MVAPICH
MPI, PGAS and Hybrid MPI+PGAS Library

HiDL
High-Performance Deep Learning

Latest version of the slides can be obtained from

http://www.cse.ohio-state.edu/~panda/sea18-dl.pdf

High Performance Distributed Deep Learning: A Beginner's Guide

A Tutorial at SEA Symposium '18

by

Dhabaleswar K. (DK) Panda

The Ohio State University

E-mail: panda@cse.ohio-state.edu

http://www.cse.ohio-state.edu/~panda

Ammar Ahmad Awan

The Ohio State University

E-mail: awan.10@osu.edu

http://www.cse.ohio-state.edu/~awan.10

Hari Subramoni

The Ohio State University

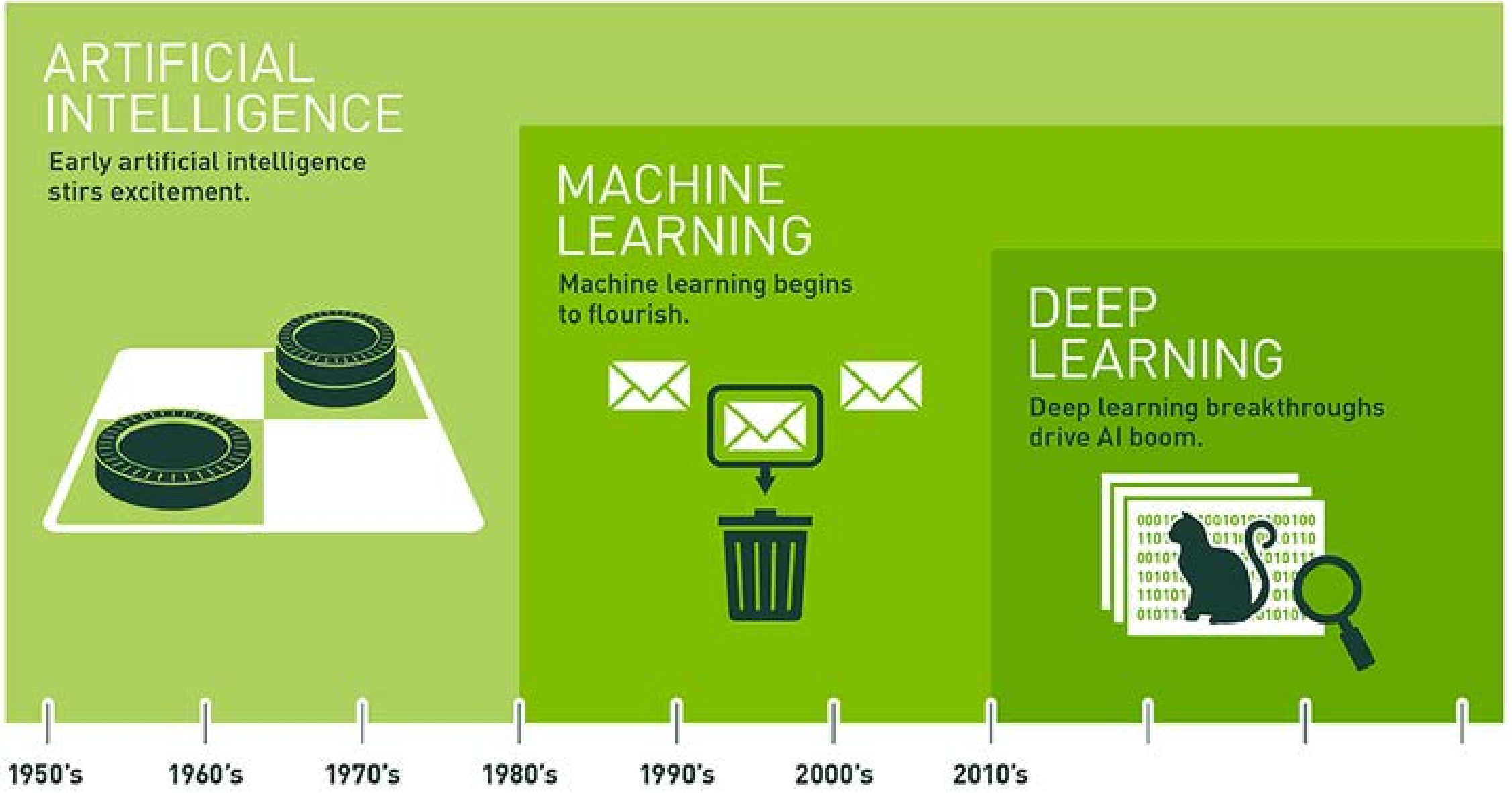
E-mail: subramon@cse.ohio-state.edu

http://www.cse.ohio-state.edu/~subramon

Outline

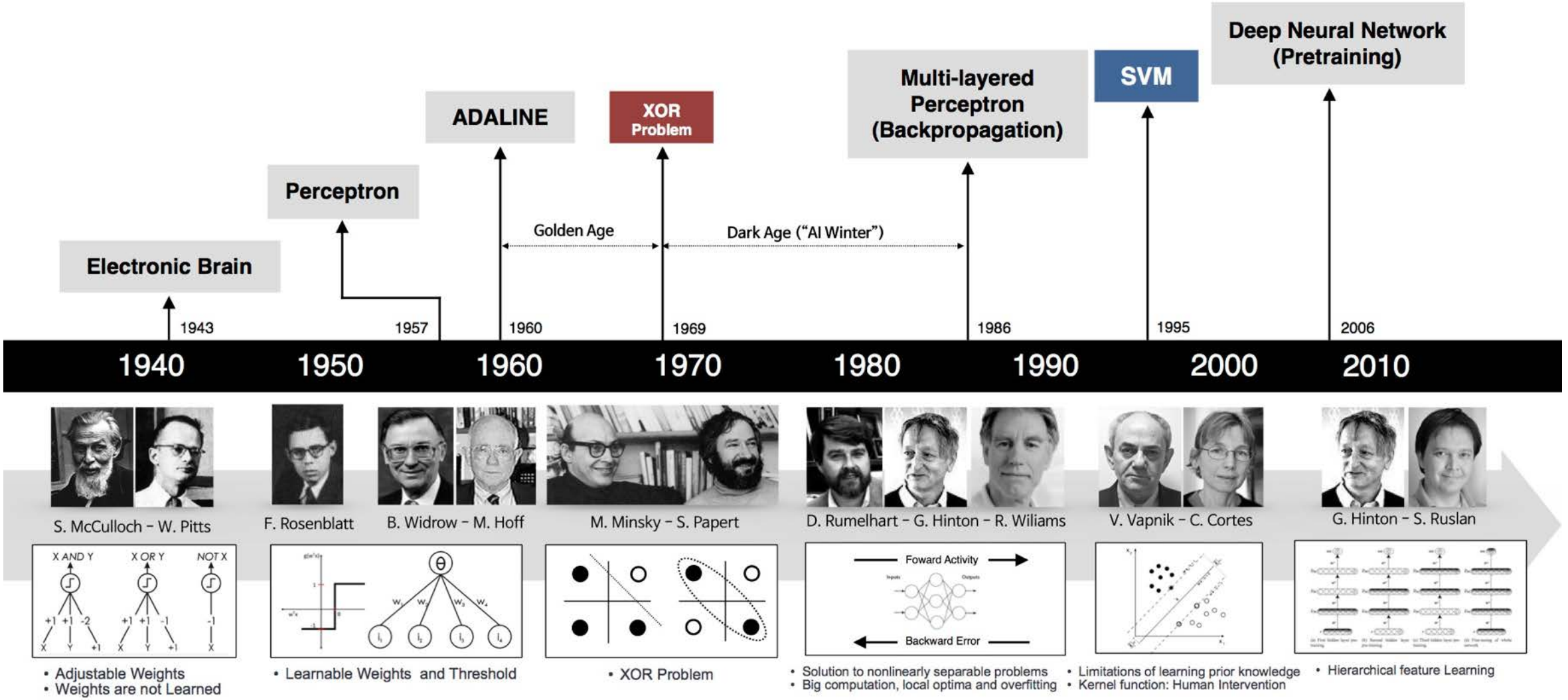
- **Introduction**
  - **The Past, Present, and Future of Deep Learning**
  - What are Deep Neural Networks?
  - Diverse Applications of Deep Learning
  - Deep Learning Frameworks
- Overview of Execution Environments
- Parallel and Distributed DNN Training
- Latest Trends in HPC Technologies
- Challenges in Exploiting HPC Technologies for Deep Learning
- Solutions and Case Studies
- Open Issues and Challenges
- Conclusion

Brief History of Deep Learning (DL)



Courtesy: http://www.zdnet.com/article/caffe2-deep-learning-wide-ambitions-flexibility-scalability-and-advocacy/

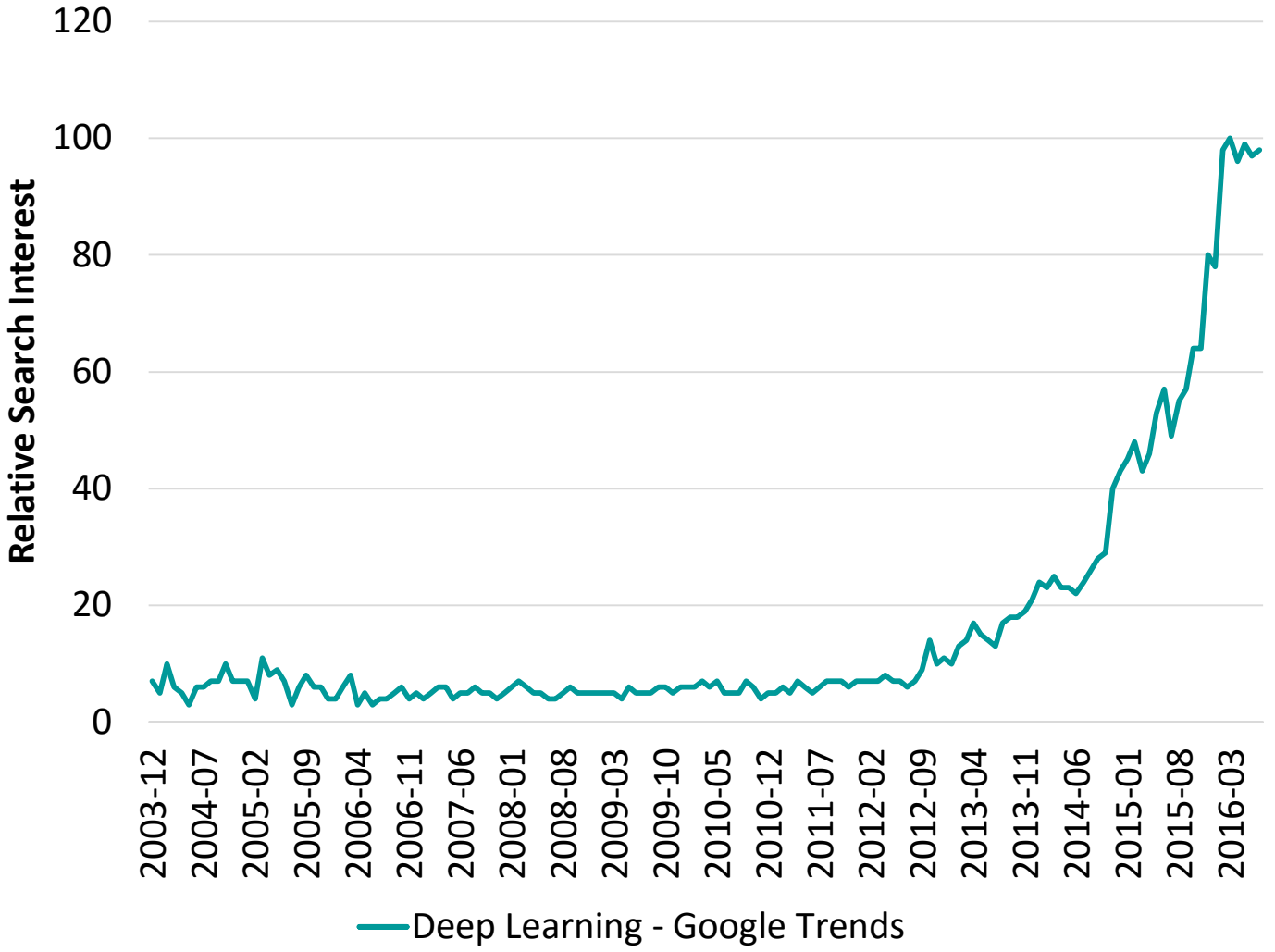
Milestones in the Development of Neural Networks



Courtesy: https://beamandrew.github.io/deeplearning/2017/02/23/deep_learning_101_part1.html

Deep Learning Resurgence

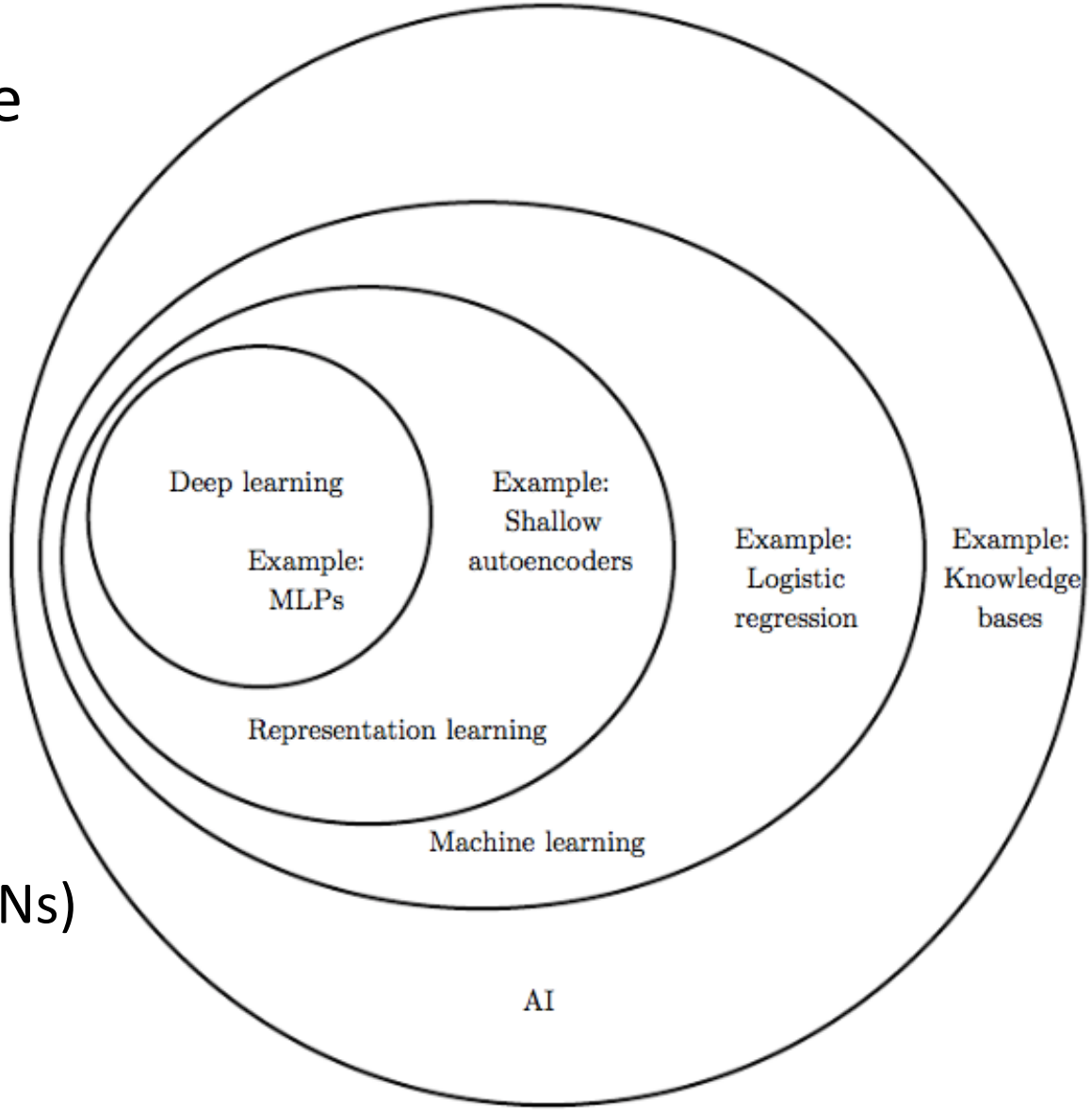
- Deep Learning is going through a resurgence
- Excellent accuracy for deep/convolutional neural networks
- Public availability of versatile datasets like MNIST, CIFAR, and ImageNet
- Widespread popularity of accelerators like NVIDIA GPUs



Courtesy: http://trends.google.com

Understanding the Deep Learning Resurgence

- Deep Learning is a sub-set of Machine Learning
  - But, it is perhaps the most radical and revolutionary subset
  - Automatic feature extraction vs. hand-crafted features
- Deep Learning
  - A renewed interest and a lot of hype!
  - Key success: Deep Neural Networks (DNNs)
  - Everything was there since the late 80s except the "**computability of DNNs**"

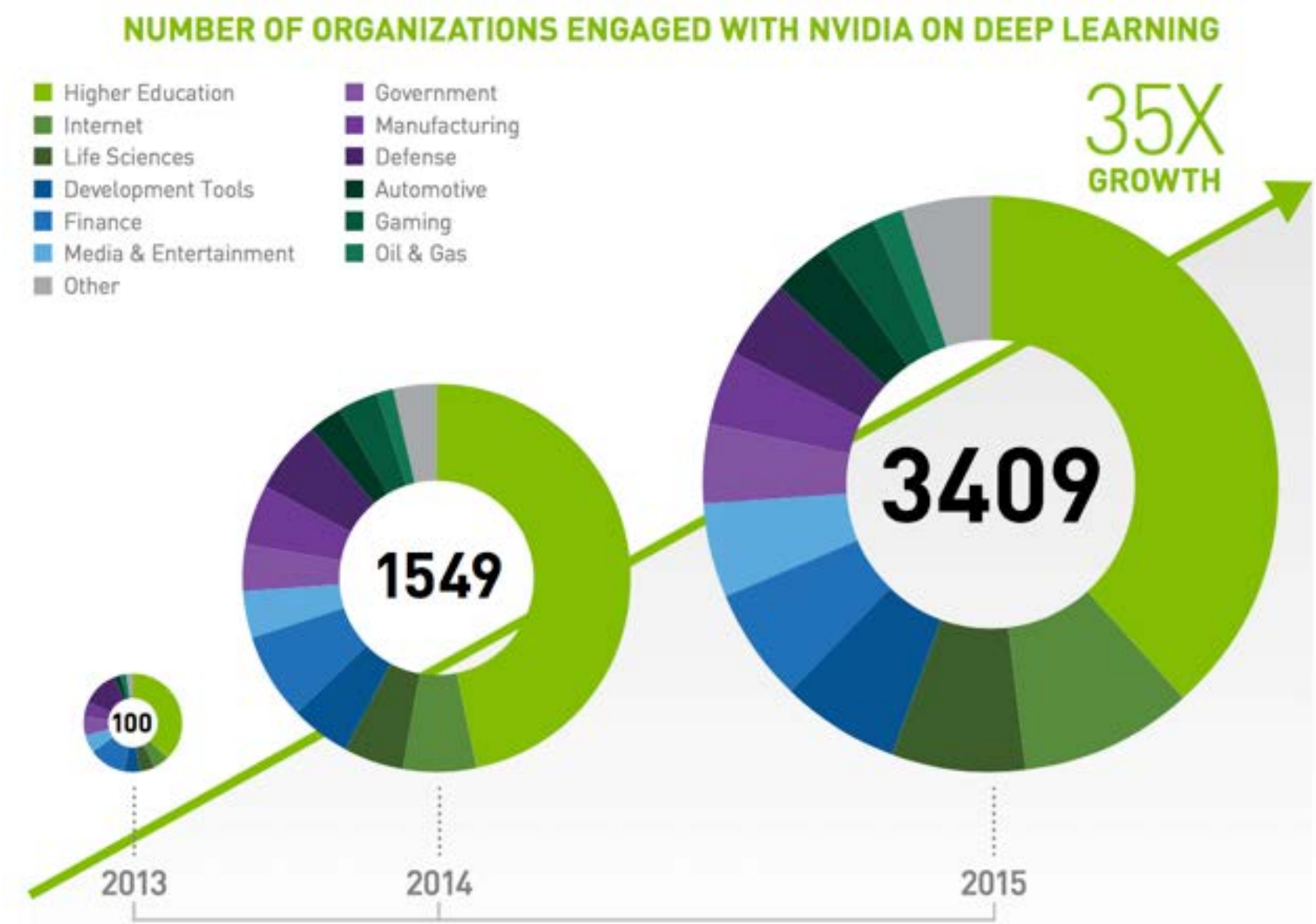


Courtesy: http://www.deeplearningbook.org/contents/intro.html

Resurgence of Deep Learning in the Many-core Era

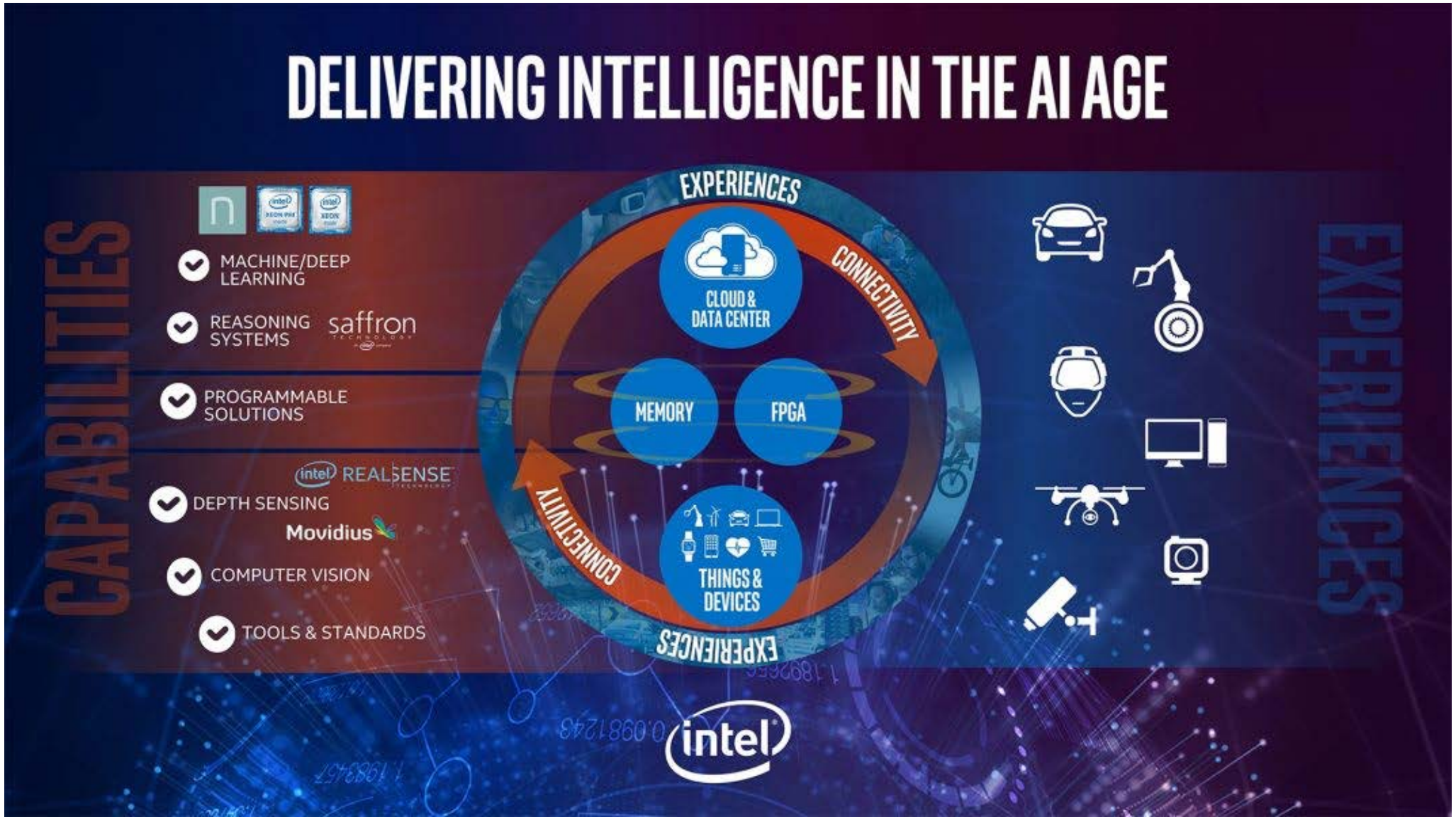
- *Computability of DNNs* made possible by modern and efficient hardware
  - Many DNN training tasks were impossible to compute!
  - GPUs are at the core of DNN training performance!
- Availability of Datasets
  - MNIST - http://yann.lecun.com/exdb/mnist/
  - CIFAR10 - https://www.cs.toronto.edu/~kriz/cifar.html
  - ImageNet - https://www.image-net.org
  - Street View House Numbers (SVHN) - http://ufldl.stanford.edu/housenumbers/
  - Several others..
- Excellent Accuracy for classical Machine Learning problems
  - Case study: 30 years of research vs. proposed Neural Machine Translation (NMT)
  - https://arxiv.org/abs/1703.01619

The Rise of GPU-based Deep Learning



Courtesy: http://images.nvidia.com/content/technologies/deep-learning/pdf/NVIDIA-DeepLearning-Infographic-v11.pdf

Intel is committed to AI and Deep Learning as well!

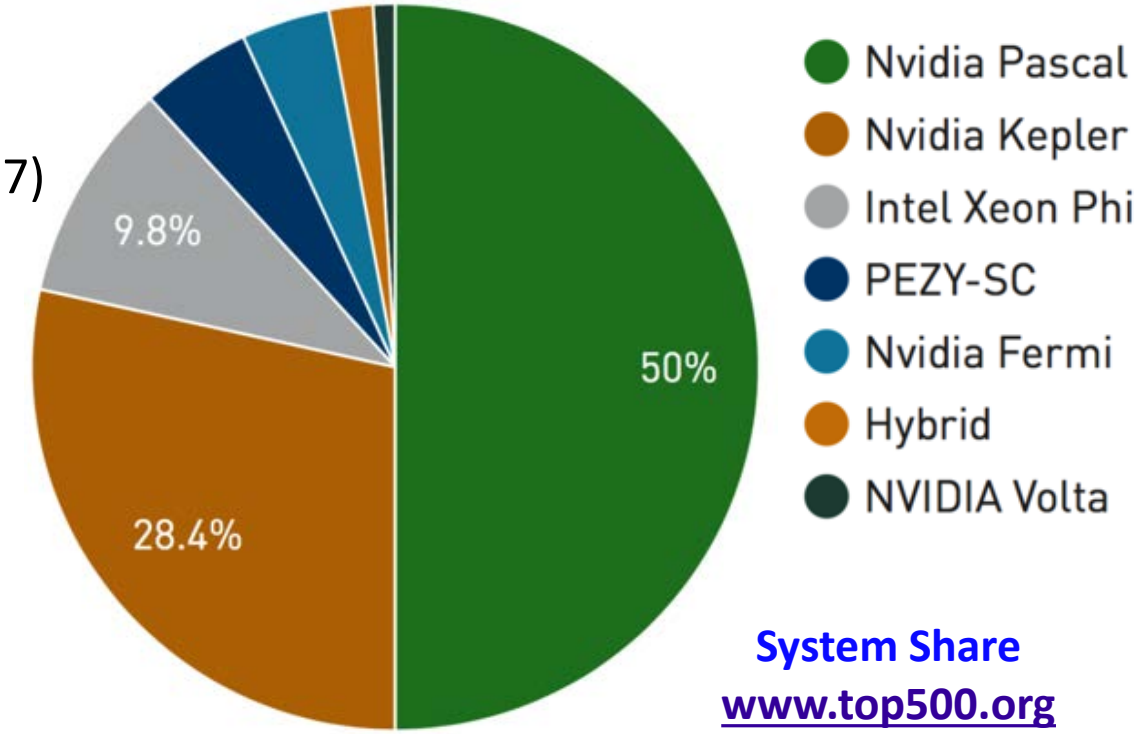


Courtesy: https://newsroom.intel.com/editorials/krzanich-ai-day/

Deep Learning, Many-cores, and HPC

- Nvidia GPUs are the main driving force for faster training of DL models
  - The ImageNet Challenge - (ILSVRC)
  - 90% of the ImageNet teams used GPUs in 2014*
  - Deep Neural Networks (DNNs) like AlexNet, GoogLeNet, and VGG are used
  - A natural fit for DL due to the throughput-oriented nature

- In the High Performance Computing (HPC) arena
  - 85/500 Top HPC systems use NVIDIA GPUs (Nov '17)
  - CUDA-Aware Message Passing Interface (MPI)
  - NVIDIA Kepler, Pascal, and Volta architecture
  - DGX-1, DGX1-V (Volta), and DGX-2
    - Dedicated DL super-computers

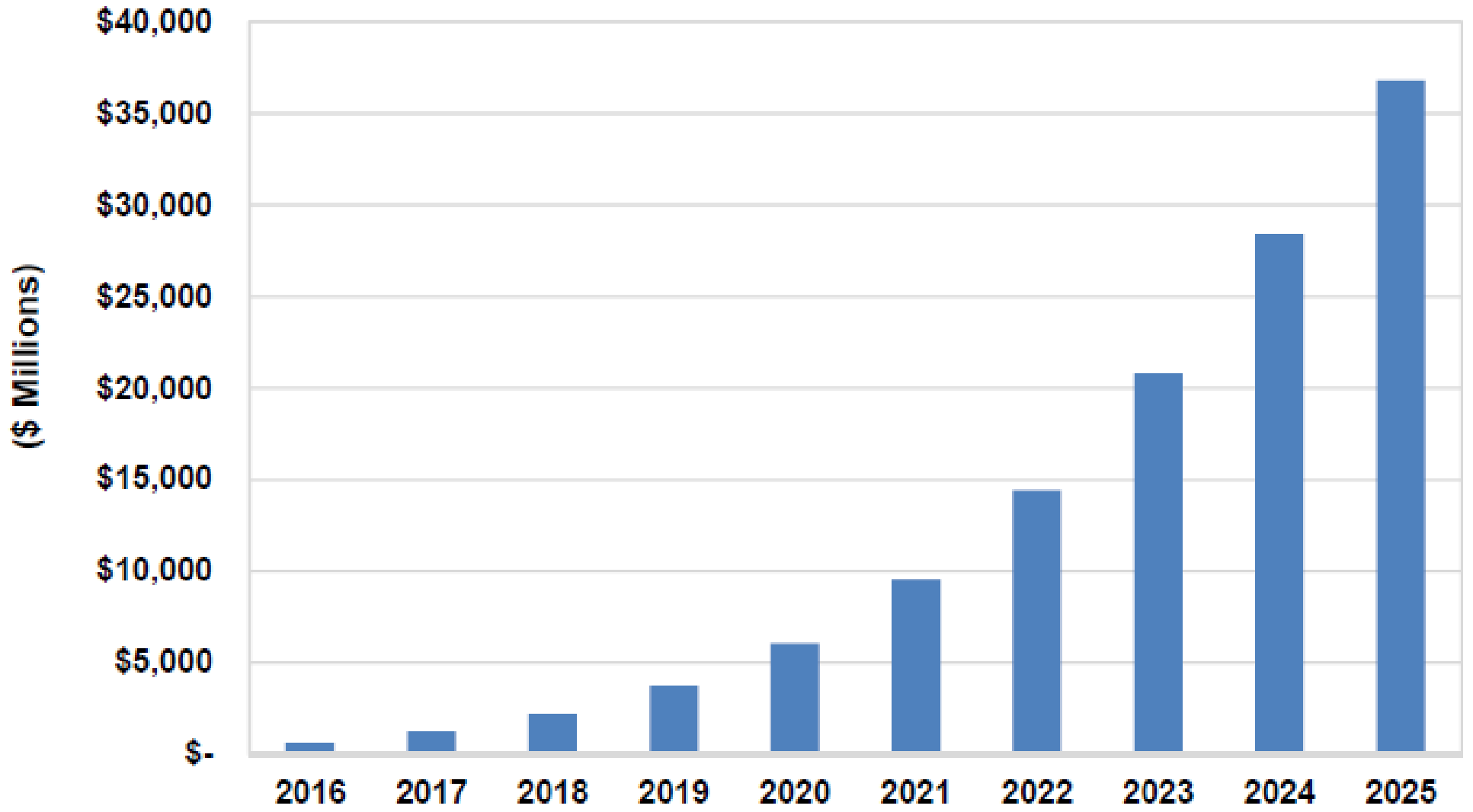


System Share
www.top500.org

*https://blogs.nvidia.com/blog/2014/09/07/imagenet/

The Bright Future of Deep Learning

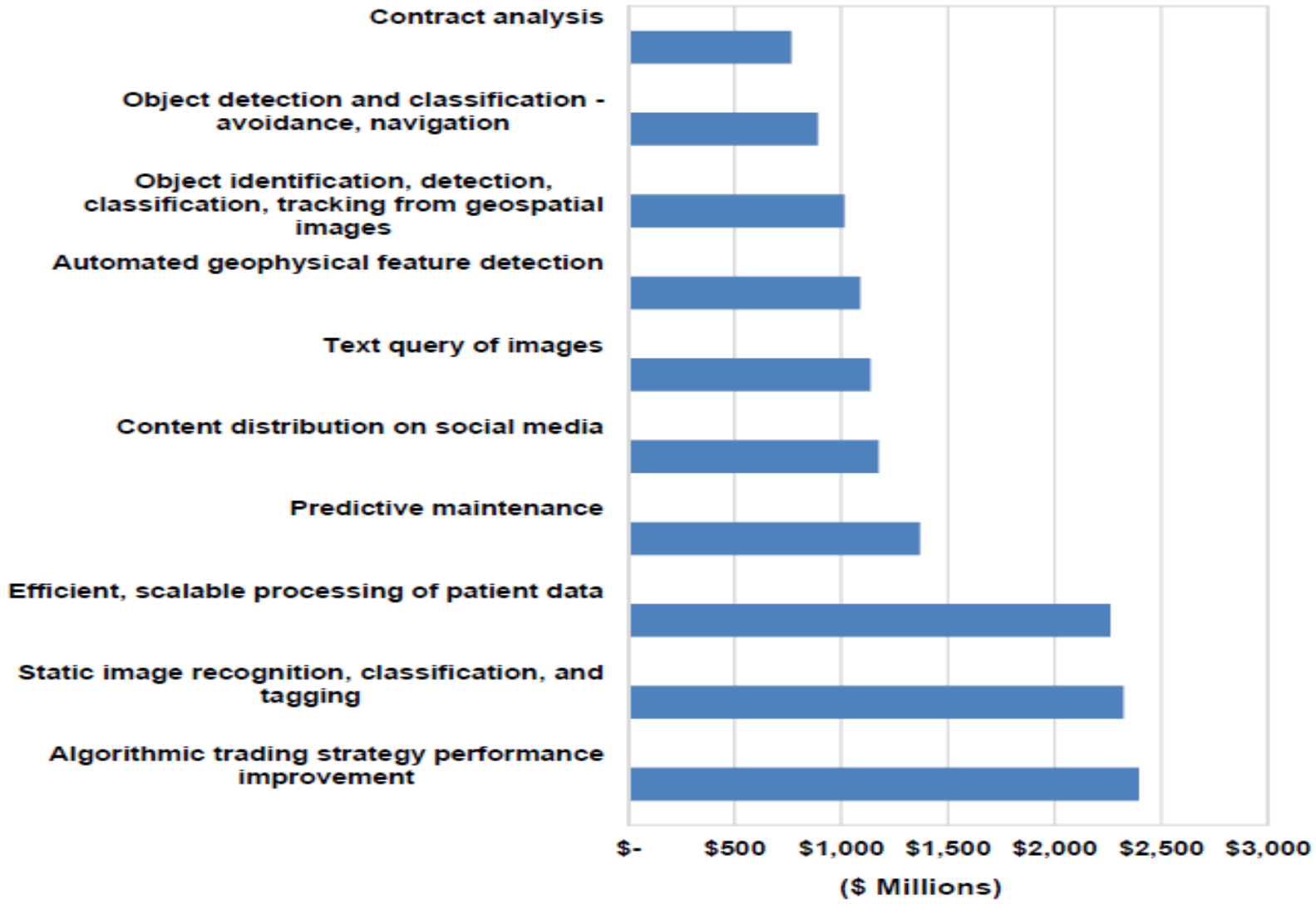
1.1 Artificial Intelligence Revenue, World Markets: 2016-2025



Courtesy: https://www.top500.org/news/market-for-artificial-intelligence-projected-to-hit-36-billion-by-2025/

