Bit Error Robustness for Energy-Efficient DNN Accelerators

David Stutz
david.stutz@mpi-inf.mpg.de
Max Planck Institute for Informatics
Saarland Informatics Campus, Germany

Nandhini Chandramoorthy
nandhini.chandramoorthy@ibm.com
IBM T. J. Watson Research Center
Yorktown Heights, New York, USA

Matthias Hein
matthias.hein@uni-tuebingen.de
University of Tübingen
Tübingen, Germany

Bernt Schiele
schiele@mpi-inf.mpg.de
Max Planck Institute for Informatics
Saarland Informatics Campus, Germany

ABSTRACT

Deep neural network (DNN) accelerators received considerable attention in past years due to saved energy compared to mainstream hardware. Low-voltage operation of DNN accelerators allows to further reduce energy consumption significantly, however, causes bit-level failures in the memory storing the quantized DNN weights. In this paper, we show that a combination of **robust fixed-point quantization**, **weight clipping**, and **random bit error training (RandBET) improves robustness against random bit errors in (quantized) DNN weights** significantly. This leads to high energy savings from *both* low-voltage operation *as well as* low-precision quantization. Furthermore, our approach generalizes across operating voltages and accelerators, as demonstrated on bit errors from profiled SRAM arrays. Without losing more than 1% in accuracy, we can reduce energy consumption on CIFAR10 by 20% for a 8-bit quantized DNN. Higher energy savings of, e.g., 30%, are possible at the cost of 2.5% accuracy, even for 4-bit DNNs.

KEYWORDS

adversarial robustness, adversarial examples, adversarial training, robust overfitting, robust generalization, flat minima

ACM Reference Format:
David Stutz, Nandhini Chandramoorthy, Matthias Hein, and Bernt Schiele. 2021. Bit Error Robustness for Energy-Efficient DNN Accelerators. In *AdvML '21: Workshop on Adversarial Learning Methods for Machine Learning and Data Mining, August 15, 2021, Singapore (virtual).* ACM, New York, NY, USA, 5 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

Energy-efficiency is important to lower carbon-dioxide emissions of deep neural network (DNN) driven applications and to enable applications in edge computing. *DNN accelerators*, i.e., specialized hardware for inference, reduce energy consumption alongside cost and space compared to mainstream GPUs. These accelerators generally feature on-chip SRAM used as scratchpads, e.g., to store DNN weights. Data access/movement constitutes a dominant component of accelerator energy consumption [43]. Reduced precision [29] is one way to reduce energy consumption at the cost of *approximate computing* [38]. Similarly, recent DNN accelerators [6, 23, 37] lower memory supply voltage to increase energy efficiency since dynamic power varies quadratically with voltage. However, operating at very low voltages causes reliability issues in SRAMs in the form of bit-level failures [14, 17] with direct impact on the stored DNN weights. The rate p of these errors increases exponentially with lowered voltage, causing devastating drops in DNN accuracy. In this paper, we aim to enable very low-voltage operation of DNN accelerators by developing DNNs robust to such bit errors in their weights, allowing DNN inference on *"approximate hardware"* [25].

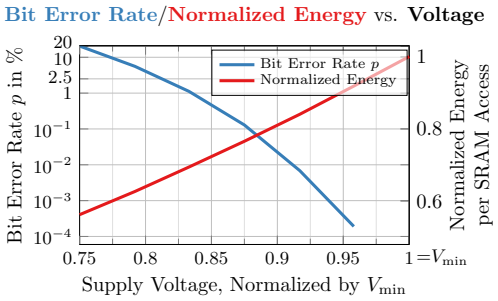
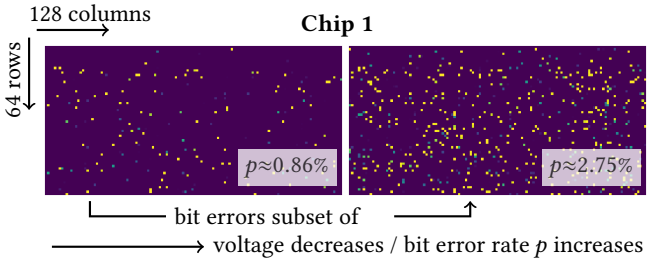


Figure 1: Average bit error rate p (blue, left y-axis) from 32 14nm SRAM arrays of size 512×64 from [6] and energy (red, right y-axis) vs voltage (x-axis). Voltage is normalized by $V_{\min}$, the minimal measured voltage for error-free operation. Reducing voltage leads to exponentially increasing bit error rates.

