Analog electronic deep networks for fast and efficient inference

Extended Abstract

Jonathan Binas*
Institute of Neuroinformatics,
U. of Zurich and ETH Zurich

Daniel Neil†
Institute of Neuroinformatics,
U. of Zurich and ETH Zurich

Giacomo Indiveri
Institute of Neuroinformatics,
U. of Zurich and ETH Zurich

Shih-Chii Liu
Institute of Neuroinformatics,
U. of Zurich and ETH Zurich

Michael Pfeiffer‡
Institute of Neuroinformatics,
U. of Zurich and ETH Zurich

ABSTRACT

We propose an efficient approach for real-time inference using deep neural networks implemented through low-power analog electronic circuits. Although analog implementations can be extremely compact, they have been largely supplanted by digital designs, partly because of device mismatch effects due to fabrication imperfections. We propose a framework that exploits the power of deep learning to compensate for this mismatch by incorporating the measured device variations as constraints in the training process. This eliminates the need for mismatch minimization strategies and allows circuit complexity and power-consumption to be reduced to a minimum. Our results, based on large-scale simulations as well as a prototype VLSI chip implementation indicate a processing efficiency comparable to current state-of-art digital implementations. This method is suitable for future technology based on nanodevices with large variability, such as memristive arrays.

1 INTRODUCTION

The large computational demands of Deep Neural Networks (DNNs) have simultaneously sparked interest in methods that make neural network inference faster and more power efficient, whether through new algorithmic inventions [8, 12, 14], dedicated digital hardware implementations [5, 6, 10], or by taking inspiration from real nervous systems [9, 15, 17–19].

With synchronous digital logic being the established standard of the electronics industry, many attempts towards hardware deep network accelerators have focused on this approach [5, 7, 11, 20]. However, the massively parallel style of computation in neural networks is not reflected in the mostly serial and time-multiplexed nature of digital systems. An arguably more natural way of building a hardware neural network emulator is to implement its computational primitives as multiple physical and parallel instances of analog computing nodes, where memory and processing elements are co-localized, and state variables are directly represented by analog currents or voltages, rather than being encoded digitally [1, 2, 4, 22–24]. By directly representing neural network operations in the physical properties of silicon transistors, such analog implementations can outshine their digital counterparts in terms of simplicity, allowing for significant advances in speed, size, and power consumption [13, 16]. The main reason why engineers have been discouraged from following this approach is that the properties of analog circuits are affected by the physical imperfections inherent to any chip fabrication process,

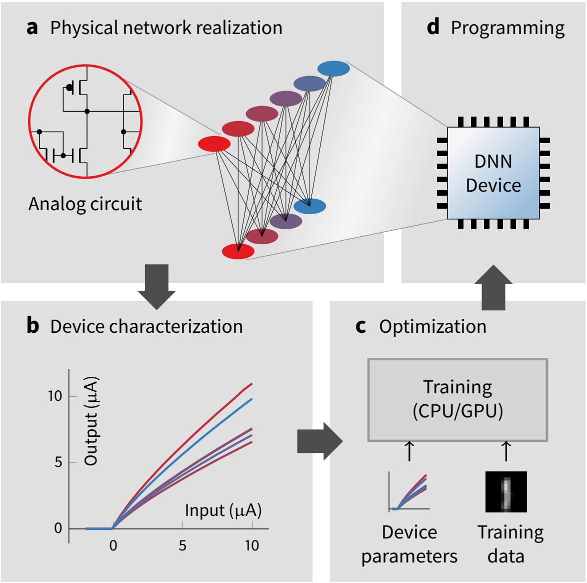

Figure 1: Implementing and training an analog electronic neural network. a) The configurable network is realized on a physical substrate by means of analog circuits, together with local memory elements storing the weight configuration. b) The transfer characteristics of individual neurons are obtained through measurements. c) Including the measured transfer characteristics in the training process allows optimization of the network for the particular device that has been measured. d) Mapping the parameters found by the training algorithm back to the device implements a neural network whose computation is comparable to the theoretically ideal network.

