Qualitative Benchmarking of Deep Learning Hardware and Frameworks: Review and Tutorial

Wei Dai and Daniel Berleant
Department of Information Science
University of Arkansas at Little Rock
Little Rock, Arkansas, USA
{wxdai, jdberleant}@ualr.edu

***Abstract*— Previous survey papers offer knowledge of deep learning hardware devices and software frameworks. This paper introduces benchmarking principles, surveys machine learning devices including GPUs, FPGAs, and TPUs, and reviews deep learning software frameworks. It also reviews these technologies with respect to benchmarking from the angles of our 7-metric approach to frameworks and 12-metric approach to hardware platforms.**

After reading the paper, the audience will understand seven benchmarking principles, generally know that differential characteristics of mainstream AI devices, qualitatively compare deep learning hardware through our 12-metric approach for benchmarking hardware, and read benchmarking results of 16 deep learning frameworks via our 7-metric set for benchmarking frameworks.

Keywords*— *Deep Learning hardware, machine learning, neural network frameworks

I. INTRODUCTION

After developing for about 75 years, deep learning technologies are still maturing. In July 2018, Gartner, an IT research and consultancy company, pointed out that deep learning technologies are in the Peak-of-Inflated-Expectations (PoIE) stage on the Gartner Hype Cycle diagram [1] as shown in Figure 2, which means deep learning networks trigger many industry projects as well as research topics [2][3].

Benchmarking is useful for both industry and academia. The definition from the Oxford English Dictionary [4] states that a benchmark is "To evaluate or check (something) by comparison with an established standard." Deep learning networks are leading technologies that extend their computing performance and capability based on flexibility, distributed architectures, creative algorithms, and large volume datasets.

Even though previous research papers provide knowledge of deep learning, it is hard to find a survey discussing qualitative benchmarks for machine learning hardware devices and deep learning software frameworks as shown in Figure 1. In this paper we introduce 12 qualitative benchmarking metrics for hardware devices and seven metrics for software frameworks in deep learning, respectively. Also, the paper provides qualitative benchmark results for major deep learning devices, and compares more than 16 deep learning frameworks.

According to [16],[17], and [18], there are seven vital characteristics for benchmarks. These key properties are:

[1] Relevance: Benchmarks should measure relatively vital features.

[2] Representativeness: Benchmark performance metrics should be broadly accepted by industry and academia.

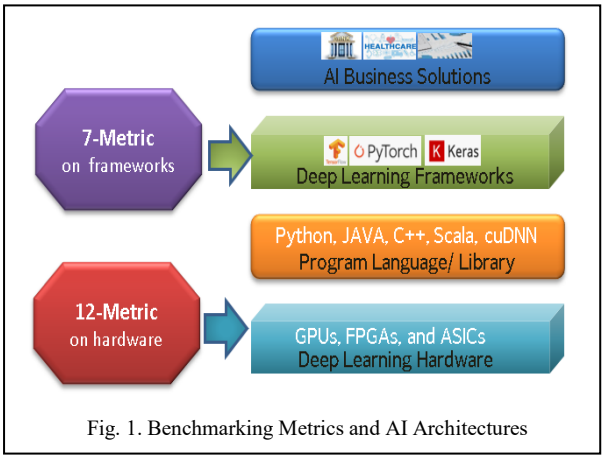

Fig. 1. Benchmarking Metrics and AI Architectures

[3] Equity: All systems should be fairly compared.

[4] Repeatability: Benchmark results can be verified.

[5] Cost-effectiveness: Benchmark tests are economical.

[6] Scalability: Benchmark tests should measure from single server to multiple servers.

[7] Transparency: Benchmark metrics should be easily to understand.

After evaluating artificial intelligence (AI) hardware and deep learning frameworks, we can discover strengths and weaknesses of deep learning technologies. So, the paper is organized as follows.

Section II reviews GPUs, FPGAs, and ASICs, qualitative metrics of benchmarking hardware, and qualitative results on benchmarking devices.

Section III introduces qualitative metrics for benchmarking frameworks and results.

Section IV presents our conclusions.

Section V discusses future work

II. MACHINE LEARNING HARDWARE

Machine Learning devices, including graphics processing units (GPUs), field-programmable gate arrays (FPGAs), and application-specific integrated circuits (ASICs), have potential to expedite machine learning algorithms because of parallel computing, high-speed internal memory, and specific libraries of hardware devices.

A. GPU Devices

GPUs are specified unitary processors that are dedicated to accelerating real time three-dimensional (3D) graphics. GPUs contain an internal cache, high speed bandwidth, and quick parallel performance. The GPU cache accelerates matrix multiplication routines because these routines do not need to access global memory.

