Gated-CNN: Combating NBTI and HCI Aging Effects in On-Chip Activation Memories of Convolutional Neural Network Accelerators

Nicolás Landeros Muñoz[1], Alejandro Valero[2], Rubén Gran Tejero[2], and Davide Zoni[1]

[1]*Dipartimento di Elettronica Informazione e Bioingegneria, Politecnico di Milano, Italy*

[2]*Department of Computer Science and Systems Engineering, Universidad de Zaragoza, Spain*

Abstract

Negative Bias Temperature Instability (NBTI) and Hot Carrier Injection (HCI) are two of the main reliability threats in current technology nodes. These aging phenomena degrade the transistor's threshold voltage (V_{th}) over the lifetime of a digital circuit, resulting in slower transistors that eventually lead to a faulty operation when the critical paths become longer than the processor cycle time. Among all the transistors on a chip, the most vulnerable transistors to such wearout effects are those used to implement SRAM storage, since memory cells are continuously degrading. In particular, NBTI ages PMOS cell transistors when a given logic value is stored for a long period (i.e., a long duty cycle), whereas HCI does the same in NMOS cell transistors not only when the stored value flips but also when it is accessed. This work focuses on mitigating aging in the on-chip SRAM memories of Convolutional Neural Network (CNN) accelerators storing activations. This paper makes two main contributions. At the software level, we quantify the aging induced by current CNN benchmarks with a characterization study of duty cycle, flip, and access patterns in every activation memory cell. Based on the insights from this study, this work proposes a novel microarchitectural technique, Gated-CNN, that ensures a uniform aging degradation of every memory cell. To do so, Gated-CNN proposes power-gating and address rotation techniques tailored to the memory demands and temporal/spatial localities exhibited by CNN applications, as well as the memory organization and management of CNN accelerators. Experimental results show that, compared to a conventional design, the average V_{th} degradation savings are at least as much as 49% depending on the type of transistor.

Keywords: Access patterns, bit flip patterns, duty cycle, Hot Carrier Injection, Negative Bias Temperature Instability, threshold voltage degradation.

1. Introduction

The end of the Dennard scaling and the Moore's Law era are the major drivers of the computer architecture community toward domain-specific accelerator chips. Unlike general-purpose processors, accelerators are optimized to handle a specific application domain, while delivering a higher performance-to-power ratio under a limited chip budget. These domains include computer vision, speech recognition, natural language processing, autonomous driving, among others, whose applications are efficiently run with machine learning algorithms like Deep Neural Networks (DNNs).

DNN accelerators have consolidated as a commodity device that complements computing platforms from high-performance to embedded systems. Since the emergence of the DianNao accelerator [1], both the academia and the industry have proposed many architectural organizations to cope with the massive convolution computation in the inference process of Convolutional Neural Networks (CNNs) [2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12]. However, like general-purpose processors, CNN accelerators are implemented using CMOS transistors and they usually demand high on-chip Static Random-Access Memory (SRAM) storage requirements to cache CNN parameters consisting of weights (synapses) and activations (neurons) of the different neural network layers.

Two of the main deleterious effects that speed up the CMOS transistor wearout are known as Negative Bias Temperature Instability (NBTI) and Hot Carrier Injection (HCI) [13]. These effects degrade the transistor's threshold voltage (V_{th}) over the lifetime of a digital circuit. Such a degradation causes an increase in the transistor's V_{th}, and therefore in the transistor's switching delay, resulting in permanent faults when the critical paths become longer than the processor cycle time.

By design, SRAM cells are particularly sensitive to both NBTI and HCI failure effects since they are continuously aging. NBTI degrades PMOS transistors when a given logic value is stored for a long period (i.e., a long duty cycle or on/off ratio), whereas HCI deteriorates not only NMOS loop inverter transistors when the stored logic value flips but also NMOS pass transistors when contents are accessed (i.e., on/off switching frequency). In addition, HCI may also degrade the drain current of both NMOS and PMOS transistors [14]. Overall, these situations are strongly related to each other, meaning that combating solely NBTI might aggravate HCI as a side effect, and

[1]Corresponding author
Email address: `alvabre@unizar.es` (A. Valero)

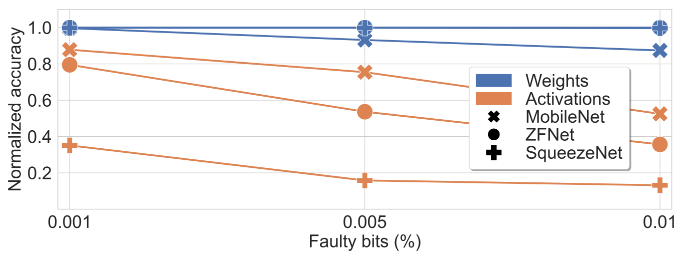


Figure 1: Normalized accuracy after injecting permanent faults in either weights or activations with respect to the original accuracy without faults. The number of faulty bits is calculated as a percentage of the largest CNN layer and it is the same for both weights and activations. The plotted results refer to the average accuracy after 100 trials.

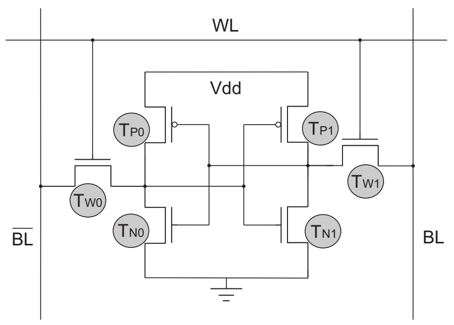


Figure 2: Implementation of a 6T SRAM cell, distinguishing between inverter loop PMOS (T_{Pi}) and NMOS (T_{Ni}) transistors, as well as NMOS (T_{Wi}) pass transistors.

vice versa.

CNN parameters are usually represented with fixed-point data types, as opposite to floating-point, with the aim to reduce the computing and energy consumption requirements of CNN inference accelerators [1, 2, 3, 4, 7, 9, 12]. Recent studies focusing on weights have shown that fixed-point data types are inherently more resilient to faults than floating-point counterparts [15]. Unlike weights, activations usually comprise a larger range of values, which can severely compromise the CNN accuracy even using fixed-point data types.

To check the previous claim, we conducted an experiment in which permanent bit faults are randomly injected at different rates in either weights or activations represented with 16-bit fixed-point data. Figure 1 plots the normalized accuracy of three different CNNs under faults with respect to the fault-free original accuracy. Activations are more vulnerable to faults than weights, even for a percentage of faulty bits as low as 0.001%. In contrast, under faulty weights, accuracy does not degrade in ZFNet and SqueezeNet regardless of the studied fault rate.

The state-of-the-art approach for aging mitigation in on-chip SRAM storage of CNN accelerators solely focuses on NBTI in memories storing weights [16]. In contrast, our work addresses not only NBTI but also HCI in on-chip memories of CNN accelerators storing activations. To do so, this paper makes two main contributions:

- We present a comprehensive characterization study of the duty cycle, flip, and access patterns that current CNN applications induce to every activation memory cell.
- Based on the previous study, we propose a novel aging-aware microarchitectural mechanism, Gated-CNN, exploiting power-gating and address rotation techniques tailored to the specific memory requirements and temporal/spatial localities of CNN applications, in addition to the memory organization and management of current CNN accelerators.

Experimental results show that, compared to a conventional design, Gated-CNN combats both NBTI and HCI aging effects with an average reduction of the ‘0’ duty cycle, flip, and access patterns for all the cells by 71, 88, and 96%, respectively. This ensures average V_{th} degradation savings in all the cell transistors at least as high as 49% depending on the type of transistor.

The rest of this paper is organized as follows. Section 2 provides a background for this work. Section 3 describes the modeled framework. Section 4 introduces the aging characterization study. Section 5 presents the proposed Gated-CNN design. Section 6 refers to the experimental evaluation. Section 7 comments on related work, and finally, Section 8 summarizes this paper.

2. Background

This section discusses how NBTI and HCI affect the transistors used to implement SRAM memory cells. Then, it introduces the state-of-the-art aging-aware mechanism for CNN accelerators. The section concludes with a description of the power-gating technique as a mean to mitigate aging.

2.1. Aging Effects

NBTI and HCI are two of the main detrimental effects that gradually increase the transistor’s threshold voltage (V_{th}) over the lifetime of a circuit. Figure 2 depicts the implementation of a typical SRAM cell consisting of 6 transistors (6T cell). The four transistors labeled as either T_{Pi} or T_{Ni} form an inverter loop storing the logic value, whereas the remaining two T_{Wi} transistors act as pass transistors, controlled by the wordline (WL) signal, to access the cell contents through the bitline (BL) and its complementary ($\overline{\text{BL}}$).

The NBTI phenomenon mainly affects PMOS transistors when a logic ‘0’ is applied to their gates. In a 6T cell, this takes place in two ways. When the cell is under a ‘0’ duty cycle, that is, when the cell is stable and stores a logic ‘0’, the transistor T_{P0} is under stress and is affected by NBTI. On the contrary, under a ‘1’ duty cycle, the counterpart transistor T_{P1} suffers from NBTI. The degradation caused by each type of duty cycle is complementary, meaning that, for a given duty cycle, the PMOS transistor not under stress is partially under recovery from the NBTI effect. In other words, under a ‘0’ (‘1’) duty cycle, T_{P1} (T_{P0}) is under a partial recovery phase. Thus, if every bit cell of a memory array experiences a balanced duty cycle ratio (50% for each logic value), the wearout effect is evenly balanced between the two PMOS transistors and minimized compared to other cells with a higher duty cycle distribution. However, a balanced duty cycle ratio is unusual in a conventional design.

$$\begin{aligned} dV_{th_{NBTI}} = A_{NBTI} \times t_{ox} \times \sqrt{C_{ox} \times (V_{dd} - V_{t_0})} \times \\ (1 - \frac{V_{ds}}{\alpha_{NBTI} \times (V_{dd} - V_{t_0})}) \times e^{\frac{V_{dd}}{t_{ox} \times E_{NBTI}} - \frac{Ea}{k \times T}} \times \\ {t_{stress}}^{0.25} \times (1 - \sqrt{etha \times \frac{t_{rec}}{t_{stress} + t_{rec}}}) \end{aligned} \tag{1}$$

Equation 1 shows the standard formula to compute the V_{th} degradation (dV_{th}) of a PMOS transistor due to the NBTI phenomenon [17]. Parameters t_{stress} and t_{rec} denote the amount of time (in seconds) that the transistor is under stress and recovery modes, respectively. In a 6T cell, T_{P0} and T_{P1} accumulate t_{stress} time when the cell stores a logic '0' and '1', respectively. Refer to Section 6.1 for further details about the remaining parameters.