[*] Corresponding author; now at MILA, Montreal
[†] Now at Benevolent AI
[‡] Now at Robert Bosch GmbH

SYSML'18, February 2018, Stanford, CA

which can lead to significant functional differences between individual devices [21].

Our work proposes a new approach, whereby rather than brute-force engineering more homogeneous circuits (e.g. by increasing transistor size or adding active stabilization mechanisms), we employ neural network training methods as an effective optimization framework to automatically compensate for the device mismatch effects.

2 COMPACT CIRCUITS FOR HIGHLY PARALLEL IMPLEMENTATIONS

The simple operations required to implement a typical neural network (we consider a multilayer perceptron architecture here) can be very efficiently realized using analog electronics. If quantities are represented as currents (current-mode design), multiplication by a constant (weighting) can be realized with as few as two transistors, addition comes for free (simply connect the wires), and rectification (e.g. ReLU activation) requires a single diode-connected transistor. To achieve a low-power solution, the circuits can be operated in the subthreshold region. The subthreshold current of a transistor is exponential in the gate voltage, rather than polynomial as is the case for above threshold operation, and can span many orders of magnitude. Thus, a system based on this technology can be operated at orders of magnitude lower currents than a digital one. In turn, this means that the device mismatch arising due to imperfections in the fabrication process can have an exponentially larger impact. Fortunately, as our method neither depends on the specific form nor the magnitude of the mismatch, it can handle a wide variety of mismatch conditions.

Specifically, we propose simple current-mode circuits for the weights and activations which, thanks to their compactness, can be used to implement a massively parallel, programmable multilayer network architecture. In our example implementations weight parameters are stored digitally and limited to three signed bits of precision. The digital memory is directly connected to analog transistors implementing the multiplication. The resulting system requires a mere 5 transistors per neuron and 11 per weight, which is substantially less than the hundreds of transistors typically required for digital multiply-accumulate units.

3 TRAINING A SYSTEM OF IMPERFECT NEURONS

The process of implementing a target functionality in a heterogeneous system of analog neurons is illustrated in Fig. 1. Once a neural network architecture with modifiable weights is implemented in silicon, the transfer characteristics of the different (mismatched) neuron instances can be measured by controlling the inputs specific cells receive and recording their output at the same time. If the transfer curves are sufficiently simple (depending on the actual implemented analog neuron circuit), a small number of discrete measurements yield sufficient information to fit a continuous, differentiable model to the hardware response. The continuous description is then used by the training algorithm, which is run on traditional computing hardware, such as CPUs or GPUs, to generate a network configuration that is tailored to the particular task and the physical device that has been characterized.

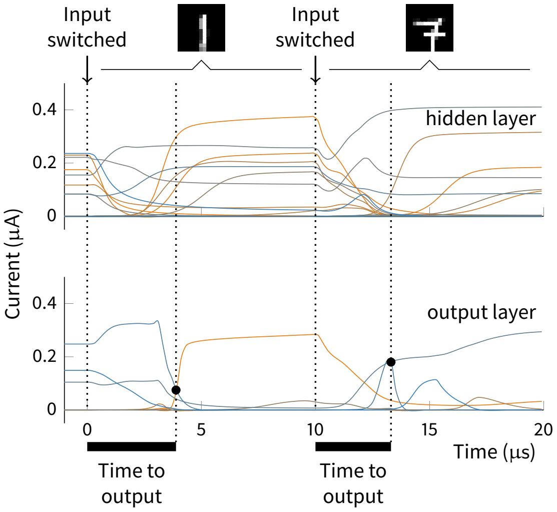

Figure 2: Analog circuit dynamics allow classification within microseconds. The curves represent the activities (currents) of all hidden (top) and output (bottom) units of a $196-50-10$ network. When a new input symbol is presented (top), the circuit converges to its new state within microseconds. Only a few units remain active, while many tend to zero, such that their soma circuits and connected synapses dissipate very little power.