Fig. 1 (left) shows the average bit error rates of SRAM arrays as supply voltage is scaled below $V_{\min}$, i.e., the measured lowest voltage at which there are no bit errors. DNNs robust to a bit error rate (blue, left y-axis) of, e.g., $p = 1\%$ allow to reduce SRAM energy by roughly 30%. To improve DNN robustness to bit errors, we first consider the impact of fixed-point quantization on robustness. While prior work [30, 32, 42] studies robustness *to* quantization, we find that the choice of quantization scheme has tremendous impact on robustness, even though accuracy is not affected. We identify a particularly **robust quantization scheme**, RQuant in Fig. 1 (right, red). Additionally, we propose aggressive **weight clipping**

AdvML '21, August 15, 2021, Singapore (virtual)

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM…$15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn



Model (CIFAR10)	RErr in %, p in %	
***Fixed* Pattern**	p=1	p=2.5
PattBET$_{0.15}$ p=2.5	8.50	7.41
***Random* Patterns**	p=1	p=2.5
PattBET$_{0.15}$ p=2.5	12.09	61.59
***Profiled* Chip** (cf. left)	p≈0.86	p≈2.75
RandBET$_{0.05}$ p=1.5	7.04	9.37

Figure 2: Left: Measured bit errors from on-chip SRAM, showing bit flip probability for a segment of 64×128 bits: yellow indicates a bit flip probability of one, violet indicates zero probability. We show measurements corresponding to two supply voltages. Right (top): RErr for training on a fixed bit error pattern (PattBET) and evaluation on the same pattern and completely random patterns. PattBET fails to generalize to lower bit error rates (in red), i.e., subsets of bit errors trained on, as well as random bit errors (i.e., other chips). Right (bottom): RErr for RandBET on the profiled bit errors shown on the left. RandBET generalizes well to the profiled bit errors, even though the pattern was not seen during training.

during training as regularization to improve robustness, Clipping in Fig. 1 (right, blue). This is in contrast to, e.g., [42, 48] ignoring weight outliers to reduce quantization range, with sole focus of improving accuracy.

Common error correcting codes (ECCs such as SECDED), cannot correct *multiple* bit errors per word (containing multiple DNN weights). However, for $p = 1\%$, the probability of two or more bit errors in a 64-bit word is 13.5%. Error detection via redundancy [37] or supply voltage boosting [6] allow error-free low-voltage operation at the cost of additional energy or space. Therefore, [23, 25] propose co-design approaches of training DNNs on *profiled* SRAM/DRAM bit errors. These approaches work as the spatial bit error patterns can be assumed fixed for a *fixed* accelerator *and* voltage. However, the random nature of variation-induced bit errors requires profiling to be carried out for each voltage, memory array and individual chip making training DNNs on profiled bit error patterns an expensive process. More importantly, the obtained DNNs do *not* generalize across voltages or to unseen bit error patterns, e.g., from other memory arrays. We propose **random bit error training (RandBET)** which, in combination with weight clipping and robust quantization, obtains robustness against completely *random* bit error patterns, see Fig. 1 (right, violet). Thereby, it generalizes across chips *and* voltages, without profiling, hardware-specific mapping or other circuit-level mitigation strategies.

2 RELATED WORK

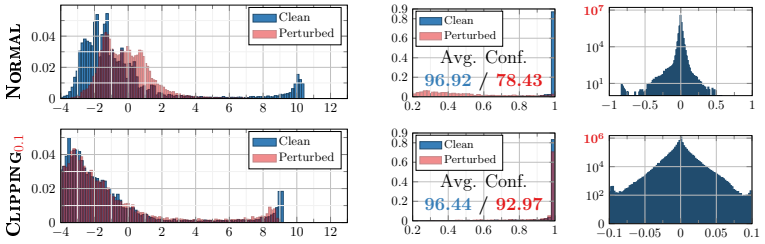
Quantization: DNN Quantization [16] is usually motivated by faster DNN inference, e.g., through fixed-point quantization and arithmetic [28, 29, 40], and energy savings. To avoid reduced accuracy, quantization is considered during training [21, 26], enabling low-bit quantization such as binary DNNs [11, 36]. While works such as [4, 30, 32, 42] study the robustness of DNNs *to* quantization, the robustness of various quantization schemes *against* random bit errors has not been studied. This is in stark contrast to our findings that quantization impacts robustness significantly. Furthermore, works such as [34, 42, 48] clip weight outliers to reduce approximation error, improving accuracy. We consider *weight clipping* independent of quantization *as regularization during training*.