On the other hand, HCI mainly affects NMOS transistors when there is a logic value transition at their gates. In a 6T cell, T_{N0} and T_{N1} transistors are affected if the stored logic value flips as a consequence of a write operation. On the other hand, every cell access (read/write operation) induces HCI wearout to the T_{W0} and T_{W1} transistors. Note that the impact of HCI is proportional to the on/off switching frequency, and, contrary to NBTI, there is no recovery phase. Therefore, HCI is minimized when the number of accesses to a cell is reduced and writes do not change the stored value.

$$dV_{th_{HCI}} = A_{HCI} \times \alpha_{HCI} \times f \times e^{\frac{V_{dd} - V_{t_0}}{t_{ox} \times E_{HCI}}} \times \sqrt{t} \tag{2}$$

The contribution of the HCI effect to the dV_{th} of an NMOS transistor is computed using the standard Equation 2 [18]. Parameter t refers to the amount of time (in seconds) that the gate transitions from '0' to '1' and vice versa. In a 6T cell, transistors T_{N0} and T_{N1} accumulate t time when the stored value flips, whereas transistors T_{W0} and T_{W1} do the same when the wordline is driven from high to low and vice versa. See Section 6.1 for further details about the remaining parameters.

2.2. *State-of-the-Art Aging-Aware Mechanism for CNN Accelerators: DNN-Life*

The state-of-the-art DNN-Life technique has faced the NBTI effect in the on-chip SRAM buffer of CNN accelerators storing weights of neural networks [16]. DNN-Life proposes to periodically invert the weights buffer contents to balance the duty cycle distribution. In particular, the technique encodes weights in such a way that bits to be written in this buffer are randomized. After a read operation, weights are decoded back to the original value before feeding the processing elements. However, by periodically bit-flipping the cell contents, DNN-Life exacerbates the HCI effect.

2.3. *Power-Gating Opportunity*

Power-gating is a well known technique to drastically reduce the power consumption of memory structures [27]. This technique can be also leveraged to mitigate aging effects.

An aging-aware power-gating configuration consists of an NMOS high-V_{th} sleep transistor connecting the 6T cell to ground. In this way, the cell ground terminal is connected to a virtual ground. When the sleep transistor drives current (active state), the cell operates as usual, yet with a ground voltage equal to the virtual ground. On the contrary, when the sleep transistor is off (*switch-off* state), the cell is disconnected from the ground and both T_{P0} and T_{P1} transistors remain partially under recovery from NBTI at the same time, since both cell nodes hold a logic '1' [28]. Of course, as long as the cell remains off, not only high duty cycle distributions are reduced to combat NBTI, but also cell contents are neither accessed nor flipped, preventing HCI wearout. Notice too that, contrary to NMOS cell transistors, the NMOS sleep transistor is resilient to HCI since it is implemented using a high-V_{th} device [29, 30].

3. Framework Overview

This section introduces the framework for our proposed aging-aware mechanism, which consists of a description of the subset of studied CNN benchmarks, and an overview of the CNN inference accelerator model used in this work as a baseline.

3.1. *Benchmarks*

We selected a variety of widely used CNNs focusing on classification and regression tasks. Table 1 summarizes the main characteristics of these benchmarks. We used colorectal histology [31] and ImageNet [32] as datasets for the image classification tasks, whereas IMDB reviews and Udacity's self-driving car simulator have been used as datasets for the sentimental classification and regression tasks, respectively.

The number of layers of the studied benchmarks largely differ among each other, from the narrowest CNN consisting of 4 layers (SentimentalNet) to the deepest CNN including 126 layers (DenseNet). Benchmarks also present disparate memory storage requirements. The smallest activation layer of every CNN consists of a few tens of KB at most and refers to either a pooling or a later convolutional layer (shown in parentheses). On the other hand, for all the CNNs except SentimentalNet, the size of the largest activation layer is in the order of MB and corresponds to the first convolutional layer. Overall, the average layer size ranges from 15 KB (SentimentalNet) to 1.32 MB (VGG16).

Finally, this work assumes 16-bit fixed-point words to represent both activations and weights, which is a common choice for most inference accelerators [1, 2, 4, 5, 6, 9]. The rightmost column in the table shows the required number of integer and fraction activation bits for each benchmark to avoid accuracy losses with respect to the top-1 accuracy assuming a 32-bit floating-point (IEEE-754) data type. Remark that the bit overprovisioning with 16-bit words is extended to the fraction part. Otherwise, devoting more than necessary bits for the integer part would translate into integer most significant bits with a 100% '0' duty cycle, thereby exacerbating NBTI in those bit cells. Notice too that, similarly to previous works [3, 8, 10], this implies our modeled baseline accelerator to dynamically adjust the number of integer and fraction bits required by each benchmark.

Table 1: Overview of the studied CNN benchmarks. Labels *Conv*, *FC*, and *DWConv* stand for convolution, fully-connected, and depth-wise convolution layers, respectively. *DWConv* layers process a different weight filter on each channel of an input image.

Benchmark	Task	Depth	Smallest layer size	Largest layer size	Average layer size	Activation representation
AlexNet [19]	Image classification	5×Conv, 3×FC, 3×MaxPooling	18 KB (*MaxPooling3*)	0.58 MB (*Conv 1*)	153 KB	4 integer bits & 4 fraction bits
ZFNet [20]	Image classification	5×Conv, 3×FC, 3×MaxPooling	18 KB (*MaxPooling3*)	2.28 MB (*Conv 1*)	324 KB	4 integer bits & 6 fraction bits
VGG16 [21]	Image classification	13×Conv, 3×FC, 5×MaxPooling	49 KB (*MaxPooling 5*)	6.4 MB (*Conv 1*)	1.32 MB	3 integer bits & 8 fraction bits
SqueezeNet [22]	Image classification	26×Conv, 3×MaxPooling, 1×GblAvgPooling	3 KB (*Conv 26*)	2.4 MB (*Conv 1*)	431 KB	6 integer bits & 4 fraction bits
MobileNet [23]	Image classification	15×Conv, 13×DWConv, 1×GblAvgPooling	49 KB (*DWConv 12*)	1.55 MB (*DWConv 1*)	340 KB	4 integer bits & 9 fraction bits
DenseNet [24]	Image classification	120×Conv, 1×MaxPooling, 3×AvgPooling, 1×FC, 1×GblAvgPooling	12 KB (*Conv 120*)	1.6 MB (*Conv 1*)	254 KB	3 integer bits & 5 fraction bits
SentimentalNet [25]	Sentimental classification	1×Conv, 1×MaxPooling, 2×FC	15 KB (*MaxPooling 1*)	30 KB (*Conv 1*)	15 KB	0 integer bits & 5 fraction bits
PilotNet [26]	Turning angle regression	5×Conv, 5×FC	2 KB (*Conv 5*)	0.13 MB (*Conv 1*)	26 KB	0 integer bits & 7 fraction bits

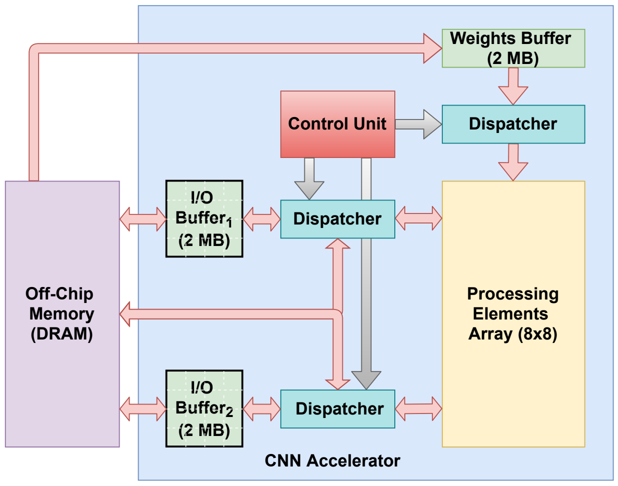


Figure 3: Overview of the baseline CNN accelerator.

3.2. 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 to speed up the inference process in CNNs, such as DaDianNao [2], Google's TPU [5], Eyeriss [6], and Bit-Tactical [9]. Figure 3 depicts the hardware organization of the baseline accelerator consisting of an 8×8 Processing Element (PE) array, a couple of 2 MB Input/Output (I/O) buffers for activation storage, a 2 MB weight buffer, dispatchers for every buffer, and a control unit. The computational and storage resources have been sized according to the domain of embedded systems [33].

The PE array is a systolic array processor conformed by 64 PEs interconnected through a two dimensional mesh. Each PE independently computes 16-bit fixed-point dot-products through partial sums with an input activation from one I/O buffer, acting as input buffer, and a weight from the weight buffer. The dataflow in the PE array corresponds to the output stationary approach described in Eyeriss [6] and SCALE-Sim [34].

The memory buffers provide intermediate storage for both activations and weights to reduce costly off-chip memory accesses. Since faulty weights have a much less impact on the network accuracy than faulty activations (see Figure 1), this work focuses on aging mitigation in the I/O buffers.

In the same way as EIE [4] and Alcolea *et al.* [11] inference accelerators, the I/O buffers swap their roles on every inference step. We define an inference step as the processing of the output activations of a neural network layer given a set of input activations from the previous layer and weights. In this way, a given I/O buffer stores even layers and the counterpart buffer stores odd layers. On the contrary, the weight buffer caches weights to be issued in the proper order by the dispatcher to the PE array. Subsequent inference steps replace old weights with those required by the current step.

The relatively small I/O buffer size implies to spill the activations to off-chip memory when a layer does not fit in 2 MB. According to Table 1, this issue only affects ZFNet, VGG16, and SqueezeNet. More precisely, a single layer of ZFNet and SqueezeNet exceed the I/O buffer size, whereas 4 out of 21 layers from VGG16 exceed this size.

