In-Memory Resistive RAM Implementation of Binarized Neural Networks for Medical Applications

Bogdan Penkovsky*, Marc Bocquet†, Tifenn Hirtzlin*, Jacques-Olivier Klein*,
Etienne Nowak‡, Elisa Vianello‡, Jean-Michel Portal† and Damien Querlioz*
*C2N, Université Paris-Saclay, CNRS, Palaiseau, France. Email: damien.querlioz@c2n.upsaclay.fr
†IM2NP, Aix-Marseille Université, Université de Toulon, CNRS, Marseille, France
‡CEA, LETI, Grenoble, France.

***Abstract*—The advent of deep learning has considerably accelerated machine learning development. The deployment of deep neural networks at the edge is however limited by their high memory and energy consumption requirements. With new memory technology available, emerging Binarized Neural Networks (BNNs) are promising to reduce the energy impact of the forthcoming machine learning hardware generation, enabling machine learning on the edge devices and avoiding data transfer over the network. In this work, after presenting our implementation employing a hybrid CMOS - hafnium oxide resistive memory technology, we suggest strategies to apply BNNs to biomedical signals such as electrocardiography and electroencephalography, keeping accuracy level and reducing memory requirements. We investigate the memory-accuracy trade-off when binarizing whole network and binarizing solely the classifier part. We also discuss how these results translate to the edge-oriented Mobilenet V1 neural network on the Imagenet task. The final goal of this research is to enable smart autonomous healthcare devices.**

I. Introduction

With recent advances in machine learning, multiple challenging tasks are now solved by computers with human-level accuracy [1], or even better [2], and smart assistance services are available for anyone. However, those services operate in the cloud, requiring energy-expensive data transfer over the network. We envision that the quality of medical services can be substantially improved with the use of machine learning, with applications such as stroke and heart attack prevention, epileptic seizure prediction, post-hospital monitoring and rehabilitation, and brain-computer interfaces for people with disabilities. Those services should be available at the edge to ensure privacy, security, and low latency. This mobility rises new challenges: maximizing battery life and making hardware as small and lightweight as possible.

The major drain of energy in modern digital electronics, especially when performing data-intensive artificial intelligence tasks, comes from data shuffling between processing logic and memory [3]. This issue can be substantially alleviated by applying in-memory computing principles, which eliminate the von Neumann bottleneck and are especially applicable for neural network architectures. This idea is particularly relevant today, with the emergence of novel non-volatile memory technologies that are fully compatible with modern CMOS processes and appear ideal for the in-memory implementation of neural networks [4], [5]. An obvious drawback of this approach is the limited amount of on-chip memory. Indeed, when talking about in-memory computing we cannot rely on external memories, thus adhering to the amount of on-chip memory becomes critical. Unfortunately, deep neural networks often require considerable amounts of memory [6].

Significant efforts have been made to reduce the memory footprint of neural networks. Compact network design aims at reducing the number of neural network parameters and operations, while maintaining accuracy [7]–[9]. These networks still use real-valued weights and activations represented as 32-bits floating point numbers. Quantized neural networks aim at reducing memory requirement by reducing the number of bits in weights and activations [10]. Eight-bit quantization is particularly successful in applications, as it usually requires no retraining [11]. Binarized neural networks (BNNs) are the extreme evolution of quantized networks with a precision reduced to a single bit [12], [13]. Beyond weight and activation, the memory footprint can also be reduced with binary representation of the inputs using stochastic sampling [14].

In this work, we address the energy efficiency challenge by proposing an in-memory implementation of binarized neural network using emerging memories. We realize that bit errors inherent to resistive memory technologies are a challenge and introduce a two transistor - two resistor approach to mitigate the issue. We validate our approach, which is adaptable to different emerging memory technologies, using measurements on a fabricated hybrid CMOS/resistive memory chip [15], [16]. We evaluate our all-binarized convolutional neural network approach on medical time-signals such as electroencephalogram and electrocardiogram. We suggest that partial binarization may sometimes be an option to minimize the accuracy loss compared to original neural networks with real weights, while substantially reducing memory requirements. Finally, we speculate about the application of the same partial binarization strategy for generic computer vision, which could have direct

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

relevance to medical imaging.