Bit Errors in DNN Accelerators: Recent work [13, 14] demonstrates that bit flips in SRAMs increase exponentially when reducing voltage below $V_{\min}$. The authors of [6] study the impact of bit flips in different layers of DNNs, showing severe accuracy degradation. Similar observations hold for DRAM [7]. To prevent accuracy drops at low voltages, [37] combines SRAM fault detection with logic to set faulty data reads to zero. [6] uses supply voltage boosting for SRAMs to ensure error-free, low-voltage operation, while [41] proposes storing critical bits in specifically robust SRAM cells. However, such methods incur power and area overhead. Thus, [23] and [25] propose co-design approaches combining training on profiled SRAM/DRAM bit errors with hardware mitigation strategies. In contrast to [23, 25], our *random bit error training* obtains robustness that generalizes across chips and voltages without expensive chip-specific profiling or hardware mitigation strategies. Furthermore, [23, 25] do not address the role of quantization and we demonstrate that these approaches can benefit from our weight clipping, as well.

Weight Robustness: Only few works consider weight robustness: [45] certify the robustness of weights with respect to L_∞ perturbations and [8] study Gaussian noise on weights. [19, 35] consider identifying and (adversarially) flipping few vulnerable bits in quantized weights. Fault tolerance, in contrast, describes structural changes such as removed units, and is rooted in early work such as [9, 33]. We study robustness against *random bit errors*, which exhibit a quite special noise pattern compared to L_∞ or Gaussian noise.

3 LOW-VOLTAGE RANDOM BIT ERRORS IN QUANTIZED DNN WEIGHTS

We assume the quantized DNN weights to be stored (linearly) on multiple memory banks, e.g., SRAM or DRAM. Following [6, 14, 23], the probability of memory bit cell failures increases exponentially as operating voltage is scaled below $V_{\min}$, i.e., the minimal voltage required for reliable operation, cf. Fig. 1. This is done intentionally to reduce energy consumption, e.g., [6, 23, 25], or adversarially by an attacker, e.g., [44]. Process variation during fabrication causes a variation in the vulnerability of individual bit cells. For a specific memory array, bit cell failures are typically approximately random and independent of each other [14]. Nevertheless, there is generally an "inherited" distribution of bit cell failures across voltages: as described in [13], if a bit error occurred at a given voltage, it is likely to occur at lower voltages, cf. Fig. 2 (left). However, across



	Model (CIFAR10) w_{max}=0.1, p in %	Err in %	RErr in % p=0.5	RErr in % p=1.5
8bit	Normal	4.37	15.42	81.74
	RQuant	**4.32**	11.28	68.65
	+Clipping	4.82	6.95	12.22
	+RandBET p=1	4.90	**6.36**	**8.65**
4bit	Clipping	**5.29**	7.71	15.79
	+RandBET p=1	5.39	**7.04**	**9.77**

Figure 3: Left: Weight clipping constraints the weights (right), thereby implicitly limiting the possible range for logits (left, blue). However, even for $w_{max} = 0.1$ the DNN is able to produce high confidences (middle, blue), suggesting that more weights are used to obtain these logits. As result, the impact of random bit errors, $p = 1\%$, on the logits/confidences (red) is reduced. Right: Average RErr of RandBET evaluated at bit error rates $p = 0.5\%$ and $p = 1.5\%$ using $m = 8$ or 4 bits. For low p, weight clipping provides sufficient robustness. For larger p, RandBET increases robustness significantly.

different SRAM arrays or different chips, the patterns or spatial distribution of bit errors is usually different and can be assumed random [6]. Throughout the paper, we use the following bit error model:

Random Bit Error Model: *The probability of a bit error is p (in %) for all weight values and bits. For a fixed memory array, bit errors are persistent across supply voltages, i.e., bit errors at probability $p' \leq p$ also occur at probability p. A bit error flips the currently stored bit. We denote random bit error injection by $BErr_p$.*

This error model captures the nature of low-voltage induced bit errors, from both SRAM and DRAM [6, 23, 25]. However, our approach in Sec. 4 is model-agnostic: the error model can be refined if extensive memory characterization results are available for individual chips. However, estimating these specifics requires testing infrastructure and introduces the risk of overfitting. Furthermore, we demonstrate that the robustness obtained using our uniform error model generalizes to profiled bit errors from real chips, cf. Fig. 2 (right).

