energy for iTUV. This is mainly due two main reasons. First, like SqueezeNet, these CNNs exhibit a high reuse of activations. Second, DenseNet and Inception are the two deepest CNNs and the execution time has a remarkable impact on the leakage expenses. Compared to DVS, the leakage overhead of F+P is compensated by the leakage savings provided by the V_{dd} reduction in all the benchmarks.

The V_{dd} reduction has a greater impact on dynamic energy savings, since these expenses grow quadratically with V_{dd}. In this sense, applications with a poor activation reuse (high dynamic consumption), such as VGG16, obtain a larger reduction of overall energy consumption. Similarly to the leakage energy, compared to DVS, the dynamic energy overhead of F+P is largely compensated in all the studied benchmarks thanks to underscaling V_{dd}. The same effect can be appreciated, for all the benchmarks except DenseNet and ResNet, comparing the dynamic contributions of iTUV and F+P. In these two applications, iTUV takes advantage over F+P because the majority of layers operate at 0.54 V without hurting the accuracy of the network. Finally, reads are more frequent than write operations for all the analyzed CNNs, since activations of a given layer are usually read several times according to the number of weight filters applied to the input data but written just once.

Overall, the average energy savings of Flip-and-Patch are by 3.2% and 10.5% compared to ideal ThUnderVolt and DVS, respectively. Such energy savings might seem relatively low; however, recall that: (i) neither timing nor energy overheads are assumed for ideal ThUnderVolt because of supply voltage adjustment at run time, and (ii) V_{dd} just reduces from 0.6 V to 0.54 V. Further supply voltage reductions are not possible since the assumed real hardware platform stops operating (see Section 2.2). Compared to a conventional accelerator powered at nominal supply voltage (0.9 V), the average energy savings are as much as 46.6%.

5.5. Sensitivity analysis

This section aims to quantify the potential impact on accuracy resulting from more challenging reliability scenarios that future technologies may face. These scenarios could be the result of either a deeper voltage underscaling beyond 0.54 V or the assumption that future technology nodes will exhibit more pronounced process variation effects.

Specifically, we have generated scenarios with two, three, and four times more faulty bit cells than the former reliability scenario assumed in previous experiments. To do this, we applied the real memory faulty bitmap (see Section 2.2) several times on the activation memories. In this way, the 4× reliability scenario consists of 1.8% of LO activations, 1.8% of HO activations, and 0.014% of L&HO activations.

Fig. 13 illustrates the raw accuracy in the former reliability scenario (1×) and prospective ones (2×, 3×, and 4×). As expected, accuracy drops with the number of faults. AlexNet, DenseNet, ResNet, and VGG19, where the ECC-based and even the baseline scheme obtain relatively moderate accuracy losses under the 1× scenario, significantly degrade the accuracy under more faults, providing a random guessing in DenseNet. On the other hand, the flipping technique alone and combined with patching maintain the original accuracy value in these applications regardless of the reliability scenario.

For the remaining benchmarks, flipping alone does not hold the original accuracy mainly due to a significant number of L&HO activations, obtaining a large accuracy degradation for Inception, SqueezeNet, VGG16, and Xception. Nevertheless, the combination of both flipping and patching boosts the accuracy close to the original value. The accuracy loss of F+P in these applications is exclusively attributed to the high number of LO activations (both original LO activations and those HO activations transformed to LO activations with the flipping approach), since the patching cache is large enough to store all the L&HO activations, including those of the 4× reliability model (see Section 4.2). On the other hand, in MobileNet and ZFNet, F+P achieves the original accuracy regardless of the number of faults.

5.6. CIFAR-10 dataset

With the aim to investigate the generality of Flip-and-Patch, this section evaluates our proposed approach under an alternative input dataset for image classification: CIFAR-10 [43]. For every studied CNN benchmark, Table 3 summarizes the minimum number of integer and fraction bits for activations that ensure the same accuracy as a 32-bit floating point (IEEE-754) representation and the obtained

