Implementing Binarized Neural Networks with Magnetoresistive RAM without Error Correction

Tifenn Hirtzlin, Bogdan Penkovsky, Jacques-Olivier Klein, Nicolas Locatelli, Adrien Vincent, Marc Bocquet, Jean-Michel Portal, Damien Querlioz

Implementing Binarized Neural Networks with Magnetoresistive RAM without Error Correction

Tifenn Hirtzlin*, Bogdan Penkovsky*, Jacques-Olivier Klein*, Nicolas Locatelli*,
Adrien F. Vincent‡, Marc Bocquet†, Jean-Michel Portal† and Damien Querlioz*
*Centre de Nanosciences et de Nanotechnologies, CNRS, Univ Paris-Sud, Université Paris-Saclay, 91129 Palaiseau, France
Email: damien.querlioz@u-psud.fr
†Institut Matériaux Microélectronique Nanosciences de Provence, Univ. Aix-Marseille et Toulon, CNRS, France
‡Laboratoire de l'Intégration du Matériau au Système, Univ. Bordeaux, Bordeaux INP, CNRS, Talence, France.

***Abstract*—One of the most exciting applications of Spin Torque Magnetoresistive Random Access Memory (ST-MRAM) is the in-memory implementation of deep neural networks, which could allow improving the energy efficiency of Artificial Intelligence by orders of magnitude with regards to its implementation on computers and graphics cards. In particular, ST-MRAM could be ideal for implementing Binarized Neural Networks (BNNs), a type of deep neural networks discovered in 2016, which can achieve state-of-the-art performance with a highly reduced memory footprint with regards to conventional artificial intelligence approaches. The challenge of ST-MRAM, however, is that it is prone to write errors and usually requires the use of error correction. In this work, we show that these bit errors can be tolerated by BNNs to an outstanding level, based on examples of image recognition tasks (MNIST, CIFAR-10 and ImageNet): bit error rates of ST-MRAM up to 0.1% have little impact on recognition accuracy. The requirements for ST-MRAM are therefore considerably relaxed for BNNs with regards to traditional applications. By consequence, we show that for BNNs, ST-MRAMs can be programmed with weak (low-energy) programming conditions, without error correcting codes. We show that this result can allow the use of low energy and low area ST-MRAM cells, and show that the energy savings at the system level can reach a factor two.**

I. Introduction

Spin Torque Magnetoresistive Random Access Memory (ST-MRAM) is currently emerging as a leading technology for embedded memory in microcontroller units [1]–[3], as well as for standalone memory [4]. ST-MRAM indeed provides a compact fast and non volatile memory, which is fully embeddable in modern CMOS, and features outstanding endurance. This unique set of features makes ST-MRAM also adapted for alternative non von Neumann computing schemes, which tightly integrate logic and memory [5]. In particular, this approach can be especially adapted for implementing hardware deep neural networks. These algorithms have become the flagship approach of modern artificial intelligence [6], but they possess a high power consumption, which is mainly caused by the von Neumann bottleneck [7].

Multiple proposals have been made to implement deep neural networks with ST-MRAM using concepts of in-memory or near-memory computing [8]–[12]. Similar ideas have been proposed for other types of resistive memory [7], [13]–[15]. The benefit of ST-MRAM is its outstanding endurance [1], [2]. However, such proposals come with an important challenge: ST-MRAMs feature bit errors. Commercial processes typically target a bit error rate (BER) of 10^{-6} [1], [2]. Such memories are therefore meant to be used with error correcting codes, as is the case of other types of resistive memories [13], [16]. Unfortunately, the bit errors in ST-MRAMs are to a large extent intrinsic, as they can originate in the basic physics of spin torque-based magnetization switching [17], [18]. They will thus not disappear, even when the technology progresses.