Unlike CPU and GPU caches arranged in sets and ways, the I/O buffers are arranged as scratchpad memories split into banks and sub-banks to provide enough bandwidth to the parallel processing in the PE array. In particular, each buffer consists of eight 256 KB banks. In turn, each bank is composed of eight 32 KB sub-banks. Activations are sub-bank interleaved and sequentially arranged bank after bank [3, 12]. This implies that the first 256 KB activations of a layer are always stored in $bank_0$. Notice too that the three most significant bits of an I/O buffer address denote the bank where the requested activation is to be found.

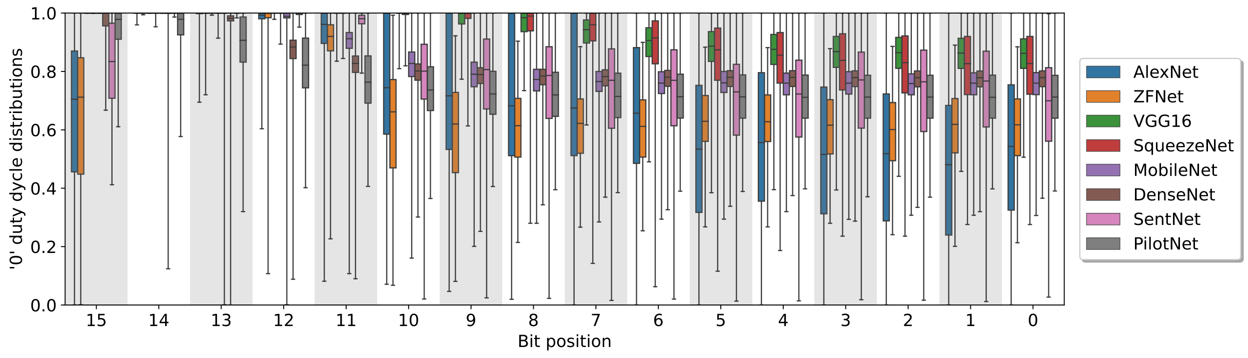


Figure 4: ‘0’ duty cycle distributions on each bit position of the 16-bit active words.

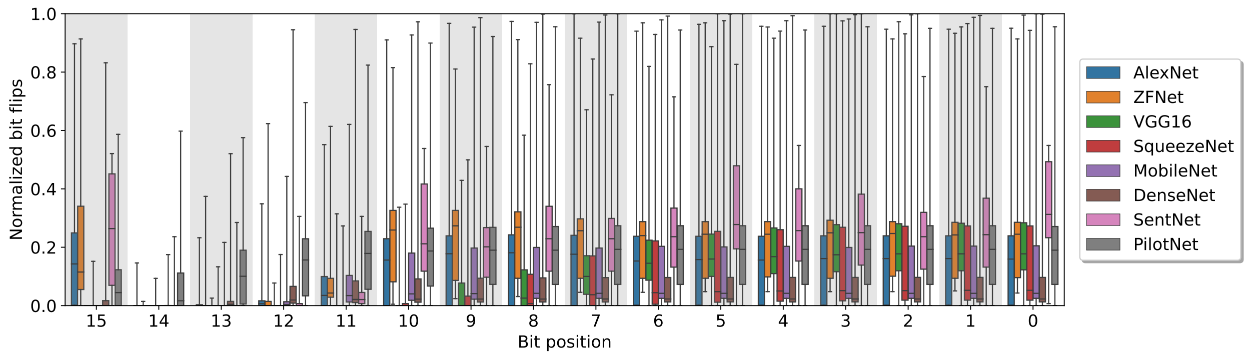


Figure 5: Normalized bit flips on each bit position of the 16-bit active words with respect to the worst-case bit cell with the highest flip peak.

Finally, all the dispatchers are capable to transmit up to eight 16-bit activations or weights per cycle. Dispatchers are driven by the control unit, which exploits control information of the currently computed layer.

4. Characterization Study

Characterizing the duty cycle, bit flip, and access patterns provides insights on the impact of both NBTI and HCI aging effects in the cell transistors of the I/O buffers. The presented results are restricted to one of the two I/O buffers. However, they are very similar for the two buffers. This study refers to the baseline approach, where neither NBTI nor HCI mitigation mechanisms are employed.

4.1. Duty Cycle Distribution

Figure 4 depicts the ‘0’ duty cycle distributions (i.e., percentage of time storing a logic ‘0’ over the total execution time) experienced in every 16-bit activation word, represented in little-endian. The complementary distributions refer to the ‘1’ duty cycle. The distributions include the ‘0’ duty cycles for every active cell storing useful bits. That is, the figure does not include the effect of duty cycles (either 100% ‘0’ or ‘1’ duty cycles) causing the maximum NBTI degradation in idle memory cells[2].

[2]For every box-and-whisker distribution plotted in this work, top and bottom box edges specify the 75th and 25th percentiles, lines within the boxes represent the median, and whiskers denote the maximum and minimum values.

Apart from AlexNet and ZFNet, a biased ‘0’ duty cycle can be seen in the sign bit (15th bit) for every benchmark. In fact, for VGG16, SqueezeNet, and MobileNet, no negative activations are written in the I/O buffers, exacerbating NBTI in the sign bit. Recall that the integer part varies in number of bits depending on the benchmark, occupying until the 9th bit at most in SqueezeNet (see the rightmost column of Table 1). The ‘0’ duty cycle is also biased in most of these bits, and particularly the most significant ones, highlighting that the majority of activations are close to zero. On the other hand, a high ‘1’ duty cycle in a few bits can be seen in benchmarks like MobileNet and DenseNet.

In contrast to sign and integer bits, fraction bits show a broaden duty cycle distribution, pushing the median toward the ideal 50% duty cycle. However, in benchmarks like VGG16 and SqueezeNet, the median ‘0’ duty cycle exceeds 90% for all the bit positions. Moreover, long ‘0’ and ‘1’ duty cycles can be seen in many bit positions for applications such as AlexNet, SqueezeNet, and SentimentalNet.

Overall, NBTI wearout, from either ‘0’ or ‘1’ duty cycles, is appreciated not only in the sign and integer bit cells, but also in many cells storing fraction bits. That is, prior aging-aware techniques for CPU architectures working at a word granularity like [35, 36, 37] would be ineffective in CNN accelerators.

4.2. Flip and Access Distributions

Figure 5 illustrates the normalized number of flips on every active bit cell with respect to the worst-case cell accumulating the highest flip peak for each benchmark. As expected from

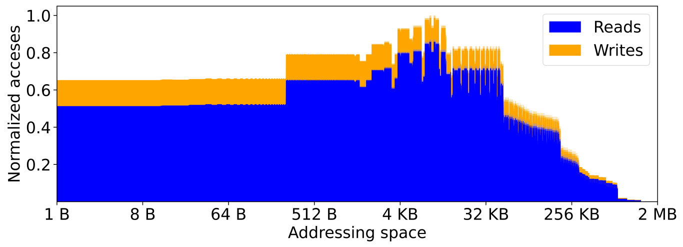


Figure 6: Normalized number of accesses (reads and writes) of DenseNet across the entire addressing space with respect to the memory address with the highest number of accesses.

the previous analysis, the sign and integer bits, where the duty cycle is generally highly biased, show a lower number of flips compared to the fraction bits, where the duty cycle is generally less biased and a higher amount of flips are observed.

The flip pattern shows that few cells suffer from a high number of flips close to the highest flip peak, since all the boxes stay below 50% of the flip distribution. However, those cells are spread across all the fraction bits for all the studied benchmarks.

Note that unpredictable CNN workload input values may cause different duty cycle and flip patterns. However, since the proposed approach combats wearout in all the cell transistors (see the next section), unpredictable workloads would benefit from the proposed technique in the same way as predictable workloads.

Figure 6 plots the normalized number of read/write accesses to every byte of an I/O buffer with respect to the byte with the highest access peak. Results are restricted to DenseNet. The remaining CNNs show similar access patterns. X axis is shown in logarithmic scale for illustrative purposes.

Most accesses are skewed toward low-order words, since every layer of a network is stored from address 0x0 onward. In addition, considering all the studied benchmarks, the average number of read operations is 201× higher than writes due to the activation reuse exhibited in convolutions. Therefore, T_{Wi} pass transistors are far more exposed to HCI degradation than T_{Ni} inverter loop transistors.

5. Proposed Approach: Gated-CNN

This section introduces the proposed Gated-CNN design. First, we show a general overview of the approach, which consists of four main modules. Then, these modules are described in detail. Finally, timing, power, energy, and area overheads are also discussed.

5.1. General Overview

Based on the previous characterization study, Gated-CNN is aimed at minimizing both NBTI and HCI aging effects in every cell transistor of the I/O buffers of CNN accelerators. To do so, the key idea of Gated-CNN is to combine a bank address rotation scheme with a bank power-gating mechanism to spread out not only flip and access patterns but also switch-off cycles to balance duty cycle distributions across all the banks.

Gated-CNN leverages the following properties of CNN applications and specific-domain accelerators:

- The memory size of activation layers largely differ not only among different CNN applications but also among layers of a given CNN (see Table 1), leading to a dynamic under-utilization of the I/O buffers.
- CNN applications expose both temporal and spatial localities in a predictable manner. Once the activations of a layer have been read in order to compute the activations of the next layer, the former activations are not reused anymore.
- CNN accelerators include two I/O buffers that alternatively exchange input and output roles forcing a given buffer to store even or odd layers (see Section 3.2).

Consequently, the dynamic under-utilization is exploited to propose a cyclic storage of activation layers in successive banks, ensuring an homogeneous bank usage. Moreover, after using an activation layer to compute the next one, those banks storing the former stale layer are powered off. Notice too that exchanging buffer roles maximizes the period of time in which these banks are powered off.

The Gated-CNN component is coupled to one I/O buffer. That is, two Gated-CNN components are required for the baseline CNN accelerator. In addition, the proposed approach is designed for an I/O buffer with 8 banks. However, it could be redesigned to support a different number of banks with minor changes.

Figure 7 depicts the architecture of the four main modules of Gated-CNN. The *effective bank* module calculates the effective bank id where a requested activation is to be found in the I/O buffer. In addition, it forwards the starting bank id of the currently stored layer to the *inter-bank rotation* module. This module identifies the starting and ending banks required by the next inference step (next layer to be stored in the buffer), and forwards this information to the *power-gating bitmap generator* module. According to such banks, the power-gating bitmap generator computes a bitmap array that states which banks should be powered on/off in the next inference step. This bitmap is forwarded to the *bitmap update* module. This module updates the on/off state of the bank power-gating sleep transistors at runtime, masking the bank wake-up latency penalty.