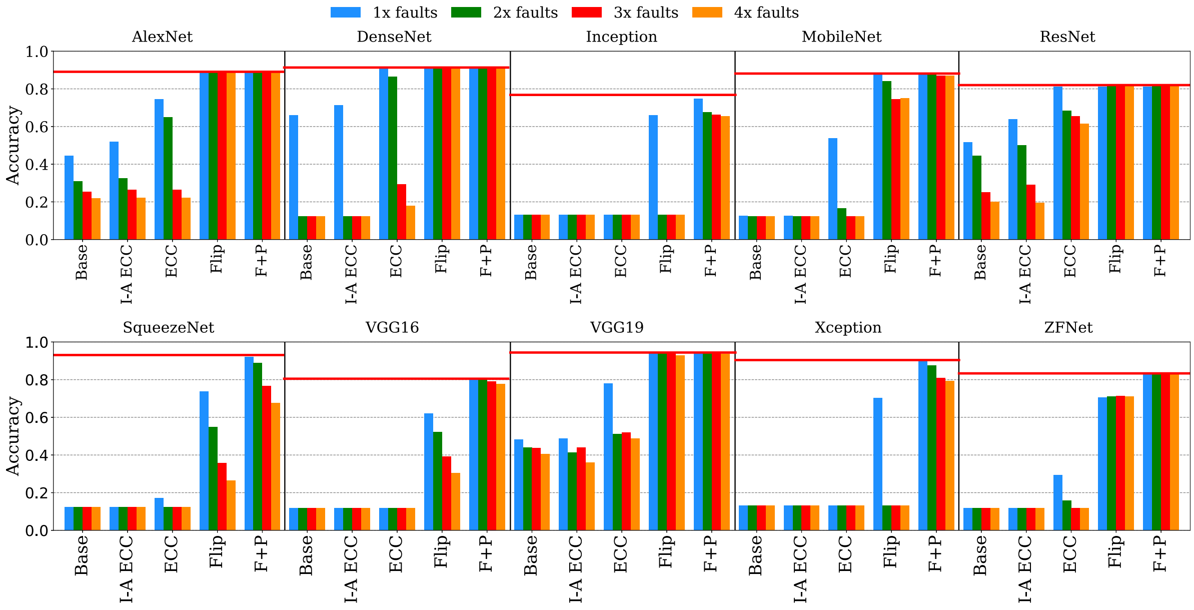


Fig. 13. Raw accuracy for different fault-tolerant approaches under 1×, 2×, 3×, and 4× reliability scenarios. The horizontal red line represents the original accuracy of each neural network.

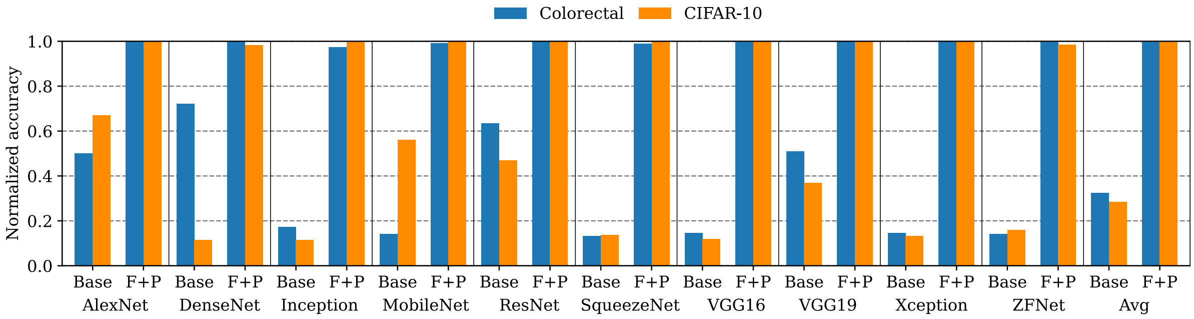


Fig. 14. Normalized accuracy of baseline and Flip-and-Patch approaches at $V_{dd} = 0.54$ V with respect to a conventional fault-free operation mode ($V_{dd} \geq V_{min}$) for colorectal and CIFAR-10 datasets.

Table 3
Activation quantization and accuracy of the studied CNN benchmarks for Colorectal and CIFAR-10 datasets.

