frontiers in Neuroscience

EDITORIAL
published: 15 April 2020
doi: 10.3389/fnins.2020.00276

Editorial: Spiking Neural Network Learning, Benchmarking, Programming and Executing

Guoqi Li[1,2†], Lei Deng[3†], Yansong Chua[4], Peng Li[3], Emre O. Neftci[5] and Haizhou Li[6]*

[1] Department of Precision Instrument, Center for Brain Inspired Computing Research, Tsinghua University, Beijing, China, [2] Beijing Innovation Center for Future Chips, Tsinghua University, Beijing, China, [3] Department of Electrical and Computer Engineering, University of California, Santa Barbara, Santa Barbara, CA, United States, [4] Huawei Technologies, Shenzhen, China, [5] Department of Cognitive Science, University of California, Irvine, Irvine, CA, United States, [6] Department of Electrical Engineering, National University of Singapore, Singapore, Singapore

Keywords: deep spiking neural networks, SNN learning algorithms, programming framework, SNN benchmarks, neuromorphics

Editorial on the Research Topic

Spiking Neural Network Learning, Benchmarking, Programming and Executing

INTRODUCTION

A spiking neural network (SNN), a type of brain-inspired neural network, mimics the biological brain, specifically, its neural codes, neuro-dynamics, and circuitry. SNNs have garnered great interest in both Artificial Intelligence (AI) and neuroscience communities given its great potential in biologically realistic modeling of human cognition and development of energy efficient, event-driven machine learning hardware (Pei et al., 2019; Roy et al., 2019). Significant progress has been made across a wide spectrum of AI fields, such as image processing, speech recognition, and machine translation. They are largely driven by the advance in Artificial Neural Networks (ANN) in systematic learning theories, explicit benchmarks with various tasks and data sets, friendly programming tools [e.g., TensorFlow (Abadi et al., 2016) and Pytorch (Paszke et al., 2019) machine learning tools], and efficient processing platforms [e.g., graphics processing unit (GPU) and tensor processing unit (TPU) (Jouppi et al., 2017)]. In comparison, SNNs are still at an early stage in these aspects. To further exploit the advantages of SNNs and attract more researchers to contribute in this field, we proposed a Research Topic in Frontiers in Neuroscience to discuss the main challenges and future prospects of SNNs, emphasizing on its "Learning algorithms, Benchmarking, Programming, and Executing." We are confident that SNNs will play a critical role in the development of energy efficient machine learning devices through algorithm-hardware co-design.

This Research Topic brings together researchers of different disciplines in order to present their recent work in SNNs. We received 22 submissions worldwide and accepted 15 papers. The scope of the accepted papers covers learning algorithms, model efficiency, programming tools, and neuromorphic hardware.

LEARNING ALGORITHMS

Learning algorithms play perhaps the most important role in AI techniques. Machine learning algorithms, in particular those for deep neural networks (DNN), have become the standard bearer in a wide spectrum of AI tasks. Some of the more common learning algorithms include backpropagation (Hecht-Nielsen, 1992), stochastic gradient descent (SGD) (Bottou, 2012), and

OPEN ACCESS

Edited by:
Timothy K. Horiuchi, University of Maryland, United States

Reviewed by:
Scott Michael Koziol, Baylor University, United States

****Correspondence:***
Guoqi Li
liguoqi@mail.tsinghua.edu.cn

†These authors have contributed equally to this work

Specialty section:
This article was submitted to Neuromorphic Engineering, a section of the journal Frontiers in Neuroscience

Received: *13 November 2019*
Accepted: *10 March 2020*
Published: *15 April 2020*

Citation:
Li G, Deng L, Chua Y, Li P, Neftci EO and Li H (2020) Editorial: Spiking Neural Network Learning, Benchmarking, Programming and Executing. Front. Neurosci. 14:276. doi: 10.3389/fnins.2020.00276

ADAM optimization (Kingma and Ba, 2014). Other techniques such as batch normalization (Ioffe and Szegedy, 2015) and distributed training (Dean et al., 2012) facilitate learning in DNNs and enable them to be applied in various real-world applications. In comparison, there are relatively fewer SNN learning algorithms and techniques. Existing SNN learning algorithms fall into three categories: unsupervised learning algorithms such as the original spike timing dependent plasticity (STDP) (Querlioz et al., 2013; Diehl and Cook, 2015; Kheradpisheh et al., 2016), indirect supervised learning such as ANN-to-SNN conversion (O'Connor et al., 2013; Pérez-Carrasco et al., 2013; Diehl et al., 2015; Sengupta et al., 2019), and direct supervised learning such as spatiotemporal backpropagation (Wu et al., 2018, 2019a,b). We note that progress in STDP research includes introducing a reward or supervision signal such as spike timing which, in combination with this third factor, dictates the weight changes (Paugam-Moisy et al., 2006; Franosch et al., 2013). Despite the progress made, no algorithm can yet train a very deep SNN efficiently, which has become almost the holy grail of our field. Below, we briefly summarize the accepted algorithm papers in this Research Topic.