In this paper, we look at the special case of Binarized Neural Networks (BNNs) [19], [20]. Theses networks have been proposed recently as a highly simplified form of deep neural networks, as both their neurons and synapses assume binary values during inference. They therefore function with reduced memory requirements with regards to standard neural networks, and use extremely simple arithmetic operations (no multiplication). They can nevertheless approach state-of-the-art performance on vision tasks on datasets such as CIFAR-10 or ImageNet [19]–[21]. Due to their simplicity and reduced resource requirements, BNNs are particularly adapted to in-memory hardware implementation [14], [22]. In this work, we look at bit error rate impact on BNNs designed with ST-MRAMs.

This works extends our prior work on the implementation of BNNs with hafnium oxide-based Resistive RAM [22], [23] to the case of ST-MRAMs by taking into account the particular mechanism of intrinsic BER in these devices. Additionally, our prior work relied on analysis of relatively simple machine learning tasks (MNIST and CIFAR-10), here we add the analysis of a much more realistic task (ImageNet).

The paper makes the following contributions:

- We simulate BNNs incorporating bit errors on the weights, and show that BERs up to 10^{-3} can be tolerated. For the first time, the resilience of BNNs is demonstrated on the large scale ImageNet image recognition task. We deduce that ST-MRAMs can be used directly without ECC (section II).
- We highlight that due to this extreme tolerance we can even reduce the programming energy of ST-MRAMs with regards to conventional applications. Based on a physical analysis, we show that a factor two in programming energy may be saved without impact on BNN accuracy (section III).

II. Binarized Neural Networks Can Tolerate ST-MRAM Bit Errors without ECC

Binarized Neural Networks are a class of neural networks, in which the synaptic weights and the neuronal states, can only take two values: $+1$ or -1 (whereas they assume real values in conventional neural networks). Therefore, the equation for neuronal activation A in a conventional neural network

$$A = f\left(\sum_i W_i X_i\right), \tag{1}$$

(X_i are inputs of the neuron, W_i the synaptic weights and f the neuronal activation function) simplifies into

$$A = \text{sign}\left(\text{POPCOUNT}_i\left(XNOR\left(W_i, X_i\right)\right) - T\right). \tag{2}$$

T is the threshold of the neuron and is learned. POPCOUNT counts the number of $1s$, and sign is the sign function.

It should be noted that the synapses also feature real weights during training. The binarized weights, which are equal to the sign of the real weights, are used in the equations of forward and backward passes, but the real weights are updated as a result of the learning rule [19]. As the real weights can be forgotten once the learning process is finished, BNNs are outstanding candidates for hardware implementation of neural network inference:

- Area and energy expensive multiplications in eq. (1) are replaced by one-bit exclusive NOR (XNOR) operations.
- Neurons and synapses require a single bit of memory.

ASIC implementations of BNNs have been proposed using purely CMOS solutions [24]. Nevertheless, the most attractive implementations propose using RRAMs or ST-MRAMs for implementing the synaptic weights, in architectures that tightly integrate logic and memory [8], [10], [11], [14], [22], [25]–[27].

However, a major challenge of RRAMs and ST-MRAMs is the presence of bit errors. In the case of ST-MRAM, this issue is traditionally solved by using ECC [1]–[3] or special programming strategies [28]. In this work, we look at the possibility to use ST-MRAMs directly to store the binarized synaptic weights, without relying on any of these techniques.

For this purpose, we perform simulations of BNNs, with bit errors added artificially. We consider three type of neural networks (Fig. 1):

- A shallow fully connected neural network with two 1024 neurons hidden layers, trained on the canonical task of MNIST handwritten digit classification.
- A deep convolutional network trained on the CIFAR-10 image recognition task. CIFAR-10 consists in recognizing 32×32 color images out of ten classes of various animals and vehicles. Our neural network features 6 convolutional layers with number of filters 384, 384, 384, 768, 768 and 1536. All convolutional layers use a kernel size of 3×3, and a stride of one. These layers are topped by three fully connected layers with 1024, 1024 and 10 neurons.
- The classic AlexNet deep convolutional neural network trained on the ImageNet recognition task [29]. ImageNet