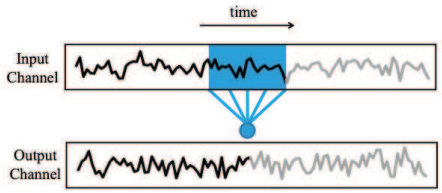

Fig. 1. Illustration of a temporal 1D convolution.

II. In-Memory Implementation of Binarized Neural Networks

A. Binarization of Neural Networks

Convolutional neural networks (CNNs) are a state-of-the-art supervised deep learning architecture [17], which consists of a feature extractor and a classifier. The feature extractor has multiple sparsely-connected convolutional layers, whereas the classifier exhibits dense, all-to-all connection topology in its layers. In both cases the elementary network unit, the artificial neuron, performs a nonlinear transformation f over a dot product between an input vector $\mathbf{x}$ and learned weights $\mathbf{w}$:

$$y = f\left(\sum_j w_j x_j + b\right), \tag{1}$$

where b is a bias term. Neural networks are trained using a gradient-descent family optimization method to minimize the error between their actual and desired outputs. Convolutional layers connection topology is based on the discrete convolution transformation between an input tensor $\mathbf{x}$ and a sliding window $\mathbf{w}$, called convolution kernel. A discrete convolution in the 1-D case (Fig. 1) is defined as

$$(\mathbf{x} * \mathbf{w})_i = \sum_{m=0}^{D_K-1} w_{i-m} x_m, \tag{2}$$

where $\mathbf{w} \in \mathbb{R}^{D_K}$, $\mathbf{x} \in \mathbb{R}^{D_F}$, and $0 \leq i < D_K + D_F - 1$. This operation can be extended to multiple dimensions in a straightforward manner. Convolutional neural networks also feature pooling layers, to reduce the spatial resolution of layers, therefore reducing the number of subsequent computations. Finally, the classifier discriminates outcomes between the known classes, based on features forwarded from the convolutional layers.

Considerable work has investigated the implementation of convolutional neural networks in hardware using emerging memory, with architectures such as ISAAC [18] or PRIME [19], using either digital or analog coding for the weights. Digital coding has substantial memory requirements. Analog coding, by contrast, requires only two devices per weight (two devices are needed for the possibility to store negative weights), but has the disadvantage of requiring complex peripherals such as analog-to-digital and digital-to-analog converters with their associated high area overhead [18], [19].

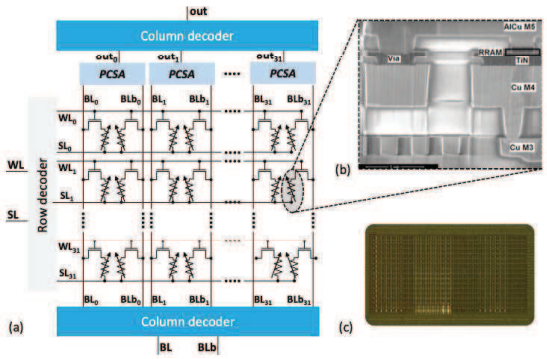

Fig. 2. (a) Schematic of a 1K synapses RRAM cells (b) Scanning Electron Microscopy image of an RRAM device integrated in the BEOL of our technology (c) Photograph of the die of our test chip with 1K synapses / 2K RRAM cells.