GPUs are universal hardware devices for deep learning. After testing neural networks including with 200 hidden layers on MNIST handwritten data sets, GPUs'

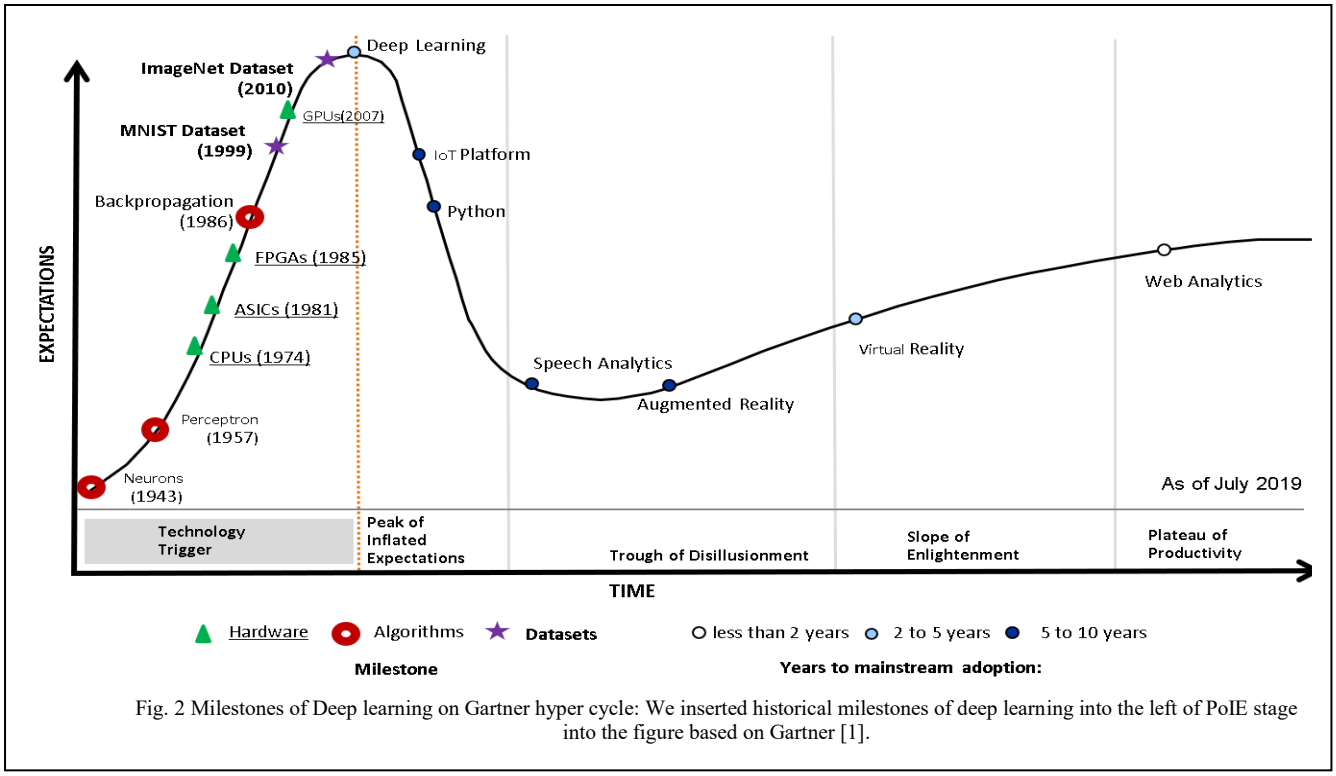

Fig. 2 Milestones of Deep learning on Gartner hyper cycle: We inserted historical milestones of deep learning into the left of PoIE stage into the figure based on Gartner [1].

performance was found to be better than CPUs [5]. The test results show NVIDIA GeForce 6800 Ultra has 3.3X speed-up compared to the Intel 3GHz P4; ATI Radeon X800 has 2.4-3.4X. In the computer industry, FLOPS means floating-point operations per second. NVIDIA GPUs increase FLOPS performance. In [6], a single NVIDIA GeForce 8800 GTX, released in November 2006, had 575 CUDA cores with 345.6 gigaflops, and its memory bandwidth was 86.4 GB/s; by September 2018, a NVIDIA GeForce RTX 2080 Ti [7] had 4,352 CUDA cores with 13.4 Teraflops, and its memory bandwidth was 616 GB/s.

B. FPGA Devices

FPGAs have dynamical hardware configurations, so hardware engineers developed FPGAs using hardware description language (HDL), including VHDL or Verilog [8][9]. However, some end-user cases are energy-sensitive scenarios, such as self-driving vehicles. FPGA devices offer better performance-per-watt than GPUs. According to [10], while comparing gigaflops per watt, FPGA devices often have 3x-4x times speed-up compared to GPUs. After comparing performances of FPGAs and GPUs [11] on ImageNet 1K data sets [12], Ovtcharov et al. confirmed that the FPGA devices named Arria 10 GX1150 handle about 233 images/sec. while device power is 25 watts. In comparison, NVIDIA K40 GPUs handle 500-824 images/sec. while device power is 235 watts. Briefly, [11] demonstrates FPGAs can process 9.3 images/joule, but these GPUs can only process 2.1-3.4 image/joule.