5.2. *Effective Bank Module*

To request a given activation to the I/O buffer, the control logic of the accelerator refers to a logical address computed as an offset (in bytes) from the first activation of the currently stored layer in the buffer (*i*-th layer). Such a first activation is stored at address 0x0 (physical base $bank_0$) by default. The input *reqBnkLyr_i* refers to the three most significant bits of the logical address, identifying the logical bank storing the requested activation (see Section 3.2).

The effective bank module translates from a logical to an effective physical bank. To do so, the 3-bit register labeled as *sBnk* contains the physical base bank id (from $bank_0$ to $bank_7$) where

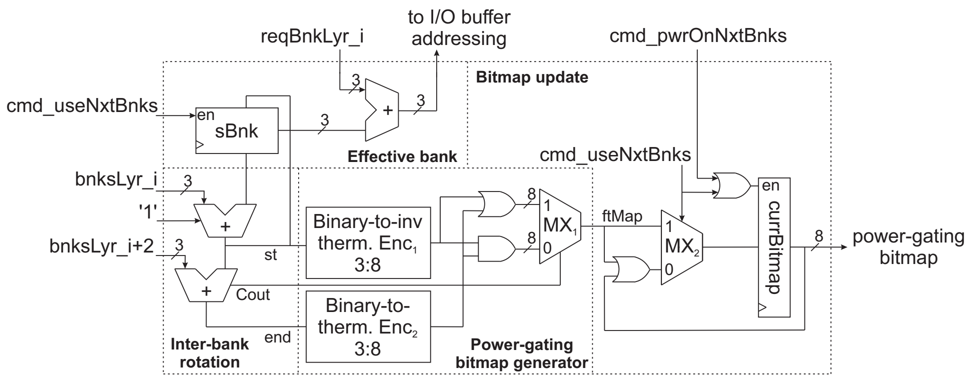


Figure 7: Proposed Gated-CNN design consisting of four main modules: effective bank, inter-bank rotation, power-gating bitmap generation, and bitmap update.

Cycle	t0	t1	t2 = t1 + 10	t3 = t2 + 1
Description	Access i-th layer	Access i-th layer and wake up banks for i+2-th layer	Access i-th layer and set bitmap for i+2-th layer	Access i+2-th layer

Inputs, registers, and intermediate signals	sBnk = 0
	bnksLyr_i = 2
	st = 3
	outEnc$_1$ = '11111000'
	bnksLyr_i+2 = 1
	end = 4
	outEnc$_2$ = '00011111'
	Cout = '0'
	ftMap = '00011000'
	(outEnc$_1$ & outEnc$_2$)
	currBitmap = '00000111'
		currBitmap = '00011111'
		(ftMap | currBitmap)
			sBnk = 3
			currBitmap = '00011000'
			(ftMap)
				sBnk = 3
				bnksLyr_i = 1
				st = 5
				outEnc$_1$ = '11100000'
				bnksLyr_i+2 = 3
				end = 0
				outEnc$_2$ = '00000001'
				Cout = '1'
				ftMap = '11100001'
				(outEnc$_1$ | outEnc$_2$)
				currBitmap = '00011000'

Commands	cmd_pwrOnNxtBnks = '0'
	cmd_useNxtBnks = '0'
		cmd_pwrOnNxtBnks = '1'
		cmd_useNxtBnks = '0'
			cmd_pwrOnNxtBnks = '0'
			cmd_useNxtBnks = '1'
				cmd_pwrOnNxtBnks = '0'
				cmd_useNxtBnks = '0'

Figure 8: Driving example of the CNN-Gated component, including the state of the inputs, registers, and intermediate signals at specific cycles of the inference process. Signals not appearing in a given cycle means that the state is preserved with respect to the previous cycle on the left.

the first activation of the *i*-th layer can be found in the buffer. This register is implemented with resilient 8T cells to address aging at the cost of a slight area increase [38, 39]. The adder outputs the current physical bank of the requested activation with the addition of the *sBnk* and *reqBnkLyr_i* bits.

Note that a layer might wrap around the I/O buffer (adder overflow or carry out bit set to '1'). That is, a layer occupying high-order and low-order banks, leaving free banks in the middle. Notice too that the adder output is concatenated with the 18 least significant bits of the logical address to properly index the 2 MB buffer with a 21-bit address (not shown in the figure).

Finally, the write enable (*en*) of the *sBnk* register is driven by the *cmd_useNxtBnks* signal. This signal is set when the physical base bank changes as a consequence of the next inference step storing activations of the new incoming layer. The subsequent section describes how the *sBnk* contents are updated.

5.3. *Inter-Bank Rotation Module*

The inter-bank rotation module cyclically changes the value in *sBnk* in a round-robin fashion throughout the banks. The next incoming layer, that is, the (*i*+2)-th layer[3], starts occupying the successive available bank with respect to the last bank used by the previous *i*-th layer. To do so, the upper adder outputs that successive bank id (*st* bits) with the addition of *sBnk*, the *bnksLyr_i* input, and the carry in bit set to logic '1'. The *bnksLyr_i* input stands for the number of banks used by the *i*-th layer, which is computed as *i-th layer size % bank size*. The *i-th layer size* parameter is derived from the software profiling of the CNN. Note that *bnksLyr_i* ranges from 0 (a single bank) to 7 (all the banks). The *st* bits will be stored in the *sBnk* register when *cmd_useNxtBnks* ='1' to properly compute an effective bank of the (*i*+2)-th layer.

The bottom adder outputs the last bank that will occupy the (*i*+2)-th layer (*end* bits). In contrast to the upper adder, this adder includes a carry out (*Cout*) bit, which flags a bank-wrapping situation for the (*i*+2)-th layer. The *st*, *end*, and *Cout* bits act as inputs for the power-gating bitmap generation module described in the next section.

5.4. *Power-Gating Bitmap Generation Module*

The power-gating bitmap generator module dynamically distinguishes between active and idle banks of the incoming (*i*+2)-th layer on every inference step. To do so, two binary-to-thermometer 3:8 encoders are required to generate a power-gating 8-bit map.

To help understand how this module works, Figure 8 shows a driving example including the state of the inputs, registers, and intermediate signals at specific cycles. At cycle *t0*, the *i*-th layer occupies $bank_0$, $bank_1$, and $bank_2$ (*sBnk* = 0 and *bnksLyr_i* = 2), whereas the (*i*+2)-th layer will occupy $bank_3$ and $bank_4$ (*bnksLyr_i*+2 = 1). The upper inverse encoder, referred to as

[3]Remember that the I/O buffers exchange roles in every inference step, meaning that one buffer stores even layers and the other odd layers (see Section 3.2).

Enc_1, is fed with the *st* bits and outputs an 8-bit map in which bits are set to '0' except those from the *st*-th bit up to the most significant bit. In the example, $st = 3$ generates the output $outEnc_1$ = '11111000'.

On the other hand, the bottom encoder, namely Enc_2, is fed with the *end* bits. In contrast to Enc_1, all the output bits of this encoder are set to '0' except those from the *end*-th bit down to the least significant bit. In the example, $end = 4$ generates the output $outEnc_2$ = '00011111'.

In order to generate a future power-gating bitmap for the (i+2)-th layer (*ftMap*), both encoding outputs are merged as follows. When *Cout* = '0', i.e., when $st \leq end$, the encoder outputs are bit-wise ANDed. In the example, $outMX_1$ = '00011000', meaning that $bank_3$ and $bank_4$ will remain powered on and the rest powered off in the next inference step at cycle *t3*. Otherwise, $st > end$, the required banks wrap around the I/O buffer. In such a case, the encoder outputs are bit-wise ORed. This is the case of the example at cycle *t3*, where *Cout* = '1' ($st = 5 > end = 0$). Finally, note that *Cout* determines the appropriate bit-wise operation by driving the MX_1 multiplexer.

5.5. Bitmap Update Module

The purpose of the bitmap update module is to establish the appropriate bitmap to the power-gating mechanism cycle by cycle. This module takes as inputs *ftMap* from the previous module, and two command signals, *cmd_pwrOnNxtBnks* and *cmd_useNxtBnks*, specifying, respectively, when the banks required by the (i+2)-th layer should be powered on and when the next inference step starts.

The 8-bit *currBitmap* register, implemented with 8T cells, stores the current power-gating bitmap applied to the banks, that is, it refers to the i-th layer. Using the example in Figure 8, this register stores the bitmap '00000111' at cycle *t0*. At cycle *t1*, those banks required by the (i+2)-th layer start to be woken up in advance. To do so, *cmd_pwrOnNxtBnks* and *cmd_useNxtBnks* are set to '1' and '0', respectively. This allows *ftMap* and *currBitmap* to be bitwise ORed, resulting in a new *currBitmap* = '00011111' (MX_1 selection bit = '0' and en = '1'), specifying not only the required banks of the i-th layer but also those of the (i+2)-th layer set to ON. Like previous works [40, 41], we assume the wake up latency of a bank to be 10 cycles. That is, the requested banks are powered on 10 cycles ahead of starting the next inference step, which ensures the wake-up latency to be out of the critical path.

At cycle *t2*, *cmd_pwrOnNxtBnks* and *cmd_useNxtBnks* are set to '0' and '1', respectively, establishing the proper bitmap for the next inference step (*currBitmap* = *ftMap* = '00011000'). That is, at this cycle, those banks no longer required are powered off. Notice too that, at the same time, the *sBnk* contents are updated accordingly. Finally, at cycle *t3*, a new *ftMap* is obtained according to the size of the subsequent (i+2)-th layer.

5.6. Timing, Energy, Power, and Area Estimations

To measure timing, energy, power, and area numbers, the proposed Gated-CNN design has been synthesized with Synopsys Design Compiler and simulated with Mentor Graphics Modelsim. The technology library corresponds to a low-power 32 nm technology available to European universities. Table 2 summarizes the results. For comparison purposes, results include the estimations for an I/O buffer and the overheads of Gated-CNN with respect to the buffer. The I/O buffer has been modeled with the CACTI-P simulation framework [42].

Table 2: Timing, energy, power, and area values for the proposed Gated-CNN approach and an I/O buffer using a 32 nm technology node.