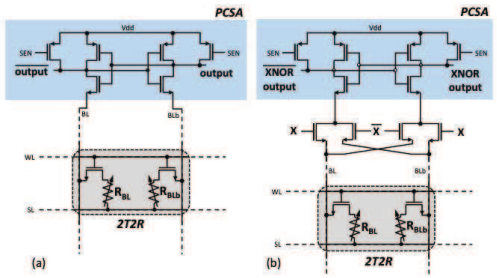

Fig. 3. (a) Schematic of the sense amplifier used to extract the binary weight from a 2T2R synapse. (b) Version augmented with an XNOR feature.

Binarizing the neural network provides an alternative route to reducing the area and energy consumption of hardware neural networks. In binarized neural networks (BNNs), we use weights with values of either $+1$ or -1, and the sign function as activation function. This allows simplifying Eq. (1) to

$$y = \text{sign}\left(\text{popcount}\left(\text{XNOR}(w_j, x_j)\right) - b\right), \tag{3}$$

where popcount is a function counting the number of 1 bits and b is a learned neuron threshold. On top of the low memory requirements of BNNs, replacing multiplication circuits with simple XNOR logic gates can greatly reduce the circuit area.

B. Implementing Binarized Neural Networks

We now introduce our technique for the energy-efficient in-memory implementation of binarized neural networks exploiting resistive memory. Our test chip uses hafnium oxide-based resistive memory, fully integrated within the back end of line (BEOL) of a commercial 130 nanometer CMOS process (Fig. 2(b)). A photograph of our die is presented in Fig. 2(c) and its simplified schematic in Fig. 2(a). The considerable challenge to implement in-memory computing with resistive memory is their inherent device variation, which leads to

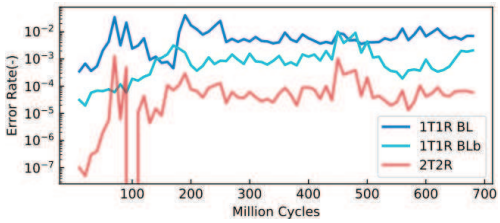

Fig. 4. Mean bit error rate comparison between 1T1R & 2T2R configuration over 10 million cycles as a function of the number of cycles that a device has been programmed.

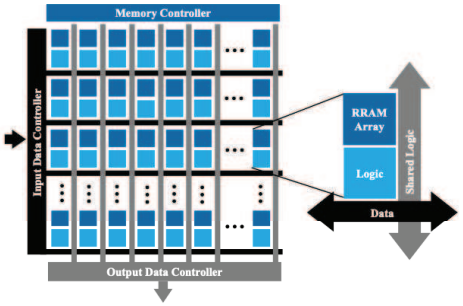

Fig. 5. Schematic of the basic architecture for implementing fully connected BNN layer from in-memory computing basic blocks.

bit errors [20], [21]. Conventional digital designs suppress these errors by relying on multiple error correcting codes (ECC). However, the use of ECC in the context of in-memory binarized neural network is not satisfying: the computation of error detection and correction is more complicated than the one of binarized neural network, and would dominate area and energy consumption. Additionally, ECC goes against the idea of integrating part of the computation within the memory array or sensing circuit, a major idea of in-memory computing. Our design therefore uses an alternative ECC-less approach to reduce the number of bit errors, by relying on a two transistor/two resistor (2T2R) architecture (Fig. 2(a)), where synaptic weights are stored in a differential fashion: by convention, a device pair programmed in the low resistance/high resistance state means a synaptic weight of $+1$, and reciprocally a pair programmed in the high resistance/low resistance state means a synaptic weight -1. Precharge sense amplifiers (PCSA, Fig. 3(a)) are used to compare the resistance states of the two devices of a pair, and therefore read the synaptic weight. An attractive possibility of the approach is the option to incorporate the XNOR operation of BNNs directly within the precharge sense amplifier, by the addition of solely four transistors (Fig. 3(b)).