C. ASIC Devices

Usually, ASIC devices have high throughout and low energy-consumption because ASICs are fabricated chips designed for special applications instead of generic tasks. While testing AlexNet, one of the convolutional neural networks, the Eyeriss consumed 278 mW [12]. Furthermore, the Eyeriss achieved 125.9 images/joule (with a batch size of N equals four) [13]. In [6], Google researchers confirm that the TPU 1.0, based on ASIC technologies, has about 15X-30X speed-up compared to GPUs or CPUs during the same period, with TOPS/watt of about 30X - 80X better.

D. Enhance Hardware Performance

Even though multiple cores, CPUs, and hyper-threading are mainstream technologies, these technologies still show weaknesses in the big data era. For example, deep learning models usually have products and matrix transpositions [5], so that these algorithms require intensive computing resources. GPUs, FPGAs, and ASICs have better computing performance with lower latency than conventional CPUs because these specialized chipsets consist of many multiple cores and on-chip memory. The memory hierarchy on these hardware devices is usually separated into two layers: 1) off-chip memory, named global memory or main memory; and 2) on-chip memory titled local memory or shared memory. After copying data from global memory, deep learning algorithms can use high-speed shared memory to expedite computing performance. Specific program libraries provide dedicated application programming interfaces (APIs) of hardware devices, abstract complex parallel programming, and increased executive performance. For instance, the CuDNN library, released by NVIDIA, can improve performance of the Apache MXNet and the Caffe on NVIDIA GPUs [14][11].

Traditionally, multi-cores, improved I/O bandwidth, and increased core clock speed can improve hardware speeds [15]. In Figure 3, Arithmetic Logic Unit (ALU), single instruction, multiple data (SIMD), and single Instruction, multiple thread (SIMT) systems concurrently execute multiply-accumulate (MACs) tasks based on shared memory and configuration files.

However, there are new algorithms to improve computing performance. GPUs are low-latency temporary storage architectures, so the Toeplitz matrix, fast Fourier transform (FFT), Winograd and Strassen algorithms can be used for improving performance of GPUs [15]. Data

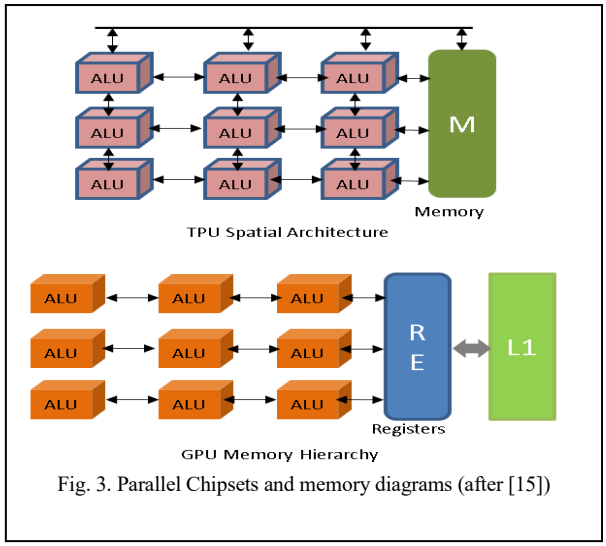

Fig. 3. Parallel Chipsets and memory diagrams (after [15])

movement consumes energy. FPGAs and ASICs are spatial architectures. These devices contain low-energy on-chip memory, so that reusable dataflow algorithms provide solutions for reducing data movements. Weight stationary dataflow, output stationary dataflow, no local reuse dataflow, and row stationary dataflow were developed for decreasing energy consumption of FPGAs and ASICs [15]. In addition, co-design of deep learning algorithms and hardware devices are other approaches. According to [15], there are two solutions. 1) Decrease precision. There are several algorithms to decrease precision of operations and operands of DNN, such as 8-bit fixed point, binary weight sharing, and log domain quantization. 2) Reduce number of operations and model size. Some algorithms need to be highlighted, such as exploiting activation statistics, network pruning algorithms, and knowledge distillation algorithms.

E. Qualitative Benchmarking Metrics on Machine Learning Hardware

GPUs, FPGAs, and ASICs can be used in different domains including cloud servers and edge devices. There are 12 qualitative benchmarking metrics we distinguish on machine learning devices as follows. In addition, the results of the benchmarks are shown in Table I.