	Gated-CNN	I/O buffer	Overhead (%)
Access time (ns)	0.19	1.88	10.11
Dyn. energy (pJ)	0.29	162.19	0.18
Leak. power (mW)	1.39	178.14	0.78
Area (mm^2)	0.003	3.378	0.09

The access time of Gated-CNN refers to its longest path delay from *sBnk* to *currBitmap* registers, which is out of the critical path. The dynamic energy of Gated-CNN refers to obtaining a new power-gating bitmap given a set of inputs, whereas these expenses for the I/O buffer are related to accessing the memory. Compared to the numbers of the I/O buffer, the overheads of Gated-CNN are minimal. In addition, both energy and power overheads are largely compensated with the benefits brought by power gating idle buffer banks.

Finally, CACTI-P has been also used to model the bank power-gating in the I/O buffers, including all the sleep transistors and interconnects. The area overhead of the power-gating technique is by 1.39% with respect to the I/O buffer, which is a similar overhead as those reported in other power-gating designs from the industry [43, 44].

6. Experimental Evaluation

This section introduces the simulation framework and aging model used to obtain the experimental results. Then, these results are quantified and discussed, including the duty cycle, bit flips, and accesses for the worst-case memory cell of the I/O buffers under both the baseline and the proposed Gated-CNN approach. Finally, a distribution of the V_{th} degradation for every transistor is also analyzed.

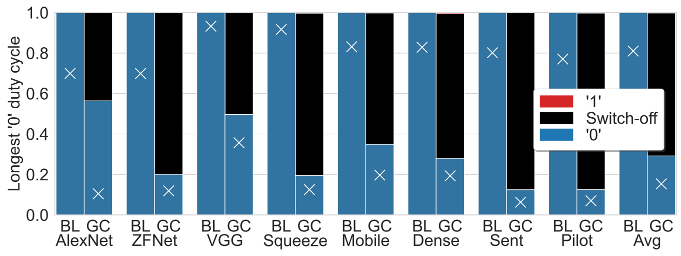
6.1. Evaluation Setup

We have extended the TensorFlow 2.5.0 simulation framework [45] to model the dataflow of the baseline CNN accelerator and on-chip memories, including the proposed Gated-CNN design, and collect processor statistics required to estimate the duty cycle, bit flip, and access distributions. The dataflow modeling establishes a cycle-accurate simulation, where the latency of the on-chip memories and Gated-CNN approach is obtained with CACTI-P and Synopsys Design Compiler, respectively (see Section 5.6). The latency of a partial sum and accumulation in a PE is assumed to be one cycle.

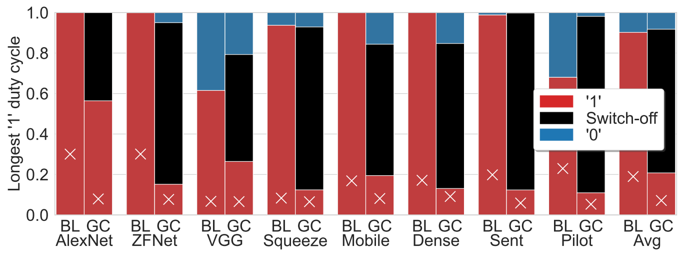
The aging model presented in Section 2.1 is integrated in TensorFlow to quantify the V_{th} degradation in every transistor according to the experienced t_{stress} and t_{rec} times (duty cycle

Table 3: Main parameters of the NBTI and HCI aging model for a 32 nm technology node. Values of all these parameters and remaining ones from Equations 1 and 2 can be found in [17, 18, 46].

Parameter	Description	Value
t_{ox}	Oxide thickness	1.65 nm
C_{ox}	Gate capacitance per unit area	4.6×10^{-20} F/nm^2
V_{dd}	Supply voltage	0.9 V
V_{t_0}	Initial threshold voltage	0.2 V
V_{ds}	Drain-source voltage	0.7 V
E_{NBTI}	Technological constant	0.2 V/nm
Ea	Activation energy	0.13 eV
k	Boltzmann constant	8.6174×10^{-5} eV/K
T	Temperature	353.15 K
α_{HCI}	Activity factor	1
f	Clock frequency	1 GHz
E_{HCI}	Technological constant	0.8 V/nm



(a) '0' duty cycle



(b) '1' duty cycle

Figure 9: Longest duty cycle distributions across all the studied benchmarks. The cross symbol represents the average duty cycles for all the cells. BL and GC refer to the baseline and Gated-CNN designs.

patterns) and t time (bit flip/access patterns) induced by each CNN application. Table 3 shows a description and value of the main parameters of the aging model for a 32 nm technology node[4].

Similarly to [16], experimental results have been obtained after the inference of 150 images for every benchmark. Such a number of inferences is enough to stabilize all the studied aging patterns.

6.2. Duty Cycle Analysis

Figure 9a plots, for all the studied benchmarks, the '0' duty cycle distribution for the worst-case cell, that is, the longest '0' duty cycle. Gated-CNN incorporates a *switch-off* state to reflect the amount of time that the worst-case cell is powered off. The cross symbol in every bar refers to the average '0' duty cycle for all the cells. BL and GC stand for baseline and Gated-CNN. Like the previous characterization study in Section 4, we conservatively assume that the baseline results only refer to active cells storing useful contents. On the contrary, Gated-CNN uses all the cells, and consequently all of them are included in the results.

As expected, the maximum '0' duty cycle distribution for the baseline approach is 100% in every benchmark. In addition, the average '0' duty cycle is also highly biased, as mentioned above. On the other hand, by powering off idle banks and uniformly distributing switch-off cycles across them, Gated-CNN largely reduces the longest '0' duty cycle beyond 50% in all the benchmarks apart from AlexNet.

The differences among benchmarks largely depend on the power-off opportunities provided by the CNN layer dimensions. In this sense, the highest '0' duty cycle reductions are seen in SentimentalNet and PilotNet due to these benchmarks have the lowest average layer size requirements (15 KB and 26 KB, respectively, see Table 1). Another factor that contributes to obtain large reductions is the fact that some layers do not fit in the I/O buffer and they are spilled to off-chip memory. Meanwhile, all the I/O buffer banks are powered off. Such a number of switch-off cycles mainly depends on the required processing time of the layer. The high '0' duty cycle reductions observed in ZFNet and SqueezeNet are mainly due to both a relatively small average layer size and a larger than 2 MB layer.

On the other hand, AlexNet does not reduce the '0' duty cycle as much as expected from the workload characteristics. This is due to the combination of the sizes of both banks and layers leading to some banks being more exercised than others (less switch-off cycles than expected), since these banks recurrently store compute-intensive layers. Nevertheless, the obtained reduction for the worst-case cell is as high as 44% in this CNN, whereas the average duty cycle reduction reaches a 90%.

Overall, Gated-CNN reduces the average highest '0' duty cycle by 71%. Taking into account all the cells, the average '0' duty cycle reduction is by 85%.

The counterpart, longest '1' duty cycle distribution, should be also considered for a complete analysis of the NBTI effect. Figure 9b shows the results. Unlike the worst-case cells of the baseline with a 100% '0' duty cycle (Figure 9a), the longest '1' duty cycle distribution for the baseline does not reach 100% in all the benchmarks. In addition, the average '1' duty cycle greatly reduces in the baseline scheme. These observations confirm the presence of more zeros than ones in the activations. For the Gated-CNN design, the percentage of the switch-off state in the distribution is the same as in the previous analysis. Like the baseline, the maximum '1' duty cycle is further reduced with the presence of zeros. This can be clearly seen in VGG16, MobileNet, and DenseNet.

In summary, Gated-CNN reduces the average longest '1' duty cycle by 79%. Taking into account all the cells, the average '1' duty cycle reduction is by 93%. Comparing the results of both duty cycle distributions, transistors T_{P0} are more exposed

[4]Interested readers may refer to the following URL containing a repository of the extended TensorFlow simulation framework with the Gated-CNN approach and aging model: `https://github.com/NicolasLanderos/Gated-CNN`.

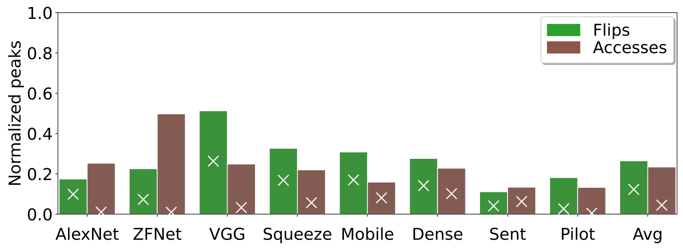


Figure 10: Normalized highest bit flip and access peaks of Gated-CNN with respect to the baseline design. The cross symbol refers to the normalized average bit flips and accesses for all the cells.

to the NBTI effect than transistors T_{P1} (see Section 2.1).

6.3. Bit Flip and Access Analysis

Figure 10 shows the normalized highest number of bit flips and accesses of the worst-case cells in the Gated-CNN design with respect to those cells of the baseline. These results only refer to active cells storing useful bits in both approaches. Otherwise, the normalized average numbers (cross symbols) would be skewed toward zero.

By rotating data across banks, Gated-CNN substantially reduces both the maximum flip and access peaks compared to the baseline. The largest peak reduction is seen in SentimentalNet. This is mostly due to every layer fits in a single bank, preventing flips in the remaining banks. The other benchmarks obtain higher peaks mostly due to larger layers occupying multiple banks. However, both the highest flip and access peaks are reduced at least as much as 49% (VGG16 and ZFNet). In addition, the average peaks do not exceed a 27% (VGG16) in any benchmark.

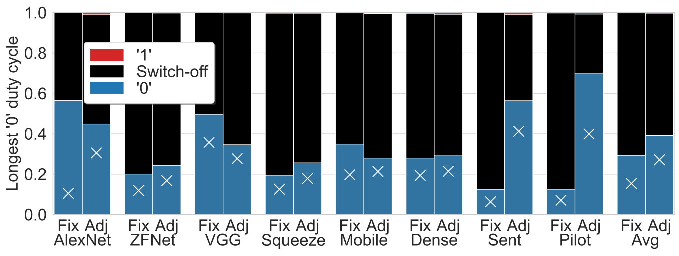
Overall, the highest flip and access peaks are reduced on average by 74%. Taking into account all the cells, the average reductions are as much as 88 and 96% for flips and accesses, respectively.