Experimental results on the fabricated test confirm that the 2T2R approach reduces the amount of bit errors. Fig. 4, for example, shows bit error rate measurements on a pair of devices within a kilobit memory array. The pair is reprogrammed 700 million times, alternating the programming in high resistance/low resistance states and low resistance/high resistance states. The weight is measured after each programming event using the on-chip precharge sense amplifier in the 2T2R case, and is compared with direct sensing when single devices are used (1T1R). We see that the 2T2R error rate is two orders of magnitude below the 1T1R error. More extensive experimental results on this test chip, involving various programming conditions and whole memory array measurements, are shown in [15], [16]. In particular, the results reported in these references indicate that the benefits of the 2T2R approach in terms of bit error rate reduction are similar to the one of formal single error correction of equivalent redundancy.

The memory arrays (Fig. 2) that we characterized can be used as basic building blocks for complete architectures implementing binarized neural networks. Fig. 5 presents an architecture to implement fully connected layers, minimizing data movement, as described in detail in [14]. The architecture incorporates RRAMs arrays with sense amplifiers augmented with XNOR to perform binary multiplication; additional logic elements are added to perform popcount operations. The devices are programmed to neural network weights obtained by off-chip training. This programming occurs before the use of the inference circuit and is managed by a memory controller. This type of architecture can be adapted for convolutional layers, with a key decision between minimizing data movement and data reuse: several works have investigated the implementation of convolutional layers from basic in-memory computing processing elements using static RAM [22] of emerging memories [18], [19], [23].

III. Biomedical Time-Signals

BNNs have been mostly investigated for computer vision tasks. Their potential for low power hardware also makes them particularly attractive for medical signal analysis. For this reason, here, we investigate BNNs for that purpose, and discuss resulting hardware implications.

A. Electroencephalography (EEG) Task

Electroencephalography is used to measure electric potentials from a human scalp surface, and has numerous applications, such as epilepsy diagnosis and prediction, brain-computer interfaces or sleep stages monitoring. As EEG suffers from low signal to noise ratio, complex analysis is usually required. For this study we utilized data from the public EEG Motor Movement/Imagery Dataset [24], [25]. Here we consider a task of motor imagery: a subject has to imagine moving their left or right fist, and our neural network aims at detecting which of the two movements has been imagined, based on six seconds time EEG measurements. The complete set contains data of 109 subjects with 64 electrodes sampled at 160 Hz. We used a subset of 105 subjects with 42 trials of left-right imaginary movements recorded. The four remaining subjects data were discarded as incomplete. The only preprocessing

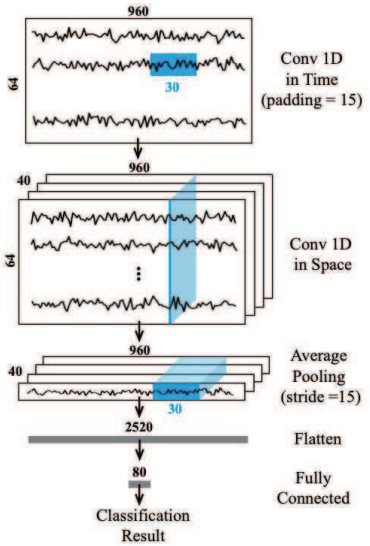

Fig. 6. End-to-end EEG classification model proposed in [27].

step performed was per-channel normalization by subtracting the mean and dividing by variance. As the dataset is not large, we used a relatively shallow neural network model, and we added small amplitude noise to each training sample for data-augmentation. As a baseline, we exploit the end-to-end EEG classification neural network model proposed in [26], [27] (Fig. 6 and Table I). This model consists of two convolutional layers followed by average pooling and a two-layer classifier. The first layer processes individual signals as 1D images, therefore performing 64 individual convolutions in time (Fig. 1) over incoming EEG time-signals. The second convolutional layer correlates obtained signals in space simultaneously over all 64 channels. We apply ReLU activations throughout the EEG model and replace them by sign in a binarized setting. The final softmax layer is necessary only for training. The real-weight version of this neural network was reported to achieve 88% accuracy [27]. To evaluate our binarization strategies, similarly to the baseline, we apply five-fold cross-validation meaning that the dataset is partitioned into five non-overlapping validation subsets not seen during the training. We report an average over five experiments where we train a new model from scratch over 1000 epochs with the Adam method [28] in each experiment.