1) *Computing Performance can be measured by FLOPS. For measuring ASICs and GPUs chipset, a quadrillion (thousand trillion) floating point operations per second (petaflops) are used in testing modern chipsets. In May 2017, Google announced Tensor Processor Unit 2.0 (TPU 2.0), which provides 11.5 petaFlops per chip[16]. TPU 3.0 released in May 2018 offers 23.0 petaFlops [17]. However, NVIDIA GeForce RTX 2080 Ti has 13.4 TeraFlops [7]. According to [10] and [18], ASICs have the best FLOPs, and GPUs are better than FPGAs.*
2) *Low Latency describes an important chipset capability [19], and is distinguished from throughout [6]. In [6][10], ASICs have the lowest latency, and FPGAs are lower than GPUs.*
3) *Energy Efficiency in Computing is highly important for edge nodes because mobile devices usually have limited power. In [6][10] ASICs have the highest energy efficiency, and FPGAs and GPUs come in second and third, respectively.*
4) *Compatibility means devices can be supported by multiple deep learning frameworks and popular programming languages. FPGAs needs specially developing libraries, so that FPGAs are not that good in compatibility. GPUs have the best compatibilities [10]. ASICs currently are second. For example, TPUs support TensorFlow, cafe, etc.*
5) *Die Size means chipset size. Dimensional size relates to chip density. ASICs are over 50 times denser than FPGAs. In [10], FPGAs are better than GPUs in comparison of chipset density.*
6) *Research Costs mean the total costs for developing devices incurred from designing architectures, developing algorithms, and deploying chip sets on hardware devices. GPUs are affordable devices [10]. ASICs are expensive, and FPGAs are between GPUs and ASICs.*
7) *Research Risks are defined by hardware architectures, development risks, and deploying chip sets. ASICs have the highest risks before scaling on markets. FPGAs are very dynamic, so that their risks are limited. GPUs are in the middle.*
8) *Upgradability is a challenge for most hardware devices. In [10], GPUs are the most flexible after deployment, and GPUs are better than FPGAs. ASICs are the most difficult to update after delivery.*
9) *Scalability means hardware devices can scale out quickly with low costs. Scalability is vital for clouds and data centres. ASICs have excellent scalability. GPUs have good scalability, but worse than ASICs. FPGAs are the lowest on this dimension.*
10) *Chip Price means price of each unit chip after industrial-scale production. In [20], FPGAs have the highest chip cost after production scale-up. ASICs have the lowest cost, and GPUs are in the middle.*
11) *Ubicomp (also named Ubiquitous Computation) means hardware devices can be used for varied use cases including either large scale clouds or low energy mobile devices. FPAGs are very flexible, so that the devices can be used in different industries and scientific fields. ASICs usually are dedicated to specific industry needs. GPUs like FPAGs can be developed for many research fields and industry domains.*
12) *Time-to-Market means the length of time from design to sale of products. According to [9],[10], and [20], FPGAs and GPUs have less development time than ASICs.*

TABLE I. QUALITATIVE BENCHMARKING HARDWARE OF MACHINE LEARNING

#	Attributes	ASICs	FPGAs	GPUs
1	Computing Performance	High	Low	Moderate
2	Low Latency	High	Moderate	Low
3	Energy efficiency	High	Moderate	Good
4	Compatibility	Low	Moderate	High
5	Die Size	High	Moderate	Low
6	Research Costs	High	Moderate	Low
7	Research Risks	High	Low	Moderate
8	Upgradability	Low	Moderate	High
9	Scalability	High	Low	Moderate
10	Chip Price	Low	High	Moderate
11	Ubicomp	Low	High	High
12	Time-to-Market	Low	High	High

III. MAINSTREAM DEEP LEARNING FRAMEWORKS

Open source deep learning frameworks allow engineers and scientists to define activation functions, develop special algorithms, train big data, and deploy neural networks on different hardware platforms from x86 servers to mobile devices.

Based on the wide variety of usages, support teams, and development interfaces, we split 18 frameworks into three sets including mature frameworks, developing frameworks, and inactive frameworks. The 10 mature frameworks can be used now to enhance training speed, improve scalable performance, and reduce development risks. The developing frameworks are not broadly used in industries or research projects, but some developing frameworks could be used in specific fields. Retired frameworks refer to inactive frameworks.

A. Mature Frameworks

1) *Caffe and Facebook Caffe2: Caffe[21] was developed by the University of California, Berkeley in C++. According to [22], Caffe can be used on FPGA platforms. Caffe2 [23] is an updated framework supported by Facebook.*

2) *Chainer Framework: Chainer [24], written in Python, can be extended to multiple nodes and GPU platformws through the CuPy and MPI4Python libraries [25][26].*

3) *DyNet Framework: DyNet [27] was written in C++. The framework can readily define dynamic computation graphs, so DyNet can help improve development speed. Currently, DyNet only supports single nodes instead of multiple nodes.*

4) *MXNet: the Apache MXNet [28][29] is a famous deep learning framework. This framework was built in C++, and MXNet supports NVIDIA GPUs through the NVIDIA CuDNN library. In [30], the GLUNO is a development interface for MXNet.*

5) *Microsoft CNTK: The Microsoft Cognitive Toolkit (Microsoft CNTK)[31][32], funded by Microsoft and written in C++, supports distributed platforms.*

6) *Google TensorFlow: In 2011, Google released DistBelief [33], but the framework was not an open source project. In 2016, the project was merged with TensorFlow[34][35], an open source deep learning framework.*