6.4. Sensitivity Study to the I/O Buffer Size

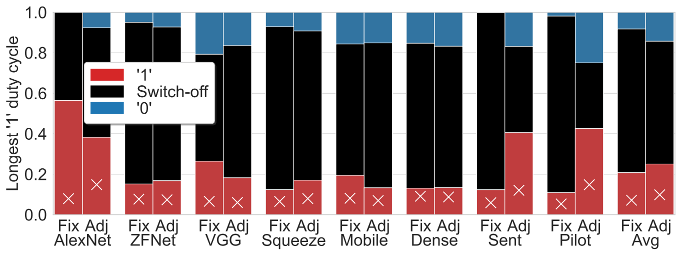
The power-off and rotation capabilities of Gated-CNN not only depend on the workload characteristics but also on the size of the I/O buffer. In this work, we have assumed a generic accelerator with a total on-chip activation storage of 4 MB (2 MB × 2 I/O buffers) capable to speed up the inference of diverse CNNs with disparate activation storage requirements. Such an overall activation storage coincides with the assumed activation storage for the DNN-Life's baseline accelerator [16], and it is much less than the storage capacity of other inference accelerators like the Google's TPU with 24 MB for activations [5].

This section evaluates the sensitivity of Gated-CNN to the I/O buffer size in terms of both the power-off and rotation capabilities. More precisely, the size of the I/O buffer is adjusted to the largest layer size of each CNN application to quantify the Gated-CNN benefits only from the workload characteristics (see Table 1). Like the proposed original approach, a number of eight same-sized banks is assumed regardless of the buffer size.

Plots in Figure 11 illustrate the previous duty cycle distributions (Fix) shown in Figure 9 and the new distributions adjusting the I/O buffers to the largest layer size (Adj). Results only refer



(a) '0' duty cycle



(b) '1' duty cycle

Figure 11: Longest duty cycle distributions for Gated-CNN. Results differentiate between a fixed 2 MB I/O buffer size (Fix) and a buffer size adjusted to the largest layer (Adj).

to the Gated-CNN design. For an adjusted size of the I/O buffer, the opportunity to switch-off banks significantly decreases in benchmarks with less than a 2 MB buffer size like SentimentalNet and PilotNet, but the longest duty cycle reductions are still at least as high as 44%. On the contrary, other benchmarks also with a smaller buffer size, like AlexNet and MobileNet, further reduce the longest duty cycle with respect to the previous results. This is mainly due to layers occupying a greater number of banks, which in turn changes the round-robin bank assignment to every layer, preventing compute-intensive layers to always remain in the same banks.

On the other hand, CNNs with more than a 2 MB buffer size increase the longest duty cycle due to huge layers are not spilled to off-chip memory anymore, reducing the number of switch-off cycles. This is the case of ZFNet and SqueezeNet. For VGG16, the reason for a duty cycle reduction is the same as explained above for AlexNet and MobileNet.

Notice too that, similarly to the longest duty cycle, the limitations of Adj in number of switch-off cycles due to either a buffer size reduction or spilling prevention translate into a higher average duty cycle over Fix in most benchmarks.

To sum up, despite adjusting the I/O buffer size to the largest layer storage requirements, the average longest '0' and '1' duty cycle reductions are as much as 63 and 76%, respectively.

Figure 12 depicts the normalized highest flip and access peaks for fixed (Figure 10) and adjusted buffer sizes. Small CNNs like SentimentalNet and PilotNet substantially increase the normalized peaks, especially flip peaks. This is mainly due to, with an adjusted buffer, layers of small CNNs occupy a larger number of banks, increasing the likelihood of a higher number of flips/accesses in a given cell. In spite of this, peak reductions are at least by 50% with respect to the baseline scheme.

The remaining CNNs show minor peak increases due to

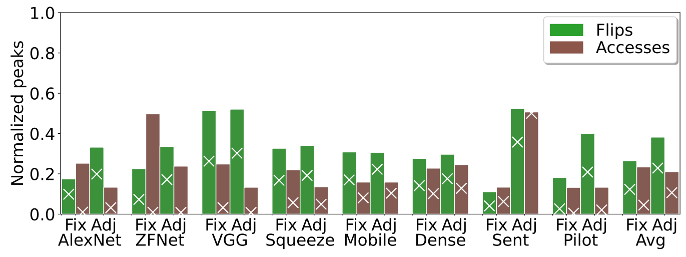
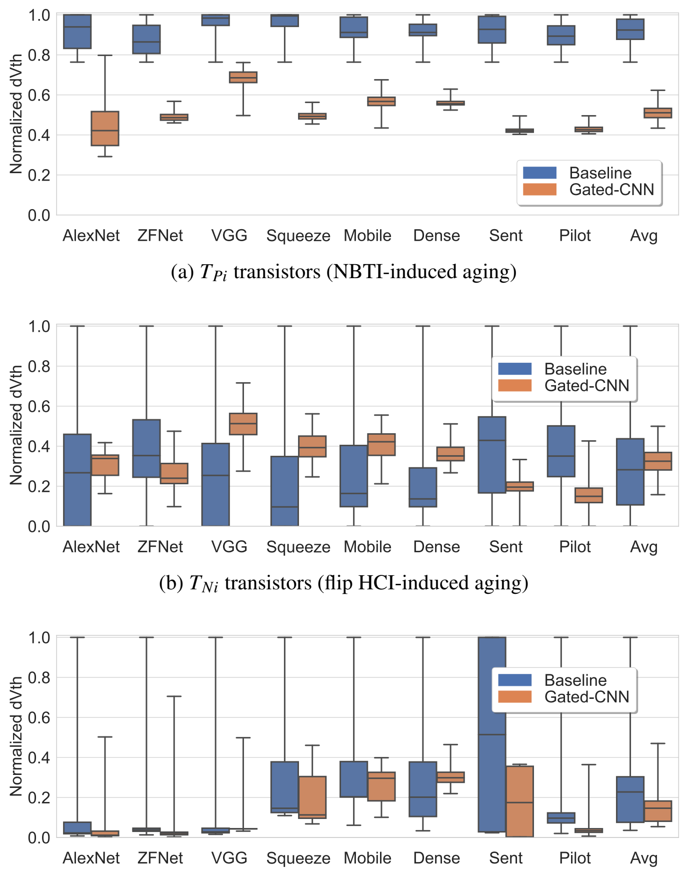


Figure 12: Normalized highest bit flip and access peaks of the Gated-CNN approach with respect to the baseline design. Results differentiate between a fixed I/O buffer size (Fix) and a buffer size adjusted to the largest layer (Adj).



(a) T_{Pi} transistors (NBTI-induced aging)

(b) T_{Ni} transistors (flip HCI-induced aging)

(c) T_{Wi} transistors (access HCI-induced aging)

Figure 13: Normalized dV_{th} for every cell transistor with respect to the maximum dV_{th} after a 3-year lifetime, distinguishing among different types of transistors (aging affects).

the same reason as stated above. On the contrary, AlexNet, VGG16, and SqueezeNet show peak reductions. This is due to the adjusted buffer size changes the bank rotation assignment in such a way that reduces the flip and access peaks with respect to the fixed 2MB buffer size.

Overall, the average maximum flip and access peak reductions are by 62 and 79%, respectively, compared to the baseline design.

6.5. Threshold Voltage Degradation

This section analyzes the threshold voltage degradation (dV_{th}) caused by the studied aging phenomena in all the transistors. The voltage degradation has been obtained for a 3-year lifetime [47]. The inference of the chosen subset of images is repeated over and over until the established lifetime is reached [18]. Results differentiate between the two types of aging effects: NBTI-induced wearout in PMOS transistors (T_{Pi}) is derived from Equation 1, whereas HCI-induced degradation in NMOS transistors is computed from Equation 2. In turn, NMOS transistors are distinguished between inverter loop (T_{Ni}) and pass transistors (T_{Wi}). Results have been obtained assuming a fixed 2 MB size for the I/O buffer.

Figure 13a shows the normalized dV_{th} distribution of the baseline and Gated-CNN with respect to the maximum dV_{th} after 3 years. The plotted results refer to the NBTI degradation affecting the T_{Pi} transistors. As observed, the proposed approach reduces the NBTI stress in all the transistors, since boxes and whiskers are located below those of the baseline in all the benchmarks. In addition, the heights of the boxes referring to Gated-CNN are shorter than those of the baseline in most benchmarks. That is, a more homogeneous NBTI degradation across all the transistors is achieved by the proposed Gated-CNN approach in at least a 50% of the cells.

The NBTI-induced dV_{th} savings for the worst-case T_{Pi} transistors range from 20 (AlexNet) to 51% (SentimentalNet). The average dV_{th} reduction for all these transistors is by 49%.

Figure 13b plots the normalized dV_{th} distribution caused by the HCI effect in the T_{Ni} transistors. Unlike the NBTI effect, some boxes of Gated-CNN stay above those of the baseline approach as a consequence of spreading out flips across NMOS transistors. In other words, HCI is more uniformly distributed across transistors, which might imply distribution percentiles above those of the baseline. Nevertheless, like the NBTI effect, the HCI distribution helps Gated-CNN to obtain thinner boxes and largely reduce the dV_{th} in the worst-case transistors.

The HCI-induced dV_{th} reduction for the worst-case T_{Ni} transistors ranges from 28 (VGG) to 67% (SentimentalNet). The average dV_{th} savings for all these transistors is by 68%

Finally, Figure 13c focuses on the HCI effect in the T_{Wi} transistors. Similarly to the previous results, Gated-CNN obtains thinner boxes with respect to the baseline in most benchmarks, whereas the worst-case dV_{th} of the transistors is substantially reduced.

The HCI-induced dV_{th} savings for the worst-case T_{Wi} transistors vary between 29 (ZFNet) and 64% (PilotNet), whereas the average dV_{th} savings for all these transistors is by 85%.

7. Related Work

Prior related work focusing on aging-aware mechanisms for on-chip memories can be classified into bit-flipping techniques, modifying the design of 6T SRAM cells, data rotation schemes, and power-gating approaches.