Current and Future Use Cases of Deep Learning

1.2 Artificial Intelligence Revenue, Top 10 Use Cases, World Markets: 2025



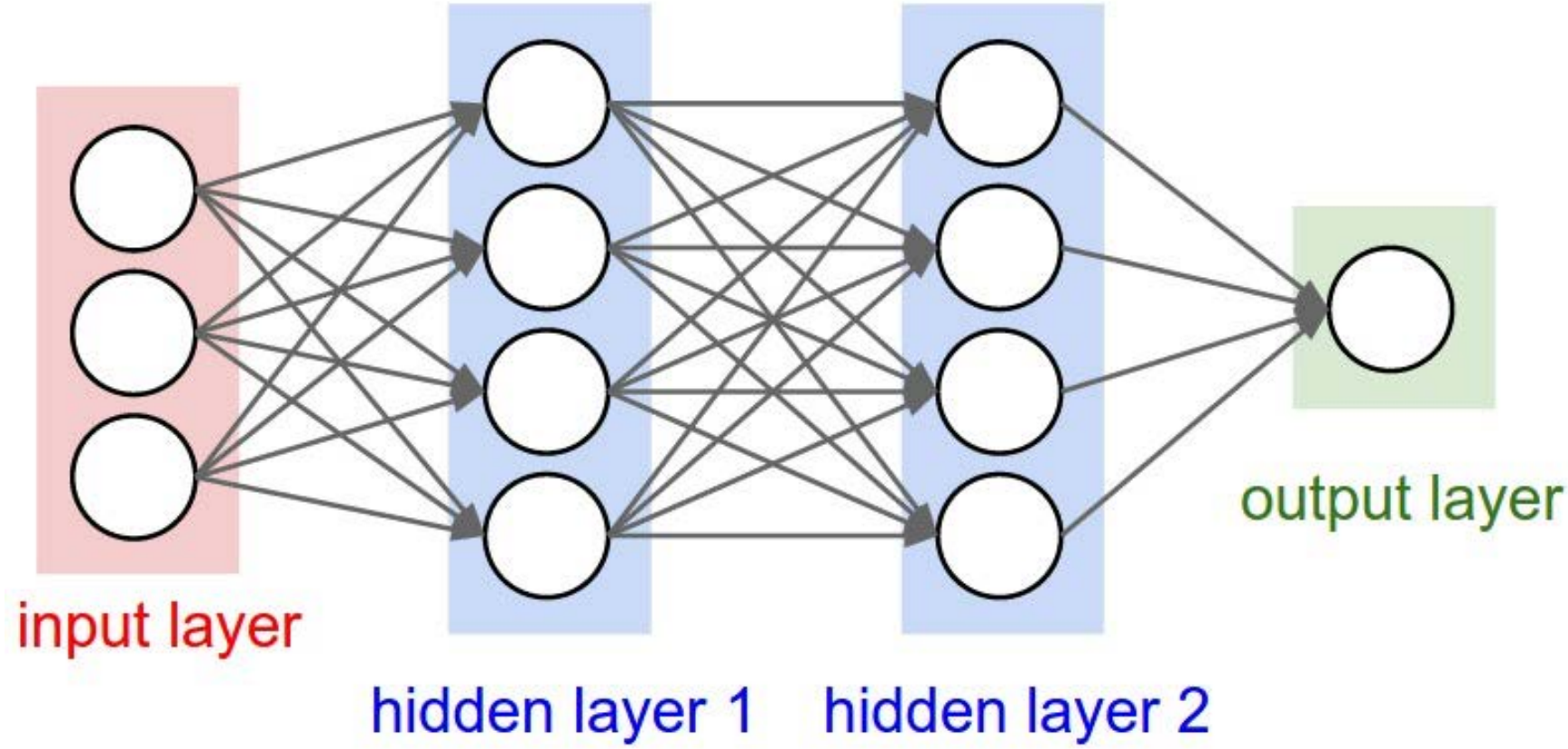
Courtesy: https://www.top500.org/news/market-for-artificial-intelligence-projected-to-hit-36-billion-by-2025/

Outline

- **Introduction**
  - The Past, Present, and Future of Deep Learning
  - **What are Deep Neural Networks?**
  - Diverse Applications of Deep Learning
  - Deep Learning Frameworks
- Overview of Execution Environments
- Parallel and Distributed DNN Training
- Latest Trends in HPC Technologies
- Challenges in Exploiting HPC Technologies for Deep Learning
- Solutions and Case Studies
- Open Issues and Challenges
- Conclusion

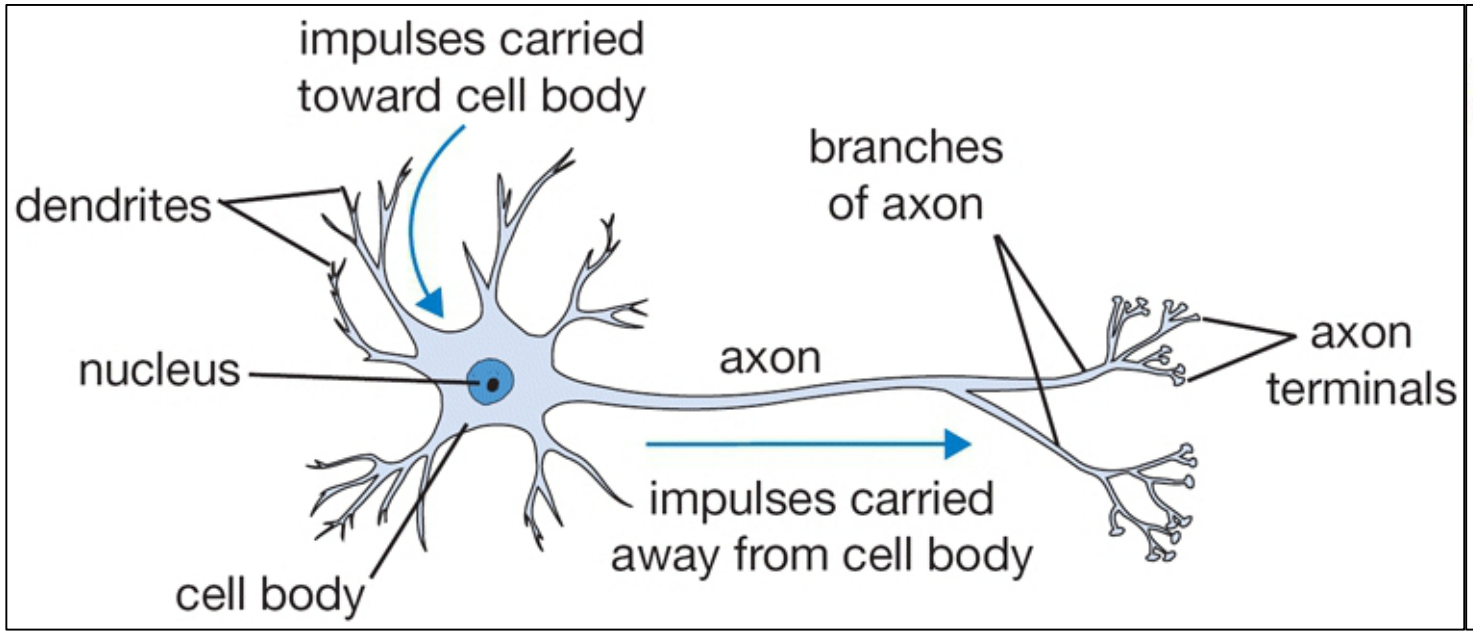
So what is a Deep Neural Network?

- Example of a 3-layer Deep Neural Network (DNN) – (input layer is not counted)

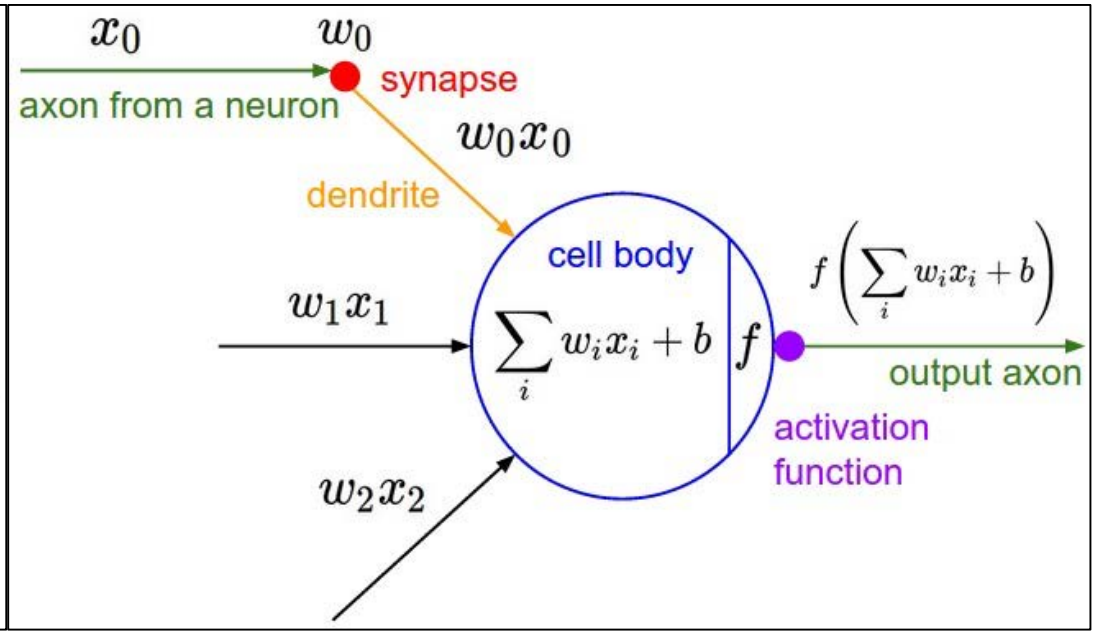


Courtesy: http://cs231n.github.io/neural-networks-1/

Graphical/Mathematical Intuitions for DNNs



Drawing of a Biological Neuron



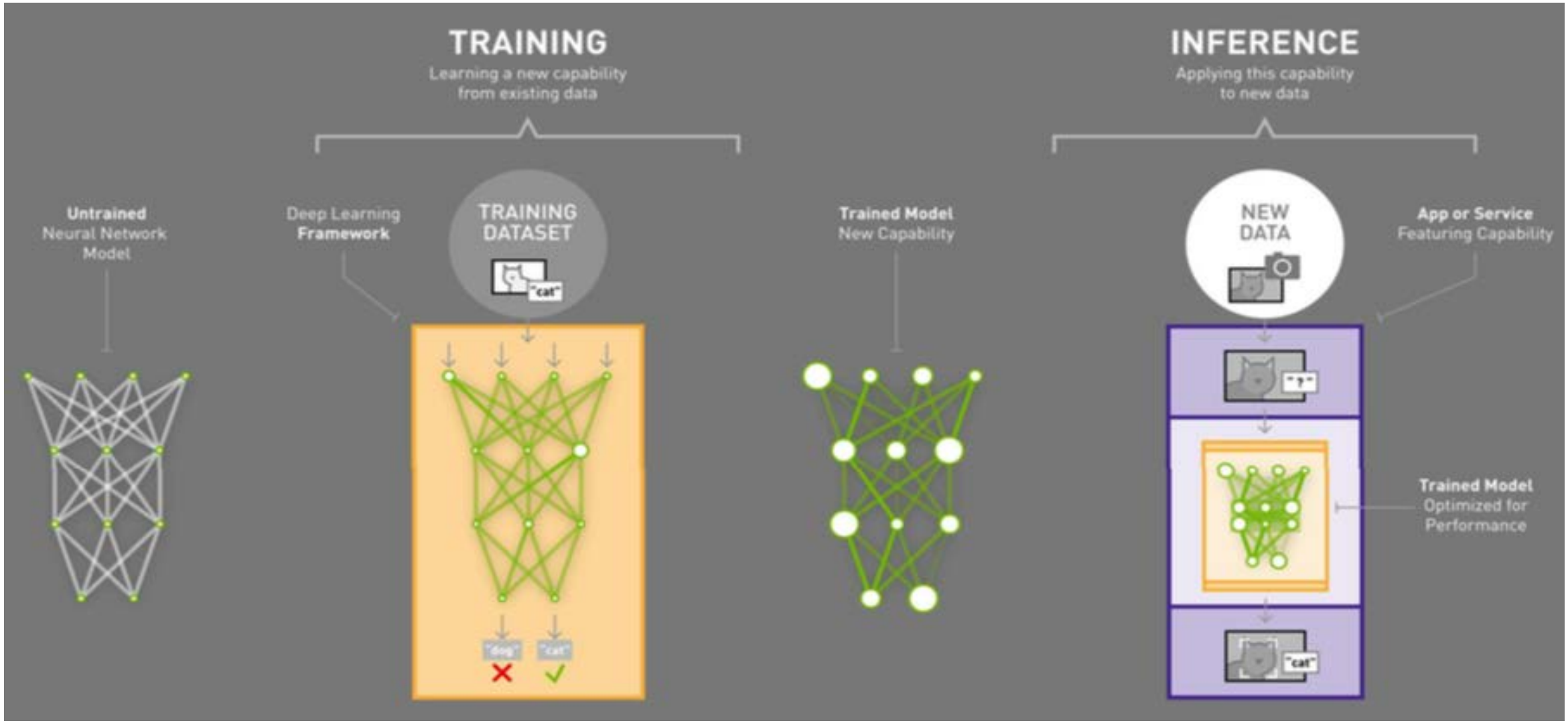
The Mathematical Model

Courtesy: http://cs231n.github.io/neural-networks-1/

Key Phases of Deep Learning

- Deep Learning has two major tasks
  1. Training of the Deep Neural Network
  2. Inference (or deployment) that uses a trained DNN
- DNN Training
  - Training is a compute/communication intensive process – can take days to weeks
  - Faster training is necessary!
- Faster training can be achieved by
  - Using Newer and Faster Hardware – But, there is a limit!
  - Can we use more GPUs or nodes?
    - The need for Parallel and Distributed Training

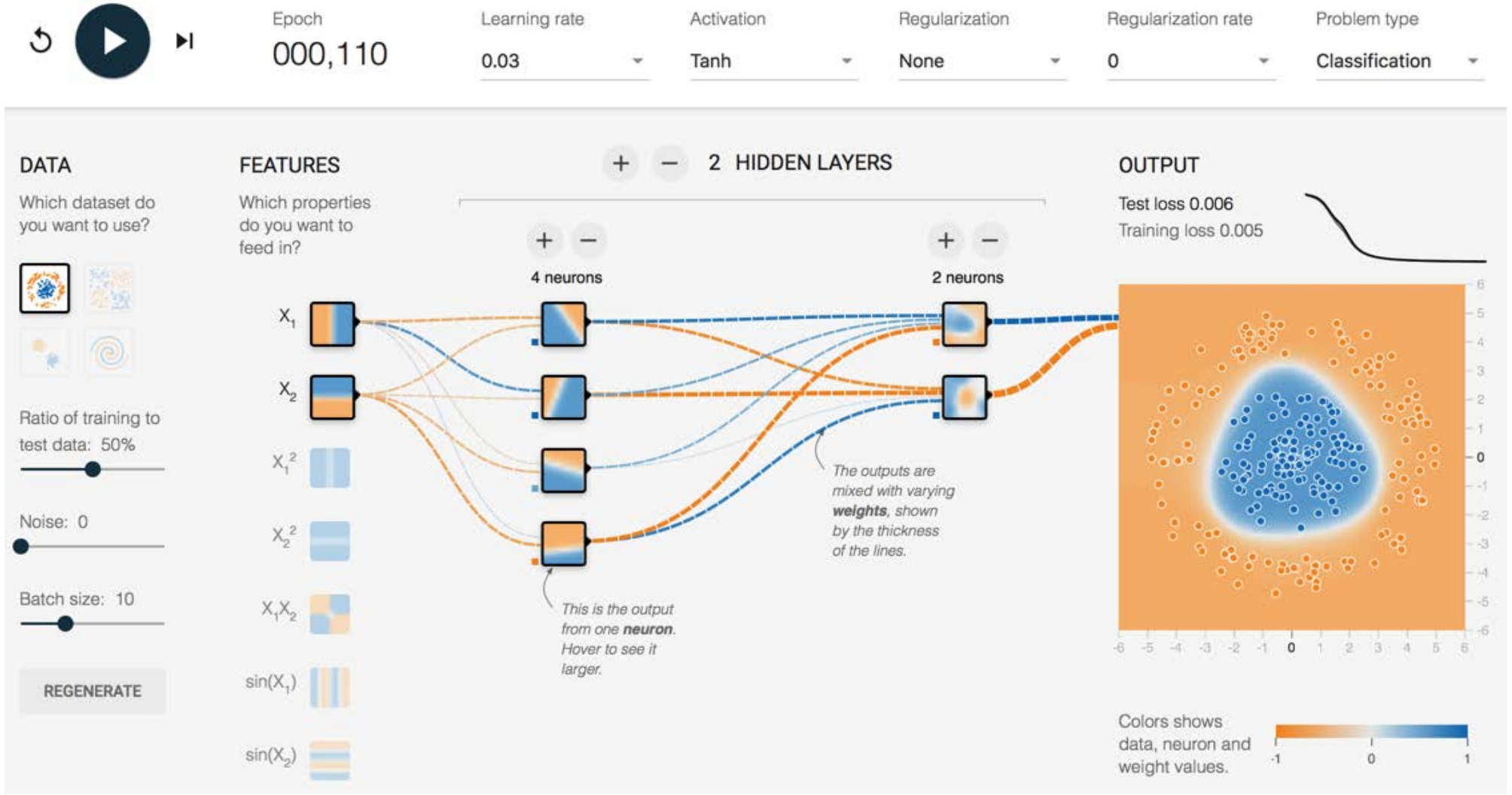
DNN Training and Inference



Courtesy: http://on-demand.gputechconf.com/gtc/2017/presentation/s7457-william-ramey-deep%20learning%20demystified_v24.pdf

TensorFlow playground (Quick Demo)

- To actually train a network, please visit: http://playground.tensorflow.org



Outline

- **Introduction**
  - The Past, Present, and Future of Deep Learning
  - What are Deep Neural Networks?
  - **Diverse Applications of Deep Learning**
  - Deep Learning Frameworks
- Overview of Execution Environments
- Parallel and Distributed DNN Training
- Latest Trends in HPC Technologies
- Challenges in Exploiting HPC Technologies for Deep Learning
- Solutions and Case Studies
- Open Issues and Challenges
- Conclusion

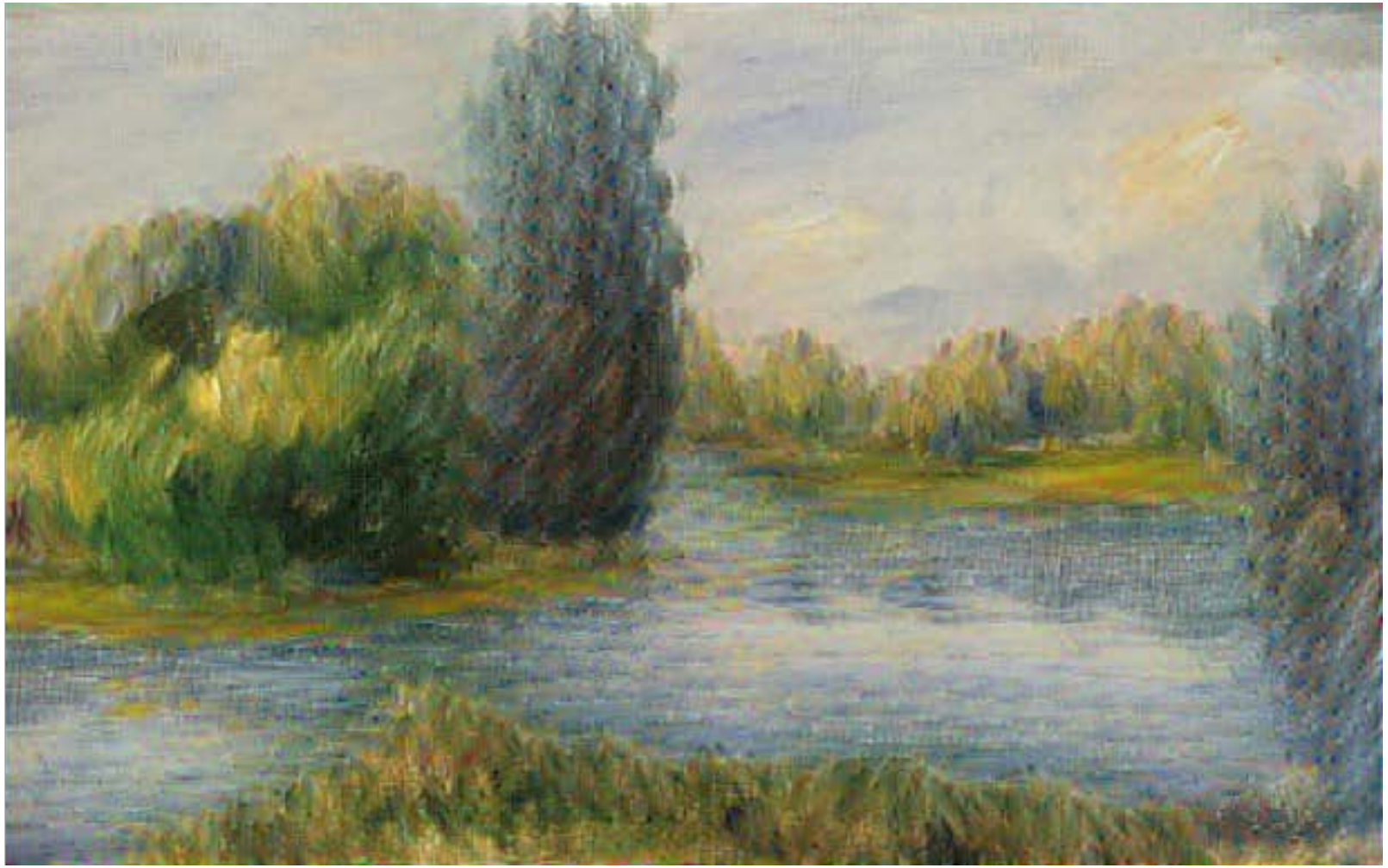
Diverse Application Areas for Deep Learning

- Vision
  - Image Classification
  - Style Transfer
  - Caption Generation
- Speech
  - Speech Recognition
  - Real-time Translation
- Text
  - Sequence Recognition and Generation
- Disease discovery
  - Cancer Detection
- Autonomous Driving
  - Combination of multiple areas like Image/Object Detection, Speech Recognition, etc.

Style Transfer

Synthesized Image

#NeuralDoodle

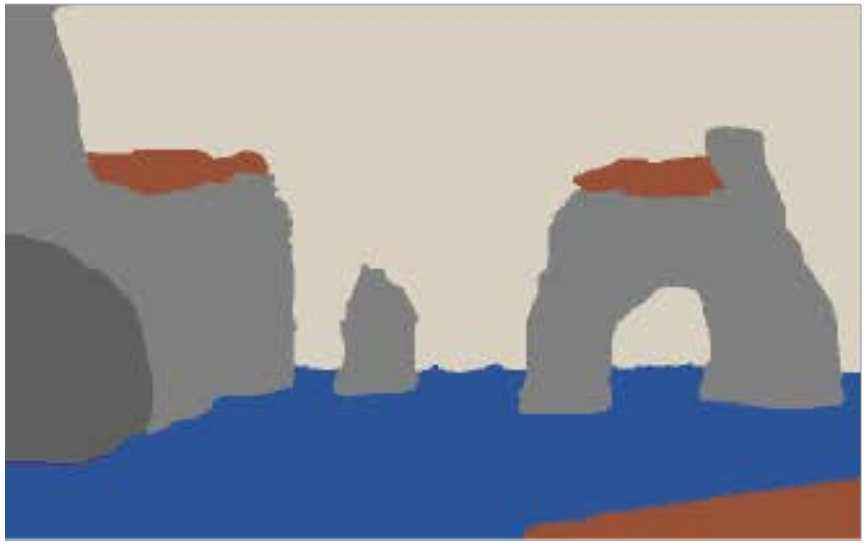


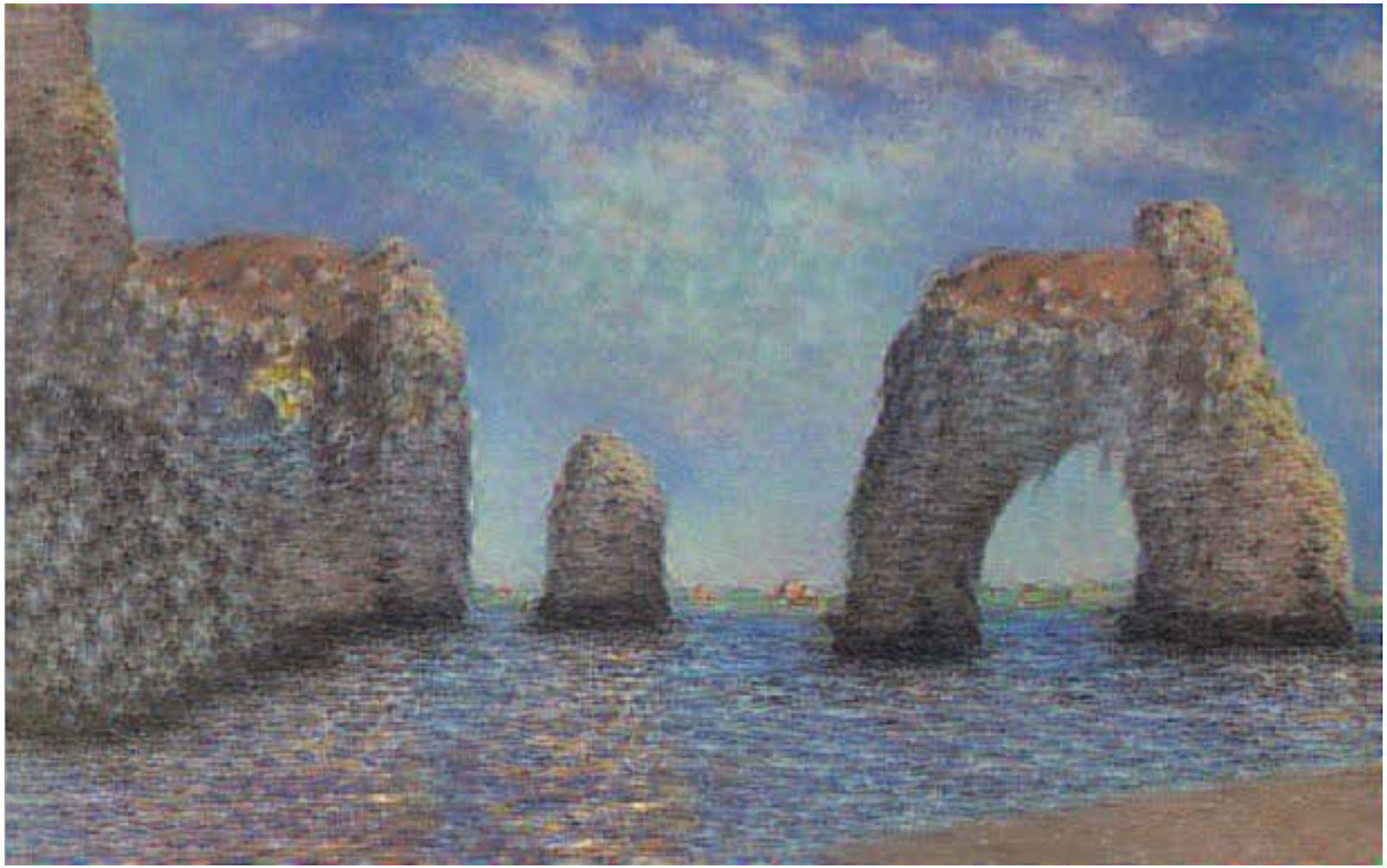
Courtesy: https://github.com/alexjc/neural-doodle

Style Transfer

Synthesized Image

#NeuralDoodle





Courtesy: https://github.com/alexjc/neural-doodle

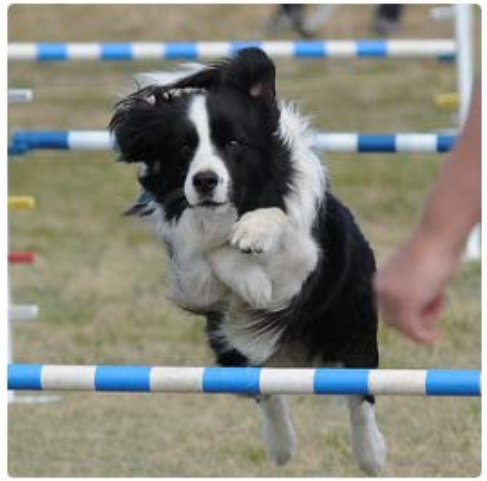
Caption Generation

"man in black shirt is playing guitar."

"construction worker in orange safety vest is working on road."

"two young girls are playing with lego toy."

"girl in pink dress is jumping in air."



"black and white dog jumps over bar."

"young girl in pink shirt is swinging on swing."

Courtesy: https://machinelearningmastery.com/inspirational-applications-deep-learning/

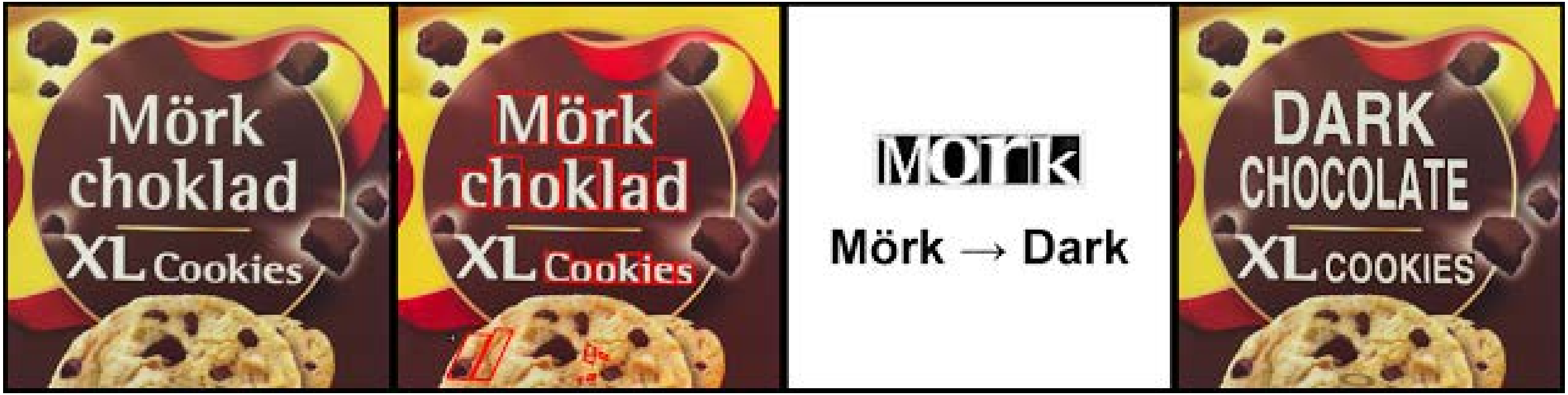
Shakespeare's Style Passage Generation

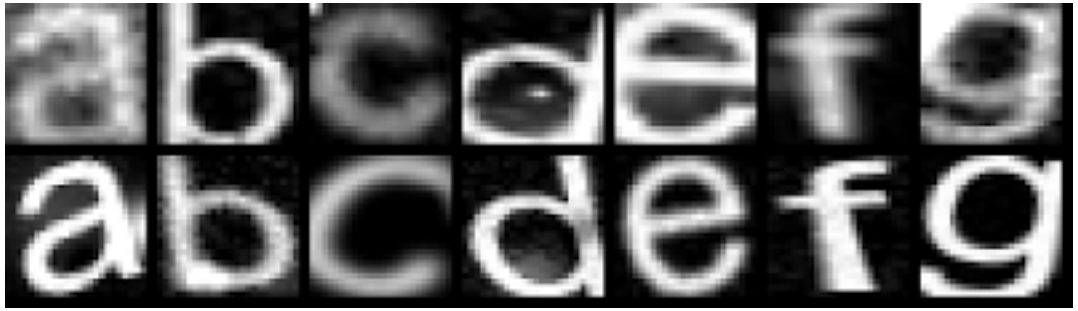
Remember, all the RNN knows are characters, so in particular it samples both speaker's names and the contents. Sometimes we also get relatively extended monologue passages, such as:

- VIOLA: Why, Salisbury must find his flesh and thought That which I am not aps, not a man and in fire, To show the reining of the raven and the wars To grace my hand reproach within, and not a fair are hand, That Caesar and my goodly father's world; When I was heaven of presence and our fleets, We spare with hours, but cut thy council I am great, Murdered and by thy master's ready there My power to give thee but so much as hell: Some service in the noble bondman here, Would show him to her wine.
- KING LEAR: O, if you were a feeble sight, the courtesy of your law, Your sight and several breath, will wear the gods With his heads, and my hands are wonder'd at the deeds, So drop upon your lordship's head, and your opinion Shall be against your honour.

Courtesy: http://karpathy.github.io/2015/05/21/rnn-effectiveness/

Machine Translation

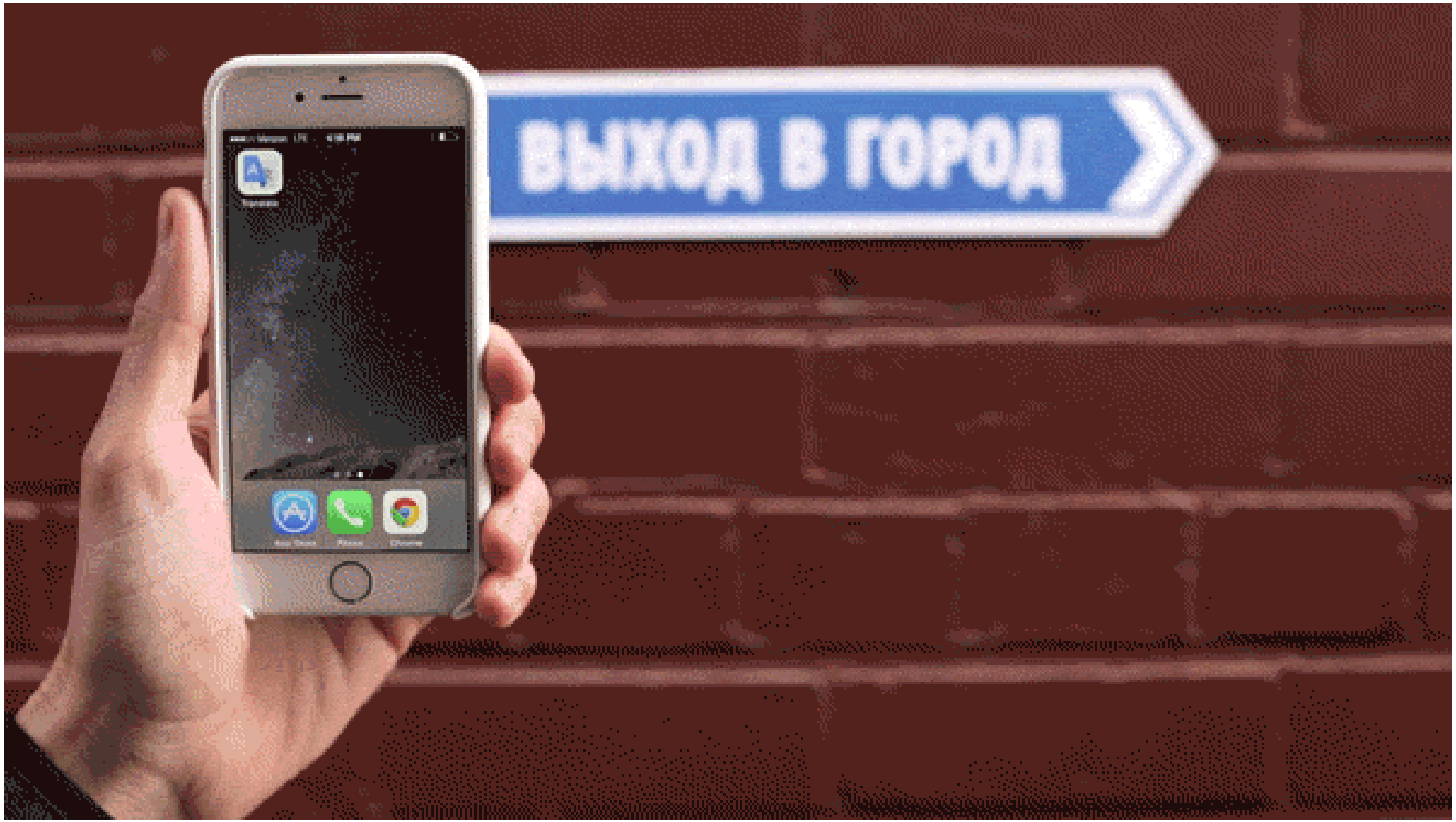




Some of the “dirty” letters we use for training. Dirt, highlights, and rotation, but not too much because we don’t want to confuse our neural net.