7) *Keras [36][37] is a Python library for TensorFlow, Theano, and Microsoft CNTK. Keras has a reasonable development interface that can help developers to quickly develop demo systems and reduce development costs and risks.*

8) *Neon and PlaidML are partially supported by Intel: Neon [38], supported by Nervana Systems and Intel, may improve performance for deep learning on diverse platforms. PLaidML[39] was released by Vertex.AI in 2017; Intel will soon fund PlaidML.*

9) *PyTorch Framework: PyTorch [40][41] written in Python can be integrated with Jupyter Notebook. Furthermore, FastAI [42] is another development interface for PyTorch.*

10) *Theano Framework: The core language of Theano [43][44] is Python with a BSD license. Lasagne [45][46] is an additional development library for Theano.*

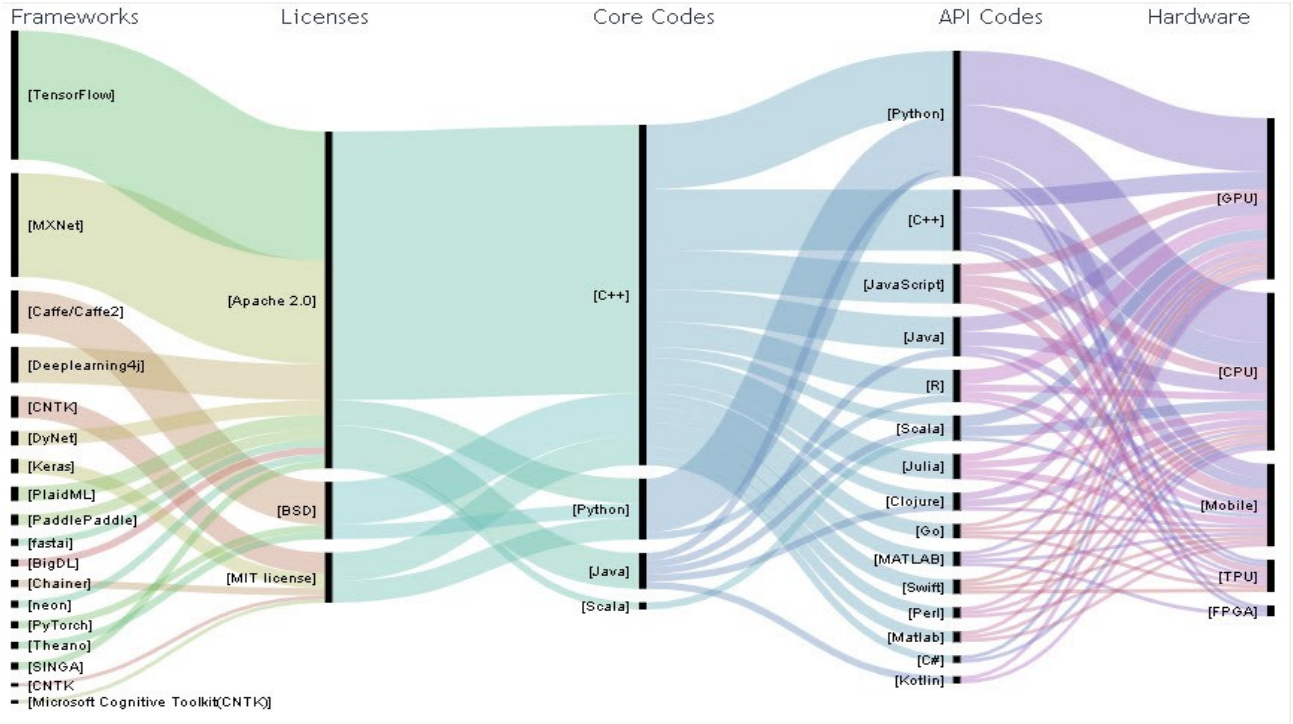

Fig. 4. Popular Deep learning Frameworks

B. Developing Frameworks

In addition, some deep learning frameworks are less frequently mentioned by academic papers because of their limited functions. For example, Apache SINGA[47] was developed in C++. The framework is supported by Apache group [44] [45]. BigDL [46][47], built with Scale codes, is a deep learning framework that can run on Apache Spark or Apache Hadoops. In [52], the authors mentioned DeepLearning4J (DL4J), which can be accelerated by cuDNN. PaddlePaddle-based AI was developed by BaiDu company with Python [53].

C. Inactive Frameworks

Torch [54], written in Lua, is inactive, so this framework will retire soon. Purine[53][54] also will be retired because the open source was no longer updated after 2014.

D. Qualitative Benchmarking Metrics for Frameworks for Deep Learning

Benchmarking Metrics on Frameworks for Deep Learning include seven qualitative metrics described next.