B. Electrocardiogram (ECG) Task

Electrocardiogram signals measure electrical changes as a result of cardiac muscle depolarization, and are used e.g. for diagnosing different kinds of arrhythmia. Typically, ECGs are recorded using 12 electrodes, which need to be properly positioned: inverting any pair of electrodes may lead to wrong diagnosis. We here consider the task of detecting such elec-

TABLE I
EEG CLASSIFICATION NETWORK ARCHITECTURE FROM [27].

	Kernels	Padding	Output shape
Conv 40	30×1	15	$961 \times 64 \times 40$
Conv 40	$1 \times 64 \times 40$	No	$961 \times 1 \times 40$
Avg. pool	30×1	No	$63 \times 1 \times 40$
Flatten	-	-	2520
FC 80	-	-	80
Softmax	-	-	2

trode inversion. We used a dataset from the Challenge Data competition [29]. The dataset contains 1000 trials of three second recordings performed at 250 Hz. Our custom CNN model is summarized in Table II. Each convolution/linear layer is followed by batch normalization and nonlinear activation. We replace hardtanh activation by a sign in a binarized setting. In addition, we also perform batch normalization of the input data. We train the model using the Adam optimizer [28] over 1000 training epochs and, as in the EEG task, we perform five-fold cross-validation five times. To address overfitting, we employ a dropout regularization with keep probability 0.95 within convolution layers and 0.85 within the classifier.

TABLE II
ECG CLASSIFICATION NETWORK ARCHITECTURE.

	Kernels	Padding	Output shape
Conv 32	$13 \times 1 \times 12$	No	$738 \times 1 \times 32$
Max. pool	2×1	No	$369 \times 1 \times 32$
Conv 32	$11 \times 1 \times 32$	No	$359 \times 1 \times 32$
Max. pool	2×1	No	$179 \times 1 \times 32$
Conv 32	$9 \times 1 \times 32$	No	$171 \times 1 \times 32$
Conv 32	$7 \times 1 \times 32$	No	$165 \times 1 \times 32$
Conv 32	$5 \times 1 \times 32$	No	$161 \times 1 \times 32$
Flatten	-	-	5152
FC 75	-	-	75
Softmax	-	-	2

TABLE III
ACCURACY COMPARISON OF CNN WITH REAL WEIGHTS, BINARIZED CNN (BNN), AND CNN WHERE ONLY THE FULLY-CONNECTED PART WAS BINARIZED. IN PARENTHESES, NUMBER OF FILTER AUGMENTATIONS.

Task	Real-weight NN	BNN	Bin. Classifier
EEG	88% [27]	84.6% (1×) 86% (11×)	87% (1×)
ECG	96.3%	92.1% (1×) 94.9% (7×)	95.9% (1×)
ImageNet Top-1	70.6% [8]	54.4% (4×) [30]	70% (1×)
ImageNet Top-5	89.5% [8]	77.5% (4×) [30]	89.1% (1×)

C. Results

We performed simulations on the ECG and EEG neural network architectures in three cases: real weights (32-bit floating point), fully binarized neural network, and a mixed situation (binarized classifier) where fully-connected layers are binarized, while convolutional layers remain real. Fig. 7 shows a detailed result in the ECG case. We see that a fully

TABLE IV
MODEL MEMORY USAGE COMPARISON BETWEEN DIFFERENT TASKS AND SAVINGS WITH CLASSIFIER BINARIZATION.