Courtesy: https://research.googleblog.com/2015/07/how-google-translate-squeezes-deep.html

Google Translate

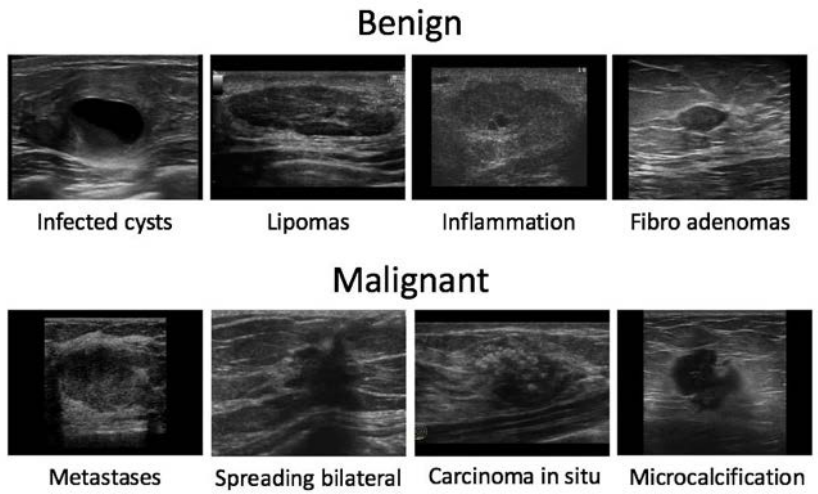


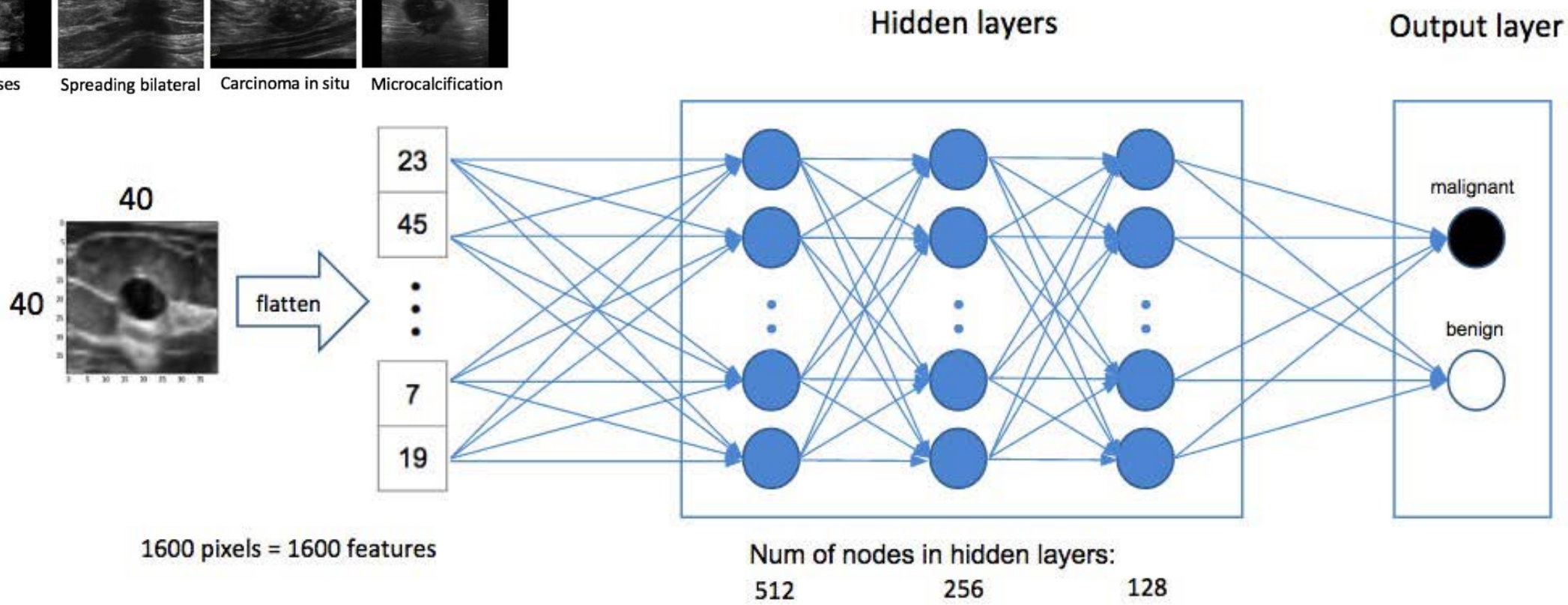
Courtesy: https://www.theverge.com/2015/1/14/7544919/google-translate-update-real-time-signs-conversations

Self Driving Cars

Courtesy: http://www.teslarati.com/teslas-full-self-driving-capability-arrive-3-months-definitely-6-months-says-musk/

Cancer Detection





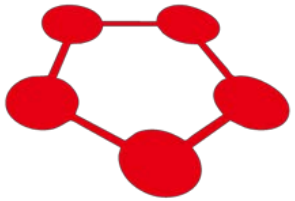
Courtesy: https://blog.insightdatascience.com/automating-breast-cancer-detection-with-deep-learning-d8b49da17950

Outline

- **Introduction**
  - The Past, Present, and Future of Deep Learning
  - What are Deep Neural Networks?
  - Diverse Applications of Deep Learning
  - **Deep Learning Frameworks**
- Overview of Execution Environments
- Parallel and Distributed DNN Training
- Latest Trends in HPC Technologies
- Challenges in Exploiting HPC Technologies for Deep Learning
- Solutions and Case Studies
- Open Issues and Challenges
- Conclusion

Deep Learning Frameworks

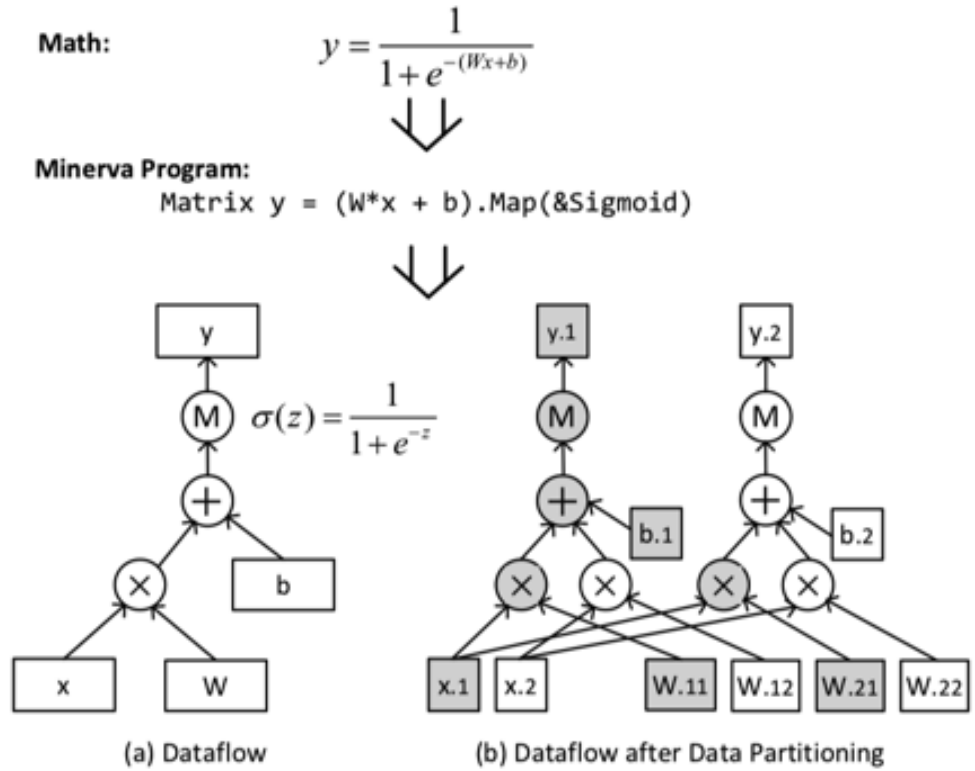
- Many Deep Learning frameworks
  - Berkeley Caffe
  - Facebook Caffe2
  - Google TensorFlow
  - Microsoft CNTK
  - Facebook Torch/PyTorch
  - Chainer/ChainerMN
  - Intel Neon/Nervana Graph



- Open Neural Net eXchange (ONNX) Format

Why we need DL frameworks?

- Deep Learning frameworks have emerged
  - hide most of the *nasty mathematics*
  - focus on the *design* of neural networks
- Distributed DL frameworks are being designed
  - We have saturated the peak potential of a single GPU/CPU/KNL
  - Parallel (multiple processing units in a single node) and/or Distributed (usually involves multiple nodes) frameworks are emerging
- Distributed frameworks are being developed along two directions
  - The HPC Eco-system: MPI-based Deep Learning
  - Enterprise Eco-system: BigData-based Deep Learning

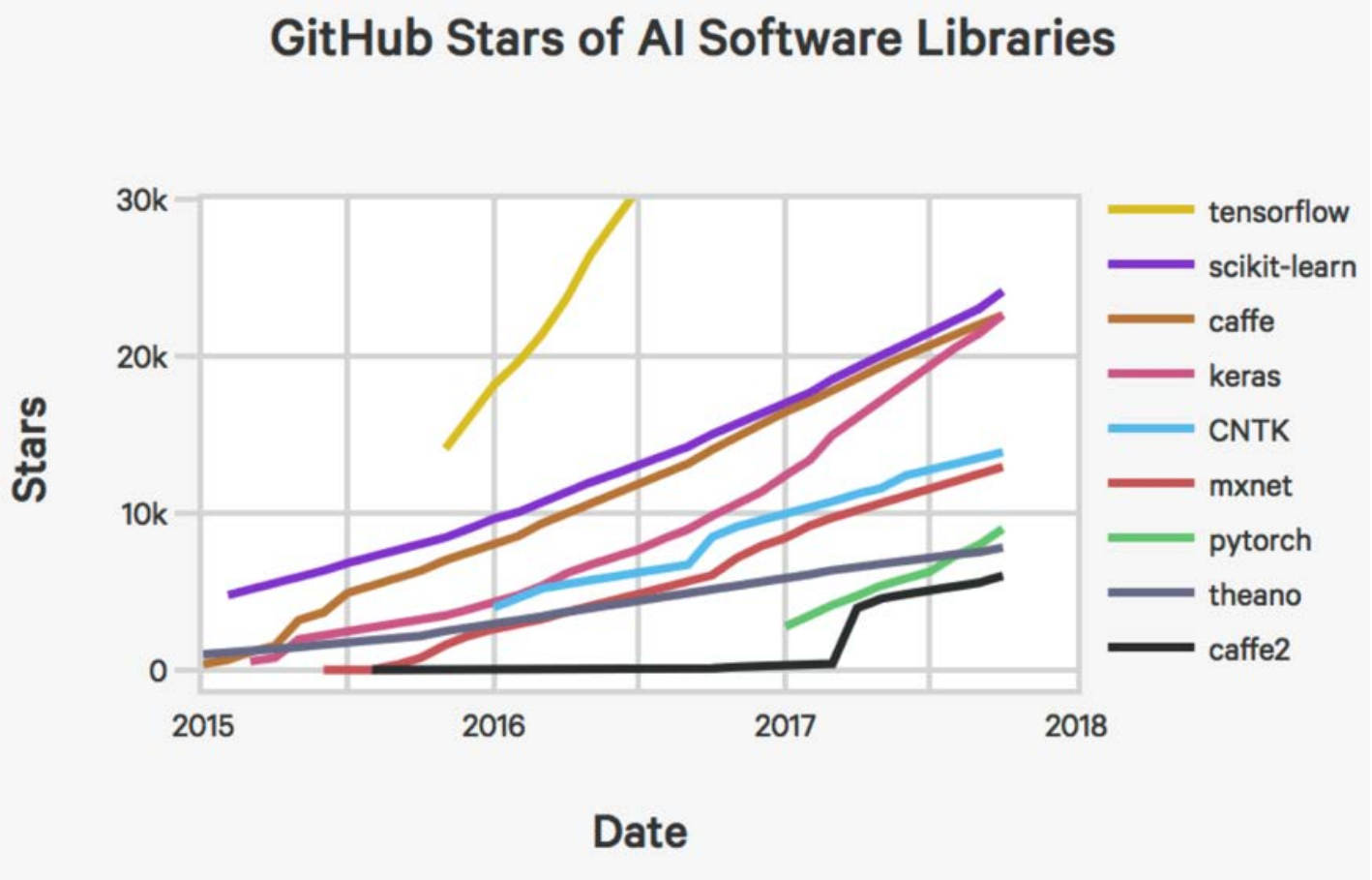


Statement and its dataflow fragment. The data and computing vertexes with different colors reside on different processes.

Courtesy: https://web.stanford.edu/~rezab/nips2014workshop/submits/minerva.pdf

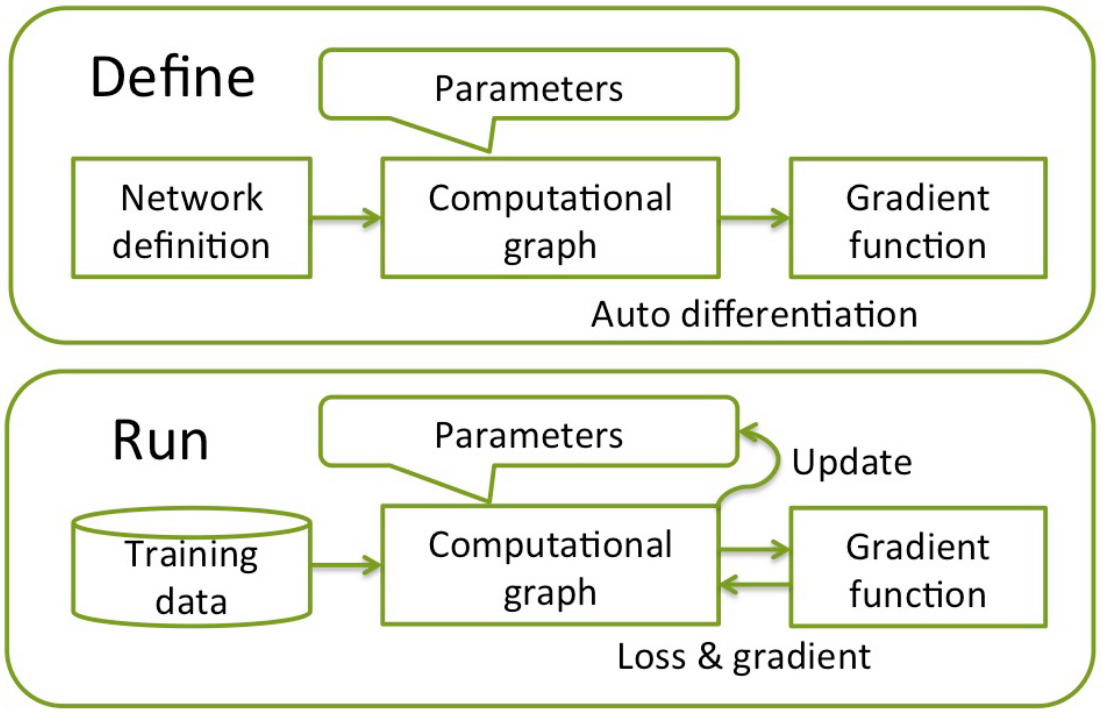
DL Frameworks and GitHub Statistics

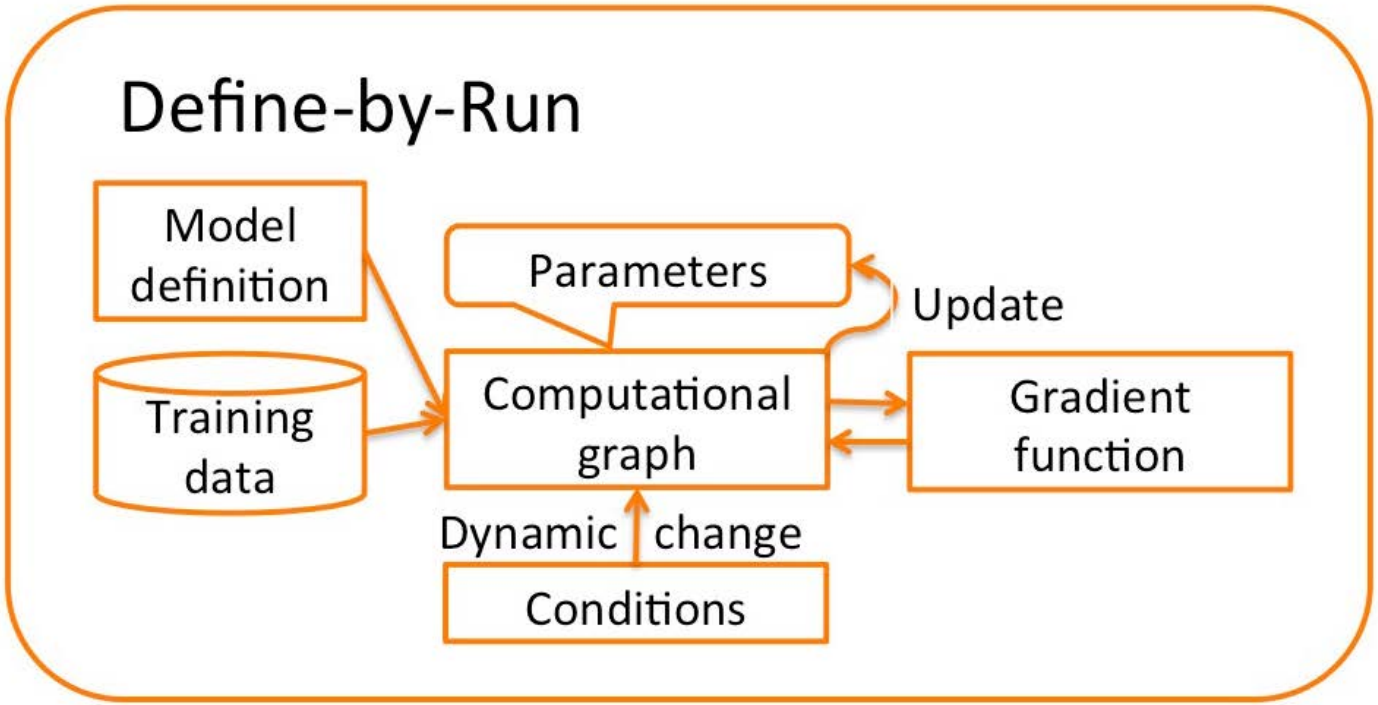
- AI Index report offers very detailed trends about AI and ML
- It also provides interesting statistics about open source DL frameworks and related GitHub statistics
- More details about DL frameworks, their features, and statistics (see Appendix)



Courtesy: http://cdn.aiindex.org/2017-report.pdf

Define-by-run frameworks vs. Define-and-run?





- Define-and-run: TensorFlow, Caffe, Torch, Theano, and others
- Define-by-run
  - PyTorch and Chainer
  - TensorFlow 1.5 (Jan, 26th) has introduced an Eager Execution (Define-by-run) mode

Courtesy: https://www.oreilly.com/learning/complex-neural-networks-made-easy-by-chainer

Berkeley (BVLC) Caffe

- One of the most popular DL framework
  - Winner of the ACM MM open source award 2014
- Yangqing Jia (BVLC)
  - Author of Caffe and Caffe2 (Facebook)
- The framework has a modular C++ backend
- C++ and Python frontends
- Caffe is a single-node but multi-GPU framework

Courtesy: http://caffe.berkeleyvision.org

Facebook Caffe2

- Caffe2 is a more versatile, diversified, and refactored framework
- Supported by Facebook
- Works on several platforms including mobile platforms
- https://github.com/caffe2/caffe2
- Main motivation
  - New Application Areas
  - Flexibility
  - Newer Platforms
    - Mobile

Courtesy: https://caffe2.ai

Google TensorFlow

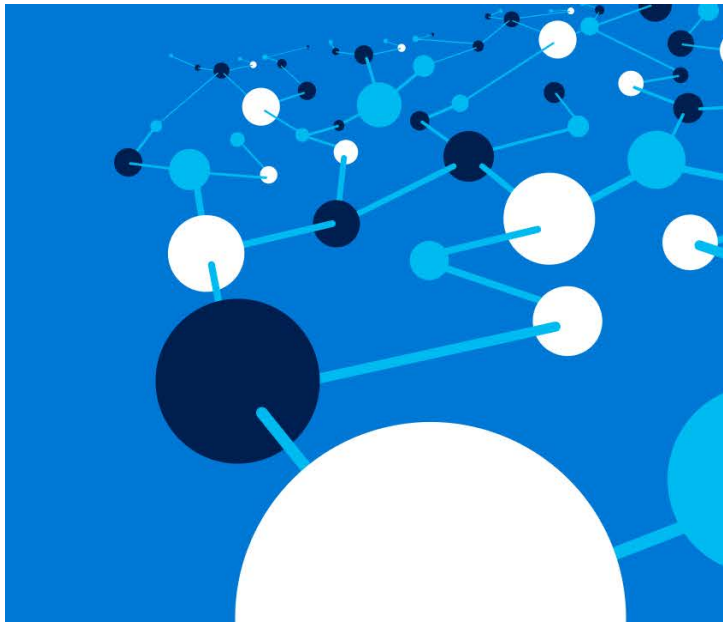
- The most widely used framework open-sourced by Google
- Replaced Google's DistBelief[1] framework
- Runs on almost all execution platforms available (CPU, GPU, TPU, Mobile, etc.)
- Very flexible but performance has been an issue
- Certain Python peculiarities like ***variable_scope*** etc.
- https://github.com/tensorflow/tensorflow

Courtesy: https://www.tensorflow.org/

[1] Jeffrey Dean et al., "Large Scale Distributed Deep Networks"
https://static.googleusercontent.com/media/research.google.com/en//archive/large_deep_networks_nips2012.pdf

Microsoft Cognitive Toolkit (CNTK)

- Formerly CNTK, now called the Cognitive Toolkit
- C++ backend
- C++ and Python frontend
- ASGD, SGD, and several others choices for Solvers/Optimizers
- Constantly evolving support for multiple platforms
- Performance has always been the “key feature”
- https://github.com/microsoft/cntk



Courtesy: https://www.microsoft.com/en-us/cognitive-toolkit/

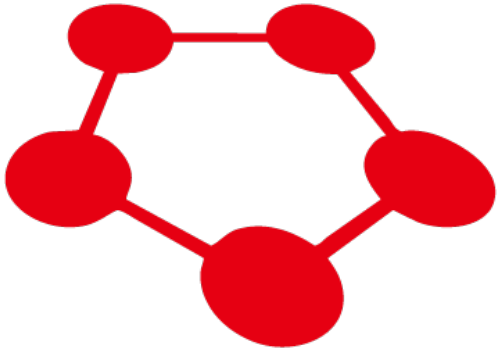
Facebook Torch/PyTorch

- Torch was written in Lua
  - Adoption wasn't wide-spread
- PyTorch is a Python adaptation of Torch
  - Gaining lot of attention
- Several contributors
  - Biggest support by Facebook
- There are/maybe plans to merge the PyTorch and Caffe2 efforts
- Key selling point is ease of expression and "define-by-run" approach

Courtesy: http://pytorch.org

Preferred Networks Chainer

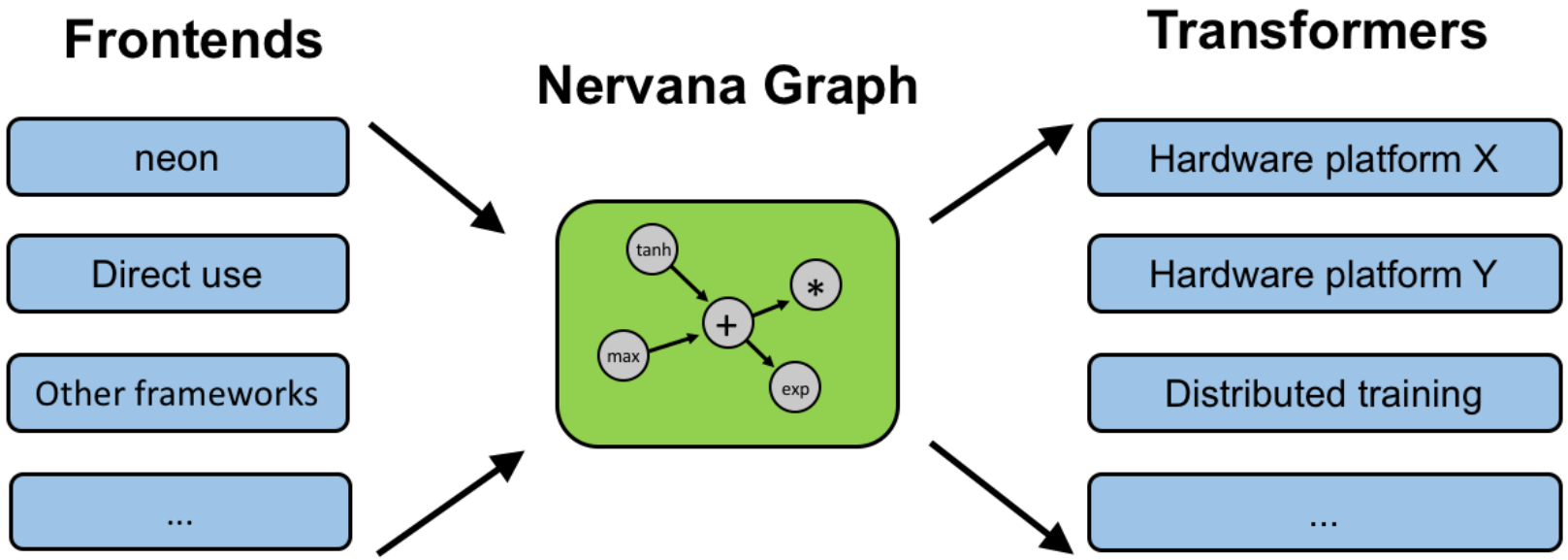
- Uses ***Define-by-run*** (Chainer, PyTorch) approach instead of ***Define-and-run*** (Caffe, TensorFlow, Torch, Theano) approach
- ***ChainerMN*** provides multi-node parallel/distributed training using Message Passing Interface (MPI) and Chainer
- **MVAPICH2 MPI** library is being used by Preferred Networks
- ChainerMN is geared towards performance
  - Focus on Speed as well as multi-node Scaling
  - *Trained ResNet-50 on 256 P100 GPUs in 15 minutes!!* [1]



1. https://arxiv.org/abs/1711.04325

Intel Neon/Nervana Graph

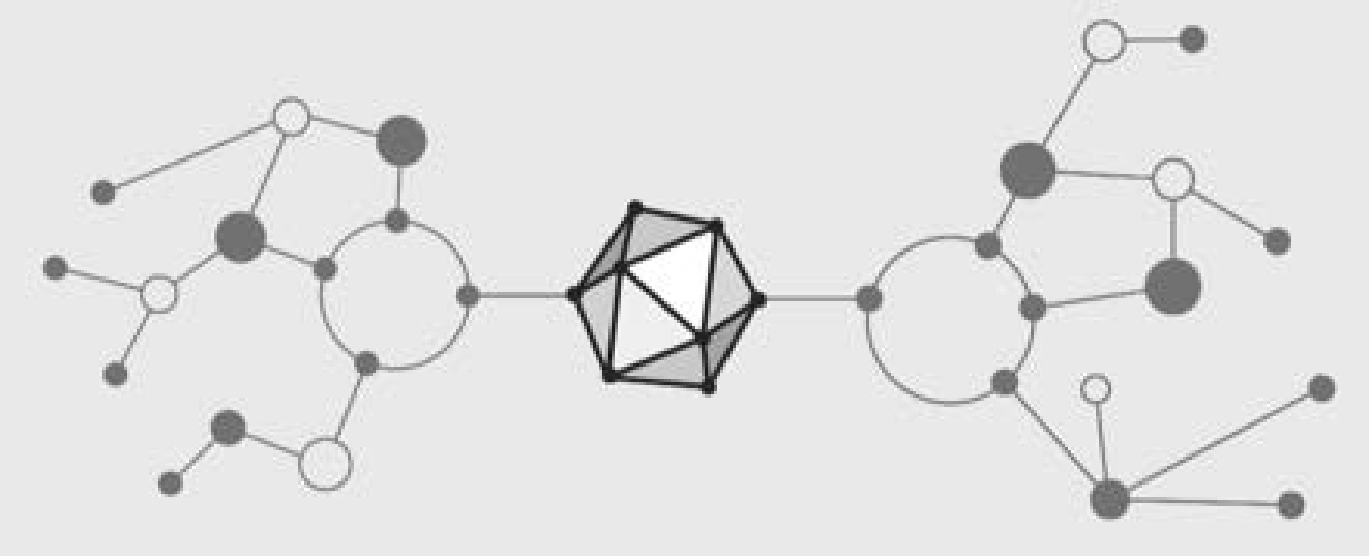
- Neon is a Deep Learning framework by Intel/Nervana
  - Works on CPUs as well as GPUs!
- Nervana Graph is like an Intermediate Representation (IR) for Neural Nets
- Nervana Graph will support various frontends and backends
  - NervanaGraph (ngraph) - https://github.com/NervanaSystems/ngraph



Courtesy: https://ai.intel.com/intel-nervana-graph-preview-release/

Open Neural Network eXchange (ONNX) Format

- ONNX- Not a Deep Learning framework but an open format to exchange "**trained**" networks across different frameworks
- Currently supported
  - Frameworks: Caffe2, Chainer, CNTK, MXNet, PyTorch
  - Convertors: CoreML, TensorFlow
  - Runtimes: NVIDIA
- https://onnx.ai
- https://github.com/onnx



Many Other DL Frameworks…

- Keras - https://keras.io
- MXNet - http://mxnet.io
- Theano - http://deeplearning.net/software/theano/
- Blocks - https://blocks.readthedocs.io/en/latest/
- Intel BigDL - https://software.intel.com/en-us/articles/bigdl-distributed-deep-learning-on-apache-spark
- The list keeps growing and the names keep getting longer and weirder ;-)
  - Livermore Big Artificial Neural Network Toolkit (LBANN) - https://github.com/LLNL/lbann
  - Deep Scalable Sparse Tensor Network Engine (DSSTNE) - https://github.com/amzn/amazon-dsstne

Outline

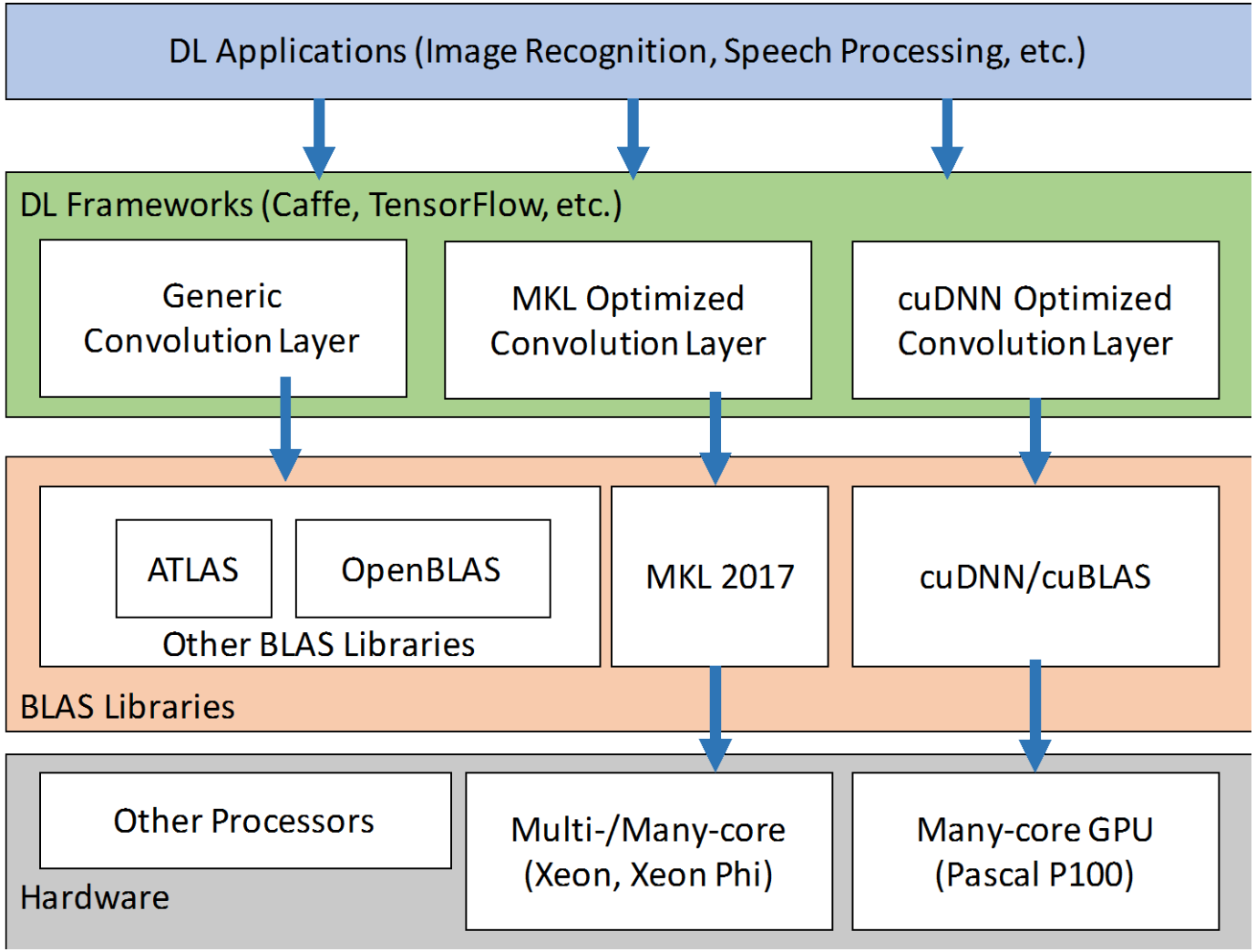
- Introduction
- **Overview of Execution Environments**
- Parallel and Distributed DNN Training
- Latest Trends in HPC Technologies
- Challenges in Exploiting HPC Technologies for Deep Learning
- Solutions and Case Studies
- Open Issues and Challenges
- Conclusion

So where do we run our DL framework?

- Early (2014) frameworks used a single fast GPU
  - As DNNs became larger, faster and better GPUs became available
  - At the same time, parallel (multi-GPU) training gained traction as well
- Today
  - Parallel training on multiple GPUs is being supported by most frameworks
  - Distributed (multiple nodes) training is still upcoming
    - A lot of fragmentation in the efforts (MPI, Big-Data, NCCL, Gloo, etc.)
  - On the other hand, DL has made its way to Mobile and Web too!
    - Smartphones - OK Google, Siri, Cortana, Alexa, etc.
    - DrivePX – the computer that drives NVIDIA's self-driving car
    - Very recently, Google announced Deeplearn.js (a DL framework in a web-browser)
    - TensorFlow playground - http://playground.tensorflow.org/

Conventional Execution on GPUs and CPUs

- My framework is faster than your framework!
- This needs to be understood in a holistic way.
- Performance depends on the entire execution environment (the full stack)
- Isolated view of performance is not helpful



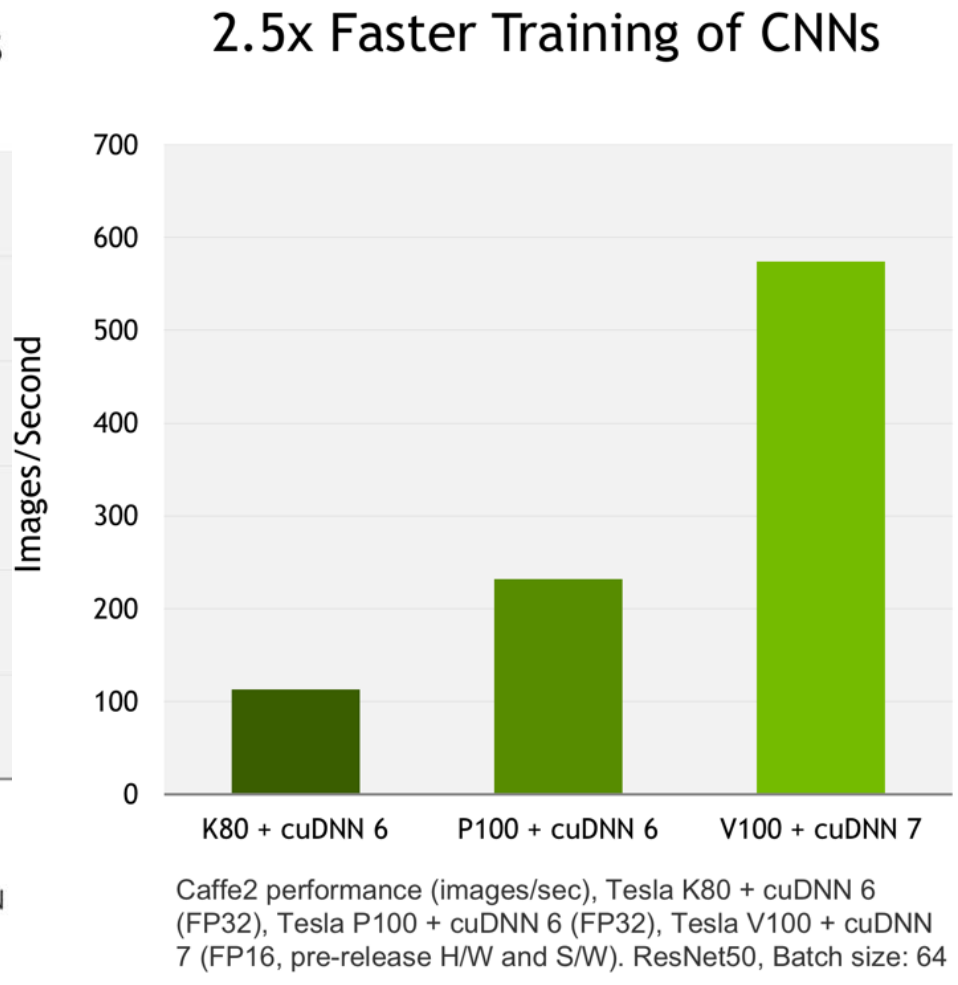
A. A. Awan, H. Subramoni, and Dhabaleswar K. Panda. "An In-depth Performance Characterization of CPU- and GPU-based DNN Training on Modern Architectures", In Proceedings of the Machine Learning on HPC Environments (MLHPC'17). ACM, New York, NY, USA, Article 8.