1) *License Type: licenses of open source software offer a variety of restrictions. Apache license 2.0 has more restrictions; MIT license requires less limitations. BSD is in the middle.*
2) *Core codes: C++ codes offer high performance, and are good for high performance projects. Python and Java provide quick development and easy maintenance. Scala is programming language useful for large-scale projects.*
3) *Interface Codes (also called API codes): API codes can reduce developing costs and enhance functions of the framwork.*
4) *Compatible hardware: The more different hardware devices a deep learning framework can run on, the better it is on this dimension.*
5) *Stability: For avoiding single points of failure, a mature framework might run on multi-server platforms rather than a single node.*
6) *Tested Deep Learning Networks: If a framwork can be officially verified by a variety of deep learning networks, then the framework is correspondingly more suitable as mainstream framework.*
7) *Tested Datasets: If a framwork was verifed by diverse datasets, we are able to know its performance, strengths and weaknesses.*

After comparing these seven metrics, there are 16 mainstreaming deep learning frameworks as shown in Figure 4 and Table II (shown after the references).

IV. Conclusions

Deep learning is an increasingly popular technology. This technology can be used in image classification, speech recognition, and language translation. In addition, deep learning technology is continually developing. Many innovative chipsets, useful frameworks, creative models, and big data sets are emerging, resulting in extending markets and usages for deep learning.

While deep learning technology is expanding, it is useful to understand the dimensions and methods for measuring deep learning hardware and software. Benchmarking principles include relevance, representativeness, equity, repeatability, affordable cost, scalability, and transparency. Major deep learning hardware platform types include GPUs, FPGAs, and ASICs. We discussed machine learning devices, and mentioned approaches that enhance performance of these devices. In addition, we listed 12 qualitative benchmarking features for comparing deep learning hardware.

Software frameworks for deep learning are diverse. We compared more than 16 mainstream frameworks through license types, compliant hardware devices, and tested networks. Popular deep learning frameworks are split into three parts: mature deep learning frameworks, developing frameworks, and retired frameworks.

V. Future Work

Deep learning technology including supporting hardware devices and software frameworks is increasing in importance, so scientists and engineers are developing new hardware and creative frameworks. We are creating a website named AI Performance (https://aiperf.org) for collecting and updating results of benchmarking hardware and frameworks. Everyone are able to access the website for sharing benchmarking deep learning.

Acknowledgment

We are grateful to Google for awarding $5,000 in computing credits for this project in 2019.

References

[1] J. H. Peter Krensky, "Hype Cycle for Data Science and Machine Learning, 2018," *Gartner Company*, 2018. [Online]. Available: https://www.gartner.com/doc/3883664/hype-cycle-data-science-machine.

[2] W. Dai, K. Yoshigoe, and W. Parsley, "Improving data quality through deep learning and statistical models," in *Advances in Intelligent Systems and Computing*, 2018.

[3] W. Dai and N. Wu, "Profiling essential professional skills of chief data officers through topic modeling algorithms," in *AMCIS 2017 - America's Conference on Information Systems: A Tradition of Innovation*, 2017, vol. 2017-Augus.

[4] O. E. D. Online, "Oxford English Dictionary Online," *Oxford English Dict.*, 2010.

[5] D. Steinkraus, I. Buck, and P. Y. Simard, "Using GPUs for machine learning algorithms," in *Eighth International Conference on Document Analysis and Recognition (ICDAR'05)*, 2005.

[6] J. D. Owens, M. Houston, D. Luebke, S. Green, J. E. Stone, and J. C. Phillips, "GPU computing," *Proc. IEEE*, vol. 96, no. 5, pp. 879–899, 2008.

[7] "Graphics Reinvented: NVIDIA GeForce RTX 2080 Ti Graphics Card," *NVIDIA*. NVIDIA COMPANY.

[8] D. Galloway, "The Transmogrifier C hardware description language and compiler for FPGAs," *Proc. IEEE Symp. FPGAs Cust. Comput. Mach.*, 1995.

[9] G. Lacey, G. W. Taylor, and S. Areibi, "Deep Learning on FPGAs: Past, Present, and Future," *arXiv Prepr. arXiv1602.04283*, 2016.

[10] BERTEN DSP, "GPU vs FPGA Performance Comparison," *Proceedings of the 2017 ACM/SIGDA International Symposium on Field-Programmable Gate Arrays - FPGA '17*, 2016.

[11] K. Ovtcharov, O. Ruwase, J. Kim, J. Fowers, K. Strauss, and E. S. Chung, "Accelerating Deep Convolutional Neural Networks Using Specialized Hardware," *Microsoft Res. Whitepaper*, 2015.

[12] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet Classification with Deep Convolutional Neural Networks," *Adv. Neural Inf. Process. Syst.*, 2012.

[13] Y. H. Chen, T. Krishna, J. S. Emer, and V. Sze, "Eyeriss: An

Energy-Efficient Reconfigurable Accelerator for Deep Convolutional Neural Networks," *IEEE J. Solid-State Circuits*, 2017.

[14] Mxn. Developers, "Apache MXNet(incubating) - A Flexible and Efficient Library for Deep Learning," *Apache*, 2018. [Online]. Available: https://mxnet.apache.org/.