Inspired by the mammalian olfactory system, Borthakur and Cleland develop an SNN model trained using STDP for signal restoration and identification. It is broadly applicable to sensor array inputs. Luo et al. propose a new weight update mechanism that adjusts synaptic weights, leading to the first wrong output spike-timing to classify input spike trains with time-sensitive information accurately. He et al. divide the learning (weight training) process into two phases: the structure formation phase using Hebb's rule, and the parameter training phase using STDP and reinforcement learning, so as to form an SNN-based associative memory system. In contrary to just training synaptic weights, Wang et al. propose training both the synaptic weights and delays using gradient descent so as to achieve better performance. Based on a structurally fixed small SNN with sparse recurrent connections, Ponghiran et al. use Q-learning to train only its output layer so as to achieve human-level performance on complex reinforcement learning tasks such as Atari games. Their research demonstrates that a small random recurrent SNN is able to provide a computationally efficient alternative to state-of-art deep reinforcement learning networks with several layers of trainable parameters. The above works have made good progress toward better performing SNN learning algorithms. We believe that further progress will be made in this field in the future.

MODEL EFFICIENCY

In recent years, hardware oriented DNN compression techniques have been proposed that offer significant memory saving and hardware acceleration (Han et al., 2015a, 2016; Zhang et al., 2016; Huang et al., 2017; Aimar et al., 2018). At present, many compression techniques are proposed that provide a trade-off between processing efficiency and application accuracy (Han et al., 2015b; Novikov et al., 2015; Zhou et al., 2016). Such an approach has also caught on in the design of SNN accelerators (Deng et al., 2019), with the following contribution in this Research Topic. Afshar et al. investigate how a hardware-efficient variant of STDP may be used for event-based feature extraction. Using a rigorous testing framework, a range of spatio-temporal kernels with different surface decaying methods, decay functions, receptive field sizes, feature numbers, and backend classifiers are evaluated. This detailed investigation provides useful insight and heuristics with regards to the trade-off between performance and complexity while using the STDP rule. Pedroni et al. study the impact of different arrangements of synaptic connectivity tables on weight storage and STDP updates for large-scale neuromorphic systems. Based on their analysis, they present an alternative formulation of STDP via a delayed causal update mechanism that permits efficient weight storage and access for both full and sparse connectivity.

Other than model complexity, several other papers focus on direct compression of SNNs. Soures and Kudithipudi propose Deep-LSM, a combination of randomly connected hidden layers and unsupervised winner-take-all layers to capture network dynamics followed by an attention modulated readout layer for classification. The connections between hidden layers and winner-take-all layers are partially trained using STDP. Their SNN model is applied in a first-person video activity recognition task, achieving state-of-the-art performance with >90% memory and operation saving compared to the long-short term memory (LSTM). Based on a single fully-connected layer with the STDP learning rule, Shi et al. propose a soft-pruning method that sets a fraction of the weights to the lower bound during training, effectively achieving >75% pruning. Srinivasan and Roy implement spiking convolutional layers comprising of binary weight kernels which are trained using probabilistic STDP, as well as non-spiking fully-connected layers which are trained using gradient descent. A residual convolutional SNN is proposed, which achieves >20x model compression.

PROGRAMMING TOOLS

Programming Tools have been one of the key components driving development in ANN research, examples of which include Theano (Al-Rfou et al., 2016), TensorFlow (Abadi et al., 2016), Caffe (Jia et al., 2014) and Pytorch (Paszke et al., 2019), MXNet (Chen et al., 2015), Keras (Chollet, 2015). These user-friendly programming tools enable researchers to build and train large-scale ANNs using only basic programming know-how. In comparison, the programming tools for SNNs are rather limited. To the best of our knowledge, only SpiNNaker (Furber et al., 2014), BindsNET (Hazan et al., 2018), and PyNN (Davison et al., 2009) provide a basic programming interface to support simple and small SNN simulations. Generally researchers have to build an SNN from the ground up which can be time-consuming and require significantly more programming know-how. Thus, developing user-friendly programming tools to efficiently deploy a large scale SNN is imperative to the advancement of our field. In this Research Topic, an open-source high-speed SNN simulation framework based on PyTorch has been proposed. SpykeTorch (Mozafari et al.) simulates convolutional SNNs with at most one spike per neuron (rank-order coding scheme), and STDP-based learning rules. Although programming tools for SNNs are still in their infancy, we believe that more research needs to be done so that the training of SNNs may approach the efficiency of training ANNs.