7.1. Bit-Flipping Approaches

A significant amount of work has addressed the NBTI wearout by periodically inverting the memory contents. The Penelope processor complements the contents of idle CPU cache blocks and registers [48]. Gebregiorgis *et al.* attack the same memory structures by identifying bit positions with an optimal NBTI signal probability and inverting the remaining bits according to such signals [49]. The iRMW mechanism flips the CPU cache contents depending on the type of write operation [50]. Similarly, recent FPGA designs have also relied on bit-flipping techniques to address aging effects in both the combinational and memory CMOS-based circuits [51]. In CNN accelerators, the state-of-the-art DNN-Life approach periodically flips the weight buffer contents using a random function. See Section 2.2 for more details.

Bit-flipping techniques effectively balance the cell duty cycles. However, these mechanisms come at the cost of aggravating the HCI effect.

7.2. SRAM Cell Circuit

Recovery Boosting focuses on NBTI mitigation in CPU memories by modifying the memory cells in such a way that both the ground voltage and the bitlines are raised to V_{dd} when cell contents are invalid [52]. Kothawade *et al.* implement CPU register files combining normally-sized transistors with NBTI-resilient up-sized transistors in a manner that the latter store the output of aging-inducing instructions [53]. Ricketts *et al.* investigate the usage of robust 8T SRAM cells together with power saving techniques to minimize the impact of NBTI [38]. Gong *et al.* employ 8T cells to implement the most significant bits from CPU integer registers, since they present highly biased '0' duty cycles, whereas 6T cells are used for the remaining bits [39]. Dounavi *et al.* take advantage of a cell aging prediction mechanism along with a repairing technique using spare cells [54].

Unfortunately, either spare cells, up-sizing, or adding more transistors to the 6T SRAM cell design imply high area and power overheads, preventing the adoption of these techniques in area and power-constrained designs like CNN accelerators.

7.3. Rotation Schemes

Dynamic Indexing identifies idle cache blocks where V_{dd} can be reduced to alleviate NBTI degradation, and uniformly spread out such idle blocks across the cache with different index update functions [55]. Colt attacks both the HCI and NBTI effects in CPU caches [56]. The former effect is minimized by applying a cache set rotation using an LFSR, whereas the latter effect is combated by periodically complementing the memory contents. Proactive Recovery establishes a suspended NBTI wearout mode on a rotating basis thanks to including spare cache memory arrays in the CPU design [57]. Other works focus on NBTI mitigation in CPU registers, where both inter and intra-register bit rotation mechanisms are performed, either splitting the register in two halves [35], introducing a barrel-shifter between the address decoder and the memory cells [36], or proposing alternative physical-to-logical register mappings [58].

These techniques cannot be directly applied to domain-specific CNN accelerators since: i) the memory organization and management in general-purpose processors is different, and ii) the memory demands and the temporal/spatial localities of general-purpose applications largely differ with respect to those of CNN applications.

7.4. Power-Gating

Power-gating has been previously exploited in CPU and GPU architectures to alleviate transistor aging. Valero *et al.* focus on CPU caches by power gating zero data bytes and rearranging the bytes of the cache blocks to uniformly distribute power-off cycles among all the bytes [37]. Such a rotation scheme is ineffective in CNN accelerators according to the characterization study shown in Section 4. In addition, the percentage of null bytes in CNN activations can be rather low (e.g., 13% in AlexNet). Finally, fine-grain power-gating schemes impose a large area overhead, making such approaches impractical for area-constrained embedded systems.

The ARGO approach exploits the observation that some GPU registers are never used, and consequently power gates such registers [40]. The RC+RAR technique also exploits this observation, and increases the power-gating opportunities by compressing entire GPU registers using a variation of the base-delta-immediate compression algorithm [59]. In addition, both ARGO and RC+RAR approaches modify the wavefront register allocation to distribute switch-off cycles among all the registers. The under-utilization of GPU register files is caused by badly programmed applications where the underlying GPU hardware is not taken into account. Contrary to such general-purpose applications, the memory under-utilization of CNN applications dynamically changes at runtime and is caused by the unique memory demands of these applications, requiring specific solutions for CNN accelerators.

8. Conclusions

Negative Bias Temperature Instability (NBTI) and Hot Carrier Injection (HCI) are two of the main aging phenomena that compromise the system's lifetime reliability. Both effects degrade the transistor's threshold voltage (V_{th}) over time, making transistors slower and eventually resulting in timing violations. These effects are especially critical in those transistors used to implement SRAM memories because they are permanently aging. In this sense, NBTI affects transistors when a cell stores a given logic value for a long period (i.e., a long duty cycle), whereas HCI manifests when the cell content is accessed and flipped.

On the other hand, the slowdown of the transistor scaling has become one of the major drivers toward domain-specific accelerators, which deliver a higher performance-to-power ratio with respect to general-purpose processors. Convolution Neural Network (CNN) accelerators have consolidated as commodity devices that speed up the inference process required on classification and regression tasks. These accelerators usually implement large on-chip SRAM memories to cache the neural network parameters consisting of weights and activations, thereby minimizing costly off-chip memory accesses.

This paper has identified those cell transistors of on-chip SRAM activation memories of CNN accelerators that age the most from the perspective of both NBTI and HCI effects. Based on this information, this work has presented Gated-CNN, a novel microarchitectural technique that reduces the largest duty cycles by power gating specific memory cells, and evenly distributes the access and flip patterns across all the cells with an address rotation mechanism.

Experimental results have shown that Gated-CNN extends the lifetime of the cell transistors with average V_{th} degradation savings at least as much as 49% depending on the type of transistor.

As for future work, we plan to characterize other aging phenomena in CNN accelerators, like time dependent dielectric breakdown and electromigration effects, and explore new microarchitectural techniques contributing to the device lifetime extension.

References

[1] T. Chen, Z. Du, N. Sun, J. Wang, C. Wu, Y. Chen, O. Temam, DianNao: A Small-Footprint High-Throughput Accelerator for Ubiquitous Machine-Learning, in: Proceedings of the 19th International Conference on Architectural Support for Programming Languages and Operating Systems, 2014, pp. 269–284.

[2] 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.

[3] 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.

[4] 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.

[5] 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.

[6] Y.-H. Chen, T. Krishna, J. S. Emer, V. Sze, Eyeriss: An Energy-Efficient Reconfigurable Accelerator for Deep Convolutional Neural Networks, IEEE Journal of Solid-State Circuits 52 (1) (2017) 127–138.

[7] D. Shin, J. Lee, J. Lee, H.-J. Yoo, 14.2 DNPU: An 8.1TOPS/W reconfigurable CNN-RNN processor for general-purpose deep neural networks, in: Proceedings of the IEEE International Solid-State Circuits Conference, 2017, pp. 240–241.

[8] 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.

[9] A. Delmas Lascorz, P. Judd, D. M. Stuart, Z. Poulos, M. Mahmoud, S. Sharify, M. Nikolic, K. Siu, A. Moshovos, Bit-Tactical: A Software/Hardware Approach to Exploiting Value and Bit Sparsity in Neural Networks, in: Proceedings of the 24th International Conference on Architectural Support for Programming Languages and Operating Systems, 2019, pp. 749–763.

[10] 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 Journal of Solid-State Circuits 54 (2019) 173–185.

[11] A. Alcolea Moreno, J. Olivito, J. Resano, H. Mecha, Analysis of a Pipelined Architecture for Sparse DNNs on Embedded Systems, IEEE Transactions on Very Large Scale Integration (VLSI) Systems 28 (9) (2020) 1993–2003.

[12] J. Sim, S. Lee, L.-S. Kim, An Energy-Efficient Deep Convolutional Neural Network Inference Processor With Enhanced Output Stationary Dataflow in 65-nm CMOS, IEEE Transactions on Very Large Scale Integration (VLSI) Systems 28 (1) (2020) 87–100.

[13] T. Alnuayri, S. Khursheed, A. L. H. Martínez, D. Rossi, Differential Aging Sensor Using Subthreshold Leakage Current to Detect Recycled ICs, IEEE Transactions on Very Large Scale Integration (VLSI) Systems 29 (12) (2021) 2064–2075.

[14] T. Nigam, B. Parameshwaran, G. Krause, Accurate product lifetime predictions based on device-level measurements, in: Proceedings of the IEEE International Reliability Physics Symposium, 2009, pp. 634–639.

[15] A. Ruospo, E. Sanchez, M. Traiola, I. O'Connor, A. Bosio, Investigating Data Representation for Efficient and Reliable Convolutional Neural Networks, Elsevier Microprocessors and Microsystems 86 (2021) 1–14.

[16] 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.

[17] R. Vattikonda, W. Wang, Y. Cao, Modeling and Minimization of PMOS NBTI Effect for Robust Nanometer Design, in: Proceedings of the 43rd Design Automation Conference, 2006, pp. 1047–1052.

[18] A. Tiwari, J. Torrellas, Facelift: Hiding and Slowing Down Aging in Multicores, in: Proceedings of the 41st Annual IEEE/ACM International Symposium on Microarchitecture, 2008, pp. 129–140.

[19] A. Krizhevsky, I. Sutskever, G. E. Hinton, ImageNet Classification with Deep Convolutional Neural Networks, Advances in Neural Information Processing Systems 25 (2012) 1097–1105.

[20] M. D. Zeiler, R. Fergus, Visualizing and Understanding Convolutional Networks, in: European Conference on Computer Vision, Springer, 2014, pp. 818–833.

[21] K. Simonyan, A. Zisserman, Very Deep Convolutional Networks for Large-Scale Image Recognition, arXiv preprint arXiv:1409.1556 (2014).

[22] 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, arXiv preprint arXiv:1602.07360 (2016).

[23] 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, CoRR abs/1704.04861 (2017). `arXiv:1704.04861`.
URL `http://arxiv.org/abs/1704.04861`

[24] F. N. Iandola, M. W. Moskewicz, S. Karayev, R. B. Girshick, T. Darrell, K. Keutzer, DenseNet: Implementing Efficient ConvNet Descriptor Pyramids, CoRR abs/1404.1869 (2014). `arXiv:1404.1869`.
URL `http://arxiv.org/abs/1404.1869`

[25] J. Brownlee, How to Predict Sentiment From Movie Reviews Using Deep Learning (Text Classification), `https://machinelearningmastery.com/predict-sentiment-movie-reviews-using-deep-learning/` (2016).

[26] M. Bojarski, D. D. Testa, D. Dworakowski, B. Firner, B. Flepp, P. Goyal, L. D. Jackel, M. Monfort, U. Muller, J. Zhang, X. Zhang, J. Zhao, K. Zieba, End to End Learning for Self-Driving Cars (2016). `arXiv:1604.07316`.