4 TOWARDS ROBUSTNESS AGAINST RANDOM BIT ERRORS

We address robustness against random bit errors in three steps: First, we analyze the impact of fixed-point quantization schemes on bit error robustness. This has been neglected both in prior work on low-voltage DNN accelerators [23, 25] and in work on quantization robustness [30, 32, 42]. This yields our **robust quantization**. On top, we propose aggressive **weight clipping** as regularization during training, enforcing a more uniformly distributed, i.e., redundant, weight distribution. We argue that the redundancy is a result of limiting weight range while encouraging large logits by minimizing the cross-entropy loss. Finally, in addition to robust quantization and weight clipping, we perform **random bit error training (RandBET)**: in contrast to the fixed bit error patterns in [23, 25], we train on completely *random* bit errors and, thus, generalize across chips and voltages.

4.1 Robust Fixed-Point Quantization

We consider quantization-aware training using a simple fixed-point quantization scheme commonly used in DNN accelerators [6]: However, we focus on the impact of quantization schemes on robustness against random bit errors, mostly neglected so far [30, 32, 42]. Let $f(x; w)$ be a DNN taking an example $x \in [0, 1]^D$, e.g., an image, and weights $w \in \mathbb{R}^W$ as input. Quantization determines how weights are represented in memory, e.g., on SRAM. In a *fixed-point quantization* scheme, m bits allow to represent 2^m distinct values. A weight $w_i \in [-q_{max}, q_{max}]$ is represented by a signed m-bit integer $v_i = Q(w_i)$ corresponding to the underlying bits. Here, $[-q_{max}, q_{max}]$ is the *symmetric* quantization range and signed integers use two's complement representation. Then, $Q : [-q_{max}, q_{max}] \mapsto \{-2^{m-1} - 1, \dots, 2^{m-1} - 1\}$ is defined as

$$Q(w_i) = \left\lfloor \frac{w_i}{\Delta} \right\rfloor, \; Q^{-1}(v_i) = \Delta v_i, \; \Delta = \frac{q_{max}}{2^{m-1} - 1} \tag{1}$$

This quantization is symmetric around zero and zero is represented exactly. Note that we consider quantizing weights only. In *global* quantization, q_{max} is chosen to accommodate all weights, i.e., $q_{max} = \max_i |w_i|$. However, it has become standard to apply quantization *per-layer* allowing to adapt q_{max} to each layer. The **per-layer, symmetric quantization is our default reference**, referred to as Normal.

To further reduce quantization error, we also consider arbitrary quantization ranges $[q_{min}, q_{max}]$ (allowing $q_{min} > 0$): we map $[q_{min}, q_{max}]$ to $[-1, 1]$ and quantize $[-1, 1]$ as above. The resulting per-layer *asymmetric* quantization has the finest granularity (i.e., lowest Δ), however, is not the most robust. Therefore, we further replace the floor operation $\lfloor w_i/\Delta \rfloor$ with proper rounding $\lceil w_i/\Delta \rfloor$. Similarly, for asymmetric quantization, we use quantization into *unsigned* integers, i.e., $Q : [q_{min}, q_{max}] \mapsto \{0, \dots, 2^m - 1\}$, instead. It is important to note that these differences have little to no impact on accuracy, while having tremendous impact on robustness against bit errors.

4.2 Weight Clipping

Weight clipping refers to constraining the weights to $[-w_{max}, w_{max}]$ *during training*, where w_{max} is a hyper-parameter. Generally, w_{max} is independent of the quantization range(s) which always adapt(s) to the weight range(s) at hand. However, weight clipping limits the maximum possible quantization range, i.e., $q_{max} \leq w_{max}$. Note that the *relative* errors induced by bit errors do *not* change through weight clipping. As the DNN's decision is usually invariant to rescaling, reducing the scale of the weights does not impact robustness. In fact, we found the mean relative error of the weights to increase with clipping, e.g., at $w_{max} = 0.1$. Thus, weight clipping does *not*