NEUROMORPHIC HARDWARES

Recent advances made in modeling SNNs *in-silico*, as demonstrated by Neurogrid of Stanford University (Benjamin et al., 2014), BrainScales of Heidelberg University (Schemmel et al., 2012), SpiNNaker of University of Manchester, Tianjic of Tsinghua University (Pei et al., 2019), IBM's TrueNorth (Akopyan et al., 2015), and Intel's Loihi (Davies et al., 2018), attest to the great potential of hardware implementation of SNNs. In this Research Topic, Shukla et al. re-model large-scale CNNs so as to mitigate hardware constraints and implement them on the IBM TrueNorth. A CNN used for car detection and counting was demonstrated, with reasonable accuracy compared to a GPU trained CNN but with much lower energy consumption. Bohnstingl et al. implement a learning-to-learn SNN on a neuromorphic chip which accelerates the learning process by extracting abstract knowledge from prior experiences. Other than conventional CMOS circuits, emerging devices such as memristors have also been studied in this Research Topic. Guo et al. propose a STDP-based greedy training algorithm for SNNs to reduce weight levels and enhance robustness toward device non-idealities. They demonstrate online learning on a resistive random access memory (RRAM) system with non-ideal behaviors. Fang et al. propose a generalized swarm intelligence model on SNN: the SI-SNN. SNNs are implemented as agents in swarm intelligence with interactive modulation and synchronization. They implement such neural dynamics on a ferroelectric field-effect transistor (FeFET) based hardware platform to solve optimization problems with high performance and efficiency.

CONCLUSIONS

In conclusion, it is believed that SNNs achieve superior performance in processing complex, sparse, and noisy spatiotemporal information with high power efficiency exploiting neural dynamics in the event-driven regime. Event-driven communication is particularly attractive in enabling energy efficient AI systems with in-memory computing that will play an important role in ubiquitous intelligence. SNN research is ongoing, and much more progress is to be expected in its learning algorithms, benchmarking framework, programming tools, and efficient hardware. Through cross-discipline exchange of ideas and collaborative research, we hope to build a truly energy-efficient and intelligent machine. This Research Topic is but a small step in this direction; we look forward to more disruptive ideas that distinguish SNNs and neuromorphic computing from the mainstream machine learning approaches in the near future.

AUTHOR CONTRIBUTIONS

All authors listed have made a substantial, direct and intellectual contribution to the work, and approved it for publication.

FUNDING

This work was partially supported by National Key R&D Program of China (No. 2018AAA0102600 and 2018YFE0200200), Beijing Academy of Artificial Intelligence (BAAI), Initiative Scientific Research Program, and a grant from the Institute for Guo Qiang, TsinghuaUniversity, and key scientific technological innovation research project by Ministry of Education, and Tsinghua–Foshan Innovation Special Fund.

REFERENCES

Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., et al. (2016). "Tensorflow: a system for large-scale machine learning," in *12th {USENIX} Symposium on Operating Systems Design and Implementation ({OSDI} 16)* (Savannah, GA), 265–283.

Aimar, A., Mostafa, H., Calabrese, E., Rios-Navarro, A., Tapiador-Morales, R., Lungu, I.-A., et al. (2018). Nullhop: a flexible convolutional neural network accelerator based on sparse representations of feature maps. *IEEE Trans. Neural Netw. Learn. Syst.* 30, 644–656. doi: 10.1109/TNNLS.2018.2852335

Akopyan, F., Sawada, J., Cassidy, A., Alvarez-Icaza, R., Arthur, J., Merolla, P., et al. (2015). Truenorth: design and tool flow of a 65 mw 1 million neuron programmable neurosynaptic chip. *IEEE Trans. Comput. Aided Des. Integr. Circuits Syst.* 34, 1537–1557. doi: 10.1109/TCAD.2015.2474396

Al-Rfou, R., Alain, G., Almahairi, A., Angermueller, C., Bahdanau, D., Ballas, N., et al. (2016). Theano: a Python framework for fast computation of mathematical expressions. *arXiv [preprint] arXiv*: 1605.02688.