DL Frameworks and Underlying Libraries

- BLAS Libraries – the heart of math operations
  - Atlas/OpenBLAS
  - NVIDIA cuBlas
  - Intel Math Kernel Library (MKL)
- Most compute intensive layers are generally optimized for a specific hardware
  - E.g. Convolution Layer, Pooling Layer, etc.
- DNN Libraries – the heart of Convolutions!
  - NVIDIA cuDNN (already reached its 7th iteration – cudnn-v7)
  - Intel MKL-DNN (MKL 2017) – recent but a very promising development

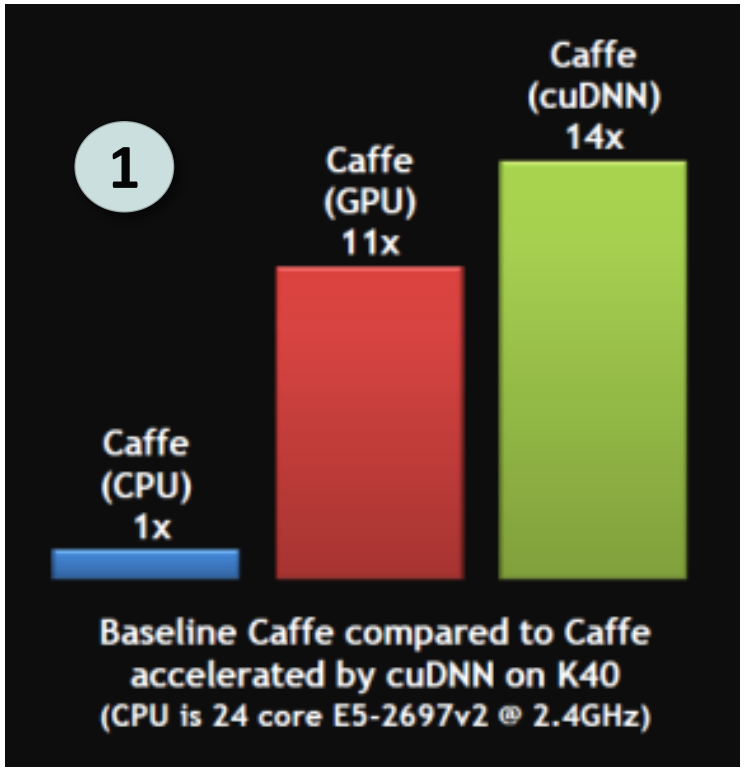
Where does the Performance come from?

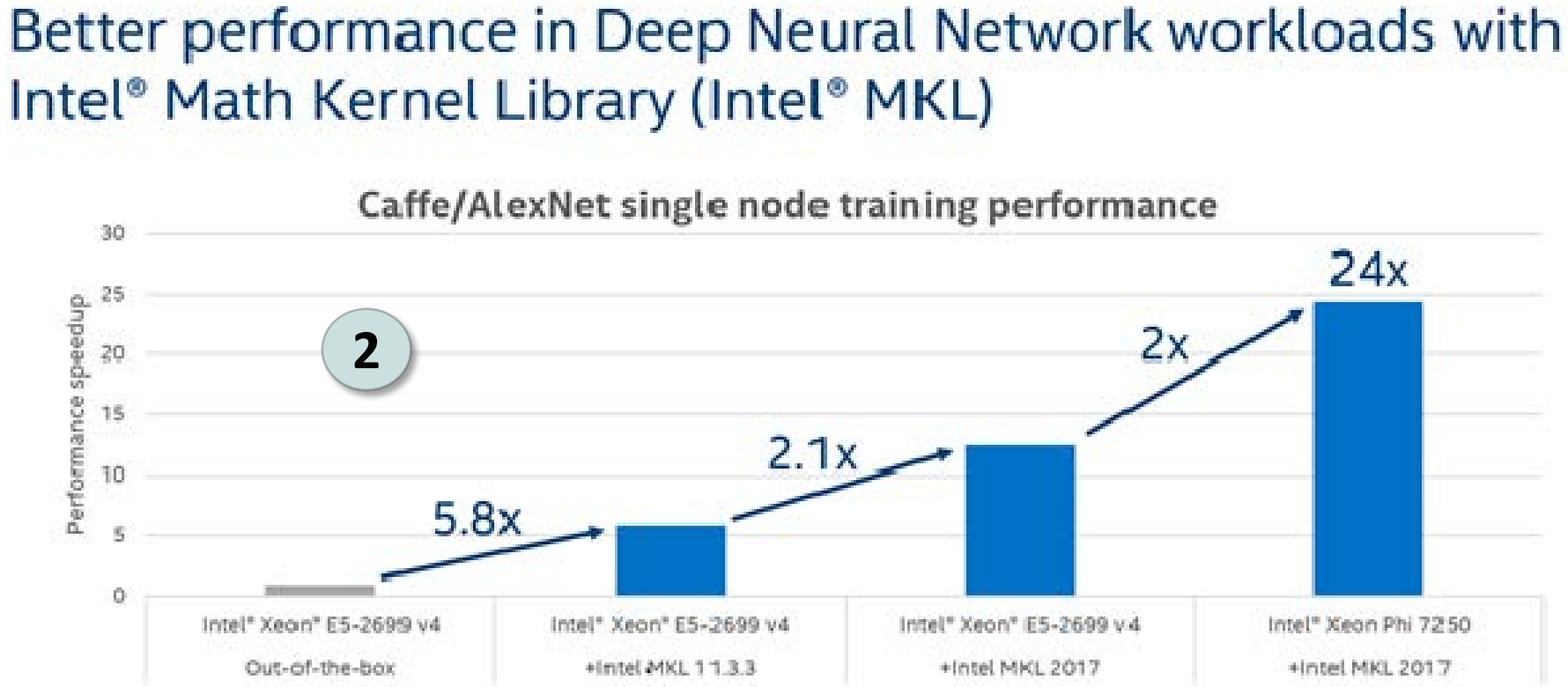
- Performance Improvements can be observed because of:
  - Faster convolutions with each successive cuDNN version
  - Faster hardware and more FLOPS as we move from:
    K-80 -> P-100 -> V-100



Courtesy: https://developer.nvidia.com/cudnn

Differences in How Vendors Report Performance

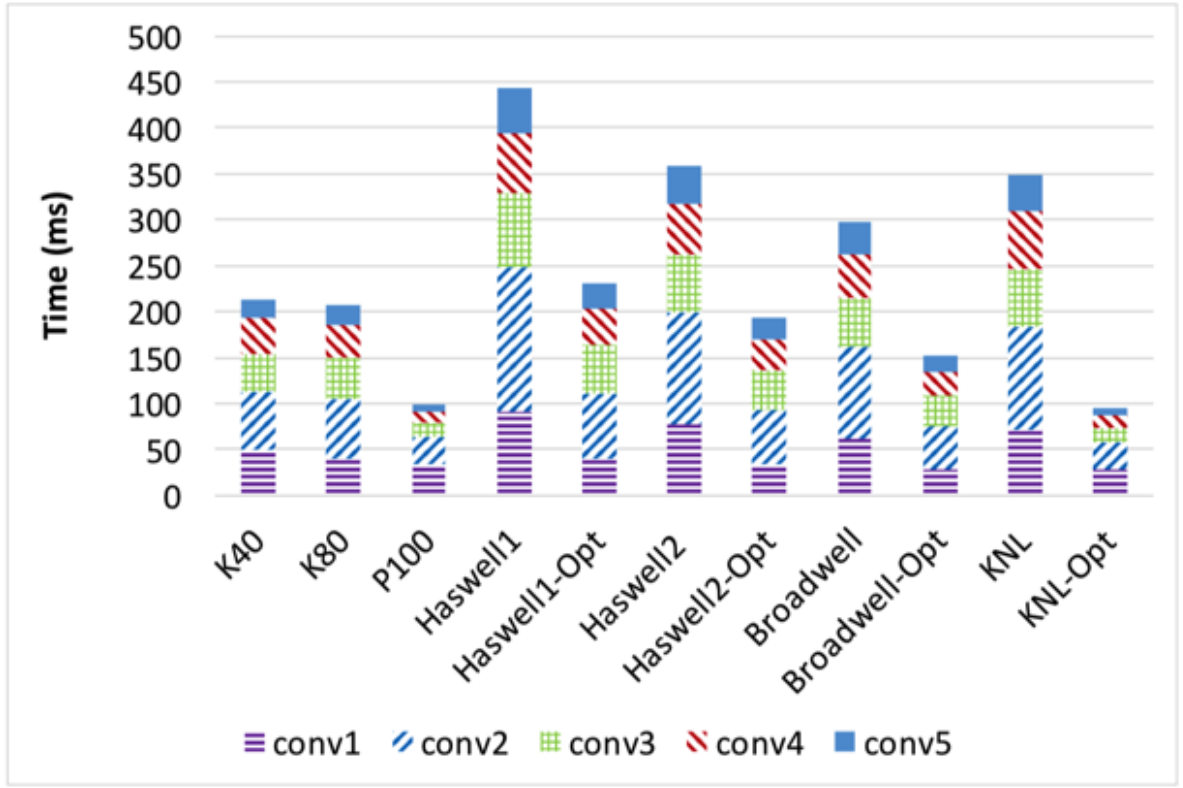




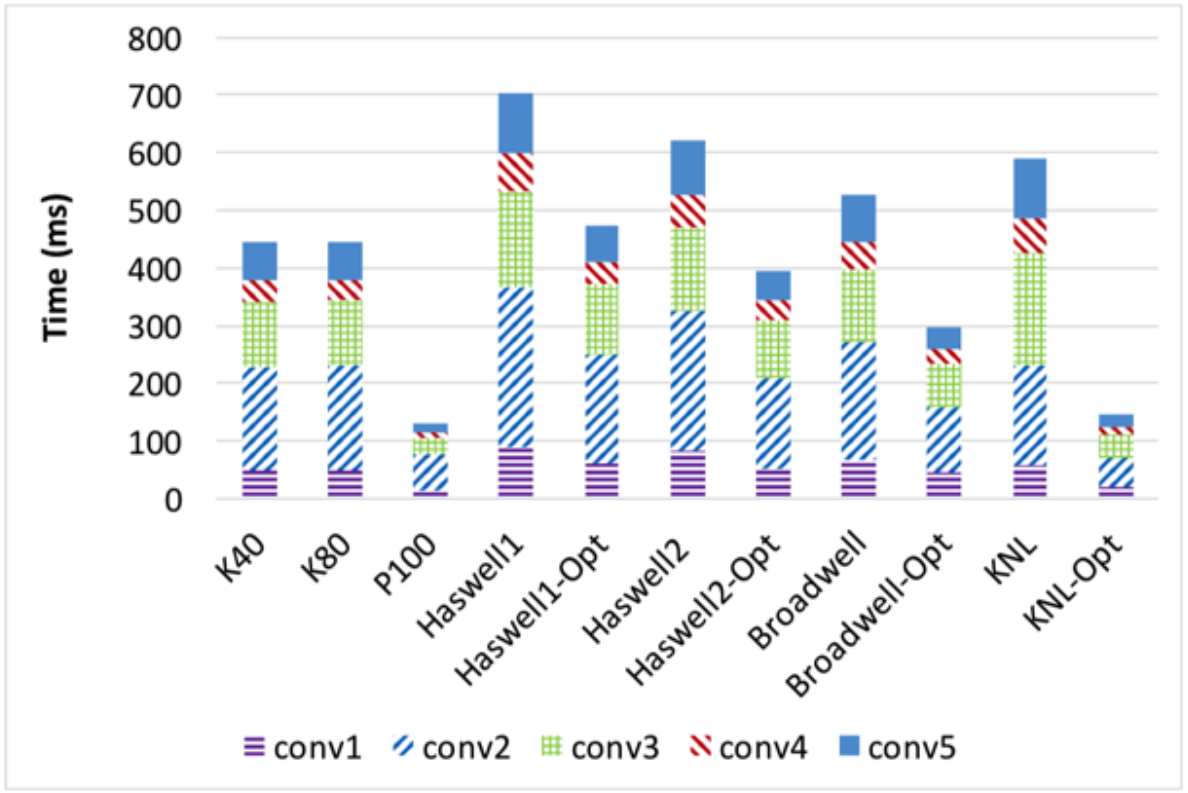
- NVIDIA reports performance [1] with the basic Caffe version (no multi-threading and no optimized MKL support)
- Intel reports [2] performance gains over the same basic Caffe version
- *Hence, the need for a holistic and fair comparison!!*

1. https://devblogs.nvidia.com/parallelforall/accelerate-machine-learning-cudnn-deep-neural-network-library/
2. https://software.intel.com/en-us/articles/introducing-dnn-primitives-in-intelr-mkl

An In-depth Comparison for CPU and GPU based Training (OSU)



(a) AlexNet: Forward Propagation



(b) AlexNet: Backward Propagation

- The full landscape for AlexNet training: Forward and Backward Pass
- ***Faster Convolutions → Faster Training***
- Key Takeaway: ***KNL-opt (CPU) is comparable to Pascal P100 (GPU)!***

A. A. Awan, H. Subramoni, and Dhabaleswar K. Panda. "An In-depth Performance Characterization of CPU- and GPU-based DNN Training on Modern Architectures", In Proceedings of the Machine Learning on HPC Environments (MLHPC'17). ACM, New York, NY, USA, Article 8.

Outline

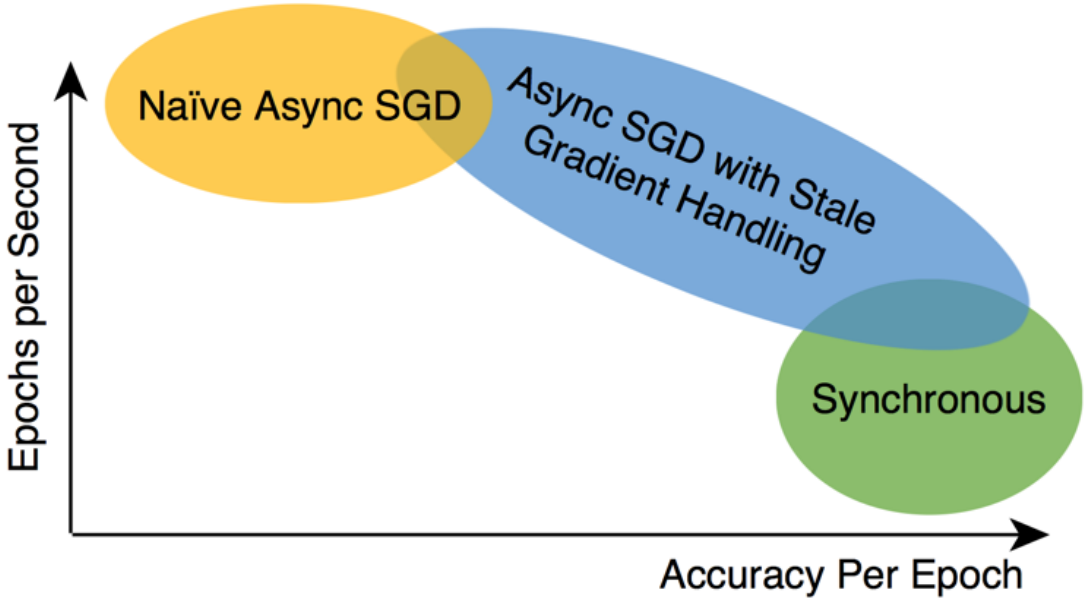
- Introduction
- Overview of Execution Environments
- **Parallel and Distributed DNN Training**
- Latest Trends in HPC Technologies
- Challenges in Exploiting HPC Technologies for Deep Learning
- Solutions and Case Studies
- Open Issues and Challenges
- Conclusion

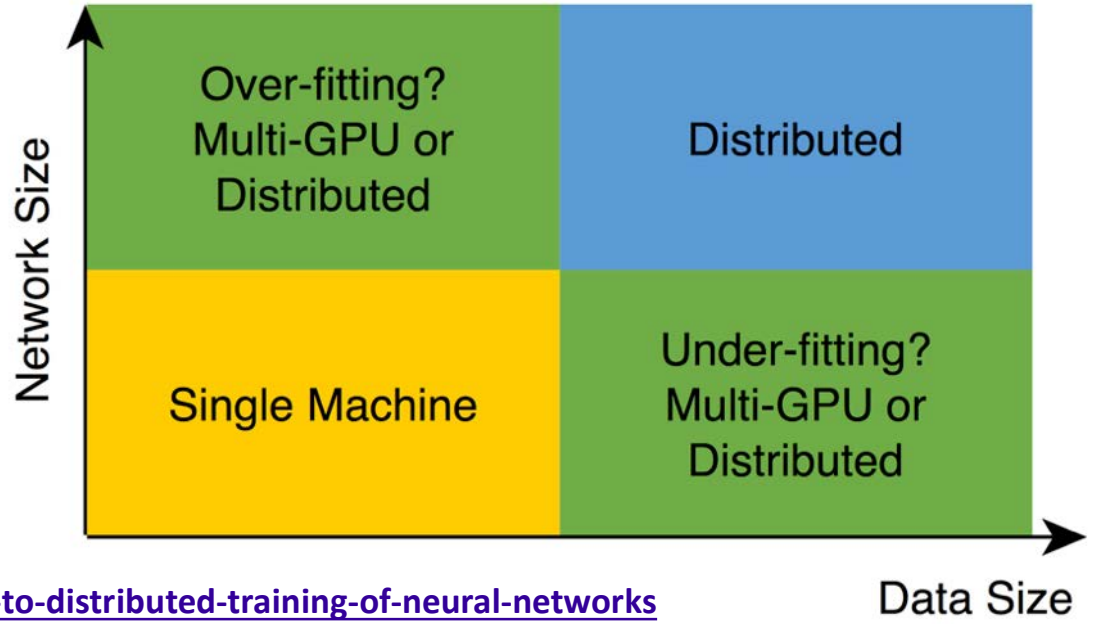
The Need for Parallel and Distributed Training

- Why do we need Parallel Training?
- Larger and Deeper models are being proposed
  - **AlexNet** to **ResNet** to **Neural Machine Translation (NMT)**
  - DNNs require a lot of memory
  - Larger models cannot fit a GPU's memory
- Single GPU training became a bottleneck
- As mentioned earlier, community has already moved to multi-GPU training
- Multi-GPU in one node is good but there is a limit to Scale-up (8 GPUs)
- **Multi-node (Distributed or Parallel) Training is necessary!!**

Batch-size, Model-size, Accuracy, and Scalability

- Increasing model-size generally increases accuracy
- Increasing batch-size requires tweaking hyper-parameters to maintain accuracy
  - Limits for batch-size
  - Cannot make it infinitely large
  - Over-fitting
- **Large batch size generally helps scalability**
  - More work to do before the need to synchronize
- Increasing the model-size (no. of parameters)
  - Communication overhead becomes bigger so scalability decreases
  - GPU memory is precious and can only fit finite model data





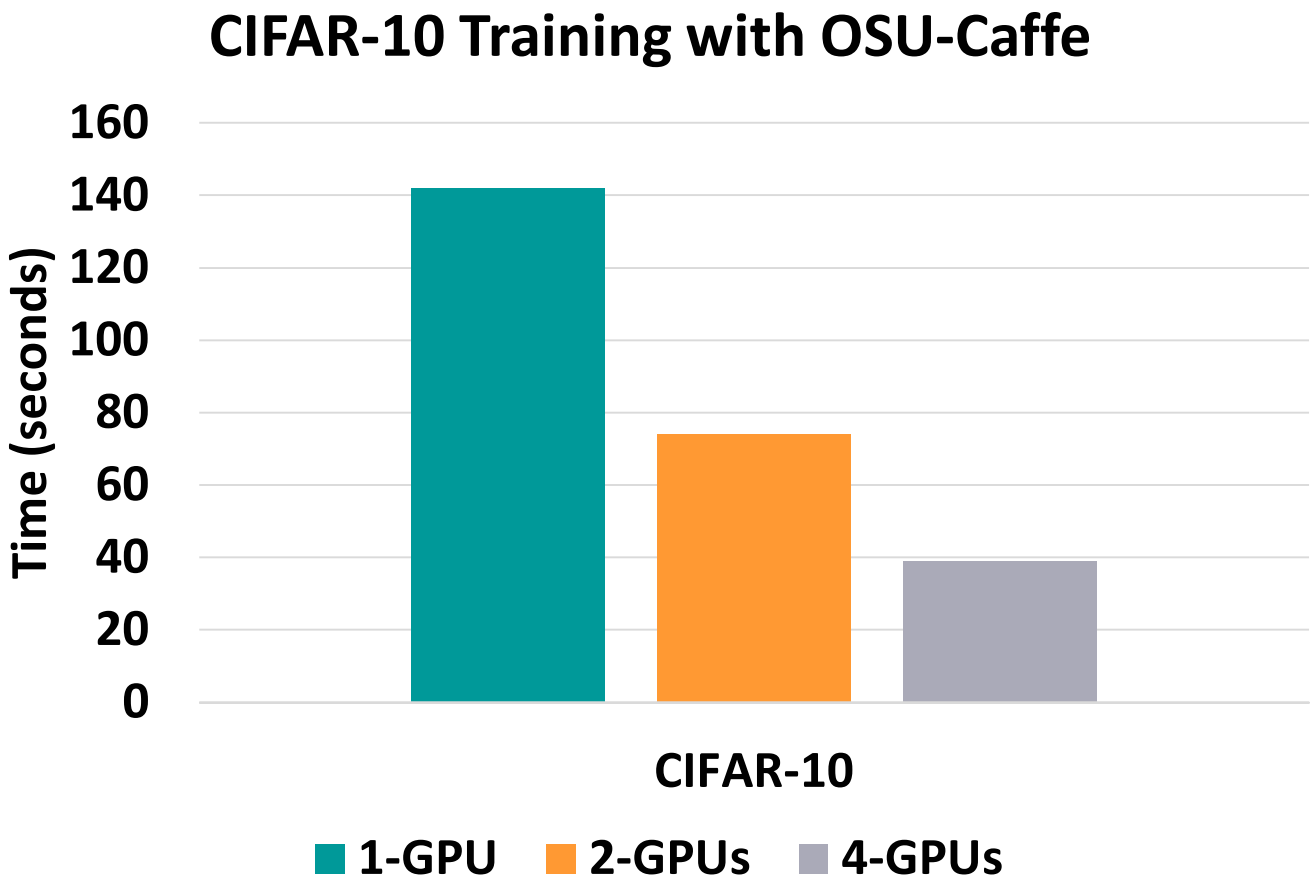
Courtesy: http://engineering.skymind.io/distributed-deep-learning-part-1-an-introduction-to-distributed-training-of-neural-networks

Benefits of Distributed Training: An Example with Caffe

- Strong scaling CIFAR10 Training with OSU-Caffe (1 –> 4 GPUs) – **Batch Size 2K**
- Large batch size is needed for scalability.
- Adding more GPUs may degrade the scaling efficiency

Run Command - (change $np from 1—4)

mpirun_rsh -np **$np** ./build/tools/caffe train -solver examples/cifar10/cifar10_quick_solver.prototxt -scal **strong**



Output: I0123 21:49:24.289763 75582 caffe.cpp:351] Avg. Time Taken: 142.101

Output: I0123 21:54:03.449211 97694 caffe.cpp:351] Avg. Time Taken: 74.6679

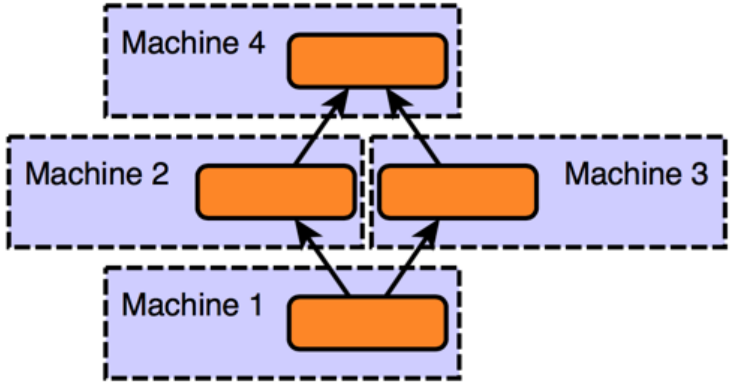
Output: I0123 22:02:46.858219 20659 caffe.cpp:351] Avg. Time Taken: 39.8109

OSU-Caffe is available from the HiDL project page (http://hidl.cse.ohio-state.edu)

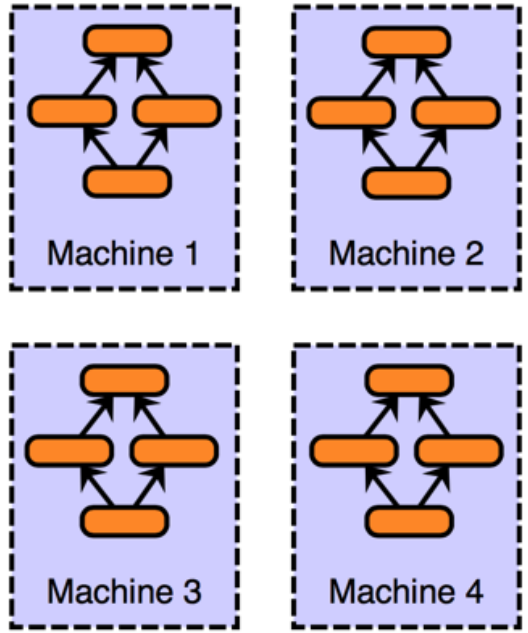
Parallelization Strategies

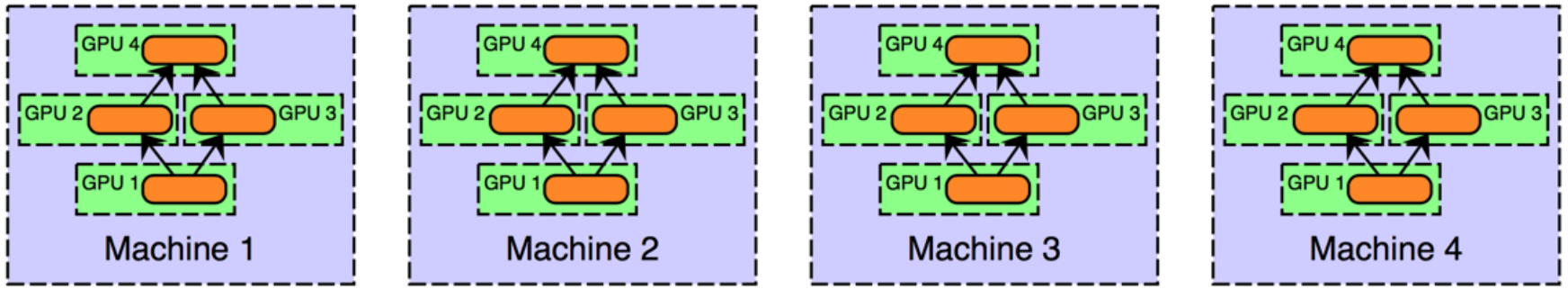
- What are the Parallelization Strategies
  - Model Parallelism
  - **Data Parallelism (Received the most attention)**
  - Hybrid Parallelism
  - Automatic Selection

Model Parallelism



Data Parallelism

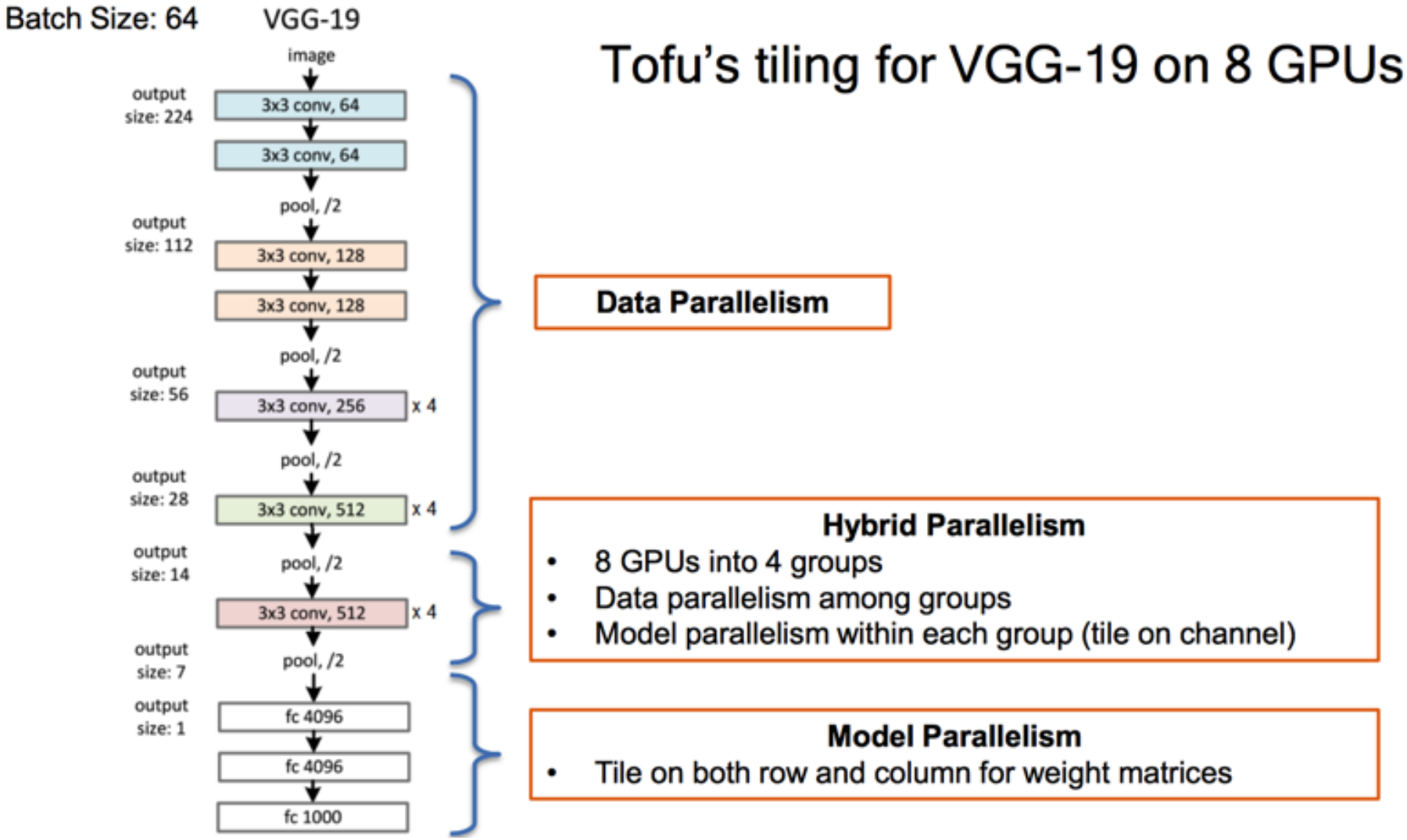




Hybrid (Model and Data) Parallelism

Courtesy: http://engineering.skymind.io/distributed-deep-learning-part-1-an-introduction-to-distributed-training-of-neural-networks

Automatic Selection of Parallelization Strategies



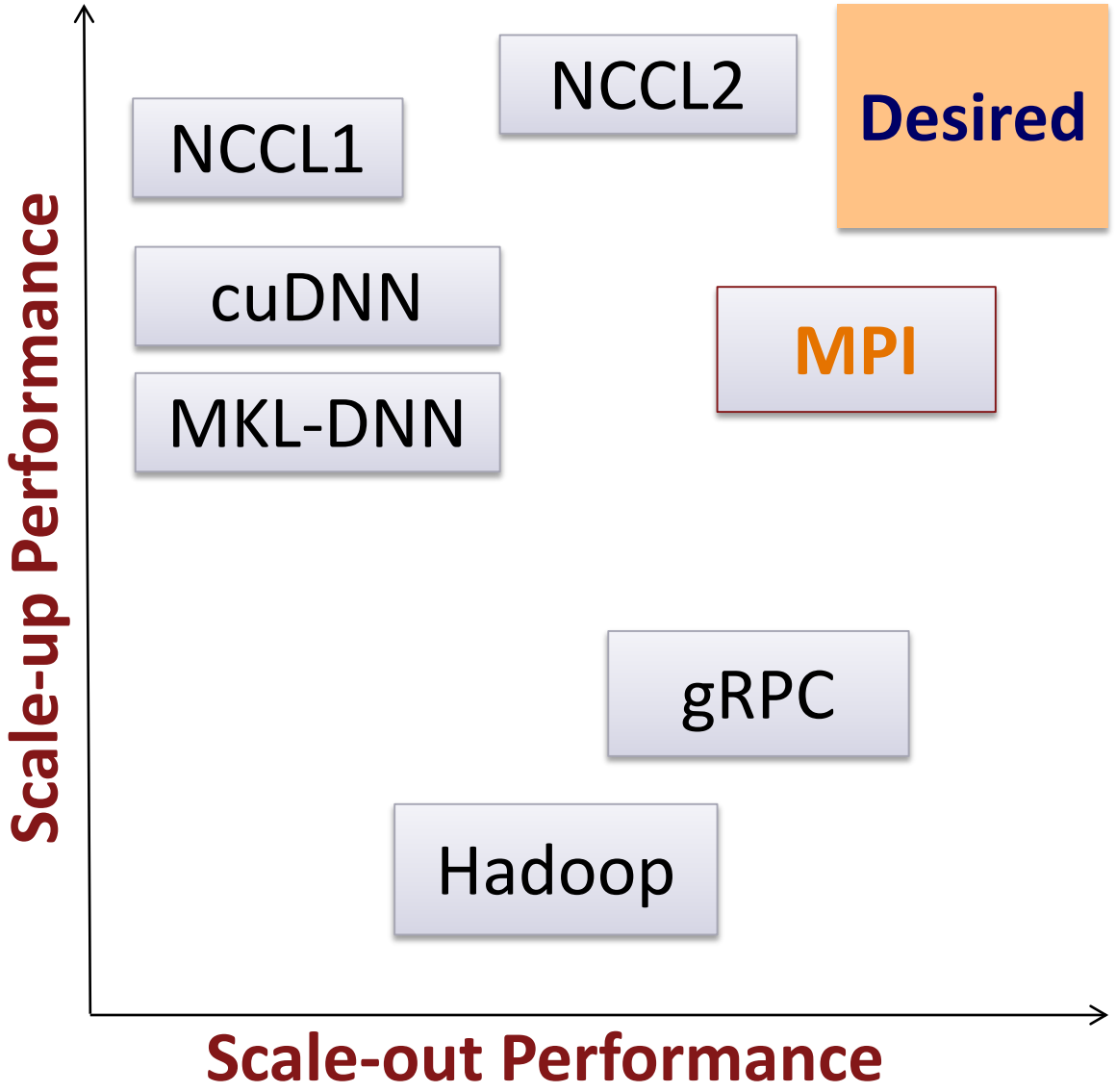
Courtesy: http://on-demand.gputechconf.com/gtc/2017/presentation/s7724-minjie-wong-tofu-parallelizing-deep-learning.pdf

Communication in Distributed Frameworks

- What are the Design Choices for Communication?
  - Established paradigms like Message Passing Interface (MPI)
  - Develop specific communication libraries like NCCL, Gloo, Baidu-allreduce, etc.
  - Use Big-Data frameworks like Spark, Hadoop, etc.
    - Still need some form of external communication for parameters (RDMA, InfiniBand, etc.)
- Focus on Scale-up and Scale-out
  - What are the challenges and opportunities?