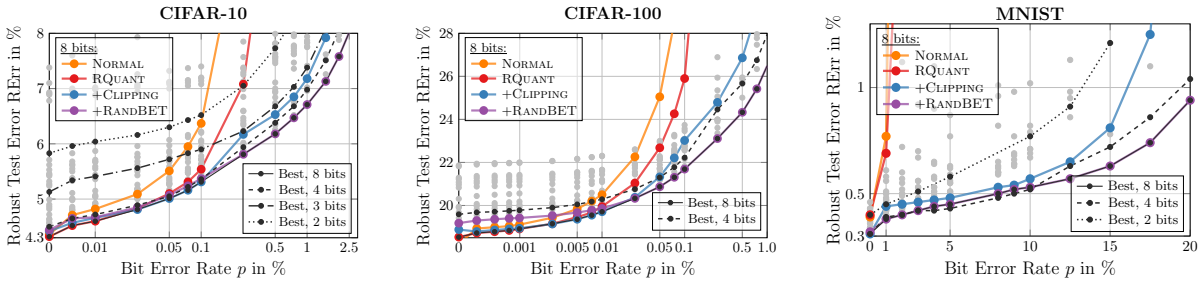


Figure 4: Robust test error (test error *after* injecting bit errors in the quantized weights, RErr, ↓, y-axis) plotted against bit error rate p (x-axis) for our robust fixed-point quantization (RQuant, orange), weight clipping (Clipping, blue) and random bit error training (RandBET, violet). In each case, we highlight the best model for each bit error rate. Additionally, we report the overall best model per bit error rate for various bit error rates, e.g., $m = 8, 4, 3$ and 2 on CIFAR10. For 8 bit and low bit error rates, Clipping is often sufficient. However, for 4 bit or higher bit error rates, RandBET is crucial to keep RErr low.

"trivially" improve robustness by reducing the scale of weights. Instead, we found that the interplay of weight clipping and minimizing the the cross-entropy loss during training is the key. High confidences can only be achieved by large differences in the logits. Because the weights are limited to $[-w_{\max}, w_{\max}]$, large logits can only be achieved using more weights in each layer to produce larger outputs. As a result, weight clipping leads to more weights being utilized, i.e., more redundancy in the weights, making them less susceptible to (bit) errors, as illustrated in Fig. 3 (left).

4.3 Random Bit Error Training (RandBET)

In *addition to* weight clipping and robust quantization, we inject random bit errors with probability p during training to further improve robustness. This results in the following learning problem:

$$\min_w \mathbb{E}[\mathcal{L}(f(x;\tilde{w}),y) + \mathcal{L}(f(x;w),y)] \quad (2)$$

$$\text{s.t.} \quad v = Q(w),\ \tilde{v} = \text{BErr}_p(v),\ \tilde{w} = Q^{-1}(\tilde{v}). \quad (3)$$

where (x, y) are labeled examples, $\mathcal{L}$ is the cross-entropy loss and $v = Q(w)$ denotes the (element-wise) quantized weights w which are to be learned. $\text{BErr}_p(v)$ injects random bit errors with rate p in v. Note that we consider both the loss on clean weights and weights with bit errors to avoid an increase in (clean) test error and stabilizes training. Note that bit error rate p implies, in expectation, pmW bit errors. We use stochastic gradient descent to optimize Eq. (3), by performing the gradient computation using the perturbed weights $\tilde{w} = Q^{-1}(\tilde{v})$ with $\tilde{v} = \text{BErr}_p(v)$, while applying the gradient update on the (floating-point) clean weights w.

5 EXPERIMENTS

We conduct experiments on MNIST and CIFAR [27] and report (clean) test error Err (lower is better, ↓), corresponding to *clean* weights, and **robust test error RErr** (↓), i.e., the **test error *after* injecting bit errors into the weights**. We report *average* RErr across 50 samples of random bit errors for a specific rate p. We use SimpleNet [18] on CIFAR10 and Wide ResNet (WRN) [46] on CIFAR100. Normal training with the standard and our robust quantization are denoted Normal and RQuant, respectively. Weight clipping with $w_{\max}$ is referred to as $\text{Clipping}_{w_{\max}}$ or together with RandBET as $\text{RandBET}_{w_{\max}}$.

Fig. 3 (right) presents RErr on CIFAR10 for $m = 8$ and $m = 4$ bit, showing that our combination of RQuant, Clipping and RandBET (trained with $p = 1\%$ bit error rate) improves robustness to random bit errors significantly, especially for high bit error rates. For smaller bit error rates, e.g., $p = 0.5\%$, Clipping with $w_{\max} = 0.1$ might be sufficient for robust operation, achieving 6.95%RErr, while RandBET is necessary at higher bit error rates, e.g., $p = 1.5\%$. For lower precision, i.e., $m = 4$ bits, the benefit of RandBET is pronounced even further, reducing RErr significantly from 15.79% to 9.77% for $p = 1.5\%$. We also emphasize that RandBET generalize to lower bit errors than trained on. This is in contrast to related work [23, 25], training on fixed bit error patterns (e.g., profiled) as demonstrated in Fig. 2 (right). RandBET also generalizes to bit errors profiled from real chips, see Fig. 2 (right).