Benchmark	Colorectal		CIFAR-10	
	Activation rep. (integer, fraction)	Accuracy	Activation rep. (integer, fraction)	Accuracy
AlexNet	4 bits, 4 bits	0.89	4 bits, 5 bits	0.71
DenseNet	3 bits, 5 bits	0.92	9 bits, 5 bits	0.84
Inception	8 bits, 6 bits	0.79	8 bits, 8 bits	0.86
MobileNet	4 bits, 9 bits	0.88	4 bits, 6 bits	0.59
ResNet	4 bits, 4 bits	0.81	5 bits, 5 bits	0.72
SqueezeNet	bits, 4 bits	0.93	4 bits, 7 bits	0.71
VGG16	3 bits, 8 bits	0.81	3 bits, 9 bits	0.80
VGG19	8 bits, 2 bits	0.94	2 bits, 8 bits	0.77
Xception	7 bits, 6 bits	0.90	7 bits, 5 bits	0.78
ZFNet	4 bits, 6 bits	0.83	4 bits, 8 bits	0.62

accuracy value under CIFAR-10. For comparison purposes, the table also includes the results for the colorectal cancer histology dataset used in the previous experiments. The new dataset consists of 6,500 test images.

As observed, for a given CNN benchmark, the number of integer and fraction bits is quite similar for the two studied datasets. Differences represent at most three bits apart from DenseNet and VGG19. On the other hand, in all the benchmarks except Inception, the accuracy value is lower for CIFAR-10 with respect to the colorectal dataset. This is mainly due to the number of training epochs has been significantly reduced for CIFAR-10. Nevertheless, this gives us the opportunity to evaluate Flip-and-Patch under CNN applications with a relatively low accuracy.

Fig. 14 shows the normalized accuracy for baseline and Flip-and-Patch approaches supplied at 0.54 V with respect to a fault-free operation mode with $V_{dd} > V_{min}$ for the two different datasets. In most benchmarks, differences between datasets are small. Without any fault-protection mechanism, the baseline scheme suffers a significant accuracy degradation. On the other hand, F+P nearly reaches the original accuracy in all the benchmarks. The most notable difference between datasets can be found in DenseNet, where the accuracy of the baseline scheme dramatically drops in CIFAR-10. This is mainly due to, in such a dataset, the required number of integer bits to represent activations scales up to 9 bits. On average, the normalized accuracy for baseline and Flip-and-Patch is 28.5% and 99.6%, respectively, for CIFAR-10.

6. Related work

This section classifies prior work into approaches focusing on CNN accelerators supplied at V_{dd} below V_{min}, patching techniques for general-purpose microprocessors, recent approaches exploiting redundant logic or algorithm-based fault tolerant strategies, and software techniques consisting of clipping algorithms.

6.1. CNN accelerators

State-of-the-art approaches for CNN accelerators focus their effort on determining which parameters of the neural networks are most sensitive to the final output. Once these parameters are identified, an attempt is made to protect them in order to avoid large accuracy deviations when reducing the supply voltage during the inference process.

ThUnderVolt characterizes which layers of the CNNs are most susceptible to faults, and consequently, each layer defines a different supply voltage in order to minimize the impact on accuracy [11]. See Section 5.4.1 for further details.

Salami et al. identify the most vulnerable CNN layers and modify the placement algorithm of an FPGA compilation process to make sure that those layers are not stored in faulty memory blocks [3]. Similarly to this work, we identify faulty blocks with a memory test prior to the deployment of the device in the field. However, unlike Salami et al., our proposed approach does not depend on the characteristics of neural networks neither alters the placement algorithm of an FPGA.

Zhang et al. expose the training process of the neural network to permanent faults with the aim to compute a set of weights that hides the impact on accuracy of faults during the inference process [10]. However, similarly to the above works, this solution is tailored to a specific neural network architecture and requires programmer intervention.

Finally, word and bit masking techniques have been proposed to deal with transient faults in weight memories [8] and registers within the PE array [44]. In particular, they reset faulty weights to zero, protect the sign bit assuming the same logic value of the adjacent bit, or protect the remaining bits assuming the same logic value of the sign bit. However, these mechanisms detect faults at bit level by monitoring circuit delays with the use of Razor double-sampling methods, which may incur significant power consumption.

6.2. Patching techniques for general-purpose processors

In the context of CPU superscalar processors, idle entries of pipeline structures like trace caches, MSHR, or store queues have been exploited as patching entries storing reliable replicas of faulty L1 cache contents [28]. This solution complicates the design and verification of the processor, since memory consistency management is propagated to such pipeline structures.

