Computational memory-based inference and training of deep neural networks

A. Sebastian[1*], I. Boybat[1,2], M. Dazzi[1,3], I. Giannopoulos[1,3], V. Jonnalagadda[1], V. Joshi[1,4], G. Karunaratne[1,3], B. Kersting[1,5], R. Khaddam-Aljameh[1,3], S. R. Nandakumar[1,4], A. Petropoulos[1,6], C. Piveteau[1,3], T. Antonakopoulos[6], B. Rajendran[4], M. Le Gallo[1], E. Eleftheriou[1]

[1]IBM Research - Zurich, Switzerland, [2]EPFL, Lausanne, Switzerland, [3]ETH Zürich, Zurich, Switzerland, [4]New Jersey Institute of Technology, Newark, NJ, USA, [5]RWTH, Aachen, Germany, [6]University of Patras, Patras, Greece.

Email: ase@zurich.ibm.com

Abstract

In-memory computing is an emerging computing paradigm where certain computational tasks are performed in place in a computational memory unit by exploiting the physical attributes of the memory devices. Here, we present an overview of the application of in-memory computing in deep learning, a branch of machine learning that has significantly contributed to the recent explosive growth in artificial intelligence. The methodology for both inference and training of deep neural networks is presented along with experimental results using phase-change memory (PCM) devices.

Keywords: In-memory computing, deep learning, PCM

Introduction

Deep neural networks (DNNs), loosely inspired by biological neural networks, consist of parallel processing units called neurons interconnected by plastic synapses. By tuning the weights of these interconnections, these networks are able to perform certain cognitive tasks remarkably well. These networks are typically trained via a supervised learning algorithm based on gradient descent. During the training phase, the input data is forward propagated through the neuron layers with the synaptic networks performing multiply-accumulate operations. The final layer responses are compared with input data labels and the errors are back-propagated. Both steps involve sequences of matrix-vector multiplications. Subsequently, the synaptic weights are updated to reduce the error. This optimization approach can take multiple days or weeks to train state-of-the-art networks on conventional computers and hence, there is a significant effort towards the design of custom ASICs based on reduced precision arithmetic and highly optimized dataflow [1,2]. However, one of the primary reasons for the inefficiency, namely the need to shuttle millions of synaptic weight values between the memory and processing units, remains unaddressed. In-memory computing is an emerging computing paradigm that addresses this challenge of processor-memory dichotomy [3,4]. For example, a computational memory unit with resistive memory (memristive) devices organized in a crossbar configuration is capable of performing matrix-vector multiply operations in-place by exploiting the Kirchhoff's circuits laws. Moreover, the computational time complexity reduces to O(1) [5,6].

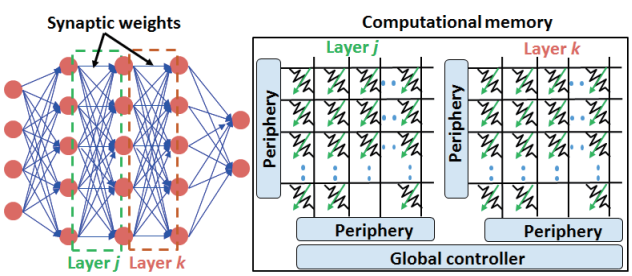

Fig. 1 The various layers of a neural network are mapped to a computational memory unit where resistive memory devices are organized in a crossbar configuration.

Key enabling properties

There are two key properties of memristive devices that facilitate computational memory-based deep learning. First, these devices can achieve a continuum of conductance values (analog storage capability) by applying suitable electrical pulses. For example, Fig. 2(a) shows the conductance values of phase-change memory (PCM) devices as a function of the applied programming current. The measurements are obtained using over 10,000 PCM devices fabricated in the 90nm technology node. Using this so-called "programing curve", it is possible to program a single PCM device to a target conductance value via iterative programming [7] (Fig. 2(b)).

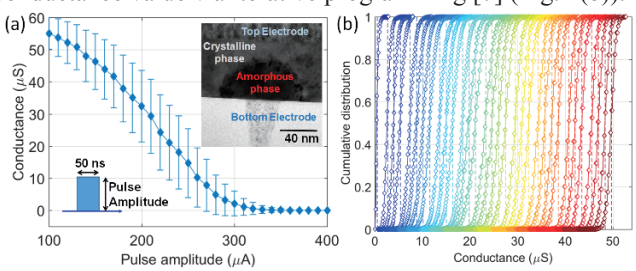

Fig. 2 (a) Mean and STD of conductance values as a function of the programming current. (b) The cumulative distribution of the PCM devices, where each device is programmed to 32 levels.