[15] V. Sze, Y. H. Chen, T. J. Yang, and J. S. Emer, "Efficient Processing of Deep Neural Networks: A Tutorial and Survey," *Proceedings of the IEEE*. 2017.

[16] "Google reveals more details about its second-gen TPU AI chips," *techcrunch*. [Online]. Available: https://www.theinquirer.net/inquirer/news/3023202/google-reveals-more-details-about-its-second-gen-tpu-ai-chips.

[17] "Google announces a new generation for its TPU machine learning hardware," *techcrunch*. [Online]. Available: https://techcrunch.com/2018/05/08/google-announces-a-new-generation-for-its-tpu-machine-learning-hardware/.

[18] M. Parker, "Understanding Peak Floating-Point Performance Claims," *Intel FPGA White Paper*, 2016.

[19] D. A. Patterson, "LATENCY LAGS BANDWITH.," *Commun. ACM*, 2004.

[20] E. Vansteenkiste, "New FPGA Design Tools and Architectures," Ghent University. Faculty of Engineering and Architecture, 2016.

[21] Y. Jia *et al.*, "Caffe: Convolutional architecture for fast feature embedding," in *Proceedings of the 22nd ACM international conference on Multimedia*, 2014, pp. 675–678.

[22] J. Xu, Z. Liu, J. Jiang, Y. Dou, and S. Li, "CaFPGA: An automatic generation model for CNN accelerator," *Microprocess. Microsyst.*, 2018.

[23] "Caffe2, GitHub Repository," 2018. [Online]. Available: https://caffe2.ai/.

[24] C. Developers, "Chainer Repository," *GitHub repository*, 2018. [Online]. Available: https://github.com/chainer.

[25] S. Tokui, K. Oono, S. Hido, and J. Clayton, "Chainer: a Next-Generation Open Source Framework for Deep Learning," in *Proceedings of Workshop on Machine Learning Systems (LearningSys) in The Twenty-ninth Annual Conference on Neural Information Processing Systems (NIPS)*, 2015.

[26] T. Akiba, K. Fukuda, and S. Suzuki, "ChainerMN: Scalable Distributed Deep Learning Framework," in *Proceedings of Workshop on ML Systems in The Thirty-first Annual Conference on Neural Information Processing Systems (NIPS)*, 2017.

[27] G. Neubig *et al.*, "Dynet: The dynamic neural network toolkit," *arXiv Prepr. arXiv1701.03980*, 2017.

[28] T. Chen *et al.*, "MXNet: A Flexible and Efficient Machine Learning Library for Heterogeneous Distributed Systems," *arXiv Prepr. arXiv1512.01274*, 2015.

[29] Mxn. J. Developers, "MXNetJS Deep Learning in Browser," *GitHub repository*, 2018. [Online]. Available: https://github.com/dmlc/mxnet.js/.

[30] "GLUON," 2018. [Online]. Available: https://gluon.mxnet.io/index.html.

[31] "Microsoft2018CNTK," 2018. [Online]. Available: https://www.microsoft.com/en-us/cognitive-toolkit/.

[32] F. Seide and A. Agarwal, "CNTK: Microsoft's open-source deep-learning toolkit," in *22nd ACM International Conference on Knowledge Discovery and Data Mining (KDD)*, 2016.

[33] J. Dean *et al.*, "Large Scale Distributed Deep Networks," *Adv. Neural Inf. Process. Syst.*, 2012.

[34] "Tensorflow: An open source library," 2018. [Online]. Available: https://www.tensorflow.org/.

[35] M. Abadi *et al.*, "TensorFlow : A System for Large-Scale Machine Learning," *Proc 12th USENIX Conf. Oper. Syst. Des. Implement.*, 2016.

[36] C. François, "Keras," *https://github.com/fchollet/keras*, 2015. [Online]. Available: https://keras.io/.

[37] Chollet François, "Keras: The Python Deep Learning library," *keras.io*. 2015.

[38] "Neon, GitHub Repository," 2018. [Online]. Available: https://github.com/NervanaSystems/neon.

[39] C. Ng, "Announcing PlaidML: Open Source Deep Learning for Every Platform," 2017.

[40] "PyTorch: An open source deep learning platform," 2018. [Online]. Available: https://pytorch.org/.

[41] A. Paszke *et al.*, "Automatic differentiation in PyTorch," *31st Conf. Neural Inf. Process. Syst.*, 2017.

[42] "FastAI, GitHub Repository," 2018. [Online]. Available: https://www.fast.ai/.

[43] "Theano," 2018. [Online]. Available: http://deeplearning.net/software/theano/.

[44] R. Al-Rfou *et al.*, "Theano: A Python framework for fast computation of mathematical expressions," *arXiv Prepr.*, 2016.

[45] "Lasagne, GitHub Repository," 2018. [Online]. Available: https://github.com/Lasagne/Lasagne.

[46] B. Van Merriënboer *et al.*, "Blocks and fuel: Frameworks for deep learning," *arXiv Prepr. arXiv1506.00619*, 2015.