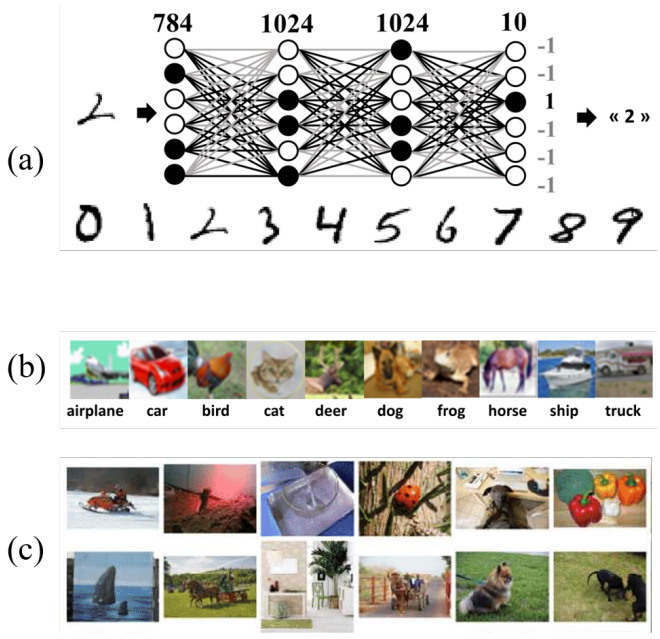

Fig. 1. (a) Fully connected neural network used for the MNIST task, and example of MNIST dataset images. (b) Examples of CIFAR-10 dataset images. (c) Examples of ImageNet dataset examples.

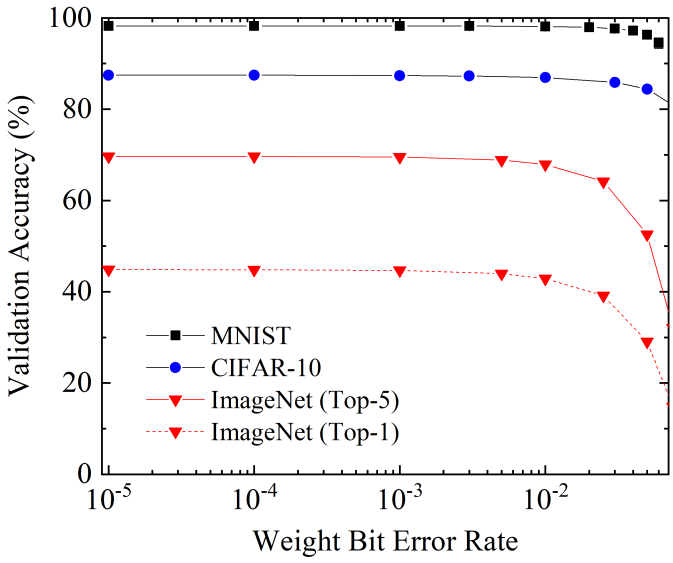

Fig. 2. Recognition rate on the validation dataset of the fully connected neural network for MNIST, the convolutional neural network for CIFAR10, and AlexNet for ImageNet (Top-5 and Top-1) accuracies as a function of the bit error rate over the weights during inference. Each experiment was repeated five times, the mean recognition rate is presented.

consists in recognizing 128×128 color images out of 1000 classes, and is a considerably harder task than CIFAR-10.

Training of the MNIST neural network is done with Python using NumPy libraries. For CIFAR-10, training is done using the TensorFlow framework. For ImageNet, we use pretrained weights [30] obtained with the Caffe framework. In all cases, a softmax function and cross-entropy loss are used during training. Adam optimizer was used for better convergence, and dropout was used in the MNIST and CIFAR-10 cases to mitigate overfitting. For inference, softmax is no longer needed and replaced by a hardmax function.

Fig. 2 shows the results of these simulations. It plots the validation accuracy obtained for the three networks, as a function of the weight BER. Each simulation was repeated five time, and the results were averaged. In the case of ImageNet, we present both the Top-5 and Top-1 accuracies.