4 EXPERIMENTAL RESULTS

Using the measured hardware characteristics as constraints during training leads to a dedicated set of parameters for each individual device. We evaluated the effectiveness of our aproach both through simulations and an actual analog VLSI prototype chip, fabricated in a 180 nm process. The fully parallel circuits are able to compute a classification within microseconds while dissipating microwatts of power (fig. 2). We obtained state-of-the-art classification results on MNIST, at an efficiency of ≈7 TOp/J (simulated system), as well as on the IRIS dataset (fabricated prototype chip). Experimental details can be found in [3].

5 CONCLUSION

We show that a few extraordinarily simple analog electronic circuits are sufficient for the exact implementation of feed-forward neural networks. To deal with the fabrication-induced transistor mismatch, making every circuit instance behave slightly differently, measured circuit characteristics are taken into account during training. The proposed method can be used with a variety of technologies suffering from similar inherent variations, such as memristive devices.

REFERENCES

[1] J. Alspector and R.B. Allen. 1987. A neuromorphic VLSI learning system. In *Proceedings of the 1987 Stanford Conference on Advanced Research in VLSI*, P. Losleben (Ed.). MIT Press, Cambridge, MA, USA, 313–349.

[2] Andreas G Andreou, Kwabena Boahen, Philippe O Pouliquen, Aleksandra Pavasovic, Robert E Jenkins, Kim Strohbehn, et al. 1991. Current-mode subthreshold MOS circuits for analog VLSI neural systems. *IEEE Transactions on neural networks* 2, 2 (1991), 205–213.

[3] Jonathan Binas, Daniel Neil, Giacomo Indiveri, Shih-Chii Liu, and Michael Pfeiffer. 2016. Precise deep neural network computation on imprecise low-power analog hardware. *arXiv preprint arXiv:1606.07786* (2016).

[4] T.H. Borgstrom, M Ismail, and S.B. Bibyk. 1990. Programmable current-mode neural network for implementation in analogue MOS VLSI. *IEE Proceedings G* 137, 2 (1990), 175–184.

[5] Lukas Cavigelli, David Gschwend, Christoph Mayer, Samuel Willi, Beat Muheim, and Luca Benini. 2015. Origami: A Convolutional Network Accelerator. In *Proceedings of the 25th edition on Great Lakes Symposium on VLSI*. ACM, 199–204.

[6] Yunji Chen, Tao Luo, Shaoli Liu, Shijin Zhang, Liqiang He, Jia Wang, Ling Li, Tianshi Chen, Zhiwei Xu, Ninghui Sun, et al. 2014. Dadiannao: A machine-learning supercomputer. In *Microarchitecture, 2014 47th Annual IEEE/ACM International Symposium on*. IEEE, 609–622.

[7] Yu-Hsin Chen, Tushar Krishna, Joel Emer, and Vivienne Sze. 2016. 14.5 Eyeriss: An energy-efficient reconfigurable accelerator for deep convolutional neural networks. In *2016 IEEE International Solid-State Circuits Conference (ISSCC)*. IEEE, 262–263.

[8] Matthieu Courbariaux, Yoshua Bengio, and Jean-Pierre David. 2015. BinaryConnect: Training Deep Neural Networks with binary weights during propagations. In *Advances in Neural Information Processing Systems*. 3105–3113.

[9] Clément Farabet, R Paz-Vicente, JA Pérez-Carrasco, Carlos Zamarreño-Ramos, Alejandro Linares-Barranco, Yann LeCun, Eugenio Culurciello, Teresa Serrano-Gotarredona, and Bernabe Linares-Barranco. 2012. Comparison between frame-constrained fix-pixel-value and frame-free spiking-dynamic-pixel convNets for visual processing. *Frontiers in Neuroscience* 6 (2012), 1–12. Issue 32.