Patching has also been used in GPU register files. GR-Guard exploits dead registers containing useless data to store useful data from faulty registers [29]. Dead registers are identified at run time with the assistance of the compiler and modifications to the instruction set. Alternatively, DC-Patch leverages the observation that registers are compressible at run time, and allocates compressed data to faulty registers, avoiding the use of defective bitcells [45].

6.3. Redundant logic and algorithm-based fault tolerance

Li et al. propose hardening mechanisms that detect and correct transient faults at run time with the addition of Triple Modular Redundant (TMR) logic to the conventional data path [46]. To reduce the TMR overhead, Libano et al. only protect those CNN layers identified as more vulnerable to faults [47].

Algorithm-Based Fault Tolerant (ABFT) approaches have been shown as an effective alternative to full redundant system solutions for CNN accelerators. These techniques compute checksums for input data, store them with the original data, perform the original and redundant computation, verify outputs, and correct transient faults at run time. To eliminate the overhead of fault correction, recent studies focus on convolution operations, detecting faults at run time with the goal of overclocking the system [48] or exploiting the inherent characteristics of fixed-point arithmetic [49]. Santos et al. have also employed ABFT strategies in the context of GPU systems [50]. Additionally, they redesign the maxpool layers of CNNs to mitigate the impact of transient faults.

6.4. Clipping algorithms

Software clipping algorithms identify long numerical deviations in CNN parameters as a consequence of faults and mitigate their contribution to the output. Ozen and Orailoglu employ regularization terms during the training process to penalize outlier weights and minimize the loss function [41]. Other works profile the CNN applications and modify their architecture, adding layers that restrict outlier parameters to a defined numerical range during the inference process [51,52].

7. Conclusions and future work

This work has explored the possibility of drastically reducing the supply voltage of activation memories in CNN inference accelerators to save energy consumption. To address the impact on CNN accuracy of bitcell permanent faults as a consequence of supply voltage underscaling, this work has proposed a couple of low-cost microarchitectural mechanisms based on flipping and patching approaches. These mechanisms are a consequence of a characterization study that identified the impact on accuracy of different fault patterns in activation memories.

The flipping technique transforms the representation of those activations with a low number of faults, whereas the patching technique provides a fault-free backup storage for those activations with a high number of faults. Contrary to state-of-the-art approaches, the proposed Flip-and-Patch technique does not add any burden to the programmer neither depend on specific characteristics of CNN applications.

Experimental results have shown that, compared to a conventional CNN accelerator supplied at a safe voltage of 0.6 V, an enhanced accelerator supplied at 0.54 V with Flip-and-Patch reduces the average energy consumption of activation memories by 10.5%, while maintaining the original (fault free) accuracy with a negligible impact on system performance (less than 0.05% for every application). Compared to the state-of-the-art ThUnderVolt approach, which dynamically adjusts the supply voltage at run time and assuming neither timing nor energy overheads for its implementation, the average energy savings are by 3.2%.

As future directions, we plan to study the applicability of Flip-and-Patch in weight memories and other AI accelerators such as those employed to speed up the inference process of transformer networks.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Data will be made available on request.

Acknowledgments

All authors acknowledge support from grants (1) PID2019-105660RB-C21 and PID2022-136454NB-C22 from Agencia Estatal de Investigación (AEI), Spain, and (2) gaZ: T58_23R research group from Dept. of Science, University and Knowledge Society, Government of Aragon, Spain. The funding agencies had no role in study design, data collection and analysis, decision to publish, or preparation of the manuscript.

References

[1] J. Leng, Y. Zu, V.J. Reddi, GPU voltage noise: Characterization and hierarchical smoothing of spatial and temporal voltage noise interference in GPU architectures, in: Proceedings of the IEEE 21st International Symposium on High Performance Computer Architecture, 2015, pp. 161–173.

[2] J. Leng, A. Buyuktosunoglu, R. Bertran, P. Bose, V.J. Reddi, Safe limits on voltage reduction efficiency in GPUs: A direct measurement approach, in: Proceedings of the 48th Annual IEEE/ACM International Symposium on Microarchitecture, 2015, pp. 294–307.