Benjamin, B. V., Gao, P., McQuinn, E., Choudhary, S., Chandrasekaran, A. R., Bussat, J.-M., et al. (2014). Neurogrid: a mixed-analog-digital multichip system for large-scale neural simulations. *Proc. IEEE* 102, 699–716. doi: 10.1109/JPROC.2014.2313565

Bottou, L. (2012). "Stochastic gradient descent tricks," in *Neural Networks: Tricks of the Trade*, eds G. Montavon, G. B. Orr, and K-R. Müller (Berlin; Heidelberg: Springer), 421–436.

Chen, T., Li, M., Li, Y., Lin, M., Wang, N., Wang, M., et al. (2015). Mxnet: a flexible and efficient machine learning library for heterogeneous distributed systems. *arXiv [preprint] arXiv*: 1512.01274.

Chollet, F. (2015). *Keras* [Online]. Available online at: https://keras.io.

Davies, M., Srinivasa, N., Lin, T.-H., Chinya, G., Cao, Y., Choday, S. H., et al. (2018). Loihi: a neuromorphic manycore processor with on-chip learning. *IEEE Micro* 38, 82–99. doi: 10.1109/MM.2018.112130359

Davison, A. P., Brüderle, D., Eppler, J. M., Kremkow, J., Muller, E., Pecevski, D., et al. (2009). PyNN: a common interface for neuronal network simulators. *Front. Neuroinform.* 2:11. doi: 10.3389/neuro.11.011.2008

Dean, J., Corrado, G., Monga, R., Chen, K., Devin, M., Mao, M., et al. (2012). "Large scale distributed deep networks," in *Advances in Neural Information Processing Systems* (Lake Tahoe, NV), 1223–1231.

Deng, L., Wu, Y., Hu, Y., Liang, L., Li, G., Hu, X., et al. (2019). Comprehensive SNN compression using ADMM optimization and activity regularization. *arXiv [preprint] arXiv*: 1911.00822.

Diehl, P. U., and Cook, M. (2015). Unsupervised learning of digit recognition using spike-timing-dependent plasticity. *Front. Comput. Neurosci.* 9:99. doi: 10.3389/fncom.2015.00099

Diehl, P. U., Neil, D., Binas, J., Cook, M., Liu, S.-C., and Pfeiffer, M. (2015). "Fast-classifying, high-accuracy spiking deep networks through weight and threshold balancing," in *2015 International Joint Conference on Neural Networks (IJCNN)* (Killarney: IEEE), 1–8.

Franosch, J. M. P., Urban, S., and van Hemmen, J. L. (2013). Supervised spike-timing-dependent plasticity: a spatiotemporal neuronal learning rule for function approximation and decisions. *Neural Comput.* 25, 3113–3130. doi: 10.1162/NECO_a_00520

Furber, S. B., Galluppi, F., Temple, S., and Plana, L. A. (2014). The spinnaker project. *Proc. IEEE* 102, 652–665. doi: 10.1109/JPROC.2014.2304638

Han, S., Liu, X., Mao, H., Pu, J., Pedram, A., Horowitz, M. A., et al. (2016). EIE: efficient inference engine on compressed deep neural network. *ACM SIGARCH Comput. Architect. News* 44, 243–254. doi: 10.1145/3007787.3001163

Han, S., Mao, H., and Dally, W. J. (2015a). Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding. *arXiv [preprint] arXiv*: 1510.00149.

Han, S., Pool, J., Tran, J., and Dally, W. (2015b)."Learning both weights and connections for efficient neural network,"in *Advances in Neural Information Processing Systems*, 1135–1143.

Hazan, H., Saunders, D. J., Khan, H., Patel, D., Sanghavi, D. T., Siegelmann, H. T., et al. (2018). Bindsnet: a machine learning-oriented spiking neural networks library in python. *Front. Neuroinform.* 12:89. doi: 10.3389/fninf.2018.00089

Hecht-Nielsen, R. (1992). "Theory of the backpropagation neural network,"" in *Neural Networks for Perception*, ed H. Wechsler (Elsevier), 65–93.

Huang, H., Ni, L., Wang, K., Wang, Y., and Yu, H. (2017). A highly parallel and energy efficient three-dimensional multilayer CMOS-RRAM accelerator for tensorized neural network. *IEEE Trans. Nanotechnol.* 17, 645–656. doi: 10.1109/TNANO.2017.2732698

Ioffe, S., and Szegedy, C. (2015). Batch normalization: accelerating deep network training by reducing internal covariate shift. *arXiv [preprint] arXiv*:1502.03167.

Jia, Y., Shelhamer, E., Donahue, J., Karayev, S., Long, J., Girshick, R., et al. (2014). "Caffe: convolutional architecture for fast feature embedding," in *Proceedings of the 22nd ACM International Conference on Multimedia* (Orlando, FL), 675–678.