Our experiments are summarized in Fig. 4 (right), plotting RErr against bit error rate p for various Clipping and RandBET models corresponding to different $w_{\max}/p$ (indicated in • gray) in comparison to Normal and RQuant. RQuant (red) clearly outperforms Normal (orange), however, RErr increases quickly even for low bit error rates. Clipping (blue) generally reduces RErr, but only the combination with RandBET (violet) can keep RErr around 6% for a bit error rate of $p \approx 0.5\%$ on CIFAR10. This corresponds to roughly 25% energy savings in Fig. 1 (left). The best model for each bit error rate p and different precisions m is shown in black (e.g., solid for $m = 8$ or dashed for $m = 4$). Even for $m = 4$ bits precision, RandBET ensures low RErr. This enables energy savings from *both* low-voltage operation *and* low precision quantization.

6 CONCLUSION

Overall, the proposed combination of **robust quantization**, **weight clipping** and **random bit error training (RandBET)** enables robust low-voltage operation *without* requiring expensive error correcting codes (ECCs) or other circuit techniques [6, 37]. Furthermore, our analysis applies both to DRAM, commonly off-chip, and SRAM, usually used as scratchpads on-chip of DNN accelerators. Compared to co-design approaches [23, 25], we do not require expert knowledge or expensive profiling infrastructure. Moreover, RandBET improves over these approaches by generalizing across chips *and* voltages. We also show that robust fixed-point quantization *only with* weight clipping can provide reasonable robustness. Finally, to further reduce energy consumption, our approach also enables low-voltage operation at low precisions, e.g., 4 bits or lower.

REFERENCES

[1] [n.d.]. Nervana Neural Network Distiller. https://github.com/nervanasystems/distiller.

[2] [n.d.]. NVIDIA TensorRT. https://developer.nvidia.com/tensorrt.

[3] Dan Alistarh, Jerry Li, Ryota Tomioka, and Milan Vojnovic. 2016. QSGD: Randomized Quantization for Communication-Optimal Stochastic Gradient Descent. *arXiv.org* abs/1610.02132 (2016).

[4] Milad Alizadeh, Arash Behboodi, Mart van Baalen, Christos Louizos, Tijmen Blankevoort, and Max Welling. 2020. Gradient ℓ_1 Regularization for Quantization Robustness. In *ICLR*.

[5] Ron Banner, Yury Nahshan, and Daniel Soudry. 2019. Post training 4-bit quantization of convolutional networks for rapid-deployment. In *NeurIPS*.

[6] Nandhini Chandramoorthy, Karthik Swaminathan, Martin Cochet, Arun Paidimarri, Schuyler Eldridge, Rajiv V. Joshi, Matthew M. Ziegler, Alper Buyuktosunoglu, and Pradip Bose. 2019. Resilient Low Voltage Accelerators for High Energy Efficiency. In *HPCA*.

[7] Kevin K. Chang, Abdullah Giray Yaalikçi, Saugata Ghose, Aditya Agrawal, Niladrish Chatterjee, Abhijith Kashyap, Donghyuk Lee, Mike O'Connor, Hasan Hassan, and Onur Mutlu. 2017. Understanding Reduced-Voltage Operation in Modern DRAM Devices: Experimental Characterization, Analysis, and Mechanisms. *PACM on Measurement and Analysis of Computing Systems* 1, 1 (2017).

[8] Nicholas Cheney, Martin Schrimpf, and Gabriel Kreiman. 2017. On the Robustness of Convolutional Neural Networks to Internal Architecture and Weight Perturbations. *arXiv.org* abs/1703.08245 (2017).

[9] Ching-Tai Chiu, Kishan Mehrotra, Chilukuri K. Mohan, and Sanjay Ranka. 1994. Training Techniques to Obtain Fault-Tolerant Neural Networks. In *Annual International Symposium on Fault-Tolerant Computing*.

[10] Jungwook Choi, Zhuo Wang, Swagath Venkataramani, Pierce I-Jen Chuang, Vijayalakshmi Srinivasan, and Kailash Gopalakrishnan. 2018. PACT: Parameterized Clipping Activation for Quantized Neural Networks. *arXiv.org* abs/1805.06085 (2018).

[11] Matthieu Courbariaux, Yoshua Bengio, and Jean-Pierre David. 2015. BinaryConnect: Training Deep Neural Networks with binary weights during propagations. In *NeurIPS*.