[3] B. Salami, O. S. Unsal, A. Cristal Kestelman, Comprehensive evaluation of supply voltage underscaling in FPGA on-chip memories, in: Proceedings of the 51st Annual IEEE/ACM International Symposium on Microarchitecture, 2018, pp. 724–736, http://dx.doi.org/10.1109/MICRO.2018.00064.

[4] C. Wilkerson, H. Gao, A.R. Alameldeen, Z. Chishti, M. Khellah, S. Lu, Trading off cache capacity for reliability to enable low voltage operation, in: Proceedings of the ACM/IEEE 35th Annual International Symposium on Computer Architecture, 2008, pp. 203–214.

[5] J. Kim, N. Hardavellas, K. Mai, B. Falsafi, J. Hoe, Multi-bit error tolerant caches using two-dimensional error coding, in: Proceedings of the 40th Annual IEEE/ACM International Symposium on Microarchitecture, 2007, pp. 197–209.

[6] J. Tan, Q. Wang, K. Yan, X. Wei, X. Fu, Saca-FI: A microarchitecture-level fault injection framework for reliability analysis of systolic array based CNN accelerator, Elsevier Future Gener. Comput. Syst. 147 (2023) 251–264.

[7] J. Lee, C. Kim, S. Kang, D. Shin, S. Kim, H.-J. Yoo, UNPU: An energy-efficient deep neural network accelerator with fully variable weight bit precision, IEEE J. Solid-State Circuits 54 (2019) 173–185.

[8] B. Reagen, P. Whatmough, R. Adolf, S. Rama, H. Lee, S.K. Lee, J.M. Hernández-Lobato, G.-Y. Wei, D. Brooks, Minerva: Enabling low-power, highly-accurate deep neural network accelerators, in: Proceedings of the 43rd International Symposium on Computer Architecture, 2016, pp. 267–278.

[9] B. Reagen, U. Gupta, L. Pentecost, P. Whatmough, S.K. Lee, N. Mulholland, D. Brooks, G.-Y. Wei, Ares: A framework for quantifying the resilience of deep neural networks, in: Proceedings of the 55th ACM/ESDA/IEEE Design Automation Conference, 2018, pp. 1–6.

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

[11] J. Zhang, K. Rangineni, Z. Ghodsi, S. Garg, ThUnderVolt: Enabling aggressive voltage underscaling and timing error resilience for energy efficient deep learning accelerators, in: Proceedings of the 55th ACM/ESDA/IEEE Design Automation Conference, 2018, pp. 1–6.

[12] Y. Toca-Díaz, N. Landeros Muñoz, R. Gran Tejero, A. Valero, On fault-tolerant microarchitectural techniques for voltage underscaling in on-chip memories of CNN accelerators, in: Proceedings of the 26th Euromicro Conference on Digital System Design, 2023, pp. 138–145.

[13] A. Krizhevsky, I. Sutskever, G.E. Hinton, ImageNet classification with deep convolutional neural networks, Adv. Neural Inf. Process. Syst. 25 (2012) 1097–1105.

[14] F.N. Iandola, M.W. Moskewicz, S. Karayev, R.B. Girshick, T. Darrell, K. Keutzer, DenseNet: Implementing efficient ConvNet descriptor pyramids, 2014, CoRR abs/1404.1869, URL http://arxiv.org/abs/1404.1869.

[15] C. Szegedy, S. Ioffe, V. Vanhoucke, A.A. Alemi, Inception-v4, inception-ResNet and the impact of residual connections on learning, in: Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, AAAI Press, 2017, pp. 4278–4284.

[16] A.G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, H. Adam, MobileNets: Efficient convolutional neural networks for mobile vision applications, 2017, CoRR abs/1704.04861, URL http://arxiv.org/abs/1704.04861.

[17] F.N. Iandola, S. Han, M.W. Moskewicz, K. Ashraf, W.J. Dally, K. Keutzer, SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and <0.5mb model size, 2016, CoRR abs/1602.07360.

[18] K. Simonyan, A. Zisserman, Very deep convolutional networks for large-scale image recognition, 2014, CoRR abs/1409.1556.

[19] F. Chollet, Xception: Deep learning with depthwise separable convolutions, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2017, pp. 1251–1258.