Interestingly, at BER up to 10^{-4}, no effect on neural network accuracy is seen: the network performs just as well as when no error is present. As ST-MRAMs can achieve BERs of 10^{-6}, [1], [2], this shows that they can be used directly, without ECC, for image recognition tasks. BERs of 10^{-3} also have practically no effect (the Top-5 accuracy on ImageNet is reduced from 69.7% to 69.5%). At a BER of 0.01, the performance starts to decrease significantly. The more difficult the task, the most substantial the performance reduction: MNIST accuracy is reduced from 98.3% to 98.1%, CIFAR-10 accuracy is reduced from 87.65% to 86.9%, ImageNet Top-5 accuracy is reduced from 69.7% to 67.9%. These results highlight the inherent error correction capability of neural network, which originates in their highly redundant nature. It should be noted that these results could be further enhanced if one knows in advance the BER at inference time, by incorporating weight errors during training [23].

III. Saving Energy by Allowing More ST-MRAM Bit Errors

The results from section II suggest that ST-MRAMs can be used in a BNN with a BER much higher than the usually targeted 10^{-6}, without error correction. We now investigate the benefit that increasing the BERs can have in terms of energy consumption, based on physical modeling of ST-MRAM cells.

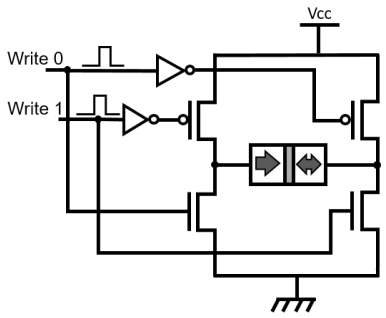

Fig. 3. Programming circuit for magnetic tunnel junctions.

ST-MRAMs are based on magnetic tunnel junctions (MTJs), nanodevices composed of stack of various materials. These materials implement two nanomagnets (reference and free magnet), separated by an oxide tunnel barrier. The reference and free magnet may have either parallel or anti-parallel magnetizations, which constitutes the memory state of the device. Due to the tunnel magnetoresistance (TMR) effect, a magnetic tunnel junction exhibits different resistance values in the parallel (R_{P}) or antiparallel (R_{AP}) states. The TMR parameter is defined by:

$$\mathrm{TMR} = \frac{R_{\mathrm{AP}} - R_{\mathrm{P}}}{R_{\mathrm{P}}}. \tag{3}$$

Due to the spin transfer torque effect, the state of a magnetic tunnel junction can be switched by applying positive or negative electrical currents to it. Unfortunately, the physics of spin torque switching is inherently stochastic [18], [31]–[35], which is a source of intrinsic bit errors. The physics of this phenomenon is now well understood. As a reasonable approximation, spin torque switching may be modeled by Sun's model [36]. The mean switching time is given by:

$$\tau = \tau_0 \frac{V_{\mathrm{c}}}{V - V_{\mathrm{c}}}, \tag{4}$$

where τ_0 is a characteristic time, and V_{c} is called the critical voltage of the junction. This equation is valid for voltages significantly higher than V_{c}.

Many works have evaluated the statistical distribution of the actual switching time of the junctions. Here, we use the model proposed in [18]. The distribution of switching time t is given by the gamma distribution:

$$f_{\mathrm{sw}}(t;\, k, \theta) = \frac{t^{k-1} \exp\left(-\frac{t}{\theta}\right)}{\Gamma(k)\theta^k}, \tag{5}$$

For the skewness k, we use the value suggested in [18] $k = 16.0$ (corresponding to a relative standard deviation of 0.25), and we take $\theta = \tau/k$.

We assume that the magnetic tunnel junctions are programmed with the standard circuit of Fig. 3, which allows applying positive or negative currents to the junction. In this work, we propose to reduce the MTJ programming time in order to reduce the required programming energy, while increasing the BER. This has been identified as the most efficient strategy to use MTJs as "approximate memory" in [9], and here we look at the impact of this strategy on BNNs.