[47] B. C. Ooi *et al.*, "SINGA: A Distributed Deep Learning Platform," *Proc. 23rd ACM Int. Conf. Multimed. - MM '15*, 2015.

[48] W. Wang *et al.*, "SINGA : Putting Deep Learning in the Hands of Multimedia Users," *Multimedia*, 2015.

[49] A. G. T. N. Daniel Dai Ted Dunning, "Apache SINGA," *Apache*, 2018. [Online]. Available: https://singa.incubator.apache.org/.

[50] "BigDL," *Apache*, 2018. [Online]. Available: https://github.com/intel-analytics/BigDL.

[51] Yiheng Wang *et al.*, "BigDL: A Distributed Deep Learning Framework for Big Data," *arXiv Prepr. arXiv1804.05839*, 2018.

[52] "Deeplearning4j: Open-source distributed deep learning for the jvm," *Apache Softw. Found. Licens.*, 2018.

[53] B. Company, "PaddlePaddle-based AI." [Online]. Available: http://en.paddlepaddle.org/.

[54] "Torch,GitHub repository," 2018. [Online]. Available: https://github.com/torch/torch7.

[55] "Purine, GitHub Repository," 2018. [Online]. Available: https://github.com/purine/purine2.

[56] M. Lin, S. Li, X. Luo, and S. Yan, "Purine: A bi-graph based deep learning framework," *arXiv Prepr. arXiv1412.6249*, 2014.

TABLE II. COMPARING POPULAR DEEP LEARNING FRAMEWORKS

#	Frameworks [a]	License Type [b]	Core Codes	API Codes	Hardware Devices	Multi-Server	Tested Networks	Related Datasets
1	BigDL	Apache 2.0	Scala	Scala	CPU/GPU	Multi-Server	VGG,Inception,ResNet,GoogleNet	ImageNet, CIFAR-10
2	Caffe/Caffe2	BSD License	C++	Python, C++ MATLAB	CPU/GPU /FPGA/Mobile	Multi-Server	LeNet, RNN	CIFAR-10,MNIST, ImageNet
3	Chainer	MIT License	Python	Python	CPU/GPU	Multi-Server	RNN	CIFAR-10, ImageNet
4	DeepLearning4j	Apache 2.0	Java	Java, Scala, Clojure, Python, Kotlin	CPU/GPU	Multi-Server	AlexNet,LeNet,Inception, ResNet, RNN, LSTM, VGG,Xception,	ImageNet
5	DyNet	Apache 2.0	C++	C++, Python	CPU/GPU	Single Node	RNN, LSTM	ImageNet
6	FastAI	Apache 2.0	Python	Python	CPU/GPU	Multi-Server	ResNet	CIFAR-10, ImageNet
7	Keras	MIT License	Python	Python, R	CPU/GPU	Multi-Server	CNN, RNN	CIFAR-10,MNIST
8	Microsoft CNTK	MIT License	C++	C++, C#, Python, Java	CPU/GPU	Multi-Server	CNN, RNN,LSTM	CIFAR-10, MNIST,ImageNet,P-VOC
9	MXNet	Apache 2.0	C++	C++, Python, Clojure, Julia, Perl, R, Scala, Java,JavaScript,Matlab	CPU/GPU /Mobile	Multi-Server	CNN, RNN,Inception	CIFAR-10, MNIST,ImageNet,P-VOC
10	Neon	Apache 2.0	Python	Python	CPU/GPU	Multi-Server	AlexNet, ResNet, LSTM	CIFAR-10, mnist,ImageNet
11	PaddlePaddle	Apache 2.0	Python	Python	CPU/GPU /Mobile	Multi-Server	AlexNet,GoogleNet,LSTM	CIFAR-10, ImageNet
12	PlaidML	Apache 2.0	C++	Python, C++	CPU/GPU	Multi-Server	Inception, ResNet, VGG, Xception, MobileNet, DenseNet, ShuffleNet, LSTM	CIFAR-10, ImageNet
13	PyTorch	BSD License	Python	Python	CPU/GPU	Multi-Server	AlexNet,Inception, ResNet, VGG, DenseNet, SqueezeNet	CIFAR-10, ImageNet
14	SINGA	Apache 2.0	C++	Python	CPU/GPU	Multi-Server	RNN, AlexNet,DenseNet, GoogleNet, Inception, ResidualNet,VGG	MNIST, ImageNet
15	TensorFlow	Apache 2.0	C++	Python, C++, Java, Go, JavaScript, R, Julia, Swift, JavaScript	CPU/GPU /TPU/Mobile	Multi-Server	AlexNet,Inception, ResNet, VGG, LeNet, MobileNet	CIFAR-10, mnist,ImageNet
16	Theano	BSD License	Python	Python (Keras)	CPU/GPU	Multi-Server	AlexNet, VGG, GoogleNet	CIFAR-10, ImageNet

[a] alphabetical order

[b] In License Type column, Apache 2.0 means the Apache 2.0 license