Model	Total params	Classifier params	Model size 32-bit / 8-bit	Bin classif. saving %
EEG	0.31M	0.2M	1.17MB / 305KB	64% / 57.8%
ECG	0.31M	0.27M	1.17MB / 305KB	84% / 75.8%
ImageNet	4.2M	1M	16.2MB / 4.1MB	20% / 7.3%

binarized neural network on average performed worse than a real weight neural network if an equivalent number of filters is used (92.1% vs. 96.3% accuracy). The introduction of more filters allows enhancing the accuracy of the BNN; however, the accuracy of the real network is not reached. Fig. 7 also shows that the network where only the classifier has been binarized is able to match the accuracy of the real neural network (within error bar), without any augmentation of the number of filters. Similar results were obtained with the EEG task, as reported in Table III.

These results have interest for hardware development. For both ECG and EEG models, most of the weights reside in fully-connected classifier layers, and the strategy of binarizing only the classifier has therefore high memory benefits. A detailed analysis of the memory savings of the different strategies is presented in Table IV. For example, the EEG model requires 0.31M total parameters, occupying 1.17MB of memory; 0.11M of parameters (406 KB) are in convolutional layers and 0.2M (789 KB or 66% of total parameters), in the classifier. Therefore, binarizing only the classifier can save about 64% compared to the 32-bit model. If we compare with a neural network quantized to eight-bit numbers, the memory saving would be 57.8% of memory. Even better memory savings are obtained with the ECG model: by binarizing only the classifier, the memory reduction is 84% compared to the original 32-bit model. If we used an eight-bits numbers quantized network as a reference, we would save 75.8% memory. The strategy of only binarizing the classifier also achieves better accuracy than a fully binarized network using an equivalent amount of memory: the binarized classifier model accuracy is by 1% better for EEG and by 1% for ECG, compared to those with all-binarized network of equivalent number of bits (11 times BNN convolution filter augmentation for EEG, 7 times augmentation for ECG BNN model). If we assume that convolutional layers can be quantized to eight-bits precision, the accuracy gap is then 2.3% for EEG (3× augmentation) and 1% for ECG (2×).

IV. PARTIAL BINARIZATION ON MOBILENET

Machine learning can be effectively applied to analyze data obtained by medical imaging, and we envision that binarized neural networks could be successful in such contexts. To evaluate the applicability of our approach to vision tasks on neural networks optimized for mobile applications, we evaluate our approach of classifier binarization on the generic ImageNet dataset [31], containing 1.2 million images representing 1000 classes. We utilize MobileNet V1, a compact network specifically designed for mobile devices with less computing power [8]. MobileNet architecture is characterized by replacing most of standard convolutions with depthwise-separable convolutions, which require less computations. In this work, we replaced the fully-connected classifier with a binarized classifier of two layers. We trained the network from scratch with stochastic gradient descent method in 255 epochs. We achieved Imagenet Top-1 accuracy close to the original MobileNet-224 (70.6% vs. 70% bin classifier) and equally, Top-5 accuracy (89.5% vs. 89.1% bin classifier) (Fig. 8), whereas fully binarizing Mobilenet V1 is associated with accuracy degradation (Table III). This result further confirms that classifiers binarize more naturally than convolutional layers.

The MobileNet-224 model has 4.2M parameters (16.2MB), 3.2M parameters (12.28MB) of which are used by convolutional layers and 1M (4.05MB), by the original single-layer classifier (Table IV). Therefore, the classifier occupies about 24% of memory. As a binarized classifier we use two layers of 5.7M binary parameters (696KB), therefore this allows us to save about 20% of memory compared to the network with all

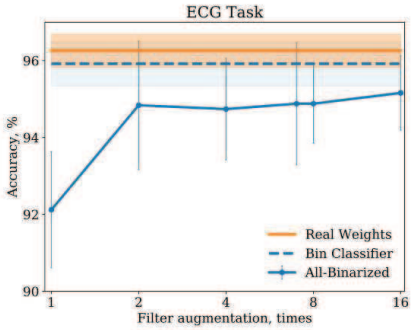

Fig. 7. Cross-validated accuracy on the ECG task, for different models. BNN accuracy is improved when increasing the number of convolution filters.