In all results, the circuit is simulated using the Cadence Spectre simulator, with the design kit of a $28\,\mathrm{nm}$ commercial technology. The magnetic tunnel junction is modeled with a Verilog-A model described in [9], parametrized with values inspired by a $32\,\mathrm{nm}$ technology using perpendicular magnetization anisotropy [37], [38]. The diameter of the junctions is $32\,\mathrm{nm}$, the thickness of the free layer is $1.3\,\mathrm{nm}$, its saturation magnetization is $1.58\,\mathrm{T}/\mu_0$. The resistance area (RA) product of the junction is $4\,\Omega{\cdot}\mu\mathrm{m}^2$, its TMR value is 150%, and its critical voltage V_{c} is $190\,\mathrm{mV}$. The junctions are programmed with a voltage of $2.0 \times V_{\mathrm{c}}$. In Monte Carlo simulations, we

consider mismatch and process variations of the transistors, as well as typical variations of MTJ parameters (5% relative standard deviation on TMR and R_P [31]).

Fig. 4 presents the correspondence between BER and programming energy using the circuit of Fig. 3. Two cases are evaluated: considering only the intrinsic stochastic effects of the MTJs (black curve), and adding the effects of transistor and MTJ variability (red curve). This curve confirms the existence of an interplay between programming energy and BER.

Figs. 5 and 6 associate the results of Figs. 2 and 4 to highlight the interplay between programming energy and BNN accuracy for the CIFAR-10 task (Fig. 5) and ImageNet (Fig. 6). The points with highest programming energy correspond to a BER of 10^{-6}.

We see that on both tasks the programming energy can be reduced approximately by a factor two with no impact on BNN accuracy. This result is not only useful in terms of energy efficiency. As the area of ST-MRAMs cells is dominated by transistor and not MTJs, this result means that we may be able to use lower area cells for BNNs (with lesser drive current) than for conventional memory applications. This strategy of reducing programming energy if one accepts increased BER is also applicable to other memory technology such as oxide-based RRAM [23], even if the underlying physics of these devices differ considerably from ST-MRAMs.

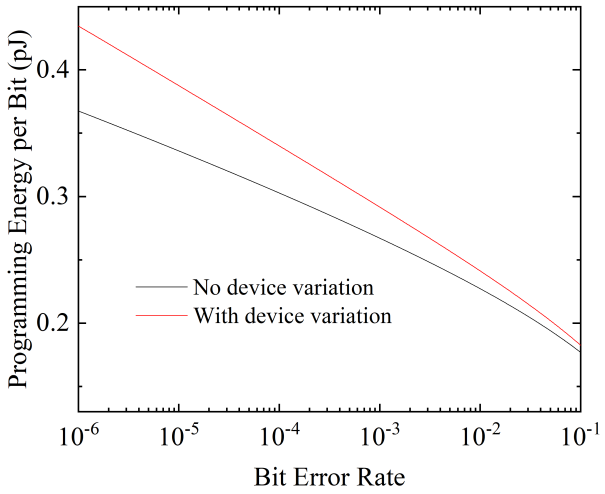

Fig. 4. ST-MRAM programming energy per bit as a function of corresponding BER, ignoring (black line) or taking into account (red line) CMOS and MTJ device variations. Results obtained by Monte Carlo simulation.

IV. Conclusion

In this work, we saw that Binarized Neural Networks can tolerate high bit error rates (up to 10^{-3}) on their binarized weights, even on difficult tasks such as ImageNet. This makes them especially appropriate for in-memory hardware implementation with emerging memories such as ST-MRAMs. Not only can we use ST-MRAMs without the use explicit error correcting codes, but we can also program ST-MRAMs with reduced energy. Based on physical modeling of ST-MRAMs,