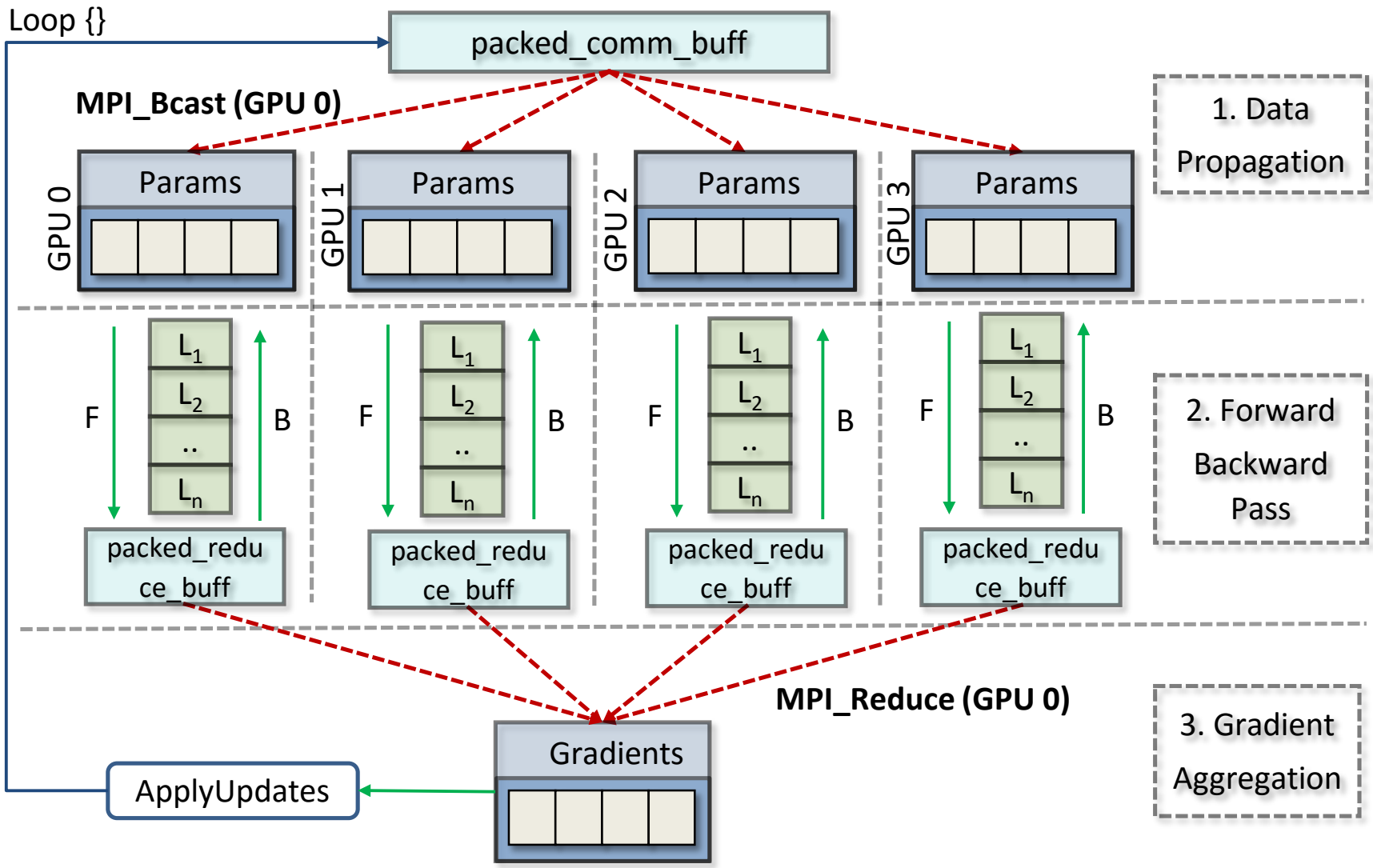
Scale-up and Scale-out

- **Scale-up**: Intra-node Communication
  - Many improvements like:
    - NVIDIA cuDNN, cuBLAS, NCCL, etc.
    - CUDA 9 Co-operative Groups
- **Scale-out**: Inter-node Communication
  - DL Frameworks – most are optimized for single-node only
  - Distributed (Parallel) Training is an emerging trend
    - **OSU-Caffe – MPI-based**
    - Microsoft CNTK – MPI/NCCL2
    - Google TensorFlow – gRPC-based/MPI/NCCL2
    - Facebook Caffe2 – Hybrid (NCCL2/Gloo/MPI)



Data Parallel Deep Learning and MPI Collectives

- Major **MPI Collectives** involved in Designing distributed frameworks
- **MPI_Bcast** – required for DNN parameter exchange
- **MPI_Reduce** – needed for gradient accumulation from multiple solvers
- **MPI_Allreduce** – use just one Allreduce instead of Reduce and Broadcast



A. A. Awan, K. Hamidouche, J. M. Hashmi, and D. K. Panda, S-Caffe: Co-designing MPI Runtimes and Caffe for Scalable Deep Learning on Modern GPU Clusters. In *Proceedings of the 22nd ACM SIGPLAN Symposium on Principles and Practice of Parallel Programming* (PPoPP '17)

Outline

- Introduction
- Overview of Execution Environments
- Parallel and Distributed DNN Training
- **Latest Trends in HPC Technologies**
- Challenges in Exploiting HPC Technologies for Deep Learning
- Solutions and Case Studies
- Open Issues and Challenges
- Conclusion

Drivers of Modern HPC Cluster Architectures

Multi-core Processors

High Performance Interconnects - InfiniBand
<1usec latency, 100Gbps Bandwidth>

Accelerators / Coprocessors high compute density, high performance/watt
>1 TFlop DP on a chip

SSD, NVMe-SSD, NVRAM

- Multi-core/many-core technologies
- Remote Direct Memory Access (RDMA)-enabled networking (InfiniBand and RoCE)
- Solid State Drives (SSDs), Non-Volatile Random-Access Memory (NVRAM), NVMe-SSD
- Accelerators (NVIDIA GPGPUs and Intel Xeon Phi)

Tianhe – 2

Titan

Stampede

Tianhe – 1A

HPC Technologies

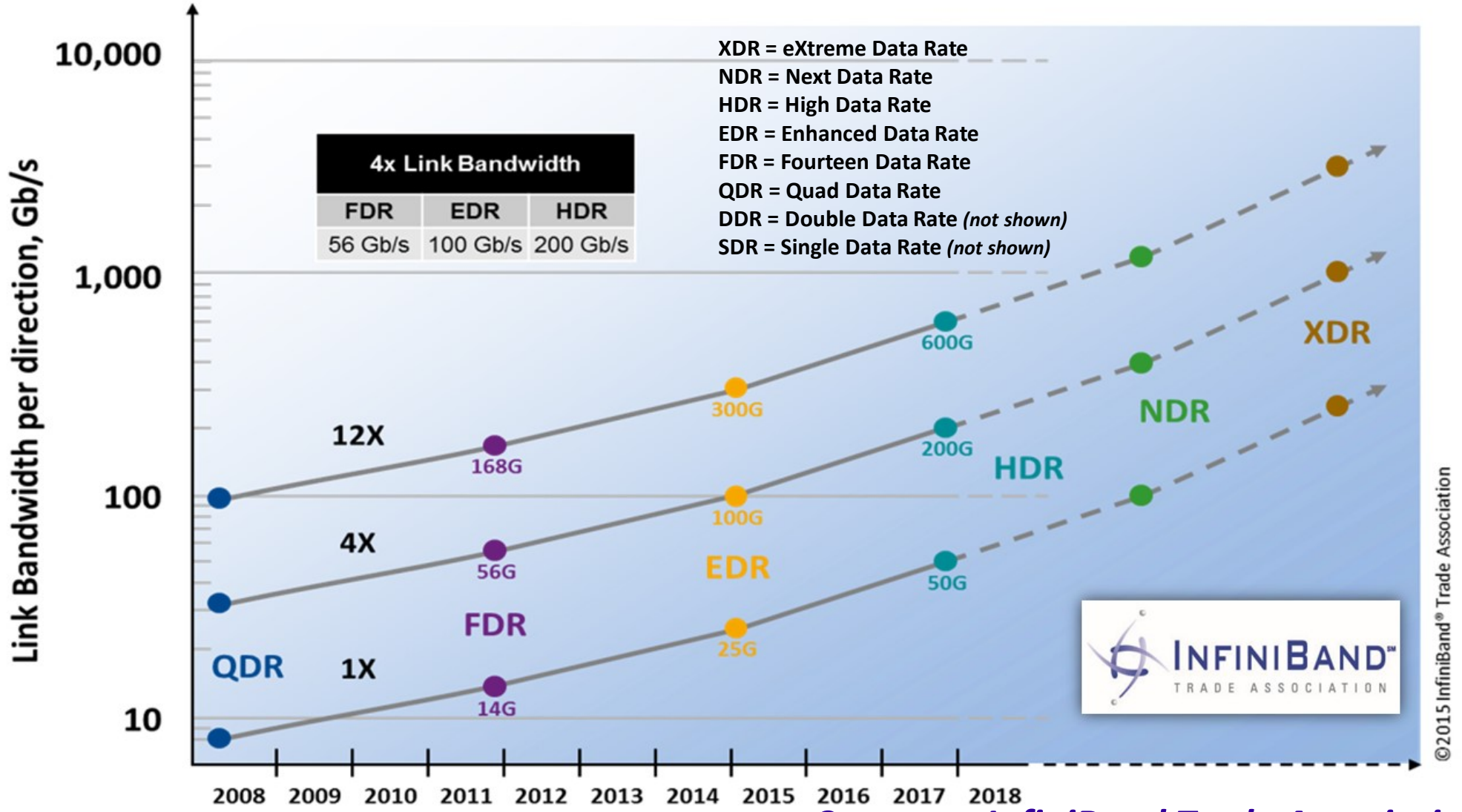
- **Hardware**
  - **Interconnects – InfiniBand, RoCE, Omni-Path, etc.**
  - **Processors – GPUs, Multi-/Many-core CPUs, Tensor Processing Unit (TPU), FPGAs, etc.**
  - **Storage – NVMe, SSDs, Burst Buffers, etc.**
- Communication Middleware
  - Message Passing Interface (MPI)
    - CUDA-Aware MPI, Many-core Optimized MPI runtimes (KNL-specific optimizations)
  - NVIDIA NCCL

Network Speed Acceleration with IB and HSE

Ethernet (1979 -)	10 Mbit/sec
Fast Ethernet (1993 -)	100 Mbit/sec
Gigabit Ethernet (1995 -)	1000 Mbit /sec
ATM (1995 -)	155/622/1024 Mbit/sec
Myrinet (1993 -)	1 Gbit/sec
Fibre Channel (1994 -)	1 Gbit/sec
InfiniBand (2001 -)	2 Gbit/sec (1X SDR)
10-Gigabit Ethernet (2001 -)	10 Gbit/sec
InfiniBand (2003 -)	8 Gbit/sec (4X SDR)
InfiniBand (2005 -)	16 Gbit/sec (4X DDR)
	24 Gbit/sec (12X SDR)
InfiniBand (2007 -)	32 Gbit/sec (4X QDR)
40-Gigabit Ethernet (2010 -)	40 Gbit/sec
InfiniBand (2011 -)	54.6 Gbit/sec (4X FDR)
InfiniBand (2012 -)	2 x 54.6 Gbit/sec (4X Dual-FDR)
25-/50-Gigabit Ethernet (2014 -)	25/50 Gbit/sec
100-Gigabit Ethernet (2015 -)	100 Gbit/sec
Omni-Path (2015 -)	100 Gbit/sec
InfiniBand (2015 -)	100 Gbit/sec (4X EDR)
InfiniBand (2016 -)	200 Gbit/sec (4X HDR)

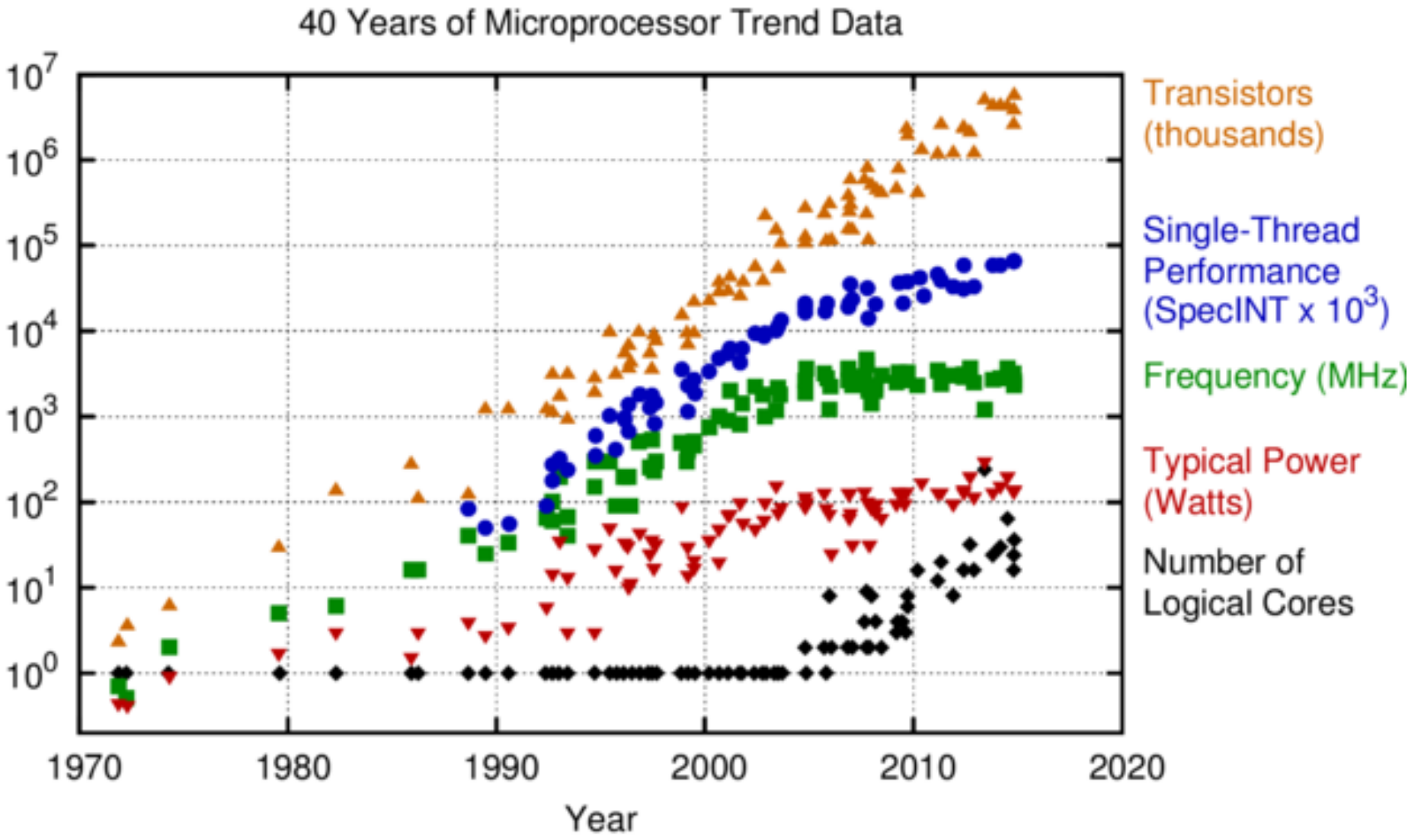
100 times in the last 15 years

InfiniBand Link Speed Standardization Roadmap



Courtesy: InfiniBand Trade Association

Trends in Microprocessor Technology



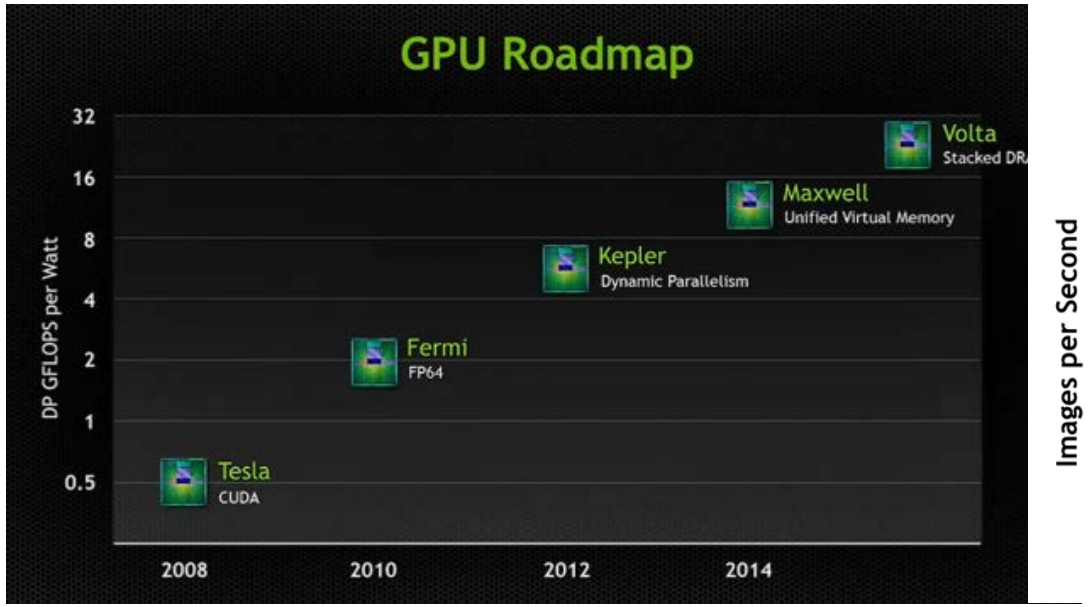
Original data up to the year 2010 collected and plotted by M. Horowitz, F. Labonte, O. Shacham, K. Olukotun, L. Hammond, and C. Batten
New plot and data collected for 2010-2015 by K. Rupp

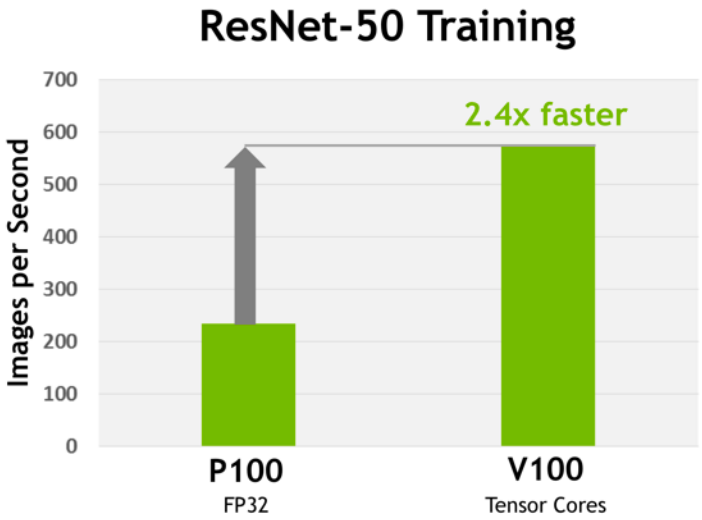
Courtesy: https://www.karlrupp.net/2015/06/40-years-of-microprocessor-trend-data/

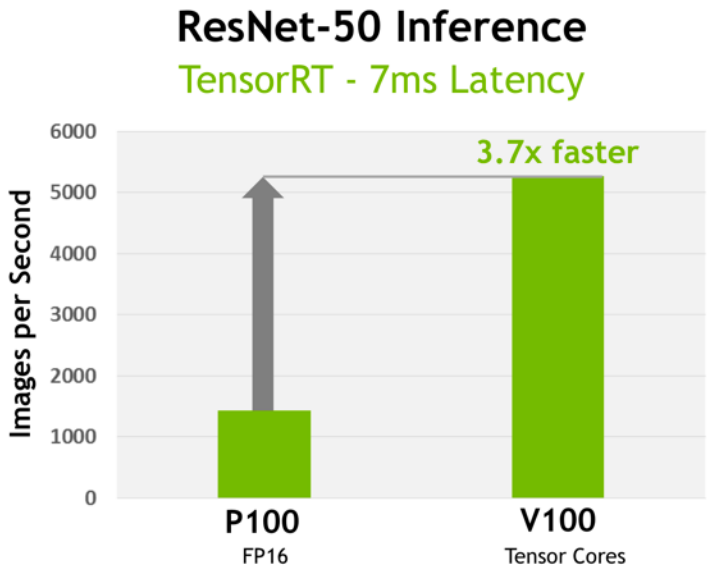
: https://www.top500.org/news/intel-spills-details-on-knights-mill-processor/

- Small, yet steady increase in single thread performance
- Rapid increase in number of transistors per chip
- Power consumption has remained more or less constant
- Rapid increase in number of cores
- Latest Intel Knights Mill expected to have DL optimized hardware

Trends in GPU Technology





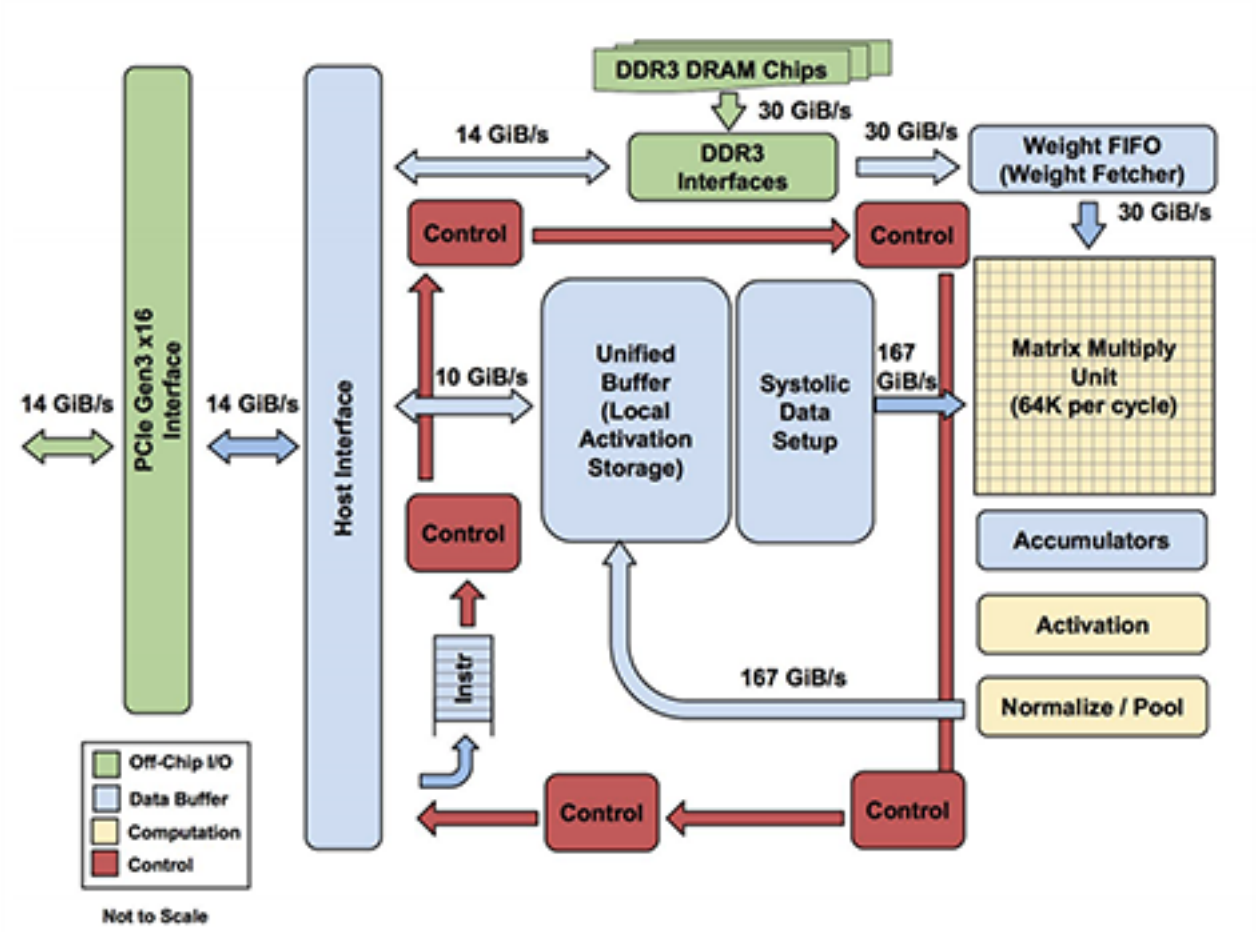


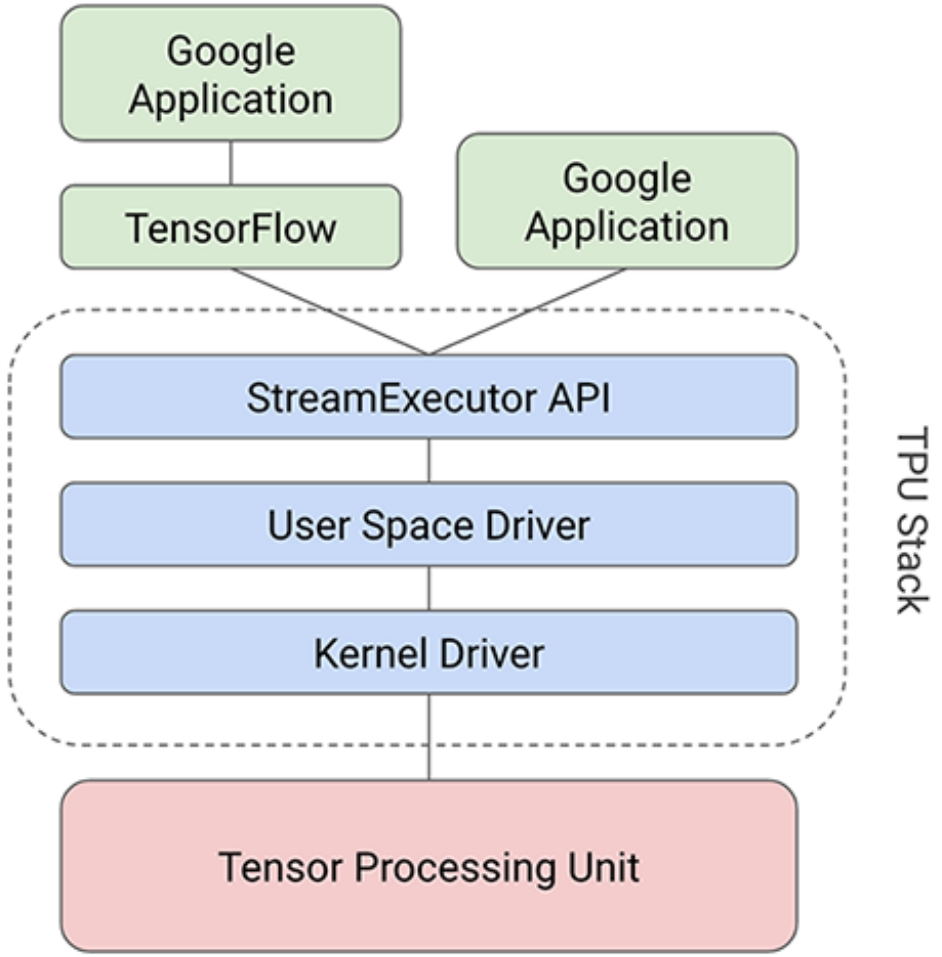
- NVIDIA Volta is optimized for Deep Learning workloads
  - has dedicated "Tensor Cores" (FP16 or half precision) for both Training and Inference
  - 2.4X faster than Pascal GPUs for ResNet-50 training

Courtesy: https://devblogs.nvidia.com/parallelforall/inside-volta/

: http://wccftech.com/nvidia-roadmap-2017-update-volta-gpu/

Google TPU





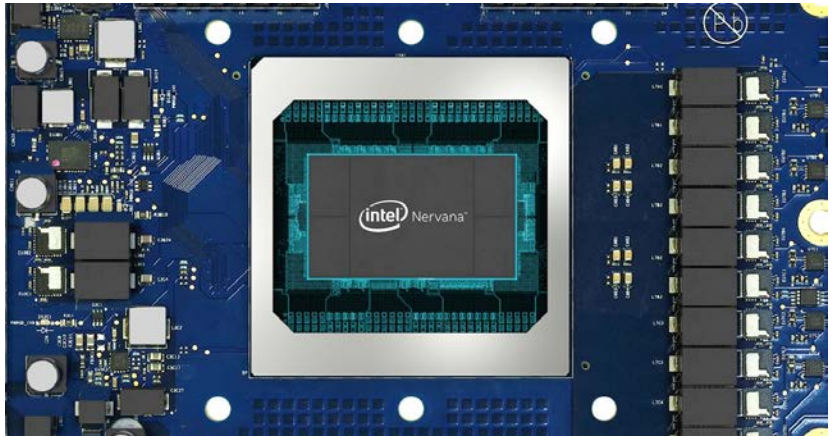
- CISC style instruction set
- Uses systolic arrays as the heart of multiply unit

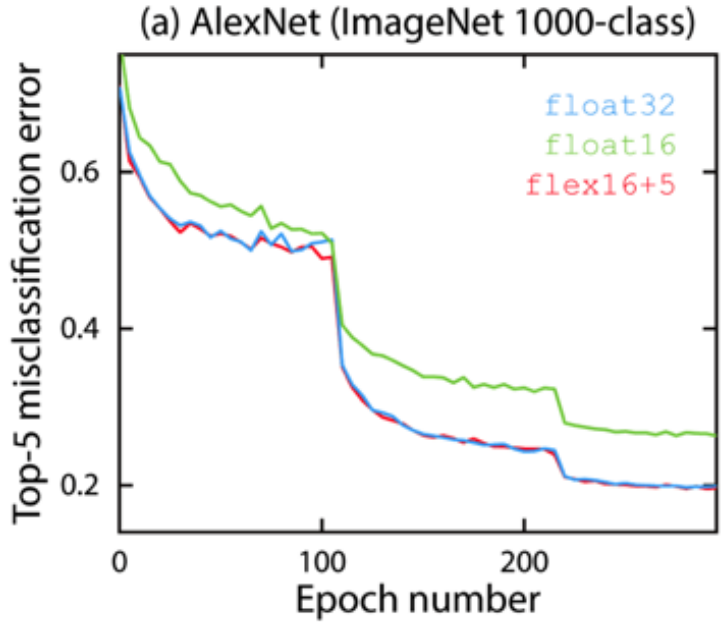
Courtesy: https://cloud.google.com/blog/big-data/2017/05/an-in-depth-look-at-googles-first-tensor-processing-unit-tpu

: https://www.nextplatform.com/2017/04/05/first-depth-look-googles-tpu-architecture/

Intel Neural Network Processor (NNP)

- Intel® Nervana™ Neural Network Processors (NNP)
  - formerly known as “Lake Crest”
- Recently announced as part of Intel’s strategy for next-gen. AI systems
- Purpose built architecture for deep learning
- 1 TB/s High Bandwidth Memory (HBM)
- Spatial Architecture
- FlexPoint format
  - Similar performance (in terms of accuracy) to FP32 while using 16 bits of storage

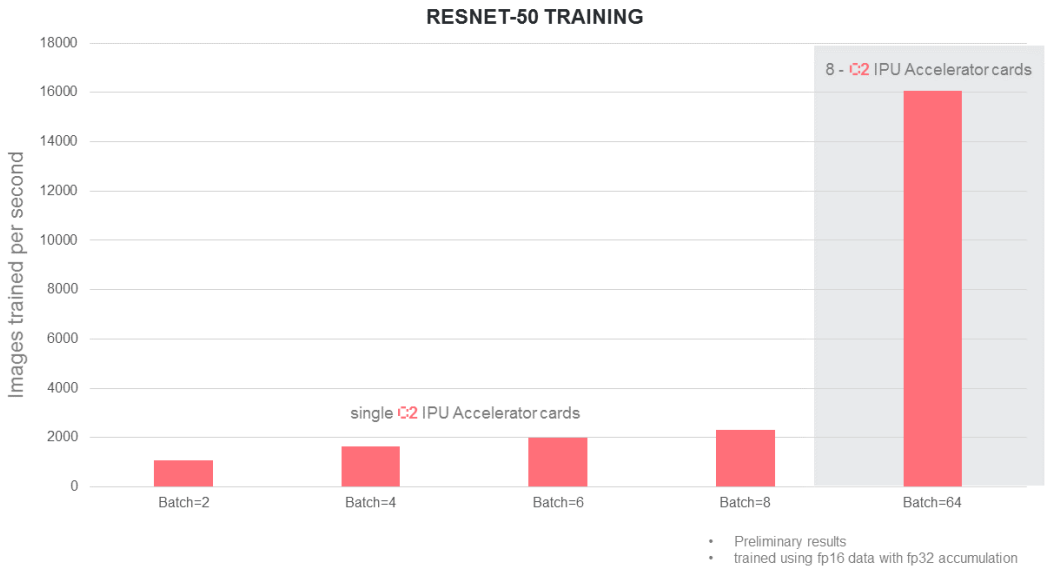


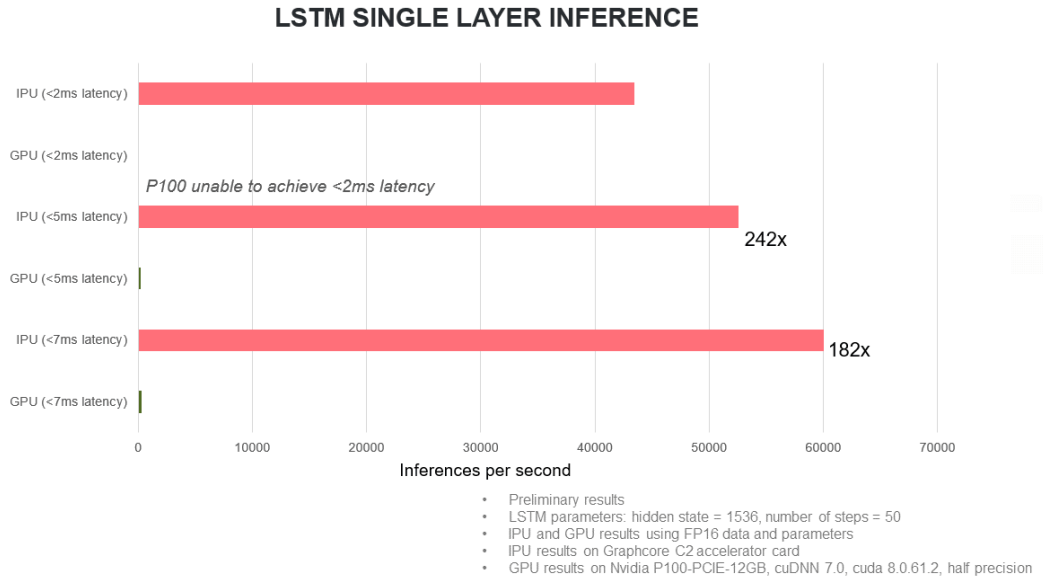


Courtesy: https://ai.intel.com/intel-nervana-neural-network-processor-architecture-update/

GraphCore – Intelligence Processing Unit (IPU)

- New processor that's the first to be specifically designed for machine intelligence workloads – an Intelligence Processing Unit (IPU)
  - Massively parallel
  - Low-precision floating-point compute
  - Higher compute density
- UK-based Startup
- Early benchmarks show 10-100x speedup over GPUs
  - Presented at NIPS 2017

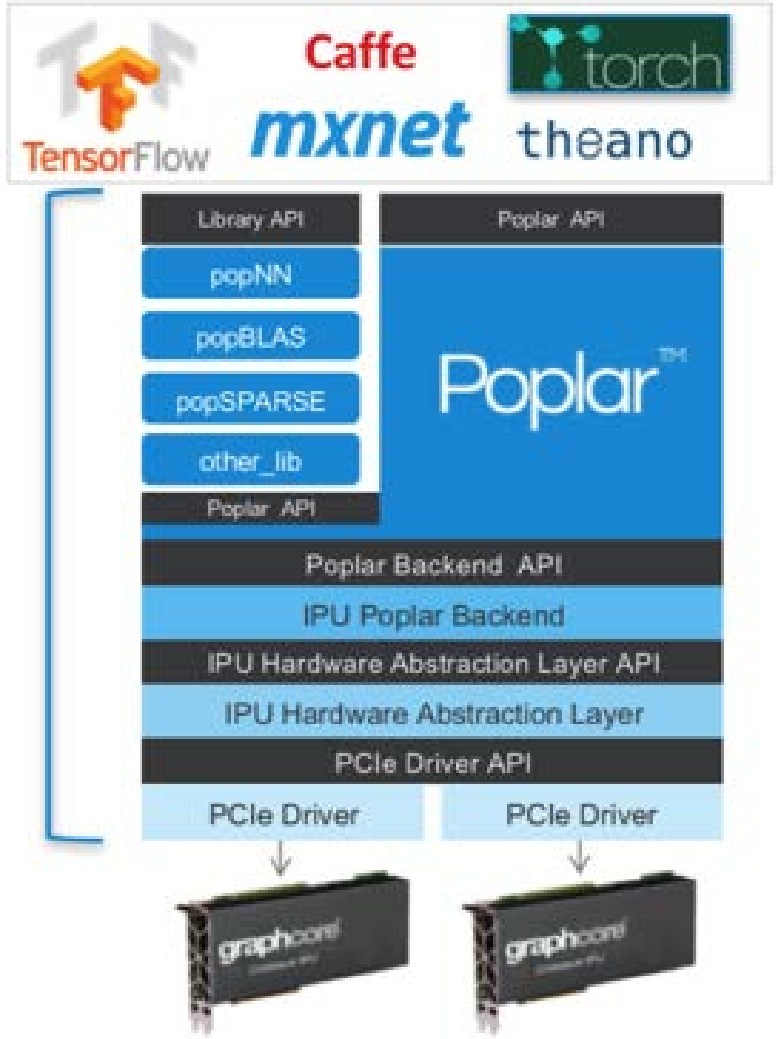




Courtesy: https://www.graphcore.ai/posts/preliminary-ipu-benchmarks-providing-previously-unseen-performance-for-a-range-of-machine-learning-applications

Poplar Graph Programming Framework

- Poplar -- graph programming framework for IPU accelerated platforms
- C++ framework that provides a seamless interface DL frameworks like Tensorflow and MXNet
- Existing applications written for Tensorflow will work out of the box on an IPU.
- Set of drivers, application libraries and debugging and analysis tools