[10] Vinayak Gokhale, Jonghoon Jin, Aysegul Dundar, Ben Martini, and Eugenio Culurciello. 2014. A 240 g-ops/s mobile coprocessor for deep neural networks. In *Computer Vision and Pattern Recognition Workshops (CVPRW), 2014 IEEE Conference on*. IEEE, 696–701.

[11] Matthew Griffin, Gary Tahara, Kurt Knorpp, Ray Pinkham, and Bob Riley. 1991. An 11-million transistor neural network execution engine. In *Solid-State Circuits Conference, 1991. Digest of Technical Papers. 38th ISSCC., 1991 IEEE International*. IEEE, 180–313.

[12] Song Han, Huizi Mao, and William J Dally. 2015. Deep Compression: Compressing Deep Neural Networks with Pruning, Trained Quantization and Huffman Coding. *arXiv preprint arXiv:1510.00149* (2015).

[13] Jennifer Hasler and Bo Marr. 2013. Finding a roadmap to achieve large neuromorphic hardware systems. *Frontiers in neuroscience* 7 (2013).

[14] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. 2015. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531* (2015).

[15] Giacomo Indiveri, Federico Corradi, and Ning Qiao. 2015. Neuromorphic Architectures for Spiking Deep Neural Networks. In *IEEE International Electron Devices Meeting (IEDM)*.

[16] Peter Masa, Klaas Hoen, and Hans Wallinga. 1994. A high-speed analog neural processor. *Micro, IEEE* 14, 3 (1994), 40–50.

[17] Paul A Merolla, John V Arthur, Rodrigo Alvarez-Icaza, Andrew S Cassidy, Jun Sawada, Filipp Akopyan, Bryan L Jackson, Nabil Imam, Chen Guo, Yutaka Nakamura, Bernard Brezzo, Ivan Vo, Steven K Esser, Rathinakumar Appuswamy, Brian Taba, Arnon Amir, Myron D Flickner, William P Risk, Rajit Manohar, and Dharmendra S Modha. 2014. A million spiking-neuron integrated circuit with a scalable communication network and interface. *Science* 345, 6197 (2014), 668–673.

[18] Daniel Neil, Michael Pfeiffer, and Shih-Chii Liu. 2016. Learning to be Efficient: Algorithms for Training Low-Latency, Low-Compute Deep Spiking Neural Networks. In *ACM Symposium on Applied Computing*.

[19] Peter O'Connor, Daniel Neil, Shih-Chii Liu, Tobi Delbruck, and Michael Pfeiffer. 2013. Real-time classification and sensor fusion with a spiking deep belief network. *Frontiers in Neuromorphic Engineering* 7 (2013). Issue 178.

[20] Seong-Wook Park, Junyoung Park, Kyeongryeol Bong, Dongjoo Shin, Jinmook Lee, Sungpill Choi, and Hoi-Jun Yoo. 2015. An energy-efficient and scalable deep learning/inference processor with tetra-parallel MIMD architecture for big data applications. *IEEE transactions on biomedical circuits and systems* 9, 6 (2015), 838–848.

[21] Marcel JM Pelgrom, Aad CJ Duinmaijer, and Anton PG Welbers. 1989. Matching properties of MOS transistors. *IEEE Journal of Solid-State Circuits* 24, 5 (Oct 1989), 1433–1439.

[22] Frank Rosenblatt. 1958. The perceptron: a probabilistic model for information storage and organization in the brain. *Psychological review* 65, 6 (nov 1958), 386–408.

[23] Srinagesh Satyanarayana, Yannis P Tsividis, and Hans Peter Graf. 1992. A reconfigurable VLSI neural network. *IEEE Journal of Solid-State Circuits* 27, 1 (Jan 1992), 67–81.

[24] Eric A Vittoz. 1990. Analog VLSI Implementation of Neural Networks. In *Proc. IEEE Int. Symp. Circuit and Systems*. New Orleans, 2524–2527.