[20] M.D. Zeiler, R. Fergus, Visualizing and Understanding Convolutional Networks, in: Springer Lecture Notes in Computer Science, vol. 8689, 2014.

[21] Y. Chen, T. Luo, S. Liu, S. Zhang, L. He, J. Wang, L. Li, T. Chen, Z. Xu, N. Sun, O. Temam, DaDianNao: A machine-learning supercomputer, in: Proceedings of the 47th Annual IEEE/ACM International Symposium on Microarchitecture, 2014, pp. 609–622.

[22] N.P. Jouppi, C. Young, N. Patil, D.A. Patterson, G. Agrawal, R. Bajwa, S. Bates, S. Bhatia, N. Boden, A. Borchers, R. Boyle, P. Cantin, C. Chao, C. Clark, J. Coriell, M. Daley, M. Dau, J. Dean, B. Gelb, T.V. Ghaemmaghami, R. Gottipati, W. Gulland, R. Hagmann, C.R. Ho, D. Hogberg, J. Hu, R. Hundt, D. Hurt, J. Ibarz, A. Jaffey, A. Jaworski, A. Kaplan, H. Khaitan, D. Killebrew, A. Koch, N. Kumar, S. Lacy, J. Laudon, J. Law, D. Le, C. Leary, Z. Liu, K. Lucke, A. Lundin, G. MacKean, A. Maggiore, M. Mahony, K. Miller, R. Nagarajan, R. Narayanaswami, R. Ni, K. Nix, T. Norrie, M. Omernick, N. Penukonda, A. Phelps, J. Ross, M. Ross, A. Salek, E. Samadiani, C. Severn, G. Sizikov, M. Snelham, J. Souter, D. Steinberg, A. Swing, M. Tan, G. Thorson, B. Tian, H. Toma, E. Tuttle, V. Vasudevan, R. Walter, W. Wang, E. Wilcox, D.H. Yoon, In-datacenter performance analysis of a tensor processing unit, in: Proceedings of the 44th Annual International Symposium on Computer Architecture, 2017, pp. 1–12.

[23] K. Seshadri, B. Akin, J. Laudon, R. Narayanaswami, A. Yazdanbakhsh, An evaluation of edge TPU accelerators for convolutional neural networks, in: Proceedings of the IEEE International Symposium on Workload Characterization, 2022, pp. 79–91.

[24] A. Samajdar, Y. Zhu, P.N. Whatmough, M. Mattina, T. Krishna, SCALE-sim: Systolic CNN accelerator, 2018, CoRR abs/1811.02883, URL http://arxiv.org/abs/1811.02883.

[25] S. Han, X. Liu, H. Mao, J. Pu, A. Pedram, M.A. Horowitz, W.J. Dally, EIE: Efficient inference engine on compressed deep neural network, in: Proceedings of the 43rd International Symposium on Computer Architecture, 2016, pp. 243–254.

[26] P. Judd, J. Albericio, T. Hetherington, T.M. Aamodt, A. Moshovos, Stripes: Bit-serial deep neural network computing, in: Proceedings of the 49th Annual IEEE/ACM International Symposium on Microarchitecture, 2016, pp. 1–12.

[27] H. Sharma, J. Park, N. Suda, L. Lai, B. Chau, V. Chandra, H. Esmaeilzadeh, Bit fusion: Bit-level dynamically composable architecture for accelerating deep neural network, in: Proceedings of the ACM/IEEE 45th Annual International Symposium on Computer Architecture, 2018, pp. 764–775.

[28] D.J. Palframan, N. Kim, M.H. Lipasti, iPatch: Intelligent fault patching to improve energy efficiency, in: Proceedings of the IEEE 21st International Symposium on High Performance Computer Architecture, 2015, pp. 428–438.

[29] J. Tan, S.L. Song, K. Yan, X. Fu, A. Marquez, D. Kerbyson, Combating the reliability challenge of GPU register file at low supply voltage, in: Proceedings of the 25th International Conference on Parallel Architectures and Compilation Techniques, 2016, pp. 3–15.

[30] A. Chatzidimitriou, G. Panadimitriou, D. Gizopoulos, S. Ganapathy, J. Kalamatianos, Assessing the effects of low voltage in branch prediction units, in: Proceedings of the IEEE International Symposium on Performance Analysis of Systems and Software, 2019, pp. 127–136.