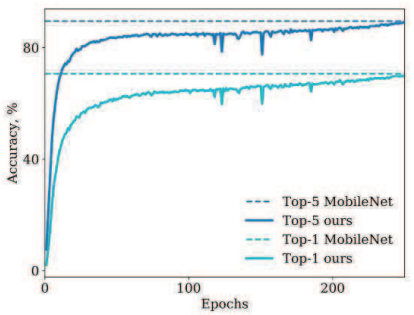

Fig. 8. Training modified MobileNet-224 model with a binarized classifier on the ImageNet 1K challenge.

32-bit weights. In case if we used 8-bit numbers as a reference, we would still spare about 7.3% memory.

V. Conclusion

In this work, we highlighted that binarized neural network can be a road to implement particularly efficient neural networks hardware. We introduced an implementation, validated by measurements on a fabricated test chip using hafnium oxide resistive memory. Our implementation is designed around the principles of in-memory computing, limiting the amount of data movement, and avoiding error correcting code altogether. As BNNs have mostly been evaluated on vision tasks, and medical signal analysis is believed to be an essential application for highly efficient AI chips, we evaluated BNNs on two sample ECG and EEG signal analysis task. We report that all-binarized neural networks can be a tool to reduce memory requirements. On the other hand, if keeping the highest accuracy is vital, we propose an alternative route where only the classifier part of the neural network is binarized. Moreover, as these neural networks are dominated by classifier, non volatile memory requirement can be considerably reduced by classifier binarization. We also evaluated the strategy of binarizing only the classifier on a vision task, on an architecture optimized for modest memory requirement (MobileNet V1). The resulting neural network matches the original MobileNet V1 network accuracy on ImageNet large-scale benchmark, however the nonvolatile memory benefits are smaller than in the EEG and ECG tasks, as MobileNet V1 is dominated by convolutions.

These first results are very encouraging to reduce efficiently memory requirement of edge devices and thus to obtain low energy hardware. This is particularly useful for medical applications where little energy is available. The results also highlight that such hardware development should be pursued in co-development with neural network architecture and in conjunction with application development, as the most efficient approaches can be considerably task-dependent.

References

[1] A. Esteva *et al.*, "Dermatologist-level classification of skin cancer with deep neural networks," *Nature*, vol. 542, no. 7639, pp. 115–118, 2017.

[2] D. Silver *et al.*, "Mastering the game of go with deep neural networks and tree search," *Nature*, vol. 529, no. 7587, p. 484, 2016.

[3] A. Pedram, S. Richardson, M. Horowitz, S. Galal, and S. Kvatinsky, "Dark memory and accelerator-rich system optimization in the dark silicon era," *IEEE Design & Test*, vol. 34, no. 2, pp. 39–50, 2017.

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

[5] G. Indiveri and S.-C. Liu, "Memory and information processing in neuromorphic systems," *Proc. IEEE*, vol. 103, no. 8, p. 1379, 2015.

[6] K. Siu, D. M. Stuart, M. Mahmoud, and A. Moshovos, "Memory Requirements for Convolutional Neural Network Hardware Accelerators," in *Proc. IISWC*. IEEE, sep 2018, pp. 111–121.

[7] F. N. Iandola *et al.*, "Squeezenet: Alexnet-level accuracy with 50x fewer parameters and¡ 0.5 mb model size," *arXiv preprint:1602.07360*, 2016.

[8] A. G. Howard *et al.*, "Mobilenets: Efficient convolutional neural networks for mobile vision applications," *preprint arXiv:1704.04861*, 2017.

[9] X. Zhang, X. Zhou, M. Lin, and J. Sun, "Shufflenet: An extremely efficient convolutional neural network for mobile devices," in *Proc. CVPR*, 2018, pp. 6848–6856.