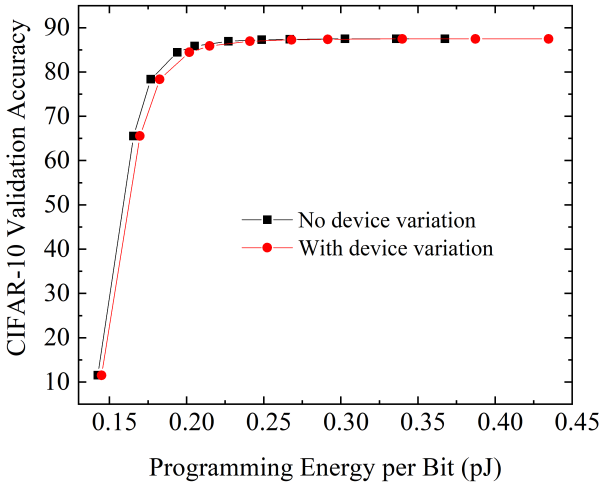

Fig. 5. Validation accuracy of the convolutional binarized neural network trained on the CIFAR-10 dataset as a function of the ST-MRAM programming energy per bit, ignoring (black line) or taking into account (red line) CMOS and MTJ device variations.

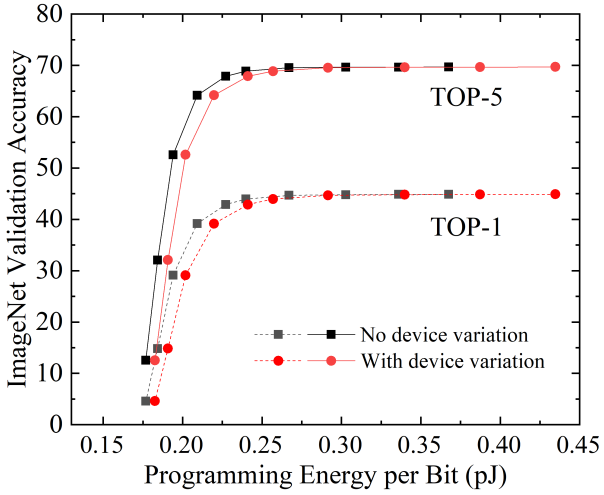

Fig. 6. Top-1 and Top5 validation accuracy of binarized AlexNet trained on the ImageNet dataset as a function of the ST-MRAM programming energy per bit, ignoring (black line) or taking into account (red line) CMOS and MTJ device variations.

we showed that a factor two in programming energy can be saved.

These results highlight that neural networks differ from more traditional forms of computation. In a way that is reminiscent of brains, which function with imperfect and unreliable basic devices, perfect device reliability might not be necessary for hardware neural networks.

Acknowledgment

This work was supported by the Agence Nationale de la Recherche grant NEURONIC (ANR-18-CE24-0009) and the European Research Council Grant NANOINFER (715872).

References

[1] O. Golonzka, J.-G. Alzate, U. Arslan, M. Bohr, P. Bai, J. Brockman, B. Buford, C. Connor, N. Das, B. Doyle *et al.*, "Mram as embedded

non-volatile memory solution for 22ffl finfet technology," in *2018 IEEE International Electron Devices Meeting (IEDM)*. IEEE, 2018, pp. 18–1.

[2] Y. K. Lee *et al.*, "Embedded stt-mram in 28-nm fdsoi logic process for industrial mcu/iot application," in *IEEE Symp. VLSI Technol.* IEEE, 2018, pp. 181–182.

[3] Y.-C. Shih *et al.*, "Logic process compatible 40-nm 16-mb, embedded perpendicular-mram with hybrid-resistance reference, sub-μa sensing resolution, and 17.5-ns read access time," *IEEE Journal of Solid-State Circuits*, 2019.

[4] T. Andre, S. M. Alam, D. Gogl, J. Barkatullah, J. Qi, H. Lin, X. Zhang, W. Meadows, F. Neumeyer, G. Viot *et al.*, "St-mram fundamentals, challenges, and outlook," in *2017 IEEE International Memory Workshop (IMW)*. IEEE, 2017, pp. 1–4.