[12] Jacob Dumford and Walter J. Scheirer. 2018. Backdooring Convolutional Neural Networks via Targeted Weight Perturbations. *arXiv.org* abs/1812.03128 (2018).

[13] Shrikanth Ganapathy, John Kalamatianos, Bradford M. Beckmann, Steven Raasch, and Lukasz G. Szafaryn. 2019. Killi: Runtime Fault Classification to Deploy Low Voltage Caches without MBIST. In *HPCA*.

[14] Shrikanth Ganapathy, John Kalamatianos, Keith Kasprak, and Steven Raasch. 2017. On Characterizing Near-Threshold SRAM Failures in FinFET Technology. In *DAC*.

[15] Alexander Goncharenko, Andrey Denisov, Sergey Alyamkin, and Evgeny Terentev. 2018. Fast Adjustable Threshold For Uniform Neural Network Quantization. *arXiv.org* abs/1812.07872 (2018).

[16] Yunhui Guo. 2018. A Survey on Methods and Theories of Quantized Neural Networks. *arXiv.org* abs/1808.04752 (2018).

[17] Zheng Guo, Andrew Carlson, Liang-Teck Pang, Kenneth Duong, Tsu-Jae King Liu, and Borivoje Nikolic. 2009. Large-Scale SRAM Variability Characterization in 45 nm CMOS. *JSSC* 44, 11 (2009).

[18] Seyyed Hossein HasanPour, Mohammad Rouhani, Mohsen Fayyaz, and Mohammad Sabokrou. 2016. Lets keep it simple, Using simple architectures to outperform deeper and more complex architectures. *arXiv.org* abs/1608.06037 (2016).

[19] Zhezhi He, Adnan Siraj Rakin, Jingtao Li, Chaitali Chakrabarti, and Deliang Fan. 2020. Defending and Harnessing the Bit-Flip based Adversarial Weight Attack. In *CVPR*.

[20] Itay Hubara, Matthieu Courbariaux, Daniel Soudry, Ran El-Yaniv, and Yoshua Bengio. 2017. Quantized Neural Networks: Training Neural Networks with Low Precision Weights and Activations. *JMLR* 18 (2017).

[21] Benoit Jacob, Skirmantas Kligys, Bo Chen, Menglong Zhu, Matthew Tang, Andrew G. Howard, Hartwig Adam, and Dmitry Kalenichenko. 2018. Quantization and Training of Neural Networks for Efficient Integer-Arithmetic-Only Inference. In *CVPR*.

[22] Yujie Ji, Xinyang Zhang, Shouling Ji, Xiapu Luo, and Ting Wang. 2018. Model Reuse Attacks on Deep Learning Systems. In *CCS*.

[23] Sung Kim, Patrick Howe, Thierry Moreau, Armin Alaghi, Luis Ceze, and Visvesh Sathe. 2018. MATIC: Learning around errors for efficient low-voltage neural network accelerators. In *DATE*.

[24] Yoongu Kim, Ross Daly, Jeremie Kim, Chris Fallin, Ji-Hye Lee, Donghyuk Lee, Chris Wilkerson, Konrad Lai, and Onur Mutlu. 2014. Flipping bits in memory without accessing them: An experimental study of DRAM disturbance errors. In *ISCA*.

[25] Skanda Koppula, Lois Orosa, Abdullah Giray Yaglikçi, Roknoddin Azizi, Taha Shahroodi, Konstantinos Kanellopoulos, and Onur Mutlu. 2019. EDEN: Enabling Energy-Efficient, High-Performance Deep Neural Network Inference Using Approximate DRAM. In *MICRO*. 166–181.

[26] Raghuraman Krishnamoorthi. 2018. Quantizing deep convolutional networks for efficient inference: A whitepaper. *arXiv.org* abs/1806.08342 (2018).

[27] Alex Krizhevsky. 2009. *Learning Multiple Layers of Features from Tiny Images*. Technical Report.

[28] Hao Li, Soham De, Zheng Xu, Christoph Studer, Hanan Samet, and Tom Goldstein. 2017. Training Quantized Nets: A Deeper Understanding. In *NeurIPS*, Isabelle Guyon, Ulrike von Luxburg, Samy Bengio, Hanna M. Wallach, Rob Fergus, S. V. N. Vishwanathan, and Roman Garnett (Eds.).

[29] Darryl Dexu Lin, Sachin S. Talathi, and V. Sreekanth Annapureddy. 2016. Fixed Point Quantization of Deep Convolutional Networks. In *ICML*.