Courtesy: https://www.graphcore.ai/posts/what-does-machine-learning-look-like

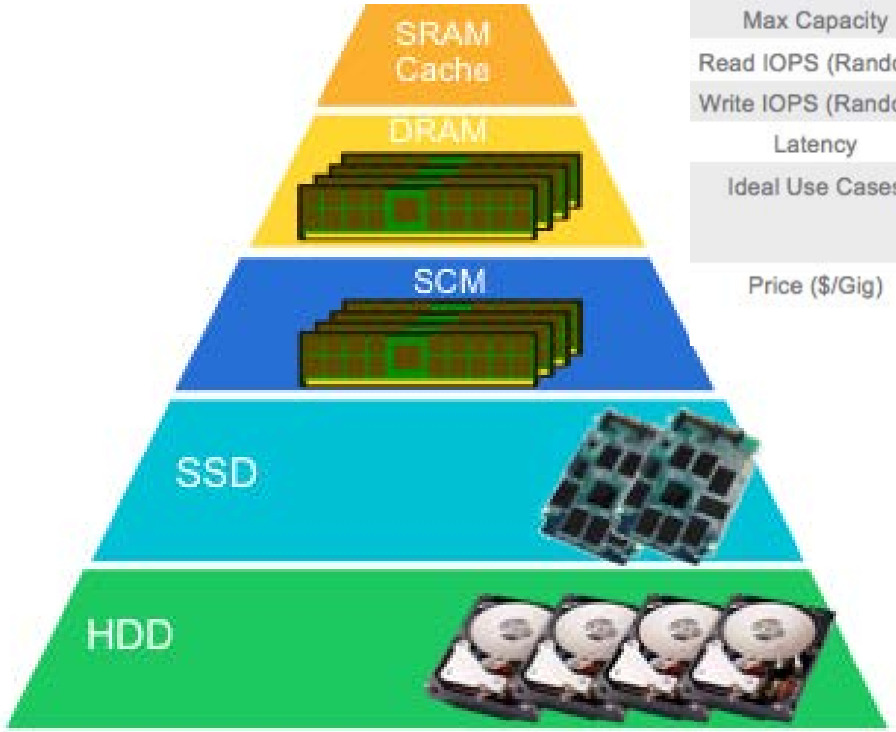
https://www.graphcore.ai/hubfs/assets/Poplar%C2%81%20technical%20overview%20NEW%20BRAND.pdf

Trends in High-Performance Storage

	NVMe	NVRAM	3D XPoint
Definition	High Speed interface for SSDs in a PCIe form factor used as **block storage**	**Non-volatile DRAM** backed up battery or super capacitor used as **byte addressable memory**	Non-volatile high performance (1000x NAND), high density (8-10X DRAM), high endurance (1000X NAND) **byte addressable memory**
Form Factor	Connects to PCIe bus	Connects to a DDR3 DIMM slot	Connects to a DDR3 DIMM slot
Max Capacity	2 TB	**16GB**	**128 GB**
Read IOPS (Random)	750,000	1.4 Million	In millions
Write IOPS (Random)	430,000	1.4 Million	In millions
Latency	15 Microsecond	10 Nanoseconds	10 Nanoseconds
Ideal Use Cases	Caching Tier: Transactional workloads **requiring high IOPS**	**Byte Addressable memory for metadata & client side caching, reduce write amplification**	**Highly Dense Byte Addressable memory for high speed caching, staging dedup/compression**
Price ($/Gig)	**$$**	**$$$**	**$$$$**

Memory

Storage



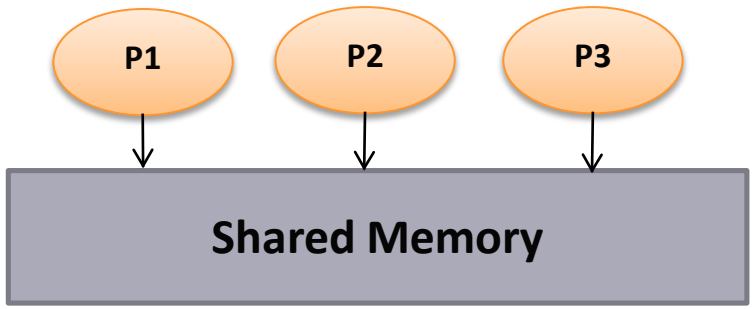
Courtesy: https://blogs.vmware.com/virtualblocks/2016/03/01/the-evolution-of-next-gen-hci-part-2-future-of-all-flash-virtual-san/

: https://www.rambus.com/blogs/mid-when-memory-and-storage-converge/

HPC Technologies

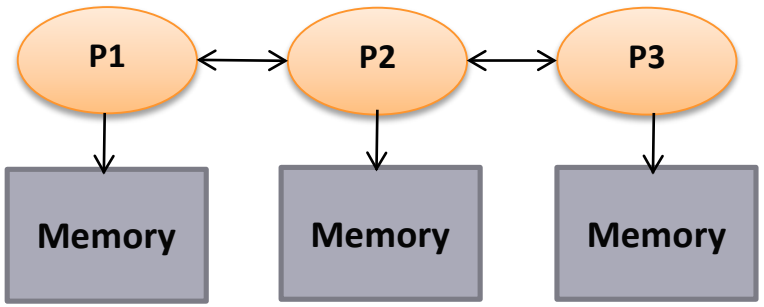
- Hardware
  - Interconnects – InfiniBand, RoCE, Omni-Path, etc.
  - Processors – GPUs, Multi-/Many-core CPUs, Tensor Processing Unit (TPU), FPGAs, etc.
  - Storage – NVMe, SSDs, Burst Buffers, etc.
- **Communication Middleware**
  - **Message Passing Interface (MPI)**
    - **CUDA-Aware MPI, Many-core Optimized MPI runtimes (KNL-specific optimizations)**
  - **NVIDIA NCCL**

Parallel Programming Models Overview



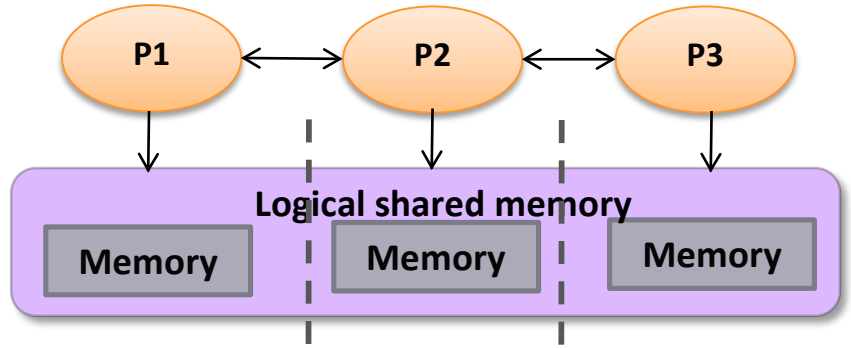
Shared Memory Model

SHMEM, DSM



Distributed Memory Model

MPI (Message Passing Interface)



Partitioned Global Address Space (PGAS)

Global Arrays, UPC, Chapel, X10, CAF, …

- Programming models provide abstract machine models
- Models can be mapped on different types of systems
  - e.g. Distributed Shared Memory (DSM), MPI within a node, etc.
- PGAS models and Hybrid MPI+PGAS models are gradually receiving importance

MPI Features and Implementations

- Major MPI features
  - Point-to-point Two-sided Communication
  - Collective Communication
  - One-sided Communication
- Message Passing Interface (MPI)
  - **MVAPICH2**
  - OpenMPI, IntelMPI, CrayMPI, IBM Spectrum MPI
  - And many more…

MVAPICH

Allreduce Collective Communication Pattern

- Element-wise Sum data from all processes and sends to all processes

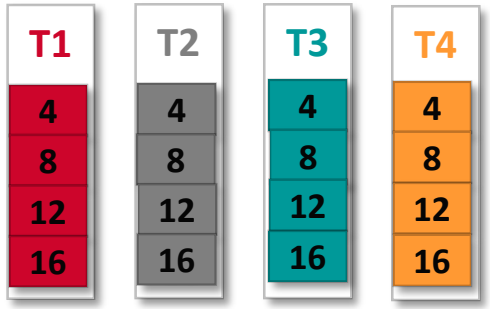
int MPI_Allreduce (const void *sendbuf, void * recvbuf, int count, MPI_Datatype datatype, MPI_Op operation, MPI_Comm comm)

Input-only Parameters	
Parameter	**Description**
sendbuf	Starting address of send buffer
recvbuf	Starting address of recv buffer
type	Data type of buffer elements
count	Number of elements in the buffers
operation	Reduction operation to be performed (e.g. sum)
comm	Communicator handle

Input/Output Parameters	
Parameter	**Description**
recvbuf	Starting address of receive buffer

Sendbuf (Before)

Recvbuf (After)



Overview of the MVAPICH2 Project

- High Performance open-source MPI Library for InfiniBand, Omni-Path, Ethernet/iWARP, and RDMA over Converged Ethernet (RoCE)
  - MVAPICH (MPI-1), MVAPICH2 (MPI-2.2 and MPI-3.1), Started in 2001, First version available in 2002
  - MVAPICH2-X (MPI + PGAS), Available since 2011
  - Support for GPGPUs (MVAPICH2-GDR) and MIC (MVAPICH2-MIC), Available since 2014
  - Support for Virtualization (MVAPICH2-Virt), Available since 2015
  - Support for Energy-Awareness (MVAPICH2-EA), Available since 2015
  - Support for InfiniBand Network Analysis and Monitoring (OSU INAM) since 2015
  - **Used by more than 2,875 organizations in 86 countries**
  - **More than 461,000 (> 0.46 million) downloads from the OSU site directly**
  - Empowering many TOP500 clusters (Nov '17 ranking)
    - **1st, 10,649,600-core (Sunway TaihuLight) at National Supercomputing Center in Wuxi, China**
    - 9th, 556,104 cores (Oakforest-PACS) in Japan
    - 12th, 368,928-core (Stampede2) at TACC
    - 17th, 241,108-core (Pleiades) at NASA
    - 48th, 76,032-core (Tsubame 2.5) at Tokyo Institute of Technology
  - Available with software stacks of many vendors and Linux Distros (RedHat and SuSE)
  - **http://mvapich.cse.ohio-state.edu**
- Empowering Top500 systems for over a decade

GPU-Aware (CUDA-Aware) MPI Library: MVAPICH2-GDR

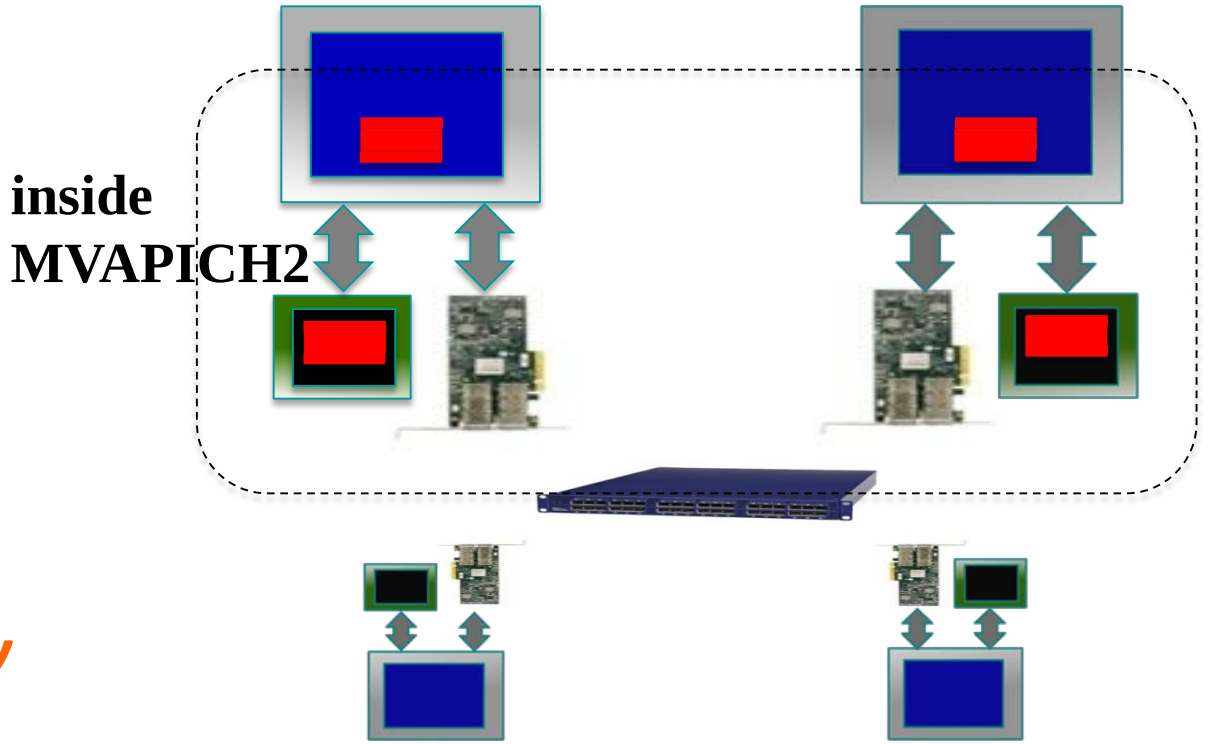
- Standard MPI interfaces used for unified data movement
- Takes advantage of Unified Virtual Addressing (>= CUDA 4.0)
- Overlaps data movement from GPU with RDMA transfers

At Sender:

```
MPI_Send(s_devbuf, size, …);
```

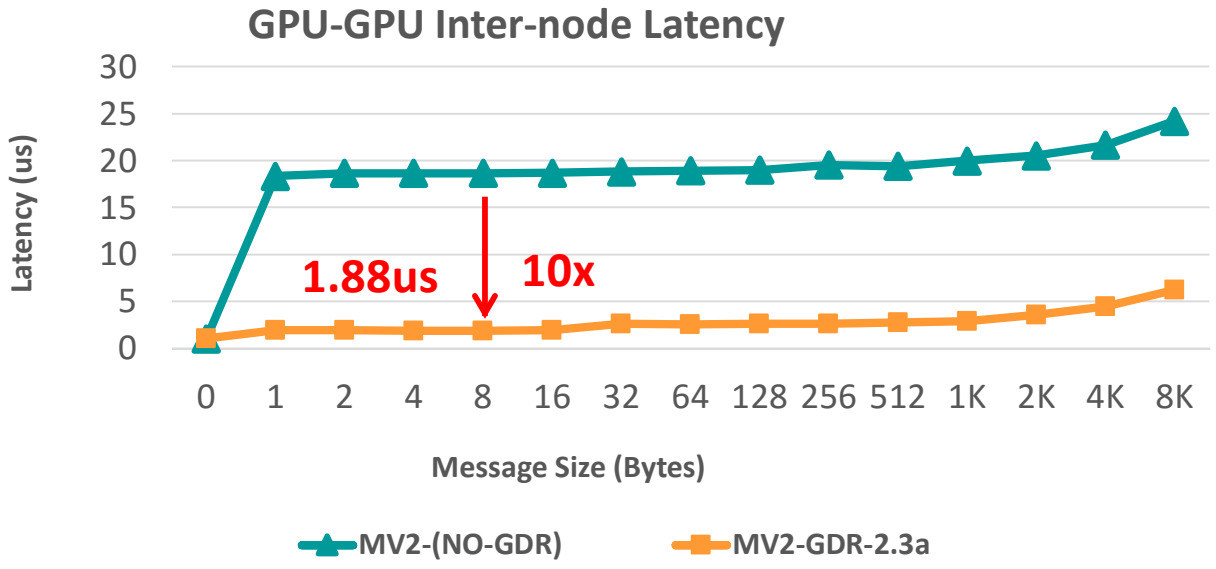
At Receiver:

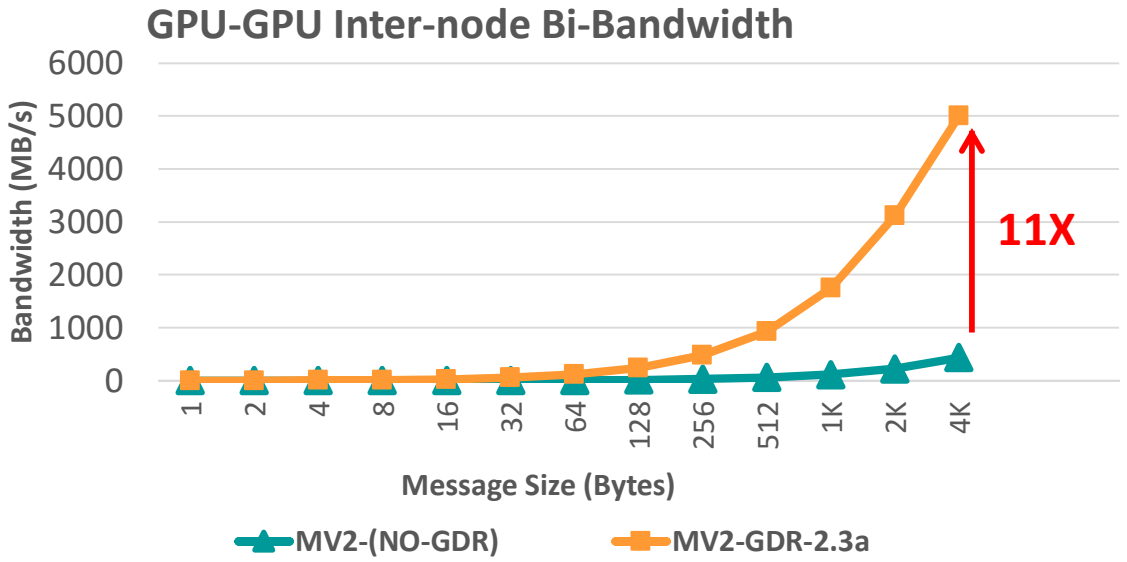
```
MPI_Recv(r_devbuf, size, …);
```

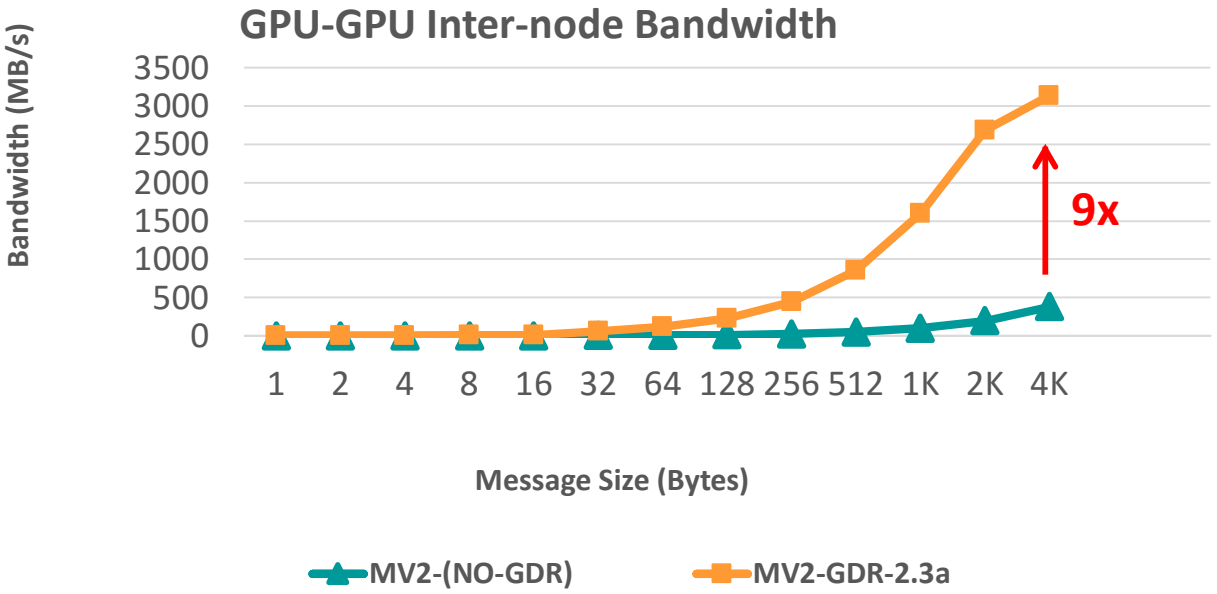


High Performance and High Productivity

Optimized MVAPICH2-GDR Design



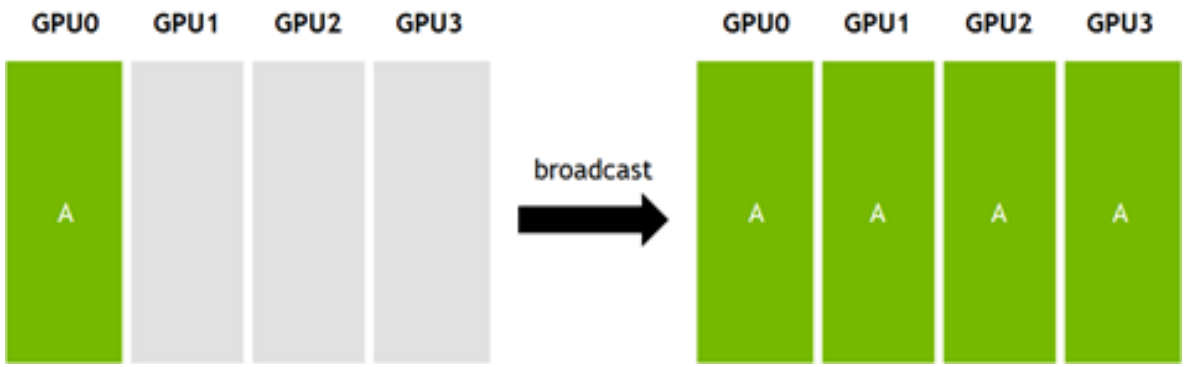


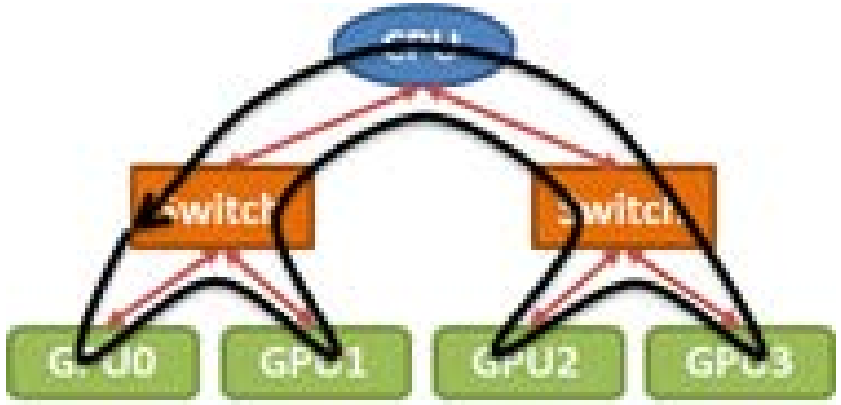


MVAPICH2-GDR-2.3a
Intel Haswell (E5-2687W @ 3.10 GHz) node - 20 cores
NVIDIA Volta V100 GPU
Mellanox Connect-X4 EDR HCA
CUDA 9.0
Mellanox OFED 4.0 with GPU-Direct-RDMA

NCCL Communication Library

- Collective Communication with a caveat!
  - GPU buffer exchange
  - **Dense Multi-GPU** systems
    (Cray CS-Storm, DGX-1)
  - MPI-like – but not MPI standard compliant
- NCCL (pronounced Nickel)
  - Open-source Communication Library by NVIDIA
  - Topology-aware, ring-based (linear) collective communication library for GPUs
  - Divide bigger buffers to smaller chunks
  - Good performance for large messages
    - Kernel-based threaded copy (Warp-level Parallel) instead of cudaMemcpy





https://devblogs.nvidia.com/parallelforall/fast-multi-gpu-collectives-nccl/

Outline

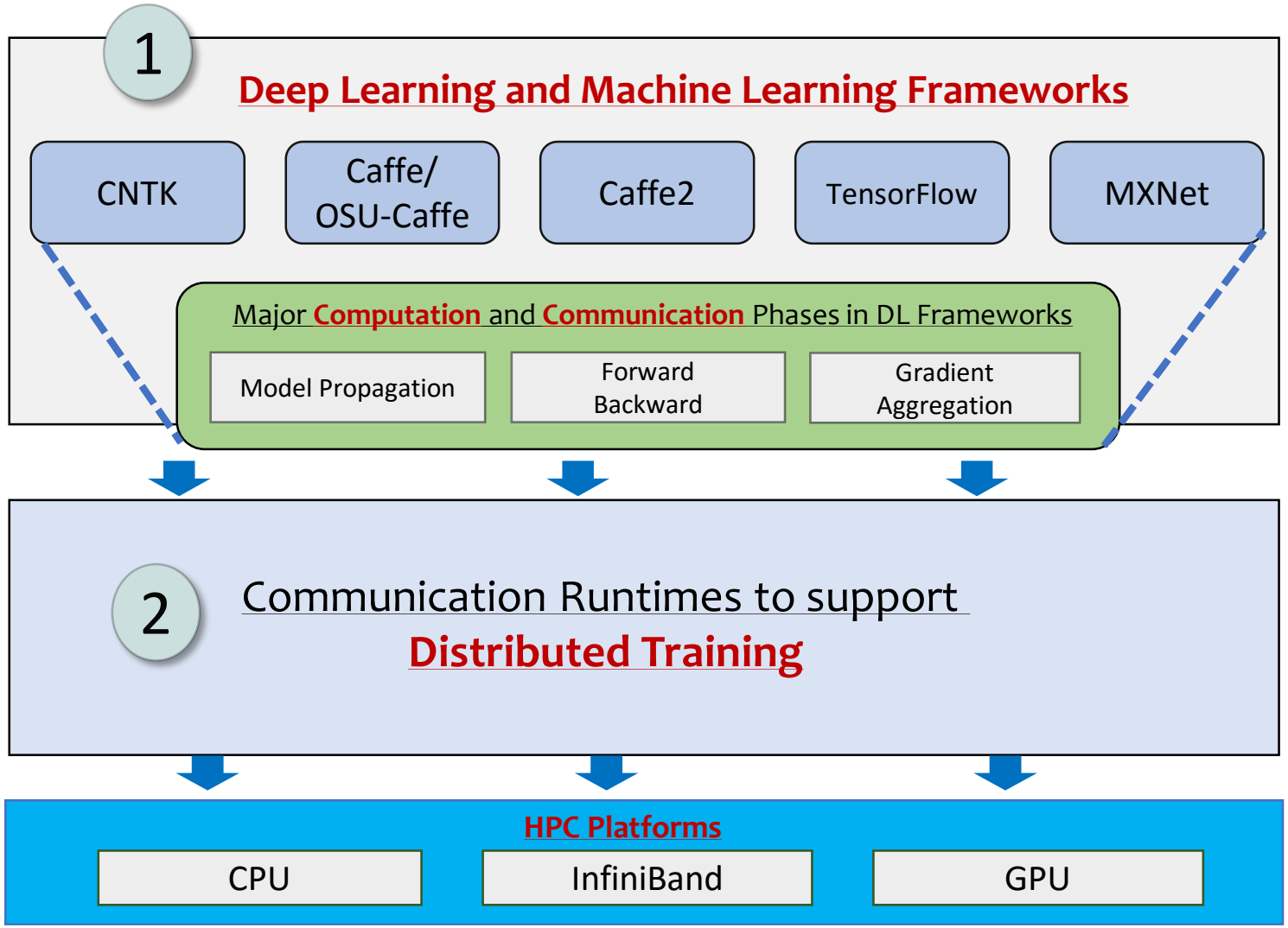
- Introduction
- Overview of Execution Environments
- Parallel and Distributed DNN Training
- Latest Trends in HPC Technologies
- **Challenges in Exploiting HPC Technologies for Deep Learning**
- Solutions and Case Studies
- Open Issues and Challenges
- Conclusion

Broad Challenge: Exploiting HPC for Deep Learning

How to efficiently scale-out a Deep Learning (DL) framework and take advantage of heterogeneous High Performance Computing (HPC) resources?

Research Challenges to Exploit HPC Technologies

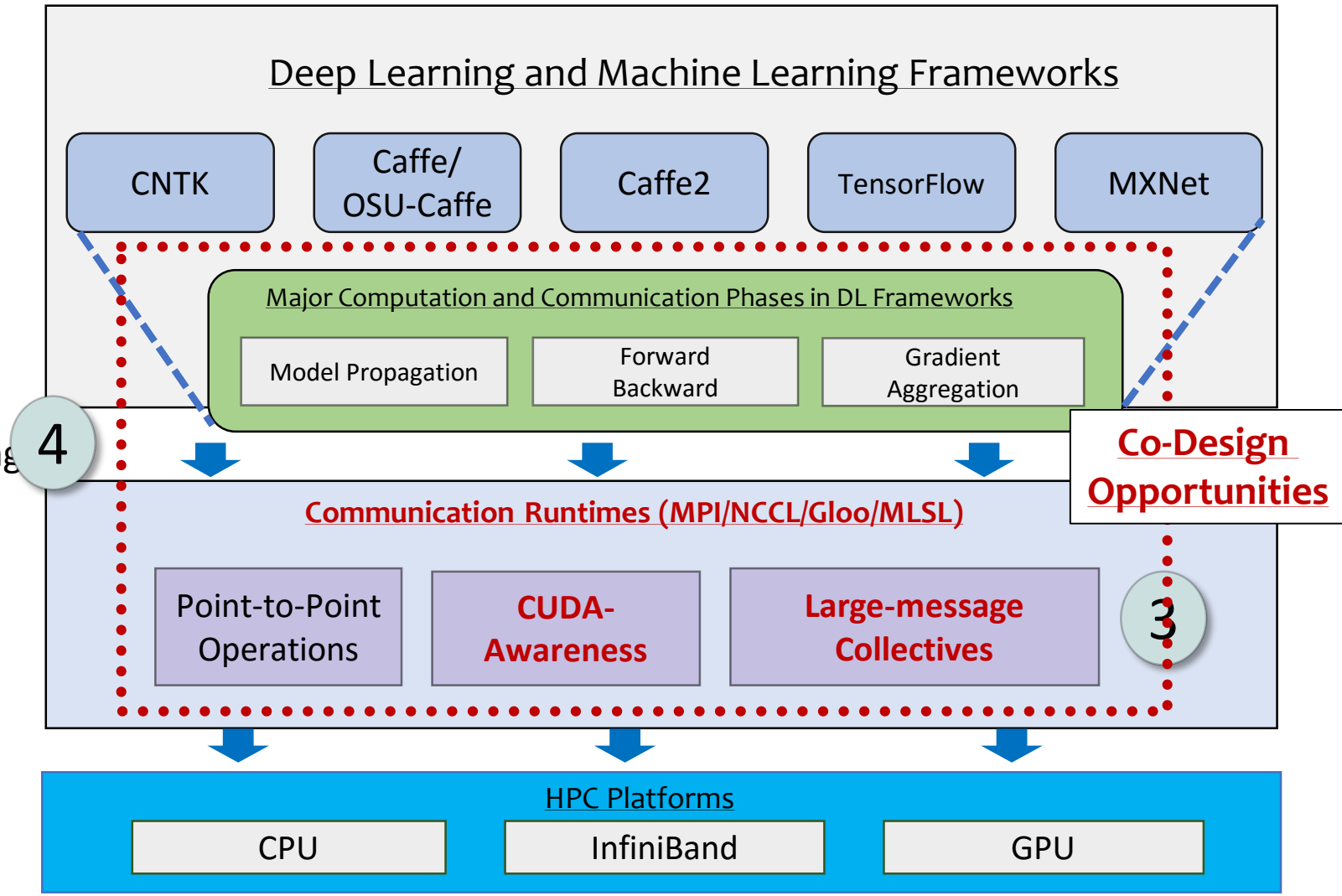
1. What are the fundamental issues in designing **DL frameworks**?
   - Memory Requirements
   - **Computation** Requirements
   - **Communication** Overhead
2. Why do we need to support **distributed training**?
   - To overcome the limits of single-node training
   - To better utilize hundreds of existing HPC Clusters



Research Challenges to Exploit HPC Technologies (Cont'd)

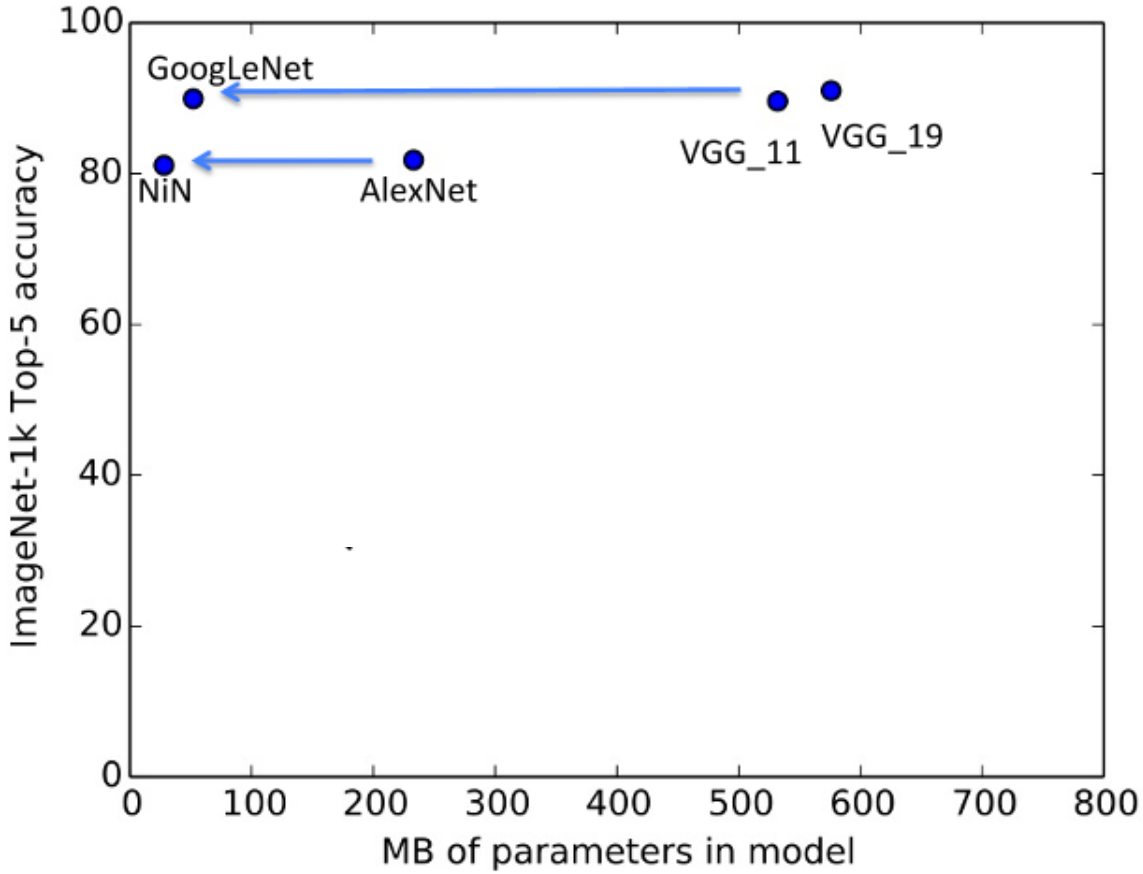
3. What are the **new design challenges** brought forward by DL frameworks for Communication runtimes?
   - Large Message **Collective Communication** and Reductions
   - GPU Buffers (**CUDA-Awareness**)

4. Can a **Co-design** approach help in achieving Scale-up and Scale-out efficiently?
   - **Co-Design** the support at **Runtime level** and Exploit it at the **DL Framework level**
   - What performance benefits can be observed?
   - What needs to be fixed at the **communication runtime** layer?



Large Message Communication and Reduction: Focus Area

- What are the new requirements and expectations for Communication Runtimes?
  - Efficiently handle very-large buffer sizes
    - Megabytes (MB) for now
    - Expect Gigabytes (GB) in future
  - New algorithms and implementations will be needed!
  - GPU buffers in existing DL frameworks
  - Importance of efficient CUDA-Aware MPI will increase even more!



Courtesy: http://arxiv.org/abs/1511.00175

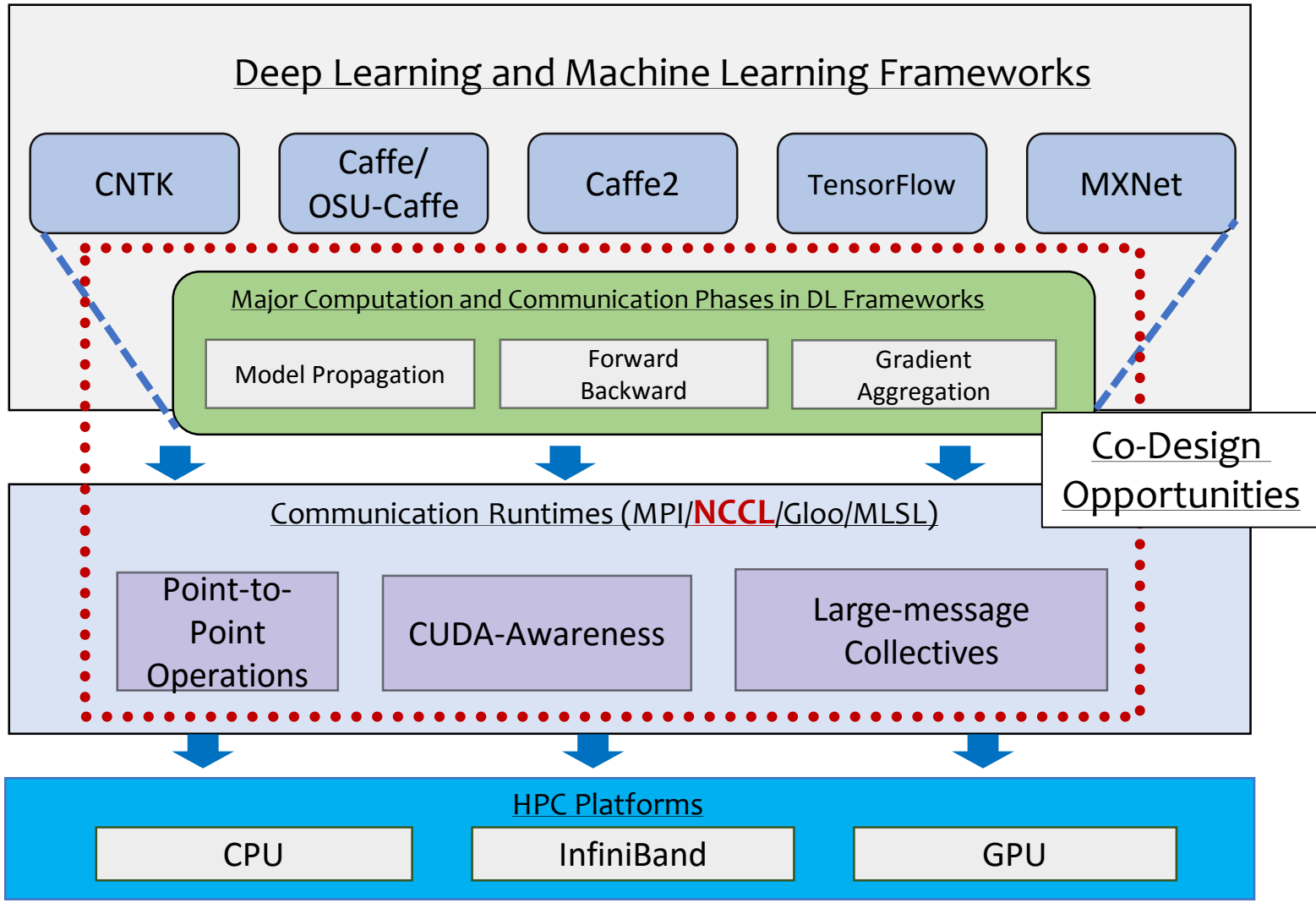
- Can MPI provide a holistic solution to this problem?