[5] W. Zhao, M. Moreau, E. Deng, Y. Zhang, J.-M. Portal, J.-O. Klein, M. Bocquet, H. Aziza, D. Deleruyelle, C. Muller *et al.*, "Synchronous non-volatile logic gate design based on resistive switching memories," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 61, no. 2, pp. 443–454, 2014.

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

[7] Editorial, "Big data needs a hardware revolution," *Nature*, vol. 554, no. 7691, p. 145, Feb. 2018.

[8] M. Natsui, T. Chiba, and T. Hanyu, "Design of mtj-based nonvolatile logic gates for quantized neural networks," *Microelectronics journal*, vol. 82, pp. 13–21, 2018.

[9] N. Locatelli, A. F. Vincent, and D. Querlioz, "Use of magnetoresistive random-access memory as approximate memory for training neural networks," in *2018 25th IEEE International Conference on Electronics, Circuits and Systems (ICECS)*. IEEE, 2018, pp. 553–556.

[10] Y. Pan, P. Ouyang, Y. Zhao, W. Kang, S. Yin, Y. Zhang, W. Zhao, and S. Wei, "A multilevel cell stt-mram-based computing in-memory accelerator for binary convolutional neural network," *IEEE Transactions on Magnetics*, no. 99, pp. 1–5, 2018.

[11] Z. He, S. Angizi, and D. Fan, "Accelerating low bit-width deep convolution neural network in mram," in *2018 IEEE Computer Society Annual Symposium on VLSI (ISVLSI)*. IEEE, 2018, pp. 533–538.

[12] S. G. Ramasubramanian, R. Venkatesan, M. Sharad, K. Roy, and A. Raghunathan, in *Proc. ISLPED*. ACM, 2014, pp. 15–20.

[13] D. Ielmini and H.-S. P. Wong, "In-memory computing with resistive switching devices," *Nature Electronics*, vol. 1, no. 6, p. 333, 2018.

[14] S. Yu, "Neuro-inspired computing with emerging nonvolatile memorys," *Proc. IEEE*, vol. 106, no. 2, pp. 260–285, 2018.

[15] D. Querlioz, O. Bichler, A. F. Vincent, and C. Gamrat, "Bioinspired programming of memory devices for implementing an inference engine," *Proc. IEEE*, vol. 103, no. 8, pp. 1398–1416, 2015.

[16] D. R. B. Ly, A. Grossi, C. Fenouillet-Beranger, E. Nowak, D. Querlioz, and E. Vianello, "Role of synaptic variability in resistive memory-based spiking neural networks with unsupervised learning," *J. Phys. D: Applied Physics*, 2018.

[17] Z. Diao, Z. Li, S. Wang, Y. Ding, A. Panchula, E. Chen, L.-C. Wang, and Yiming Huai, "Spin-transfer torque switching in magnetic tunnel junctions and spin-transfer torque random access memory," *J. Phys. Cond. Mat.*, vol. 19, no. 16, p. 165209, 2007.

[18] A. F. Vincent, N. Locatelli, J.-O. Klein, W. Zhao, S. Galdin-Retailleau, and D. Querlioz, "Analytical Macrospin Modeling of the Stochastic Switching Time of Spin-Transfer Torque Devices," *IEEE Trans. Electron Dev.*, vol. 62, no. 1, pp. 164–170, Jan. 2015.

[19] M. Courbariaux, I. Hubara, D. Soudry, R. El-Yaniv, and Y. Bengio, "Binarized neural networks: Training deep neural networks with weights and activations constrained to+ 1 or-1," *arXiv preprint arXiv:1602.02830*, 2016.

[20] M. Rastegari, V. Ordonez, J. Redmon, and A. Farhadi, "Xnor-net: Imagenet classification using binary convolutional neural networks," in *Proc. ECCV*. Springer, 2016, pp. 525–542.

[21] X. Lin, C. Zhao, and W. Pan, "Towards accurate binary convolutional neural network," in *Proc. NIPS*, 2017, pp. 345–353.