[27] S. Kaxiras, Z. Hu, M. Martonosi, Cache Decay: Exploiting Generational Behavior to Reduce Cache Leakage Power, in: Proceedings of the 28th Annual International Symposium on Computer Architecture, 2001, pp. 240–251.

[28] A. Calimera, E. Macii, M. Poncino, Analysis of NBTI-Induced SNM

Degradation in Power-Gated SRAM Cells, in: Proceedings of the IEEE International Symposium on Circuits and Systems, 2010, pp. 785–788.

[29] T. Siddiqua, S. Gurumurthi, Enhancing NBTI Recovery in SRAM Arrays Through Recovery Boosting, IEEE Transactions on Very Large Scale Integration (VLSI) Systems 20 (4) (2012) 616–629.

[30] M. T. Rahman, D. Forte, J. Fahrny, M. Tehranipoor, ARO-PUF: An Aging-Resistant Ring Oscillator PUF Design, in: Proceedings of the Design, Automation & Test in Europe Conference & Exhibition, 2014, pp. 1–6.

[31] 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, Scientific Reports 6 (2016) 27988.

[32] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, L. Fei-Fei, ImageNet: A Large-Scale Hierarchical Image Database, in: IEEE Conference on Computer Vision and Pattern Recognition, 2009, pp. 248–255.

[33] A. Yazdanbakhsh, K. Seshadri, B. Akin, J. Laudon, R. Narayanaswami, An Evaluation of Edge TPU Accelerators for Convolutional Neural Networks (2021). `arXiv:2102.10423`.

[34] A. Samajdar, Y. Zhu, P. N. Whatmough, M. Mattina, T. Krishna, SCALE-Sim: Systolic CNN Accelerator, CoRR abs/1811.02883 (2018). `arXiv:1811.02883`.
URL `http://arxiv.org/abs/1811.02883`

[35] H. Amrouch, T. Ebi, J. Henkel, Stress Balancing to Mitigate NBTI Effects in Register Files, in: Proceedings of the 43rd Annual IEEE/IFIP International Conference on Dependable Systems and Networks, 2013, pp. 1–10.

[36] S. Kothawade, K. Chakraborty, S. Roy, Analysis and Mitigation of NBTI Aging in Register File: An End-To-End Approach, in: Proceedings of the 12th International Symposium on Quality Electronic Design, 2011, pp. 1–7.

[37] A. Valero, N. Miralaei, S. Petit, J. Sahuquillo, T. M. Jones, On Microarchitectural Mechanisms for Cache Wearout Reduction, IEEE Transactions on Very Large Scale Integration (VLSI) Systems 25 (3) (2017) 857–871.

[38] A. Ricketts, J. Singh, K. Ramakrishnan, N. Vijaykrishnan, D. K. Pradhan, Investigating the impact of nbti on different power saving cache strategies, in: Proceedings of the Design, Automation & Test in Europe Conference & Exhibition, 2010, pp. 592–597.

[39] N. Gong, J. S., J. Wang, A. B., K. Sekar, R. Sridhar, Hybrid-Cell Register Files Design for Improving NBTI Reliability, Elsevier Microelectronics Reliability 52 (9–10) (2012) 1865–1869.

[40] M. Namaki-Shoushtari, A. Rahimi, N. Dutt, P. Gupta, R. K. Gupta, ARGO: Aging-aware GPGPU Register File Allocation, in: Proceedings of the International Conference on Hardware/Software Codesign and System Synthesis, 2013, pp. 1–9.

[41] S. Lee, K. Kim, G. Koo, H. Jeon, M. Annavaram, W. W. Ro, Improving Energy Efficiency of GPUs Through Data Compression and Compressed Execution, IEEE Transactions on Computers 66 (5) (2017) 834–847.

[42] S. Li, K. Chen, J. H. Ahn, J. B. Brockman, N. P. Jouppi, CACTI-P: Architecture-Level Modeling for SRAM-based Structures with Advanced Leakage Reduction Techniques, HP Laboratories, Palo Alto, CA, USA. Technical Report HPL-2012-187 (2012).

[43] H. Pilo, V. Ramadurai, G. Braceras, J. Gabric, S. Lamphier, Y. Tan, A 450ps Access-Time SRAM Macro in 45nm SOI Featuring a Two-Stage Sensing-Scheme and Dynamic Power Management, in: Proceedings of the IEEE International Solid-State Circuits Conference - Digest of Technical Papers, 2008, pp. 378–621.

[44] H. Pilo, C. A. Adams, I. Arsovski, R. M. Houle, S. M. Lamphier, M. M. Lee, F. M. Pavlik, S. N. Sambatur, A. Seferagic, R. Wu, M. I. Younus, A 64Mb SRAM in 22nm SOI technology featuring fine-granularity power gating and low-energy power-supply-partition techniques for 37% leakage reduction, in: Proceedings of the IEEE International Solid-State Circuits Conference - Digest of Technical Papers, 2013, pp. 322–323.

[45] 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). `arXiv:1603.04467`.

[46] F. Oboril, M. B. Tahoori, ExtraTime: Modeling and Analysis of Wearout due to Transistor Aging at Microarchitecture-Level, in: Proceedings of the 42nd Annual IEEE/IFIP International Conference on Dependable Systems and Networks, 2012, pp. 1–12.

[47] E. Mintarno, V. Chandra, D. Pietromonaco, R. Aitken, R. W. Dutton, Workload-Dependent NBTI and PBTI Analysis for a sub-45nm Commercial Microprocessor, in: IRPS, 2013, pp. 1–6.

[48] J. Abella, X. Vera, A. González, Penelope: The NBTI-Aware Processor, in: Proceedings of the 40th Annual IEEE/ACM International Symposium on Microarchitecture, 2007, pp. 85–96.

[49] A. Gebregiorgis, M. Ebrahimi, S. Kiamehr, F. Oboril, S. Hamdioui, M. B. Tahoori, Aging Mitigation in Memory Arrays Using Self-controlled Bit-flipping Technique, in: Proceedings of the 20th Asia South Pacific Design Automation Conference, 2015, pp. 231–236.

[50] S. Ganapathy, R. Canal, A. González, A. Rubio, iRMW: A Low-Cost Technique to Reduce NBTI-Dependent Parametric Failures in L1 Data Caches, in: Proceedings of the 32nd IEEE Interntional Conference on Computer Design, 2014, pp. 68–74.

[51] M. H. Mottaghi, M. Sedighi, M. Saheb Zamani, FIFA: A Fully Invertible FPGA Architecture to Reduce BTI-Induced Aging Effects, IEEE Transactions on Computers (Early Access) (2021) 1–11.

[52] T. Siddiqua, S. Gurumurthi, Recovery Boosting: A Technique to Enhance NBTI Recovery in SRAM Arrays, in: Proceedings of the IEEE Computer Society Annual Symposium on VLSI, 2010, pp. 393–398.

[53] S. Kothawade, D. M. Ancajas, K. Chakraborty, S. Roy, Mitigating NBTI in the Physical Register File through Stress Prediction, in: Proceedings of the IEEE 30th International Conference on Computer Design, 2012, pp. 345–351.

[54] H.-M. Dounavi, Y. Sfikas, Y. Tsiatouhas, Aging Prediction and Tolerance for the SRAM Memory Cell and Sense Amplifier, Springer Journal of Electronic Testing 37 (2021) 65–82.

[55] A. Calimera, M. Loghi, E. Macii, M. Poncino, Dynamic Indexing: Leakage-Aging Co-Optimization for Caches, IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems 33 (2) (2014) 251–264.

[56] E. Gunadi, A. A. Sinkar, N. S. Kim, M. H. Lipasti, Combating Aging with the Colt Duty Cycle Equalizer, in: Proceedings of the 43rd Annual IEEE/ACM International Symposium on Microarchitecture, 2010, pp. 103–114.

[57] J. Shin, V. Zyuban, P. Bose, T. M. Pinkston, A Proactive Wearout Recovery Approach for Exploiting Microarchitectural Redundancy to Extend Cache SRAM Lifetime, in: Proceedings of the 35th International Symposium on Computer Architecture, 2008, pp. 353–362.

[58] I. Tuzov, P. Andreu, L. Medina, T. Picornell, A. Robles, P. López, J. Flich, C. Hernández, Improving the Robustness of Redundant Execution with Register File Randomization, in: Proceedings of the IEEE/ACM International Conference on Computer Aided Design, 2021, pp. 1–9.

[59] A. Valero, F. Candel, D. Suárez-Gracia, S. Petit, J. Sahuquillo, An Aging-Aware GPU Register File Design Based on Data Redundancy, IEEE Transactions on Computers 68 (1) (2019) 4–20.

Nicolás Landeros Muñoz received the B.S. degree in Engineering Sciences and Electronic Engineering from the Federico Santa María Technical University, Valparaíso, Chile, in 2019. He is currently working toward the M.S. degree in Computer Science and Engineering at Politecnico di Milano, Italy. His research interests include the design of machine learning accelerators with a focus on reliability.

Alejandro Valero received the Ph.D. degree in Computer Engineering from the Universitat Politècnica de València, Spain, in 2013. From 2013 to 2015, he was a visiting researcher with Northeastern University, Boston, MA, USA, and the University of Cambridge, UK. Since 2016, he has been a professor with the Department of Computer Science and Systems Engineering, Universidad de Zaragoza, Spain. His research interests include

GPU 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.

Rubén Gran Tejero graduated in Computer Science from the University of Zaragoza, Spain. He received his Ph.D. from the Polytechnic University of Catalonia (UPC), Spain, in 2010. He is currently an Associate Professor in the Department of Computer Science and Systems Engineering at the University of Zaragoza. Previously, from 2010 to 2021, he was an Assistant Professor in the same institution. He has been Visiting Research Associate in the University of Illinois at Urbana-Champaign (UIUC) in 2011 and 2013. 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.

Davide Zoni is an Assistant Professor at Politecnico di Milano, Italy. He published more than 50 papers in journals and conference proceedings. His research interests include RTL design and optimization of complex embedded systems with emphasis on low power methodologies and hardware security. He filed two patents on cyber-security and won the Switch2Product competition in 2019. He is also the principal investigator of LAMP (see `www.lamp-platform.org`).