Outline

- Introduction
- Overview of Execution Environments
- Parallel and Distributed DNN Training
- Latest Trends in HPC Technologies
- Challenges in Exploiting HPC Technologies for Deep Learning
- **Solutions and Case Studies**
- Open Issues and Challenges
- Conclusion

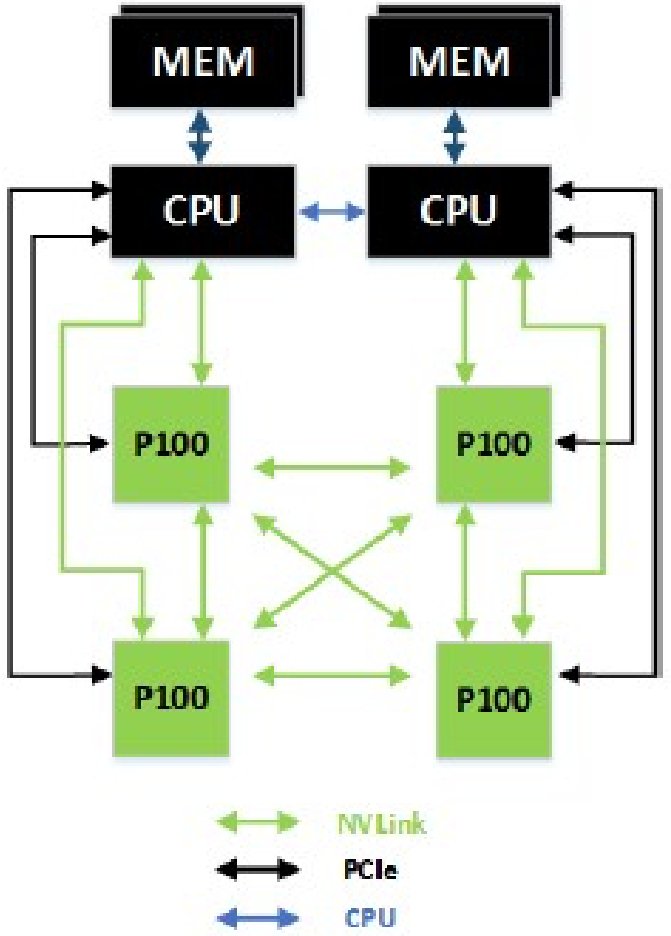
Solutions and Case Studies: Exploiting HPC for DL

- **NVIDIA NCCL**
- Baidu-allreduce
- Facebook Gloo
- Co-design MPI runtimes and DL Frameworks
  - MPI+NCCL for CUDA-Aware CNTK
  - OSU-Caffe
- TensorFlow (Horovod)
- Scaling DNN Training on Multi-/Many-core CPUs
- PowerAI DDL



NVIDIA NCCL

- NCCL is a collective communication library
  - NCCL 1.x is only for Intra-node communication on a single-node
- NCCL 2.0 supports inter-node communication as well
- Design Philosophy
  - Use Rings and CUDA Kernels to perform efficient communication
- NCCL is optimized for dense multi-GPU systems like the DGX-1 and DGX-1V



Fully connected quad

120 GB/s per GPU bidirectional for peer traffic

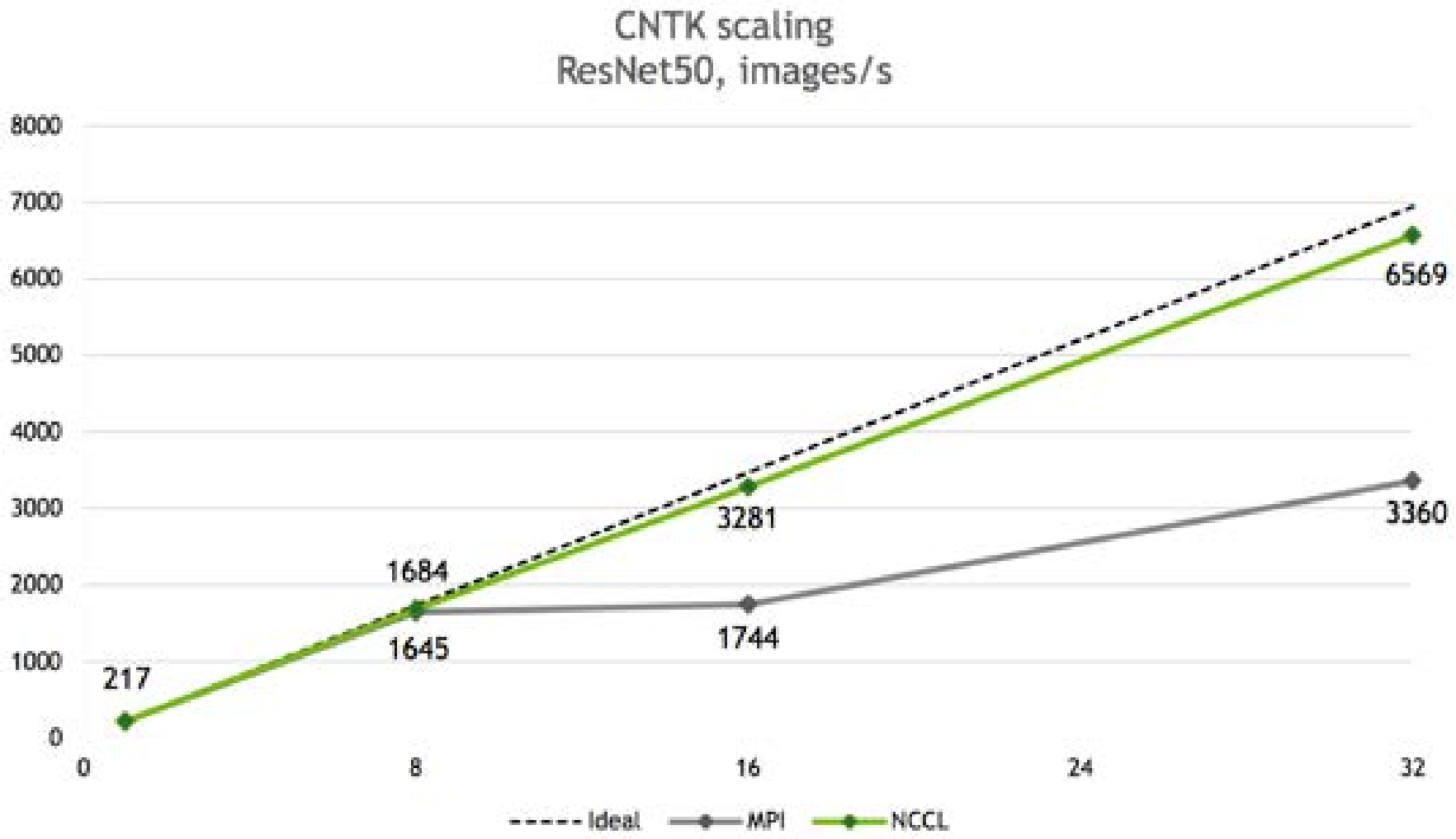
40 GB/s per GPU bidirectional to CPU

Direct Load/store access to CPU Memory

High Speed Copy Engines for bulk data movement

Courtesy: https://www.nextplatform.com/2016/05/04/nvlink-takes-gpu-acceleration-next-level/

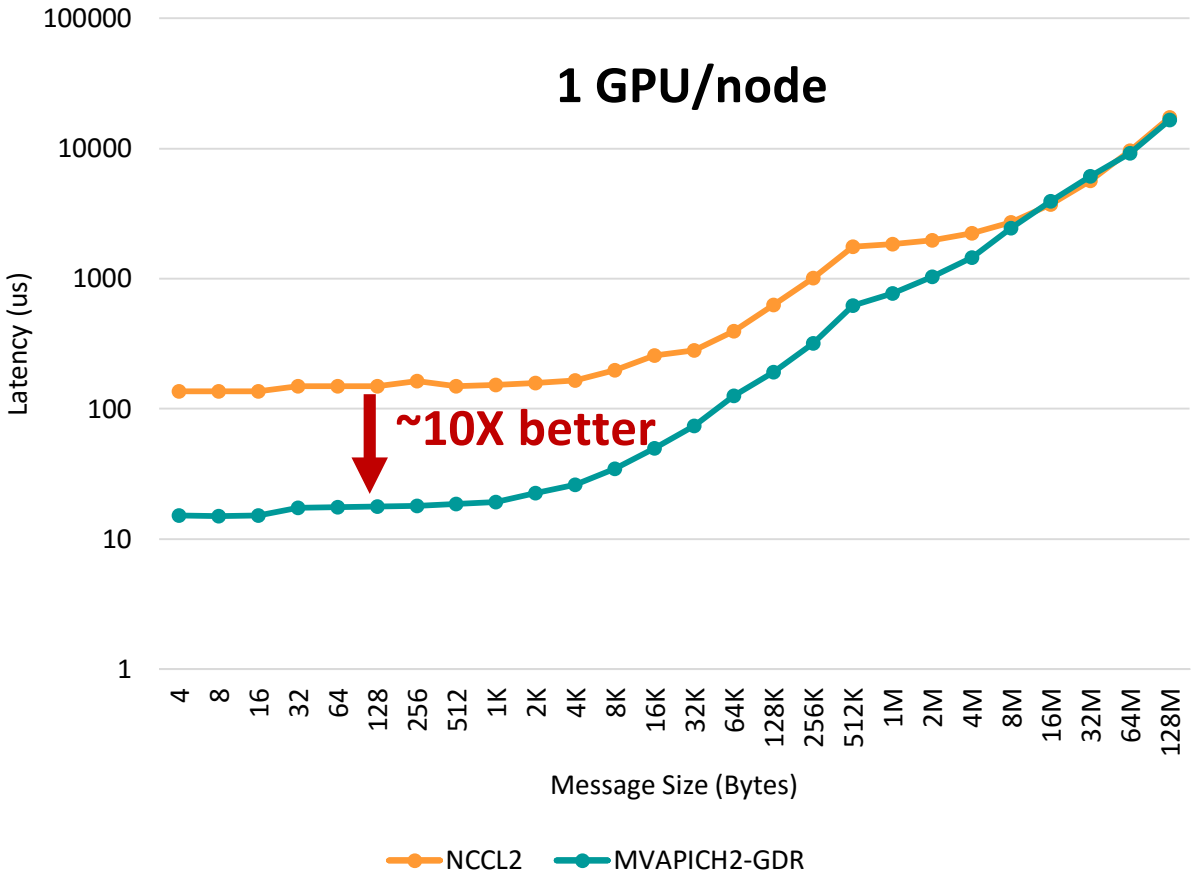
NCCL2: Multi-node GPU Collectives

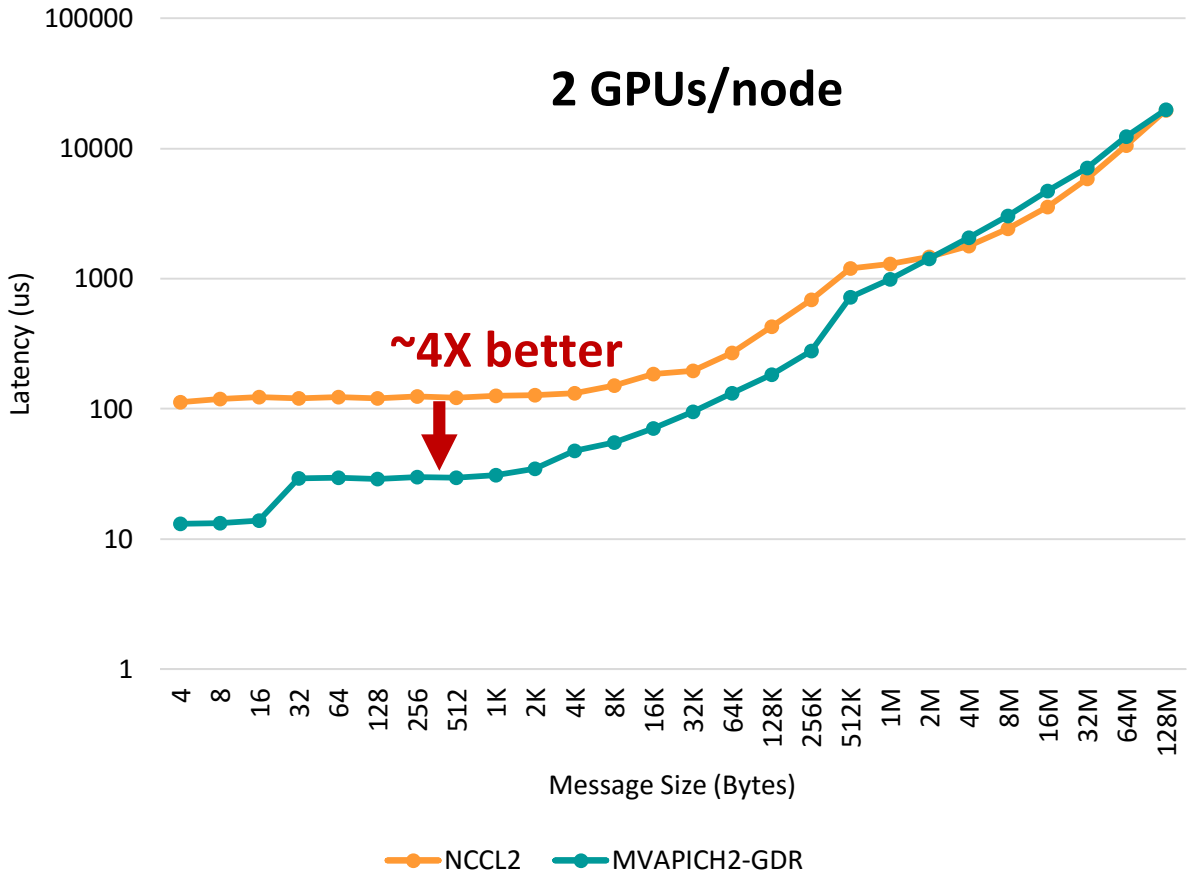


Courtesy: http://on-demand.gputechconf.com/gtc/2017/presentation/s7155-jeaugey-nccl.pdf

MVAPICH2-GDR vs. NCCL2 – Broadcast Operation

- Optimized designs in MVAPICH2-GDR 2.3b* offer better/comparable performance for most cases
- MPI_Bcast (MVAPICH2-GDR) vs. ncclBcast (NCCL2) on 16 K-80 GPUs



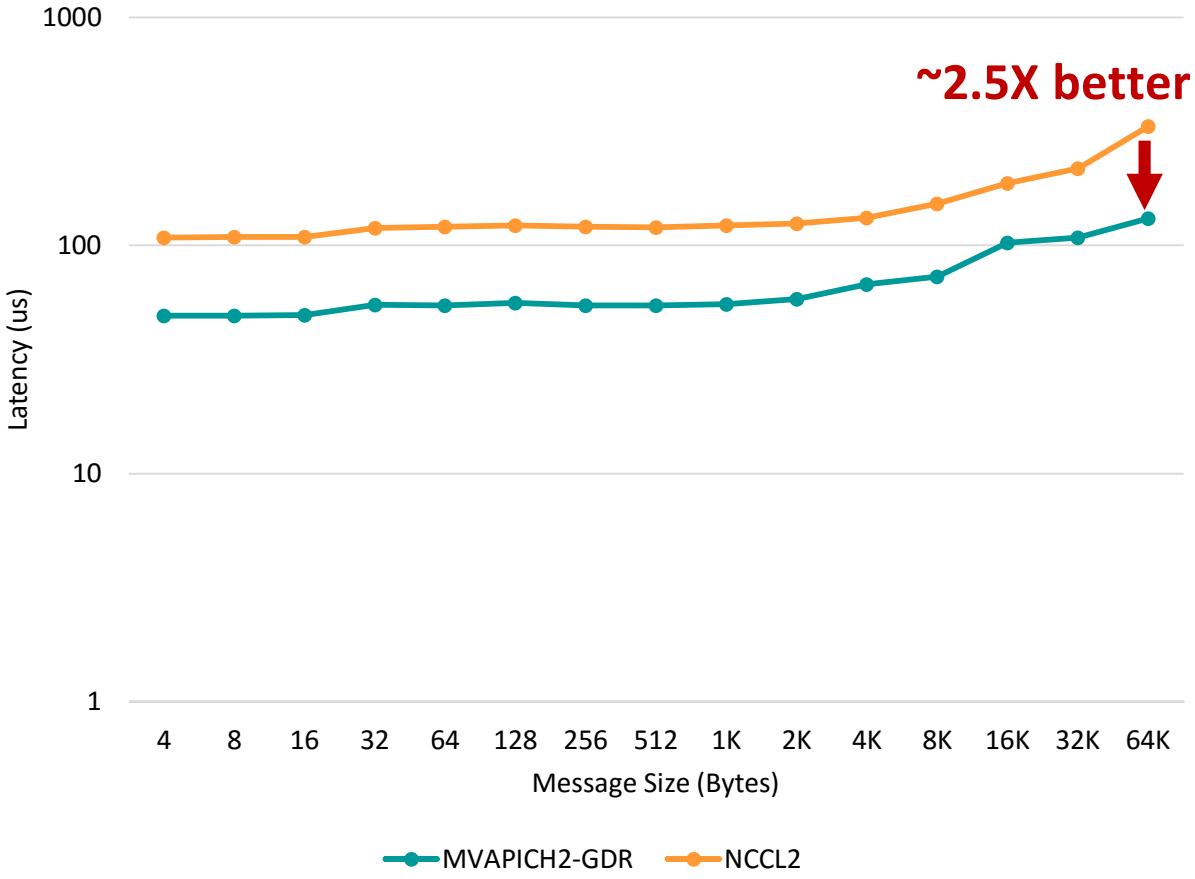


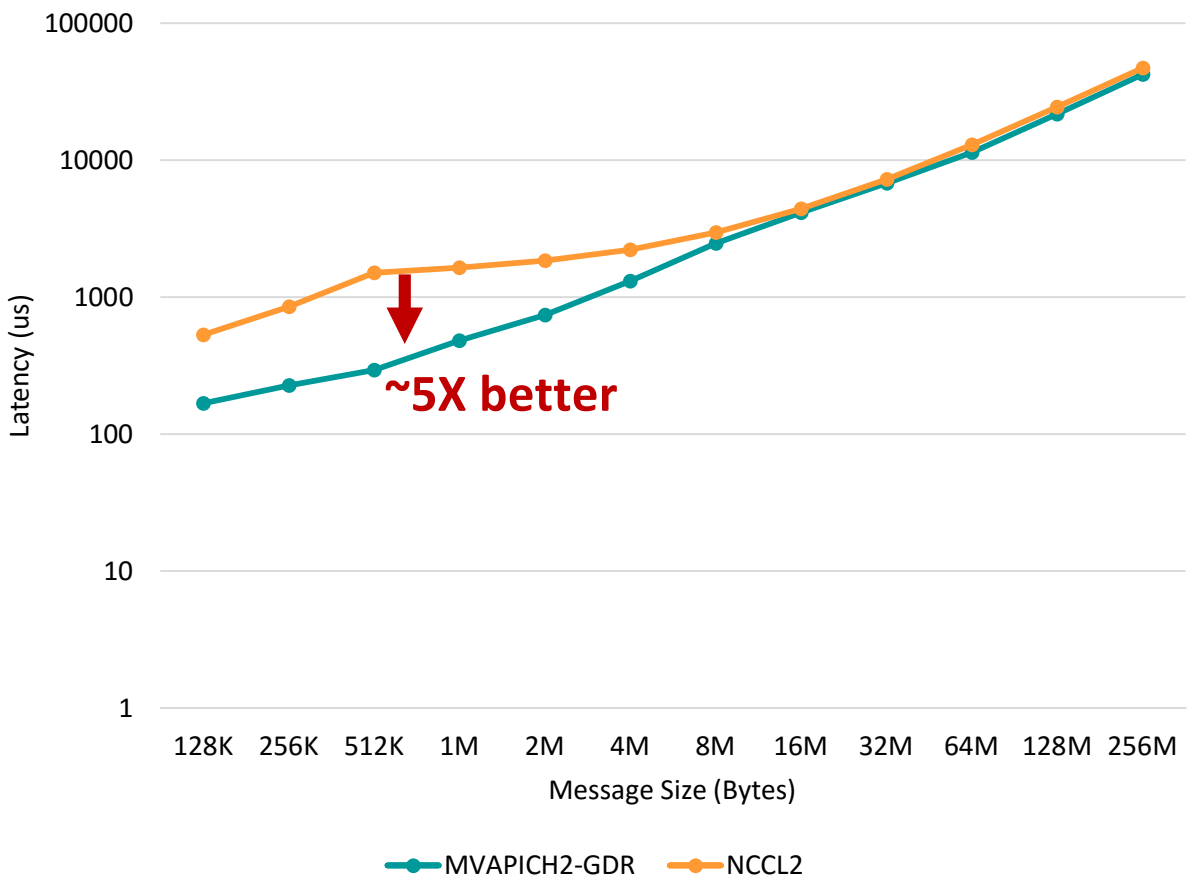
***Will be available with upcoming MVAPICH2-GDR 2.3b**

Platform: Intel Xeon (Broadwell) nodes equipped with a dual-socket CPU, 2 K-80 GPUs, and EDR InfiniBand Inter-connect

MVAPICH2-GDR vs. NCCL2 – Reduce Operation

- Optimized designs in MVAPICH2-GDR 2.3b* offer better/comparable performance for most cases
- MPI_Reduce (MVAPICH2-GDR) vs. ncclReduce (NCCL2) on 16 GPUs



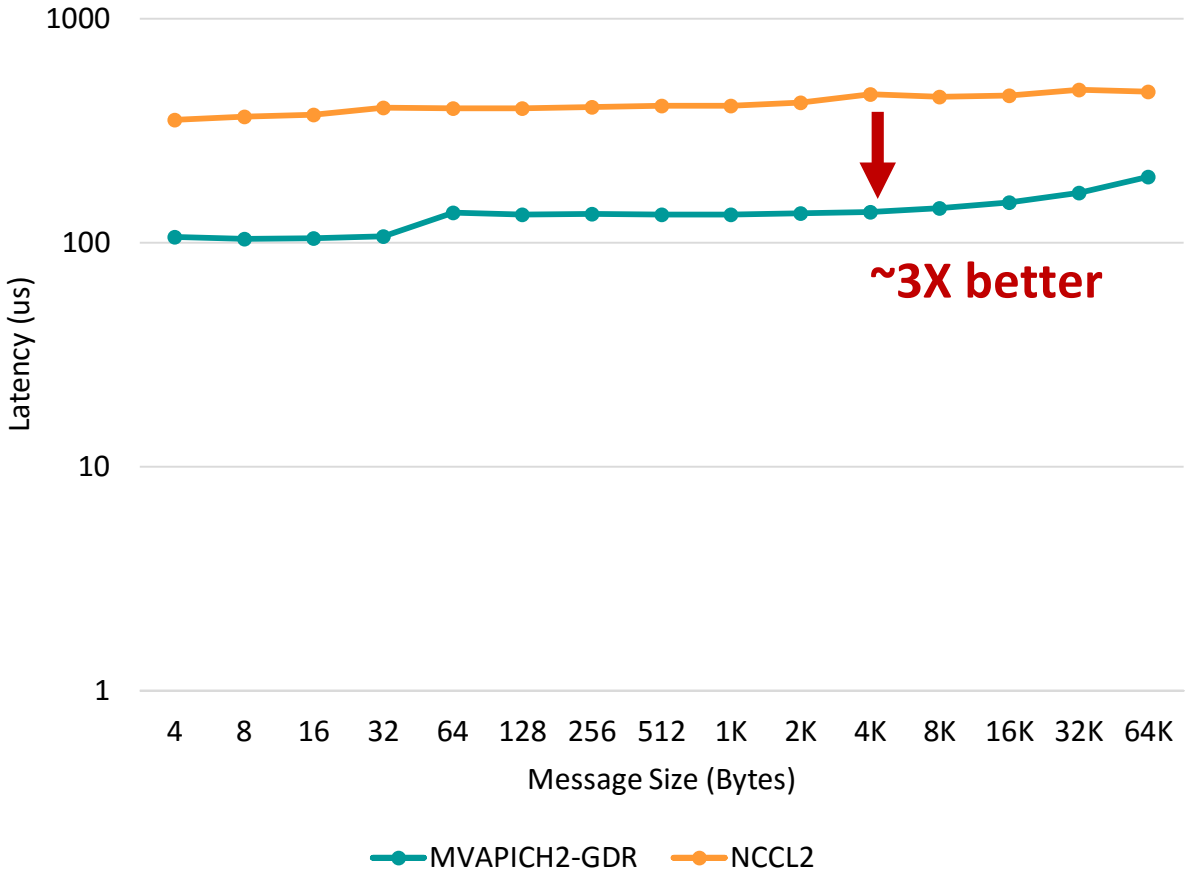


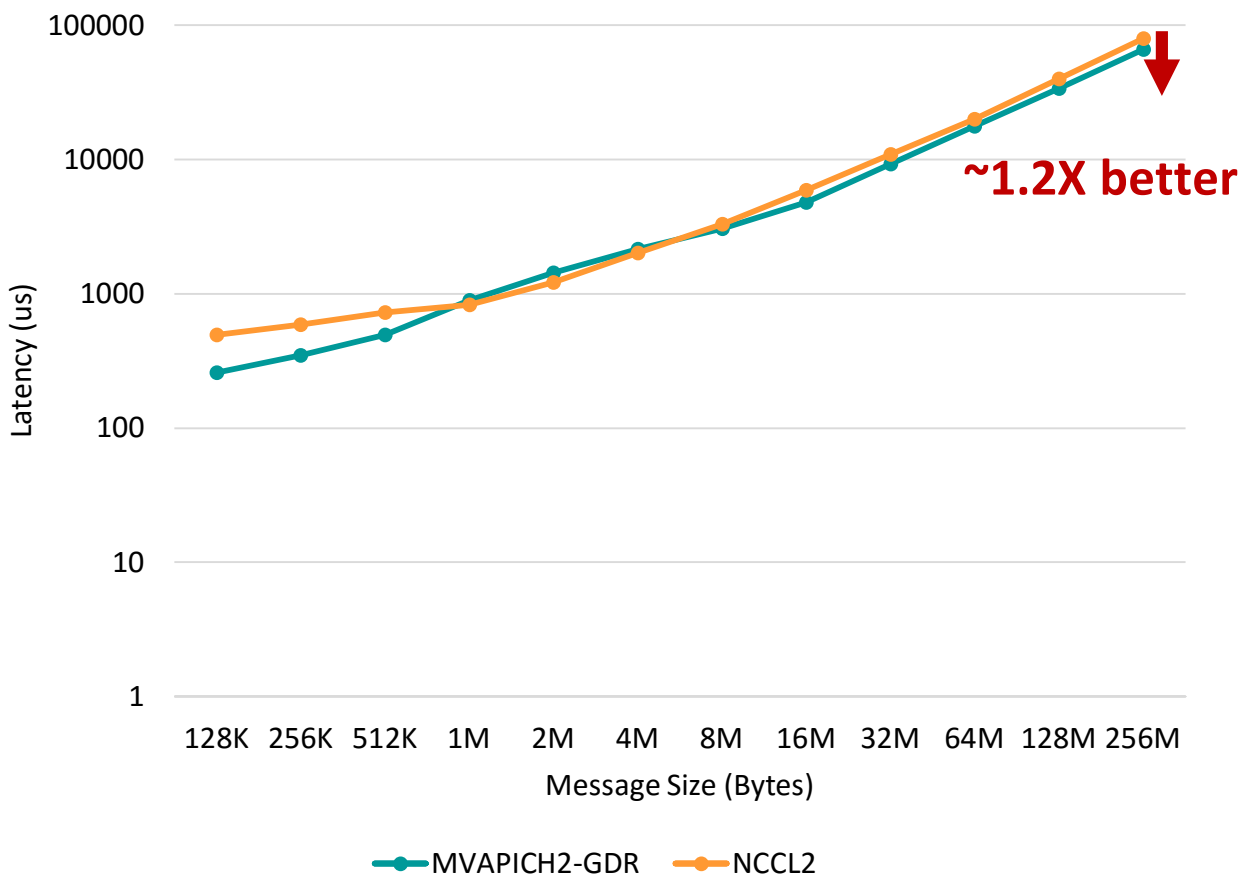
*Will be available with upcoming MVAPICH2-GDR 2.3b

Platform: Intel Xeon (Broadwell) nodes equipped with a dual-socket CPU, 1 K-80 GPUs, and EDR InfiniBand Inter-connect

MVAPICH2-GDR vs. NCCL2 – Allreduce Operation

- Optimized designs in MVAPICH2-GDR 2.3b* offer better/comparable performance for most cases
- MPI_Allreduce (MVAPICH2-GDR) vs. ncclAllreduce (NCCL2) on 16 GPUs



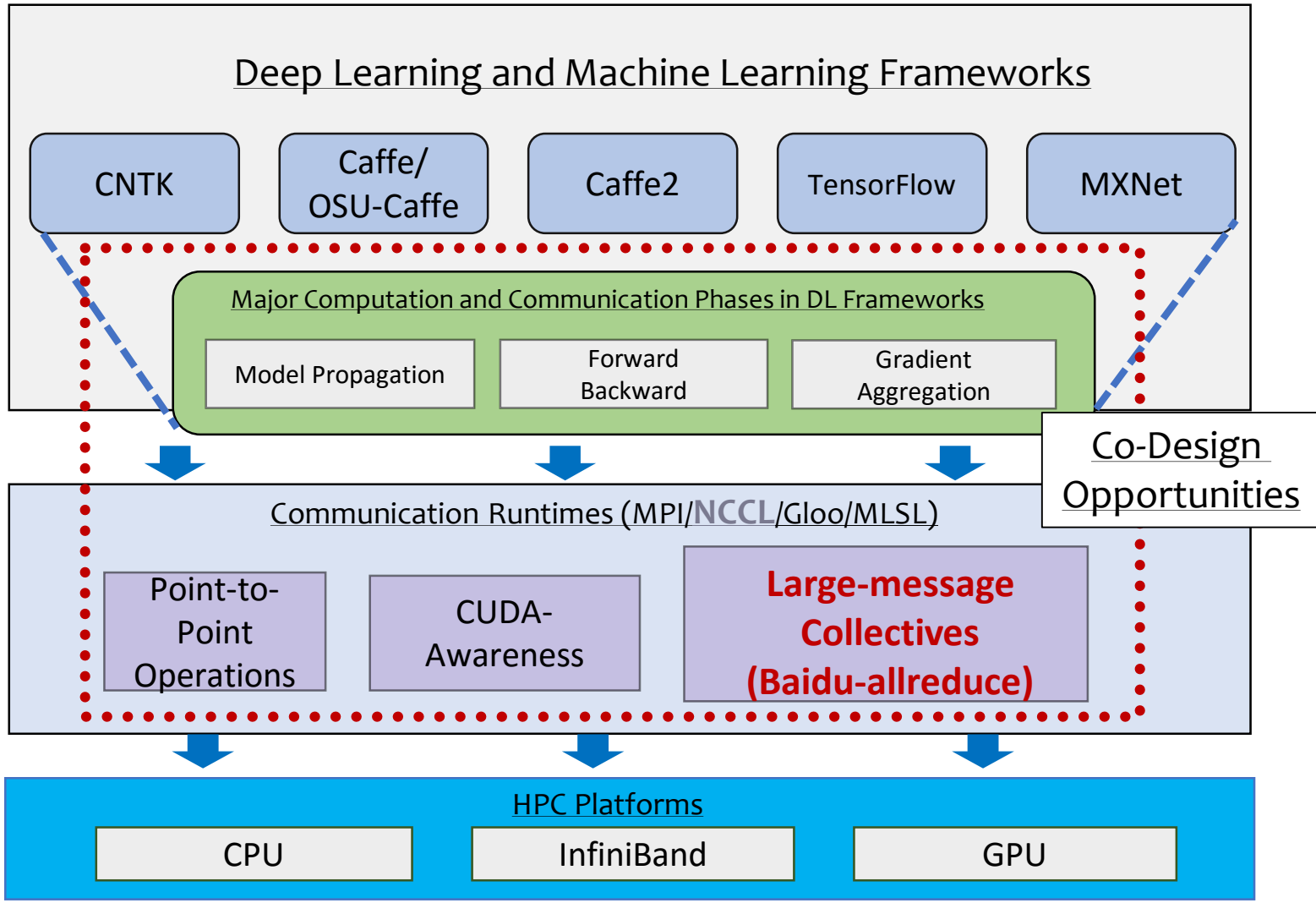


***Will be available with upcoming MVAPICH2-GDR 2.3b**

Platform: Intel Xeon (Broadwell) nodes equipped with a dual-socket CPU, 1 K-80 GPUs, and EDR InfiniBand Inter-connect

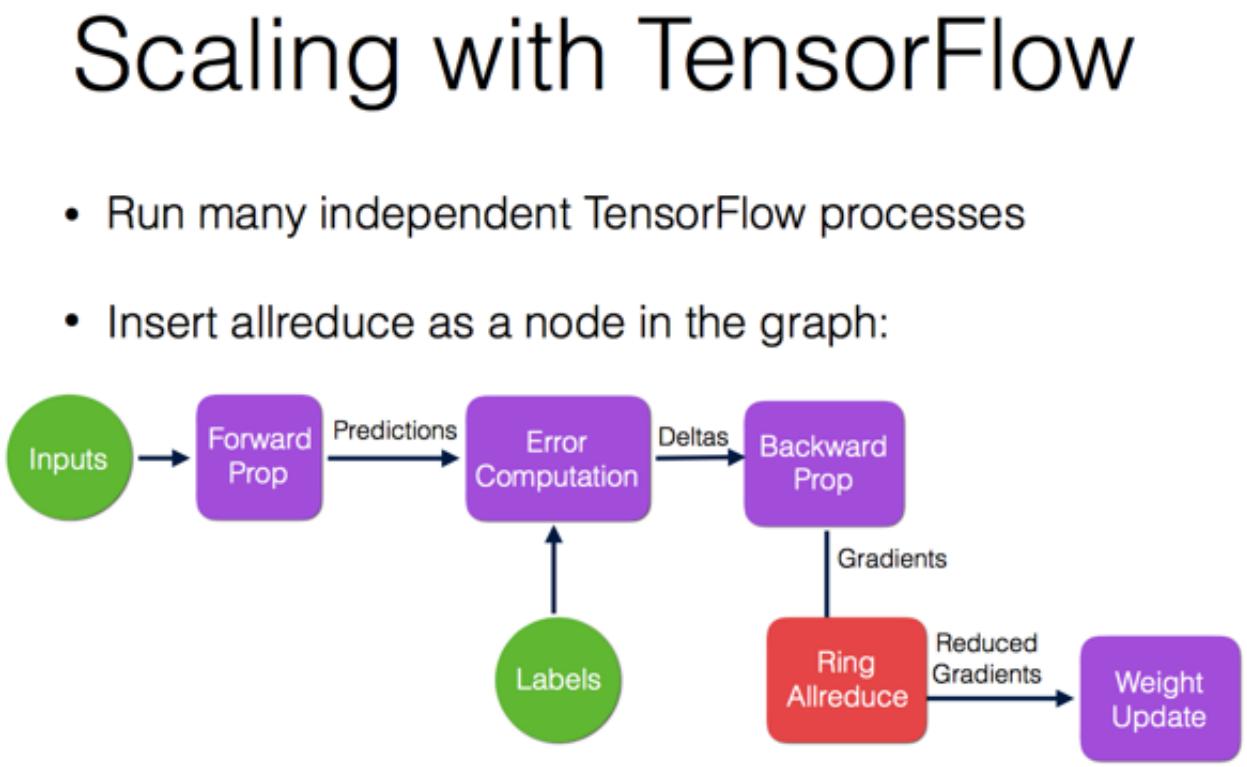
Solutions and Case Studies: Exploiting HPC for DL

- NVIDIA NCCL
- **Baidu-allreduce**
- Facebook Gloo
- Co-design MPI runtimes and DL Frameworks
  - MPI+NCCL for CUDA-Aware CNTK
  - OSU-Caffe
- TensorFlow (Horovod)
- Scaling DNN Training on Multi-/Many-core CPUs
- PowerAI DDL



Baidu-allreduce in TensorFlow

- Baidu uses large message Allreduce collectives
- Evaluation with OpenMPI Allreduce showed performance degradation
- Proposed Solution:
  - Implement a Ring-Allreduce algorithm on top of point to point MPI primitives (Send/Recv) at the application level
- 2.5-3X better than OpenMPI Allreduce
- Used in the Deep Speech 2 paper*

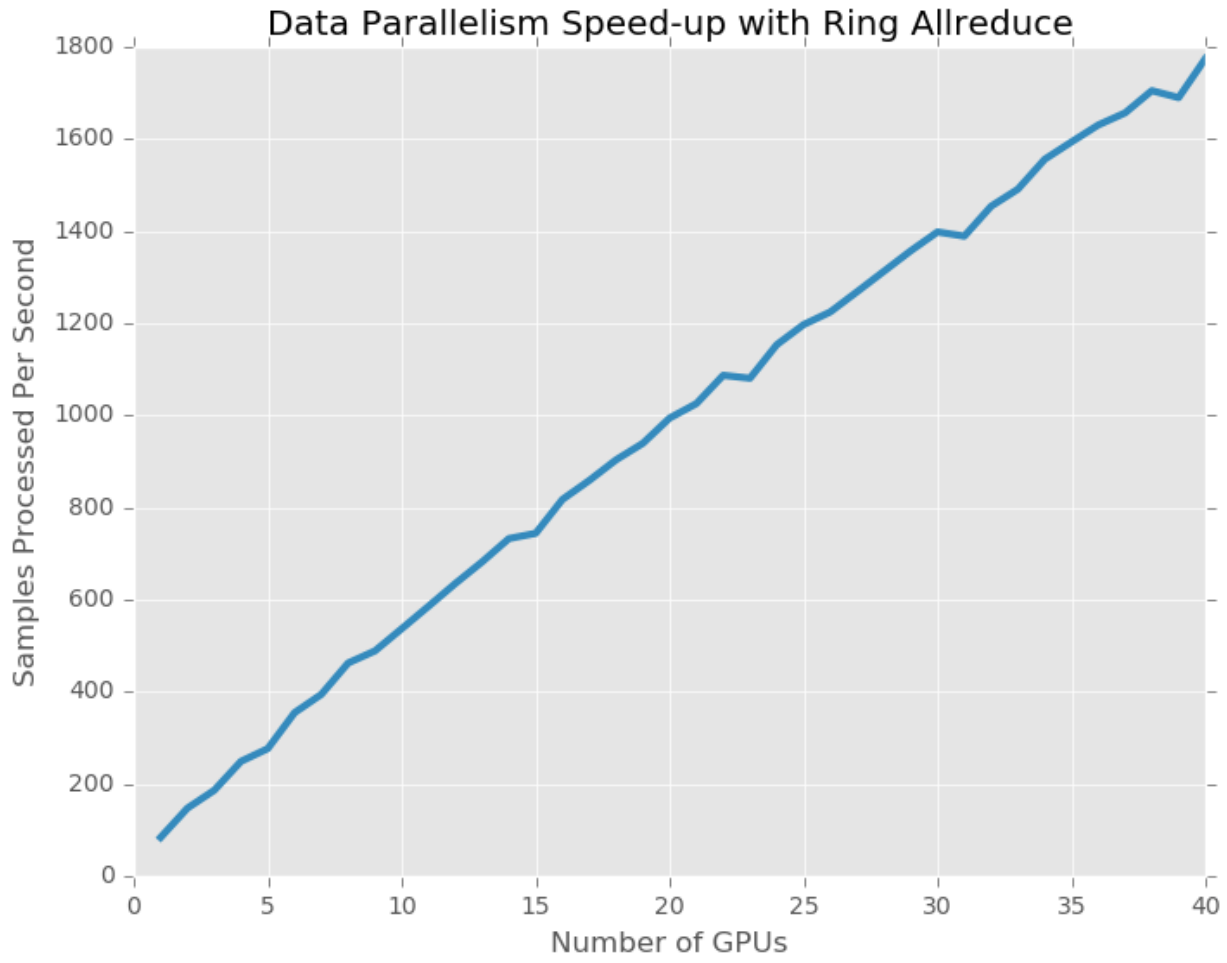


Courtesy: http://on-demand.gputechconf.com/gtc/2017/presentation/s7543-andrew-gibiansky-effectively-scakukbg-deep-learning-frameworks.pdf

*Amodei, Dario et al. "Deep Speech 2: End-to-End Speech Recognition in English and Mandarin." *ICML* (2016).