[30] Paul Merolla, Rathinakumar Appuswamy, John V. Arthur, Steven K. Esser, and Dharmendra S. Modha. 2016. Deep neural networks are robust to weight binarization and other non-linear distortions. *arXiv.org* abs/1606.01981 (2016).

[31] Kit Murdock, David Oswald, Flavio D. Garcia, Jo Van Bulck, Daniel Gruss, and Frank Piessens. 2020. Plundervolt: Software-based Fault Injection Attacks against Intel SGX. In *SP*.

[32] Abhishek Murthy, Himel Das, and Md. Ariful Islam. 2019. Robustness of Neural Networks to Parameter Quantization. *arXiv.org* abs/1903.10672 (2019).

[33] Chalapathy Neti, Michael H. Schneider, and Eric D. Young. 1992. Maximally fault tolerant neural networks. *TNN* 3, 1 (1992), 14–23.

[34] Eunhyeok Park, Dongyoung Kim, and Sungjoo Yoo. 2018. Energy-Efficient Neural Network Accelerator Based on Outlier-Aware Low-Precision Computation. In *ISCA*.

[35] Adnan Siraj Rakin, Zhezhi He, and Deliang Fan. 2019. Bit-Flip Attack: Crushing Neural Network With Progressive Bit Search. In *ICCV*.

[36] Mohammad Rastegari, Vicente Ordonez, Joseph Redmon, and Ali Farhadi. 2016. XNOR-Net: ImageNet Classification Using Binary Convolutional Neural Networks. In *ECCV*.

[37] Brandon Reagen, Paul N. Whatmough, Robert Adolf, Saketh Rama, Hyunkwang Lee, Sae Kyu Lee, José Miguel Hernández-Lobato, Gu-Yeon Wei, and David M. Brooks. 2016. Minerva: Enabling Low-Power, Highly-Accurate Deep Neural Network Accelerators. In *ISCA*.

[38] Adrian Sampson, Werner Dietl, Emily Fortuna, Danushen Gnanapragasam, Luis Ceze, and Dan Grossman. 2011. EnerJ: Approximate Data Types for Safe and General Low-Power Computation. *SIGPLAN Not.* 46, 6 (2011).

[39] Frank Seide, Hao Fu, Jasha Droppo, Gang Li, and Dong Yu. 2014. 1-bit stochastic gradient descent and its application to data-parallel distributed training of speech DNNs. In *INTERSPEECH*.

[40] Sungho Shin, Yoonho Boo, and Wonyong Sung. 2017. Fixed-point optimization of deep neural networks with adaptive step size retraining. In *ICASSP*.

[41] Gopalakrishnan Srinivasan, Parami Wijesinghe, Syed Shakib Sarwar, Akhilesh Jaiswal, and Kaushik Roy. 2016. Significance driven hybrid 8T-6T SRAM for energy-efficient synaptic storage in artificial neural networks. In *DATE*.

[42] Wonyong Sung, Sungho Shin, and Kyuyeon Hwang. 2015. Resiliency of Deep Neural Networks under Quantization. *arXiv.org* abs/1511.06488 (2015).

[43] Vivienne Sze, Yu-Hsin Chen, Tien-Ju Yang, and Joel S. Emer. 2017. Efficient Processing of Deep Neural Networks: A Tutorial and Survey. *IEEE* 105, 12 (2017).

[44] Adrian Tang, Simha Sethumadhavan, and Salvatore J. Stolfo. 2017. CLKSCREW: Exposing the Perils of Security-Oblivious Energy Management. In *USENIX*.

[45] Tsui-Wei Weng, Pu Zhao, Sijia Liu, Pin-Yu Chen, Xue Lin, and Luca Daniel. 2020. Towards Certificated Model Robustness Against Weight Perturbations. In *AAAI*.

[46] Sergey Zagoruyko and Nikos Komodakis. 2016. Wide Residual Networks. In *BMVC*.

[47] Shuchang Zhou, Zekun Ni, Xinyu Zhou, He Wen, Yuxin Wu, and Yuheng Zou. 2016. DoReFa-Net: Training Low Bitwidth Convolutional Neural Networks with Low Bitwidth Gradients. *arXiv.org* abs/1606.06160 (2016).

[48] Bohan Zhuang, Chunhua Shen, Mingkui Tan, Lingqiao Liu, and Ian D. Reid. 2018. Towards Effective Low-Bitwidth Convolutional Neural Networks. In *CVPR*.