[31] ARM, ARM11 MPCore™ Processor. Revision: r2p0. Technical Reference Manual, Tech. Rep., ARM Limited, 2008.

[32] J.N. Kather, C.-A. Weis, F. Bianconi, S.M. Melchers, L.R. Schad, T. Gaiser, A. Marx, F.G. Zöllner, Multi-class texture analysis in colorectal cancer histology, Nat. Sci. Rep. 6 (2016).

[33] M.A. Hanif, M. Shafique, DNN-life: An energy-efficient aging mitigation framework for improving the lifetime of on-chip weight memories in deep neural network hardware architectures, in: Proceedings of the Design, Automation & Test in Europe Conference & Exhibition, 2021, pp. 729–734.

[34] S. Salamin, G. Zervakis, O. Spantidi, I. Anagnostopoulos, J. Henkel, H. Amrouch, Reliability-aware quantization for anti-aging NPUs, in: Proceedings of the Design, Automation & Test in Europe Conference & Exhibition, 2021, pp. 1460–1465.

[35] N. Landeros Muñoz, A. Valero, R. Gran Tejero, D. Zoni, Gated-CNN: Combating NBTI and HCI aging effects in on-chip activation memories of convolutional neural network accelerators, Elsevier J. Syst. Archit. 128 (2022) 1–13.

[36] R. Balasubramonian, A.B. Kahng, N. Muralimanohar, A. Shafiee, V. Srinivas, CACTI 7: New tools for interconnect exploration in innovative off-chip memories, ACM Trans. Archit. Code Optim. (ISSN: 1544-3566) 14 (2) (2017).

[37] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G.S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mane, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viegas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, X. Zheng, TensorFlow: Large-scale machine learning on heterogeneous distributed systems, 2016, CoRR abs/1603.04467, arXiv:1603.04467.

[38] A. Parashar, P. Raina, Y.S. Shao, Y.-H. Chen, V.A. Ying, A. Mukkara, R. Venkatesan, B. Khailany, S.W. Keckler, J. Emer, Timeloop: A systematic approach to DNN accelerator evaluation, in: Proceedings of the IEEE International Symposium on Performance Analysis of Systems and Software, 2019, pp. 304–315.

[39] L. Mei, H. Liu, T. Wu, H.E. Sumbul, M. Verhelst, E. Beigne, A uniform latency model for DNN accelerators with diverse architectures and dataflows, in: Proceedings of the Design, Automation & Test in Europe Conference & Exhibition, 2022, pp. 220–225.

[40] T. Hotfilter, P. Schmidt, J. Höfer, F. Kreß, T. Harbaum, J. Becker, An analytical model of configurable systolic arrays to find the best-fitting accelerator for a given DNN workload, in: Proceedings of the DroneSE and RAPIDO: System Engineering for Constrained Embedded Systems, 2023, pp. 73–78.

[41] E. Ozen, A. Orailoglu, SNR: Squeezing numerical range defuses bit error vulnerability surface in deep neural networks, ACM Trans. Embed. Comput. Syst. 20 (5s) (2021).

[42] J. Park, D. Shin, N. Chang, M. Pedram, Accurate modeling and calculation of delay and energy overheads of dynamic voltage scaling in modern high-performance microprocessors, in: Proceedings of the ACM/IEEE International Symposium on Low-Power Electronics and Design, 2010, pp. 419–424.

[43] A. Krizhevsky, Learning Multiple Layers of Features from Tiny Images, Tech. Rep., University of Toronto, 2009.

[44] B. Salami, O.S. Unsal, A.C. Kestelman, On the resilience of RTL NN accelerators: Fault characterization and mitigation, in: Proceedings of the 30th International Symposium on Computer Architecture and High Performance Computing, 2018, pp. 322–329.

[45] A. Valero, D. Suárez-Gracia, R. Gran-Tejero, DC-patch: A microarchitectural fault patching technique for GPU register files, IEEE Access 8 (2020) 173276–173288.

[46] G. Li, S.K.S. Hari, M. Sullivan, T. Tsai, K. Pattabiraman, J. Emer, S.W. Keckler, 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, 2017.