The second key property is the accumulative behavior, which for PCM arises from the crystallization dynamics [8]. The conductance of these devices progressively increases with the successive application of crystallizing pulses albeit with significant randomness and nonlinearity (see Fig. (3)). For deep learning, the analog storage capability facilitates the forward and backward propagations while the accumulative behavior is typically exploited for synaptic weight updates.

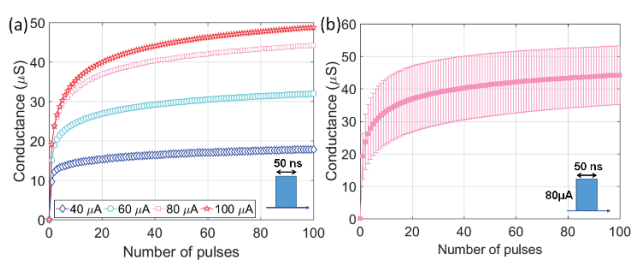

Fig. 3 (a) The mean conductance values as a function of the number of programming pulses. (b) The STD shows the significant randomness associated with the accumulative behavior.

DNN inference

Deep learning inference refers to just the forward propagation in a DNN once the weights have been learned. When using computational memory for inference, the key challenges are the inaccuracies associated with programming the devices to a specified synaptic weight as well as drift, noise etc. associated with the conductance values. Due to these reasons, the synaptic

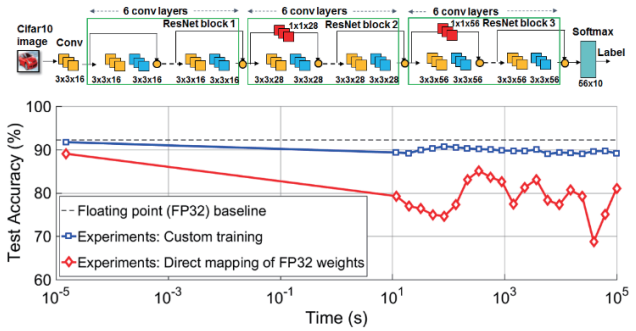

Fig. 4 Experimental results on ResNet-20 using the CIFAR-10 dataset. The classification accuracies obtained via the direct mapping and custom training approaches are compared to the FP32 baseline.

weights that are obtained by training in high precision arithmetic (eg. 32-bit floating point) cannot be mapped directly to computational memory. However, it can be shown that by customizing the training procedure to make it aware of the device-level non-idealities, it is possible to obtain synaptic weights that are suitable for being mapped to computational memory [9]. Fig. 4 shows mixed hardware/software experimental results using a prototype multi-level PCM chip. The synaptic weights are mapped to PCM devices organized in a 2-PCM differential configuration (541,812 PCM devices in total). It can be seen that the custom training scheme approaches the FP32 baseline, whereas the direct mapping approach fails to deliver sufficient accuracy. The slight temporal decline in accuracy is attributed to the conductance drift exhibited by PCM devices [10]. However, in spite of the drift, a classification accuracy of close to 90% is maintained over a significant duration of time.

DNN training

Recent DL research shows that when training DNNs, it is possible to perform the forward and backward propagations rather imprecisely while the gradients need to be accumulated in high precision [11]. This observation makes the DL training problem amenable to the mixed-precision in-memory computing approach that was recently proposed [12]. The computational memory unit is used to store the synaptic weights and to perform the forward and backward passes, while the weight changes are accumulated in high precision (Fig. 5(a)) [13]. When the accumulated weight exceeds a certain threshold, pulses are applied to the corresponding memory devices to alter the synaptic weights. This approach was tested using the handwritten digit classification problem based on the MNIST data set. A two-layered neural network was employed with 2-PCM devices in differential configuration (approx. 400,000 devices) representing the synaptic weights. Resulting test accuracy after 20 epochs of training was approx. 98% (Fig. 5(b)). After training, inference on this network was performed for over a year with marginal