Jouppi, N. P., Young, C., Patil, N., Patterson, D., Agrawal, G., Bajwa, R., et al. (2017). "In-datacenter performance analysis of a tensor processing unit," in *Proceedings of the 44th Annual International Symposium on Computer Architecture* (Toronto, ON), 1–12.

Kheradpisheh, S. R., Ganjtabesh, M., and Masquelier, T. (2016). Bio-inspired unsupervised learning of visual features leads to robust invariant object recognition. *Neurocomputing* 205, 382–392. doi: 10.1016/j.neucom.2016.04.029

Kingma, D. P., and Ba, J. (2014). Adam: a method for stochastic optimization. *arXiv [preprint] arXiv*:1412.6980.

Novikov, A., Podoprikhin, D., Osokin, A., and Vetrov, D. P. (2015). "Tensorizing neural networks," in *Advances in Neural Information Processing Systems* (Montreal, QC), 442–450.

O'Connor, P., Neil, D., Liu, S.-C., Delbruck, T., and Pfeiffer, M. (2013). Real-time classification and sensor fusion with a spiking deep belief network. *Front. Neurosci.*7:178. doi: 10.3389/fnins.2013.00178

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., et al. (2019). "PyTorch: an imperative style, high-performance deep learning library," in *Advances in Neural Information Processing Systems* (Vancouver, BC), 8024–8035.

Paugam-Moisy, H., Martinez, R., and Bengio, S. (2006). *A supervised learning approach based on STDP and polychronization in spiking neuron networks*, IDIAP, EPFL.

Pei, J., Deng, L., Song, S., Zhao, M., Zhang, Y., Wu, S., et al. (2019). Towards artificial general intelligence with hybrid Tianjic chip architecture. *Nature* 572, 106–111. doi: 10.1038/s41586-019-1424-8

Pérez-Carrasco, J. A., Zhao, B., Serrano, C., Acha, B., Serrano-Gotarredona, T., Chen, S., et al. (2013). Mapping from frame-driven to frame-free event-driven vision systems by low-rate rate coding and coincidence processing–application to feedforward ConvNets. *IEEE Trans. Pattern Anal. Mach. Intell.* 35, 2706–2719. doi: 10.1109/TPAMI.2013.71

Querlioz, D., Bichler, O., Dollfus, P., and Gamrat, C. (2013). Immunity to device variations in a spiking neural network with memristive nanodevices. *IEEE Trans. Nanotechnol.* 12, 288–295. doi: 10.1109/TNANO.2013.22 50995

Roy, K., Jaiswal, A., and Panda, P. (2019). Towards spike-based machine intelligence with neuromorphic computing. *Nature* 575, 607–617. doi: 10.1038/s41586-019-1677-2

Schemmel, J., Grübl, A., Hartmann, S., Kononov, A., Mayr, C., Meier, K., et al. (2012). "Live demonstration: a scaled-down version of the brainscales wafer-scale neuromorphic system," in *2012 IEEE International Symposium on Circuits and Systems* (Seoul: IEEE), 702–702.

Sengupta, A., Ye, Y., Wang, R., Liu, C., and Roy, K. (2019). Going deeper in spiking neural networks: vgg and residual architectures. *Front. Neurosci.* 13:95. doi: 10.3389/fnins.2019.00095

Wu, J., Chua, Y., Zhang, M., Yang, Q., Li, G., and Li, H. (2019a). "Deep spiking neural network with spike count based learning rule," in *2019 International Joint Conference on Neural Networks (IJCNN)* (Budapest: IEEE), 1–6.

Wu, Y., Deng, L., Li, G., Zhu, J., and Shi, L. (2018). Spatio-temporal backpropagation for training high-performance spiking neural networks. *Front. Neurosci.* 12:331. doi: 10.3389/fnins.2018.00331

Wu, Y., Deng, L., Li, G., Zhu, J., Xie, Y., and Shi, L. (2019b). "Direct training for spiking neural networks: faster, larger, better," in *Proceedings of the AAAI Conference on Artificial Intelligence* (Honolulu, HI), 1311–1318.

Zhang, S., Du, Z., Zhang, L., Lan, H., Liu, S., Li, L., et al. (2016). "Cambricon-x: an accelerator for sparse neural networks," in *2016 49th Annual IEEE/ACM International Symposium on Microarchitecture (MICRO)* (Taipei: IEEE), 1–12.

Zhou, S., Wu, Y., Ni, Z., Zhou, X., Wen, H., and Zou, Y. (2016). Dorefa-net: training low bitwidth convolutional neural networks with low bitwidth gradients. *arXiv [preprint] arXiv*: 1606.06160.

Conflict of Interest: The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.