[47] F. Libano, B. Wilson, J. Anderson, M.J. Wirthlin, C. Cazzaniga, C. Frost, P. Rech, Selective hardening for neural networks in FPGAs, IEEE Trans. Nucl. Sci. 66 (1) (2019) 216–222.

[48] T. Marty, T. Yuki, S. Derrien, Safe overclocking for CNN accelerators through algorithm-level error detection, IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst. 39 (12) (2020) 4777–4790.

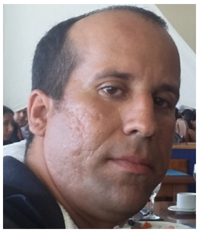
[49] S.K.S. Hari, M.B. Sullivan, T. Tsai, S.W. Keckler, Making convolutions resilient via algorithm-based error detection techniques, IEEE Trans. Dependable Secure Comput. 19 (4) (2022) 2546–2558.

[50] F.F.d. Santos, P.F. Pimenta, C. Lunardi, L. Draghetti, L. Carro, D. Kaeli, P. Rech, Analyzing and increasing the reliability of convolutional neural networks on GPUs, IEEE Trans. Reliab. 68 (2) (2019) 663–677.

[51] L.-H. Hoang, M.A. Hanif, M. Shafique, FT-ClipAct: Resilience analysis of deep neural networks and improving their fault tolerance using clipped activation, in: Proceedings of the 23rd Conference on Design, Automation and Test in Europe, 2020, pp. 1241–1246.

[52] Z. Chen, G. Li, K. Pattabiraman, A low-cost fault corrector for deep neural networks through range restriction, in: Proceedings of the 51st Annual IEEE/IFIP International Conference on Dependable Systems and Networks, 2021, pp. 1–13.

Yamilka Toca-Díaz received the B.S. and M.S. degrees in Computer Science from Universidad de Camagüey, Cuba, in 2007 and 2013, respectively. She is currently working toward the Ph.D. degree in Computer Engineering at the Department of Computer Science and Systems Engineering, Universidad de Zaragoza, Spain. Her research interests include the design of machine learning accelerators with a focus on reliability.



Reynier Hernández Palacios graduated in Computer Science at University of Camagüey, Cuba. He received the M.S. degree in Applied Computer Science from the same institution in 2016. From 2009 to 2015, he was a part-time Instructor Professor at the University of Camagüey, where he is currently a researcher and software developer. He is affiliated to the Unión de Informáticos de Cuba, where he teaches courses, conferences, and workshops on a regular basis. He has led research in the area of Artificial Intelligence related to Tourism, Health, and Patrimony. His research interests include deep learning, cloud computing, parallel and highly scalable architectures, and algorithmic efficiency.

Rubén Gran Tejero graduated in Computer Science from University of Zaragoza, Spain. He received his Ph.D. from Polytechnic University of Catalonia (UPC), Spain, in 2010. He is currently an Associate Professor in the Department of Computer Science and Systems Engineering at University of Zaragoza. He has been Program Committee Member of several conferences and workshops in the area: IPDPS, HPCS, and PMBS. His research interests include hard real-time systems, hardware for reducing worst-case execution time and energy consumption, efficient processor microarchitecture, and effective programming for parallel and heterogeneous systems.

Alejandro Valero received the Ph.D. degree in Computer Engineering from Universitat Politècnica de València, Spain, in 2013. From 2013 to 2015, he was a visiting researcher with Northeastern University, Boston, MA, USA, and University of Cambridge, UK. From 2016 to 2021, he was an Assistant Professor with the Department of Computer Science and Systems Engineering, Universidad de Zaragoza, Spain. Since 2021, he has been an Associate Professor with the same department and institution. His Ph.D. research was recognized with multiple awards, including the 2012 Intel Doctoral Student Honor Award and the Gold Medal in the 2013 ACM Student Research Competition (SRC) held in ICS-27. He has been Technical Program Committee Member of several conferences, workshops, and research competitions, including DATE, ICCD, PMBS, and ACM SRC Grand Finals. His research interests include GPU and ASIC architectures, memory hierarchy design, energy efficiency, and fault tolerance. Prof. Valero is a member of the Aragon Institute of Engineering Research (I3A) and the HiPEAC European NoE.