[10] I. Hubara, M. Courbariaux, D. Soudry, R. El-Yaniv, and Y. Bengio, "Quantized neural networks: Training neural networks with low precision weights and activations," *JMLR*, vol. 18, no. 1, pp. 6869–6898, 2017.

[11] N. P. Jouppi *et al.*, "In-datacenter performance analysis of a tensor processing unit," in *Proc. ISCA*. IEEE, 2017, pp. 1–12.

[12] 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:1602.02830*, 2016.

[13] 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.

[14] T. Hirtzlin, B. Penkovsky, M. Bocquet, J. O. Klein, J. M. Portal, and D. Querlioz, "Stochastic Computing for Hardware Implementation of Binarized Neural Networks," *IEEE Access*, vol. 7, p. 76394, 2019.

[15] 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.

[16] T. Hirtzlin *et al.*, "Digital biologically plausible implementation of binarized neural networks with differential hafnium oxide resistive memory arrays," *arXiv preprint:1908.04066*, 2019.

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

[18] A. Shafiee *et al.*, "Isaac: A convolutional neural network accelerator with in-situ analog arithmetic in crossbars," *Comput. Archit. News*, vol. 44, no. 3, pp. 14–26, 2016.

[19] P. Chi *et al.*, "Prime: A novel processing-in-memory architecture for neural network computation in reram-based main memory," in *Comput. Archit. News*, vol. 44, no. 3. IEEE Press, 2016, pp. 27–39.

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

[21] D. R. B. Ly *et al.*, "Role of synaptic variability in resistive memory-based spiking neural networks with unsupervised learning," *J. Phys. D: Applied Physics*, 2018.

[22] H. Valavi, P. J. Ramadge, E. Nestler, and N. Verma, "A mixed-signal binarized convolutional-neural-network accelerator integrating dense weight storage and multiplication for reduced data movement," in *Symp. VLSI Circ.* IEEE, 2018, pp. 141–142.

[23] E. Giacomin, T. Greenberg-Toledo, S. Kvatinsky, and P.-E. Gaillardon, "A robust digital rram-based convolutional block for low-power image processing and learning applications," *IEEE TCAS I*, vol. 66, no. 2, pp. 643–654, 2019.

[24] G. Schalk, D. McFarland, T. Hinterberger, N. Birbaumer, and J. Wolpaw, "BCI2000: A General-Purpose Brain-Computer Interface (BCI) System," *IEEE Trans. Biomed. Eng.*, vol. 51(6), pp. 1034–1043, 2004.

[25] A. Goldberger *et al.*, "PhysioBank, PhysioToolkit, and PhysioNet: Components of a New Research Resource for Complex Physiologic Signals," *Circulation*, vol. 101(23), pp. e215–e220, 2000.

[26] R. T. Schirrmeister *et al.*, "Deep learning with convolutional neural networks for EEG decoding and visualization," *Human Brain Mapping*, vol. 38, no. 11, pp. 5391–5420, 2017.

[27] H. Dose, J. S. Møller, H. K. Iversen, and S. Puthusserypady, "An end-to-end deep learning approach to MI-EEG signal classification for BCIs," *Expert Systems with Applications*, vol. 114, pp. 532–542, 2018.

[28] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint:1412.6980*, 2014.

[29] Challenge data, "Electrode inversion detection in ECGs," 2015, Accessed 11.12.2019. [Online]. Available: http://archive.is/67k4P

[30] H. Phan, D. Huynh, Y. He, M. Savvides, and Z. Shen, "Mobinet: A mobile binary network for image classification," *arXiv preprint:1907.12629*, 2019.

[31] O. Russakovsky *et al.*, "ImageNet Large Scale Visual Recognition Challenge," *Int. J. Comput. Vis.*, vol. 115, no. 3, pp. 211–252, 2015.