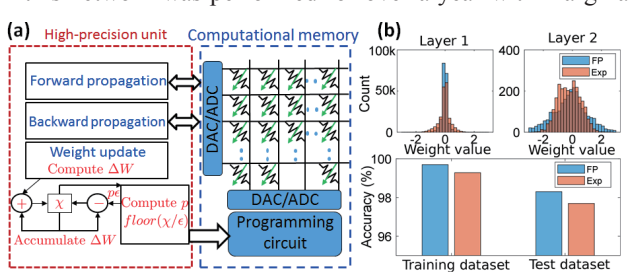

Fig. 5 (a) Schematic illustration of the mixed-precision architecture for training DNNs. (b) The synaptic weight distributions and classification accuracies are compared between the experiments and floating point baseline.

reduction in the test accuracy [14].

The crossbar topology also facilitates the estimation of the gradient and the in-place update of the resulting synaptic weight all in O(1) time complexity [15,16]. By obviating the need to perform gradient accumulation externally, this approach could yield better performance than the mixed-precision approach. However, significant improvements to the memristive technology, in particular the accumulative behavior, is needed to apply this to a wide range of DNNs.

Outlook

Our system-level studies show that even with today's PCM technology, we can achieve significantly higher performance compared to conventional approaches for both inference and training. This performance improvement is expected to be substantially higher with future generations of PCM devices. Phase change materials are known to undergo reversible phase transition down to nanoscale dimensions and on the order of nanoseconds. Moreover, the retention time, which is a key requirement for traditional memory applications is less important for DL training and this could enable the exploration of new material classes such as elemental Antimony [17]. However, there are also numerous roadblocks associated with using PCM devices for computational purposes. One key challenge is the structural relaxation of the melt-quenched amorphous phase that results in a conductance drift which was mentioned earlier. There are also temperature-induced conductance variations as well as 1/f noise that reduces the precision associated with the matrix-vector multiply operations. One very promising research avenue towards addressing the challenge of conductance variations is that of projected phase-change memory where a shunt path is provided for read current to bypass the amorphous phase-change material [18,19]. Recently, it was shown that it is possible to achieve remarkably high precision in-memory scalar multiplication (equivalent to 8-bit fixed point arithmetic) using projected PCM devices [20]. Another challenge is the limited endurance of PCM devices, which is relatively high (approx. 10^9 to 10^{12}) and sufficient for inference applications but may not be adequate for certain training applications. The limited endurance and various other non-idealities associated with the accumulative behavior such as limited dynamic range, nonlinearity and stochasticity can be partially circumvented with multi-memristive synaptic architectures [21]. Hence, in spite of the various challenges, computational memory-based co-processors for DL are expected to usher in a new era of non-von Neumann computing.

Acknowledgements We would like to thank our colleagues at IBM Research – Zurich, IBM T. J. Watson Research Center and IBM Research – Almaden for their contributions to this work. We acknowledge partial financial support from ERC grant 682675.

References

[1] B. Fleischer, *VLSI*, 2018 [2] N. P. Jouppi, *ISCA*, 2017 [3] A. Sebastian, *Nature Comm.*, 2017 [4] D. Ielmini, *Nature Electr.*, 2018 [5] G. W. Burr, *Adv. Phys. X*, 2017 [6] A. Sebastian, *J. Appl. Phys.*, 2018 [7] N. Papandreou, *ISCAS*, 2011 [8] A. Sebastian, *Nature Comm.*, 2014 [9] A. F. Murray, *IEEE Trans. Neural Networks*, 1993 [10] M. Le Gallo, *Adv. Electr. Mat.*, 2018 [11] I. Hubara, *J. Machine Learning Research*, 2017 [12] M. Le Gallo, *Nature Electr.*, 2018 [13] S. R. Nandakumar, *ISCAS*, 2018 [14] https://analog-ai-demo.mybluemix.net/ [15] G. W. Burr, *IEEE TED*, 2015 [16] W. Haensch, *Proc. IEEE*, 2019 [17] M. Salinga, *Nature Mat.*, 2018 [18] S. Kim, *IEDM*, 2013 [19] W. Koelmans, *Nature Comm.*, 2015 [20] I. Giannopoulos, *IEDM*, 2018 [21] I. Boybat, *Nature Comm.*, 2018