[22] M. Bocquet, T. Hirztlin, J.-O. Klein, E. Nowak, E. Vianello, J.-M. Portal, and D. Querlioz, "In-memory and error-immune differential rram implementation of binarized deep neural networks," in *IEDM Tech. Dig.* IEEE, 2018, p. 20.6.1.

[23] T. Hirtzlin, M. Bocquet, J.-O. Klein, E. Nowak, E. Vianello, J.-M. Portal, and D. Querlioz, "Outstanding bit error tolerance of resistive ram-based binarized neural networks," *arXiv preprint arXiv:1904.03652*, 2019.

[24] K. Ando *et al.*, "Brein memory: A 13-layer 4.2 k neuron/0.8 m synapse binary/ternary reconfigurable in-memory deep neural network accelerator in 65 nm cmos," in *Proc. VLSI Symp. on Circuits*. IEEE, 2017, p. C24.

[25] X. Sun, X. Peng, P.-Y. Chen, R. Liu, J.-s. Seo, and S. Yu, "Fully parallel rram synaptic array for implementing binary neural network with (+ 1,- 1) weights and (+ 1, 0) neurons," in *Proc. ASP-DAC*. IEEE Press, 2018, pp. 574–579.

[26] X. Sun, S. Yin, X. Peng, R. Liu, J.-s. Seo, and S. Yu, "Xnor-rram: A scalable and parallel resistive synaptic architecture for binary neural networks," *algorithms*, vol. 2, p. 3, 2018.

[27] T. Tang, L. Xia, B. Li, Y. Wang, and H. Yang, "Binary convolutional neural network on rram," in *Proc. ASP-DAC*. IEEE, 2017, pp. 782–787.

[28] Y. Lakys, W. S. Zhao, T. Devolder, Y. Zhang, J.-O. Klein, D. Ravelosona, and C. Chappert, "Self-enabled error-free switching circuit for spin transfer torque mram and logic," *IEEE Transactions on Magnetics*, vol. 48, no. 9, pp. 2403–2406, 2012.

[29] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Proc. NIPS*, 2012, pp. 1097–1105.

[30] J. Yue, "Xnor-net-pytorch," https://github.com/jiecaoyu/XNOR-Net-PyTorch, 2017.

[31] D. Worledge *et al.*, "Switching distributions and write reliability of perpendicular spin torque MRAM," in *IEDM Tech. Dig.*, 2010, pp. 12.5.1–12.5.4, 00018.

[32] Z. Wang, Y. Zhou, J. Zhang, and Y. Huai, "Bit error rate investigation of spin-transfer-switched magnetic tunnel junctions," *Appl. Phys. Lett.*, vol. 101, no. 14, p. 142406, Oct. 2012.

[33] A. F. Vincent *et al.*, "Spin-transfer torque magnetic memory as a stochastic memristive synapse for neuromorphic systems," *IEEE T. Biomed. Circ. S.*, vol. 9, no. 2, pp. 166–174, 2015.

[34] A. Ranjan, S. Venkataramani, X. Fong, K. Roy, and A. Raghunathan, "Approximate storage for energy efficient spintronic memories," in *Proc. DAC*, Jun. 2015, pp. 1–6, 00002.

[35] F. Sampaio *et al.*, "Approximation-aware Multi-Level Cells STT-RAM cache architecture," in *Proc. CASES*, 2015, pp. 79–88.

[36] J. Z. Sun, "Spin-current interaction with a monodomain magnetic body: A model study," *Phys. Rev. B*, vol. 62, no. 1, pp. 570–578, 2000, 00656.

[37] Khvalkovskiy *et al.*, "Basic principles of STT-MRAM cell operation in memory arrays," *J. Phys. D*, vol. 46, no. 7, p. 074001, Feb. 2013.

[38] K. C. Chun *et al.*, "A Scaling Roadmap and Performance Evaluation of In-Plane and Perpendicular MTJ Based STT-MRAMs for High-Density Cache Memory," *IEEE JSSC*, vol. 48, no. 2, pp. 598–610, 2013.