Data Parallel Training with Baidu-allreduce

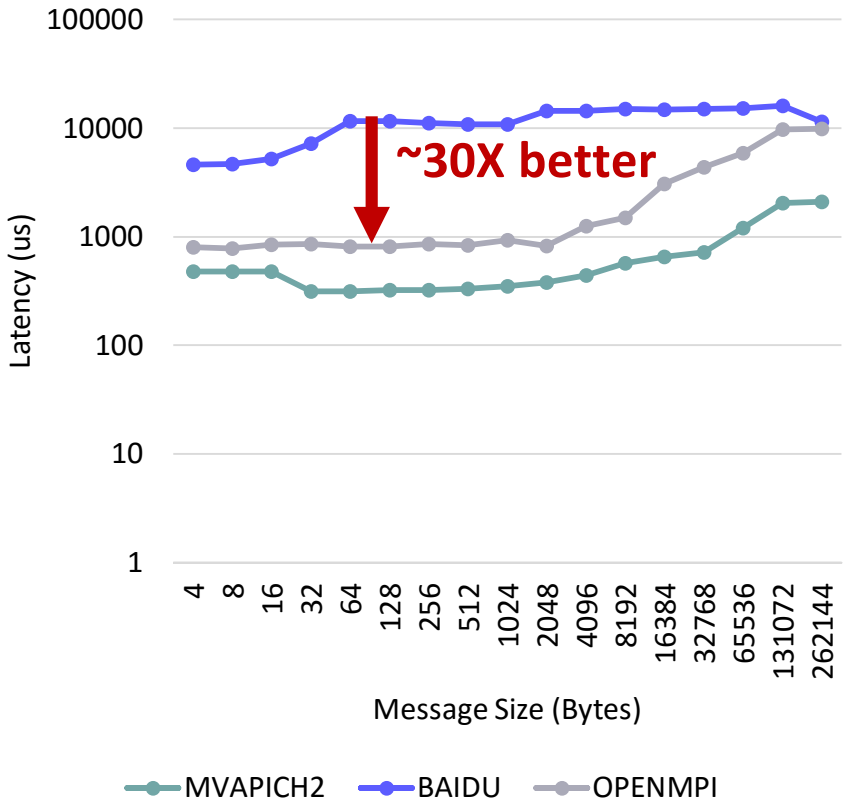
- Near-linear speedup for DNN training throughput (samples/second)
- The Allreduce design has been integrated in a TensorFlow contribution
- Details of the design are available from the Github site: https://github.com/baidu-research/tensorflow-allreduce

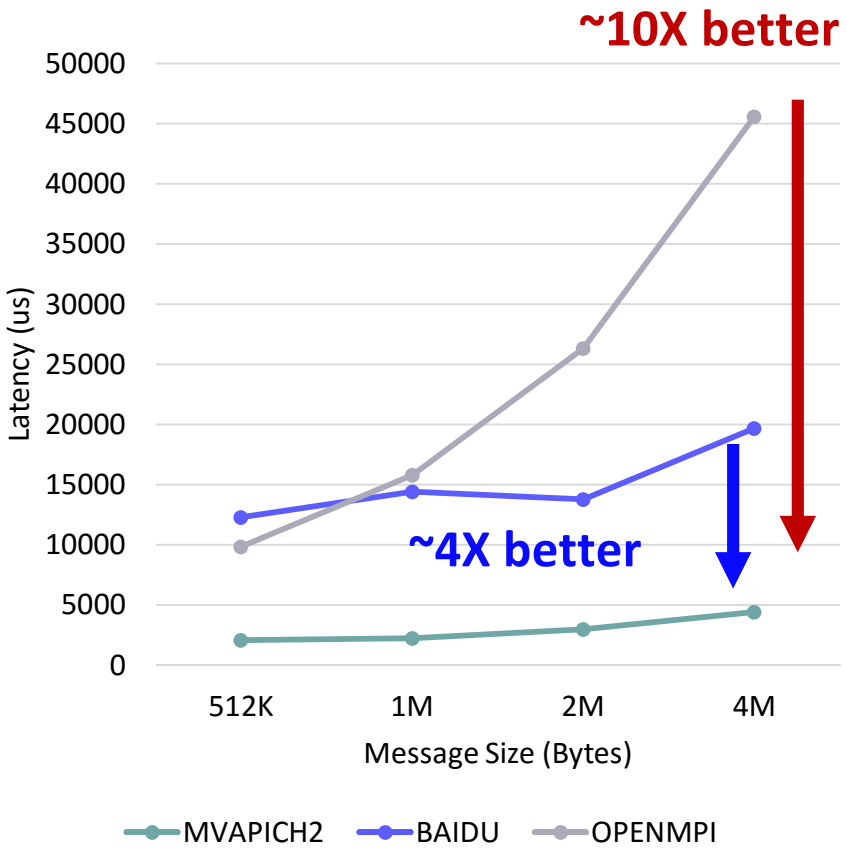


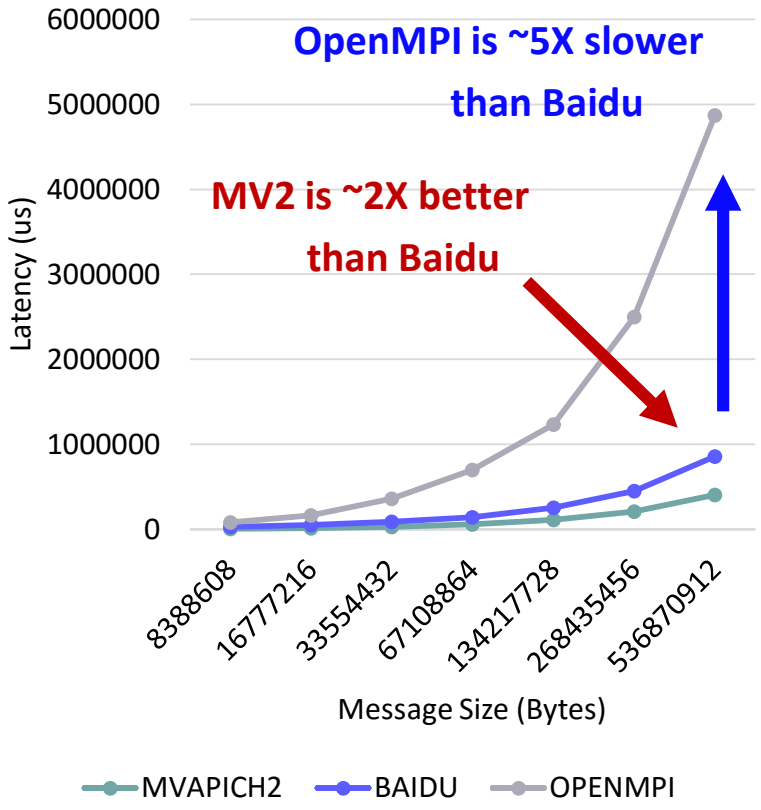
Courtesy: http://research.baidu.com/bringing-hpc-techniques-deep-learning/

MVAPICH2: Allreduce Comparison with Baidu and OpenMPI

- 16 GPUs (4 nodes) MVAPICH2-GDR vs. Baidu-Allreduce and OpenMPI 3.0



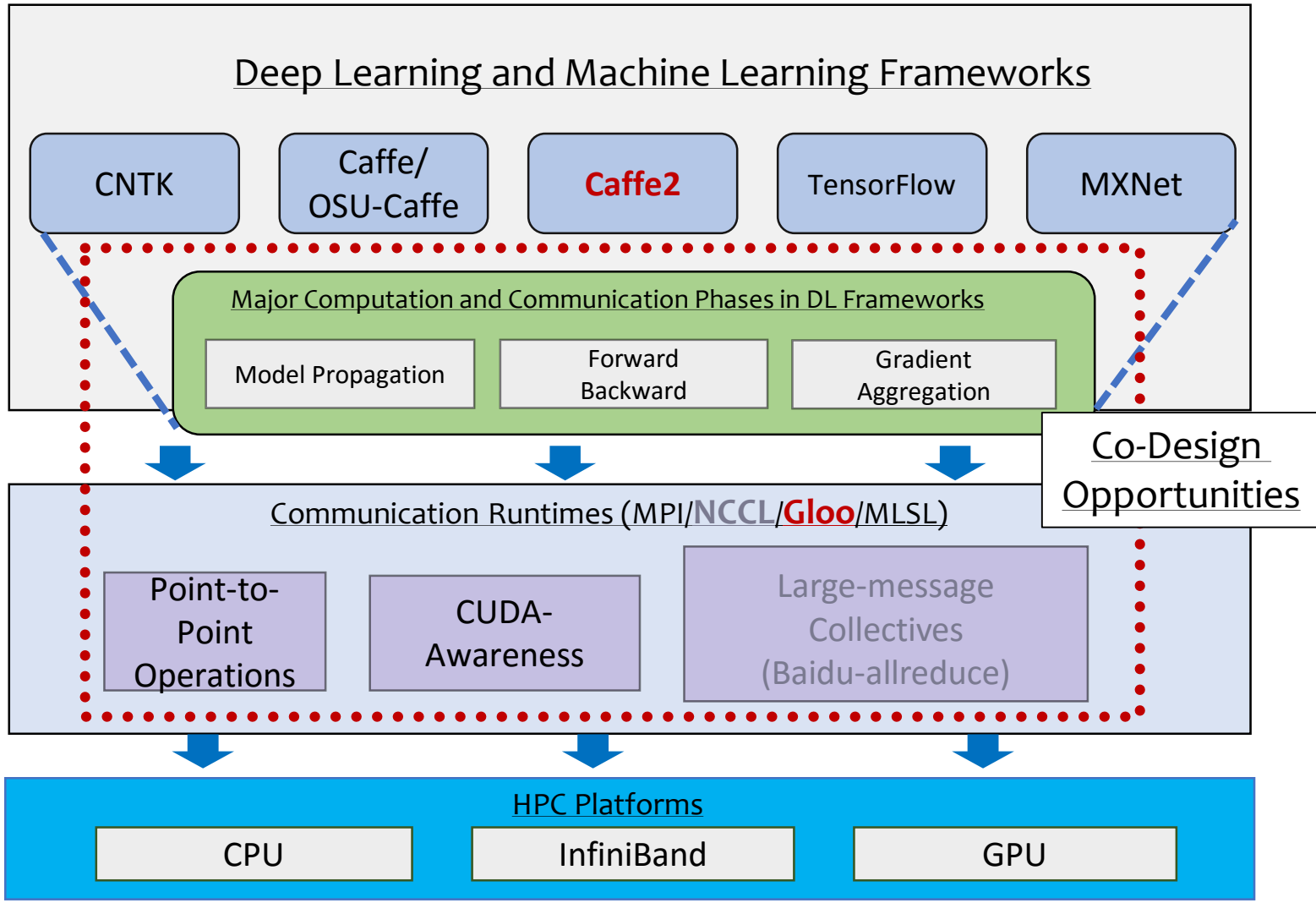




**Available with MVAPICH2-GDR 2.3a*

Solutions and Case Studies: Exploiting HPC for DL

- NVIDIA NCCL
- Baidu-allreduce
- **Facebook Gloo**
- Co-design MPI runtimes and DL Frameworks
  - MPI+NCCL for CUDA-Aware CNTK
  - OSU-Caffe
- TensorFlow (Horovod)
- Scaling DNN Training on Multi-/Many-core CPUs
- PowerAI DDL



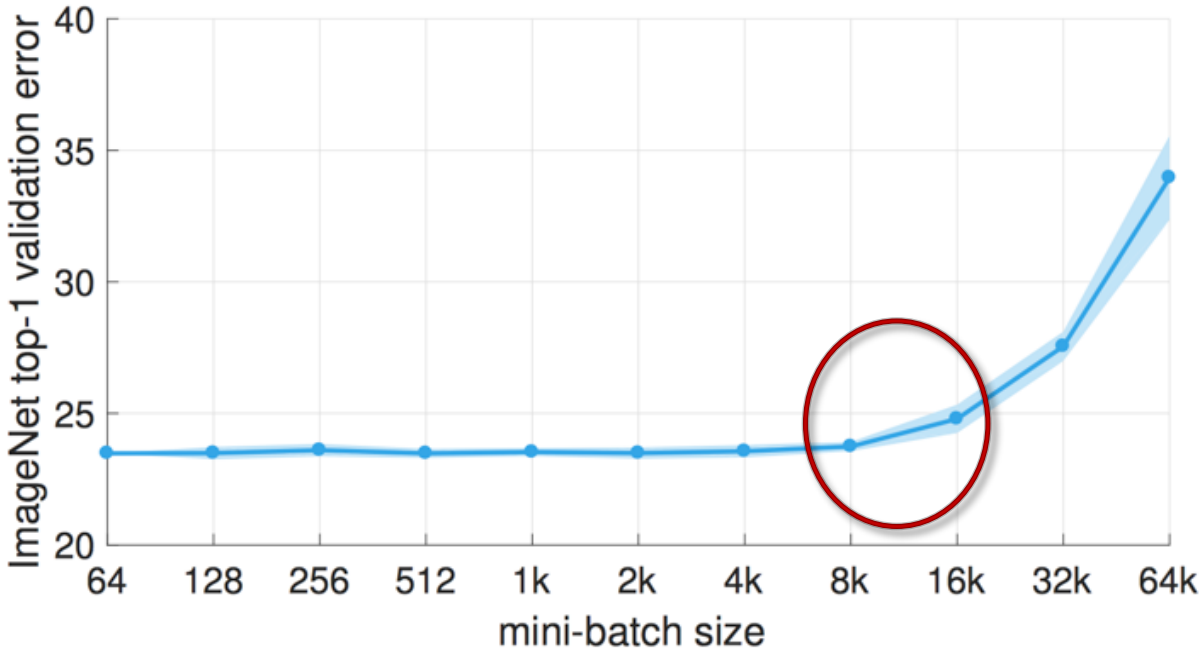
Facebook Caffe2

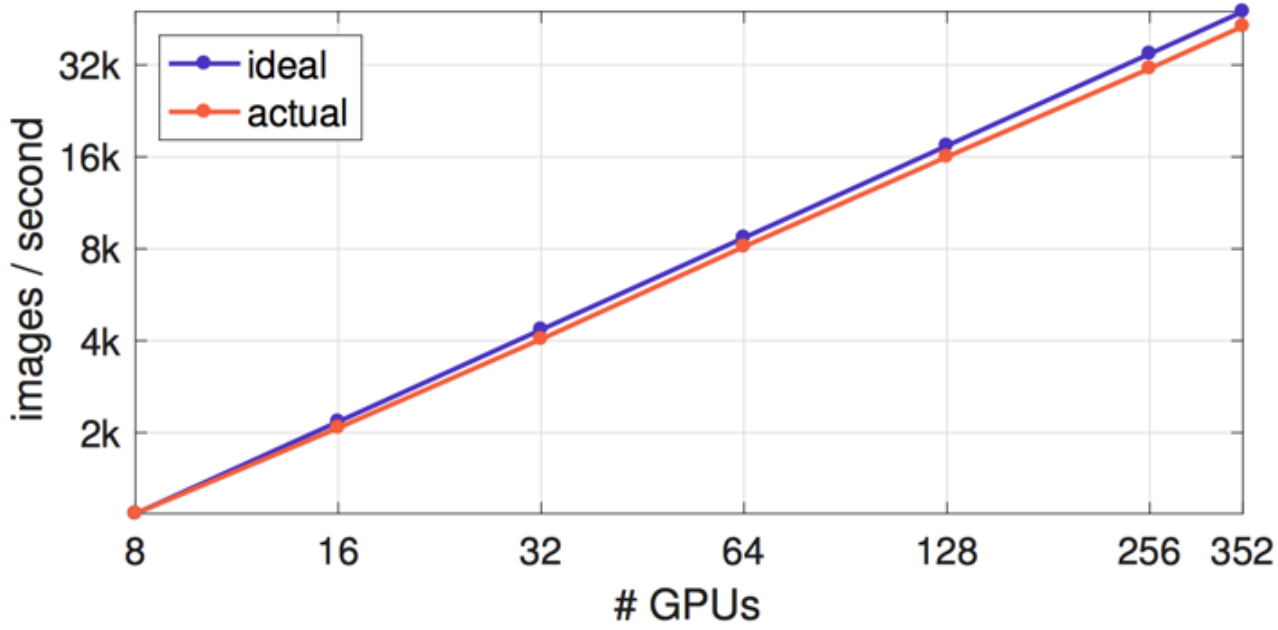
- Caffe2 (by Facebook) allows the use of multiple communication back-ends
  - Gloo – Multi-node design from the beginning
  - NCCL – Multi-node support added recently in v2
- Gloo – Performance evaluation studies not available yet
- Design principles are similar to MPI and NCCL
- In essence, Gloo is an application level implementation of collective algorithms for Reduce, Allreduce, etc.
- Details and code available from: https://github.com/facebookincubator/gloo

Gloo

- Gloo comes with a number of collective algorithms useful for machine learning applications
  - Barrier
  - Broadcast
  - Allreduce
- Transport of data between participating machines is abstracted so that IP can be used at all times, or InfiniBand (or RoCE) when available
- If InfiniBand transport is used, GPUDirect can be used to accelerate cross machine GPU-to-GPU memory transfers
- Implementation that works with system memory buffers, and one that works with NVIDIA GPU memory buffers. (CUDA-Aware)

Facebook: Training ImageNet in 1 Hour



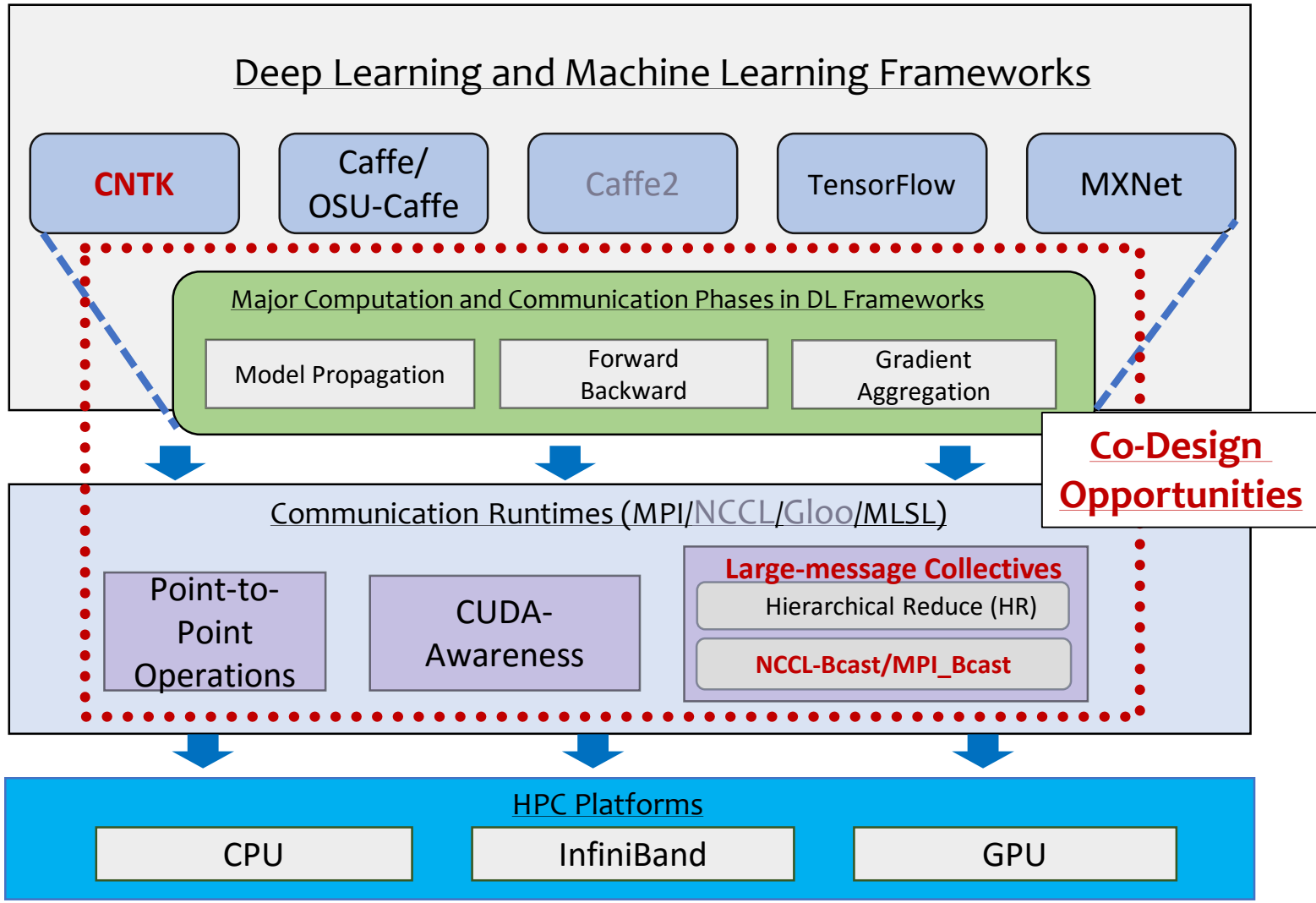


- Near-linear Scaling for ~256 Pascal GPUs (Facebook Big Basin Servers with 8 GPUs/node)
- Explored large batch-size training with ResNet-50
  - ***8K batch-size seems to be the sweet-spot.***

Courtesy: https://research.fb.com/publications/imagenet1kin1h/

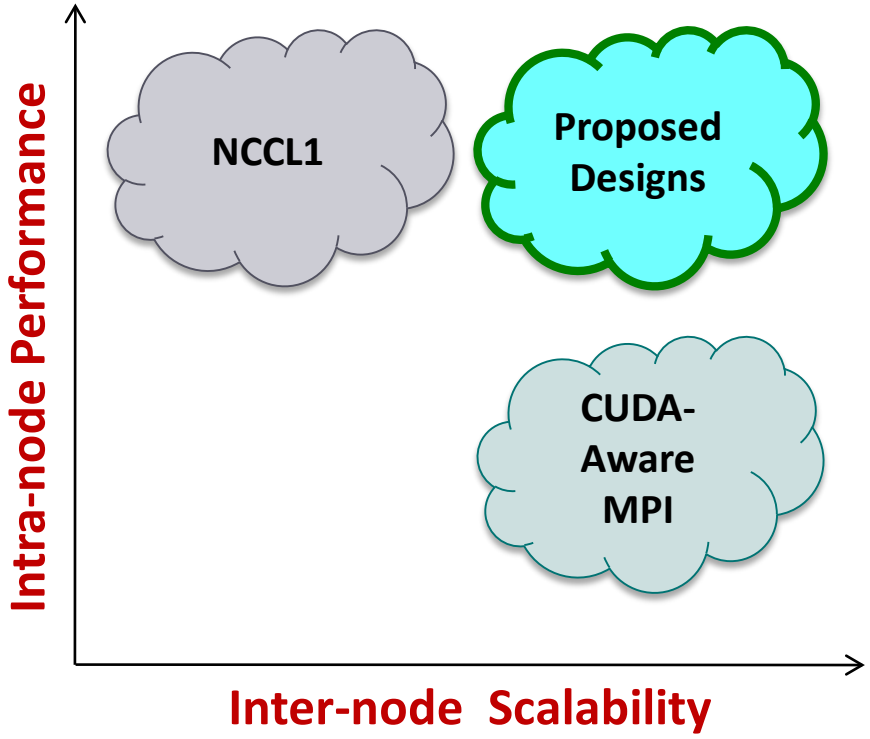
Solutions and Case Studies: Exploiting HPC for DL

- NVIDIA NCCL
- Baidu-allreduce
- Facebook Gloo
- **Co-design MPI runtimes and DL Frameworks**
  - **MPI+NCCL for CUDA-Aware CNTK**
  - OSU-Caffe
- TensorFlow (Horovod)
- Scaling DNN Training on Multi-/Many-core CPUs
- PowerAI DDL



MPI+NCCL: Can we exploit NCCL to accelerate MPI?

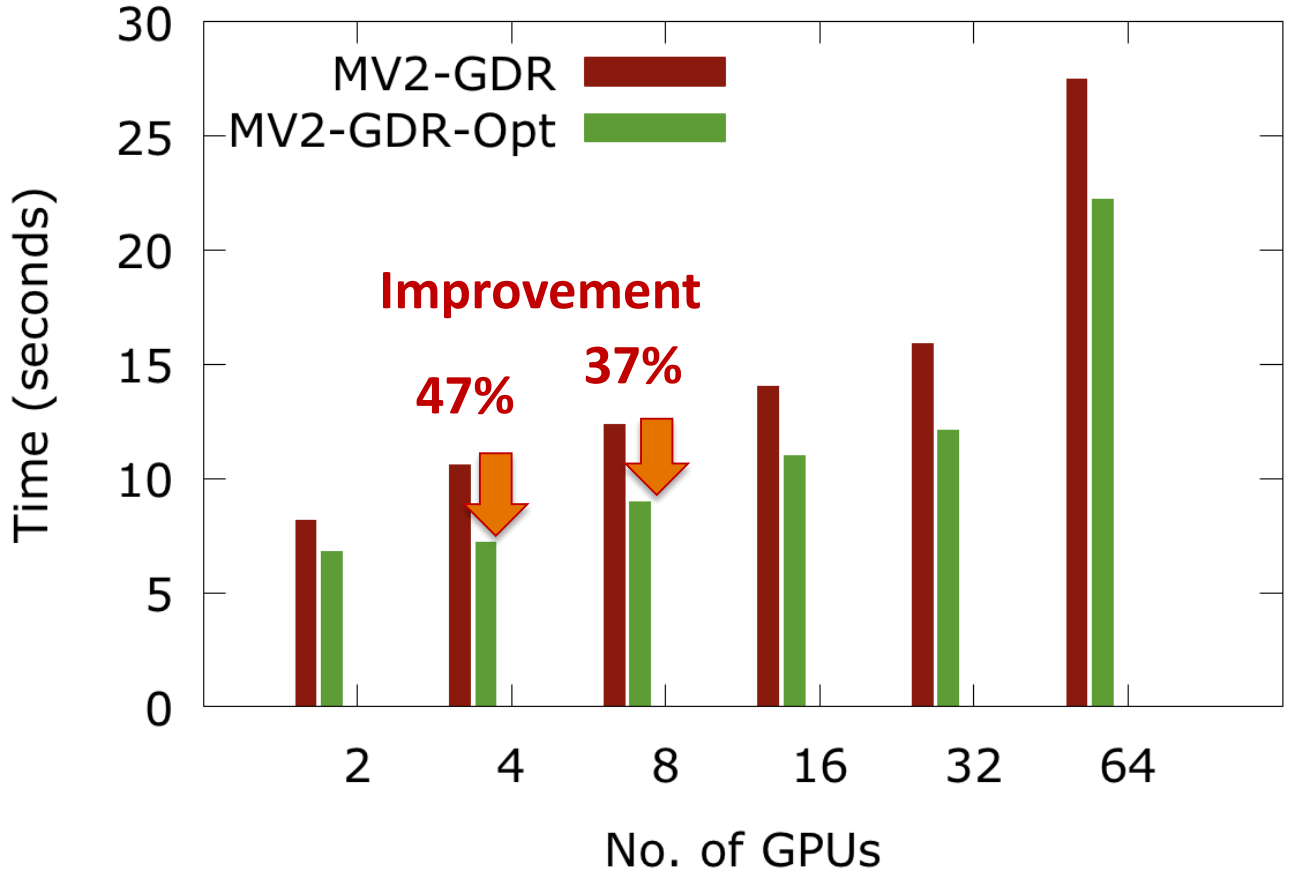
- CUDA-Aware MPI provides excellent performance for small and medium message sizes
- NCCL has overhead for small messages but provides excellent performance for large messages
- Can we have designs that provide good performance for intra-node communication and inter-node scalability?
  - Exploit NCCL1 for intra-node inter-GPU communication
  - Design and utilize existing Inter-node communication in MVAPICH2-GDR



A. A. Awan, K. Hamidouche, A. Venkatesh, and D. K. Panda, Efficient Large Message Broadcast using NCCL and CUDA-Aware MPI for Deep Learning. In *Proceedings of the 23rd European MPI Users' Group Meeting* (EuroMPI 2016). [Best Paper Nominee]

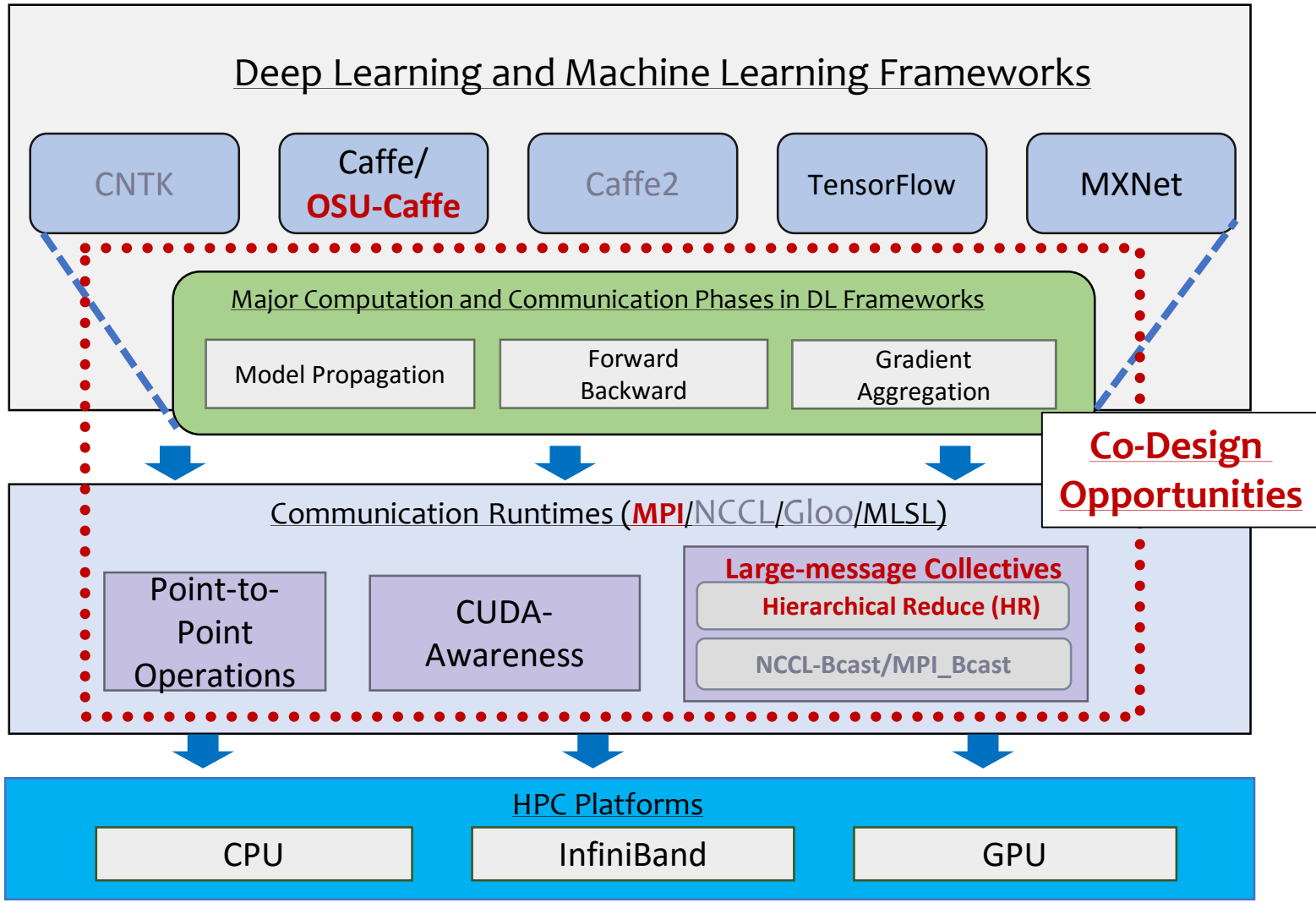
Application Performance with Microsoft CNTK (64 GPUs)

- Microsoft CNTK is a popular and efficient DL framework
- CA-CNTK is a CUDA-Aware version developed at OSU
- Proposed Broadcast provides up to **47%** improvement in Training time for the **VGG** network



Solutions and Case Studies: Exploiting HPC for DL

- NVIDIA NCCL
- Baidu-allreduce
- Facebook Gloo
- **Co-design MPI runtimes and DL Frameworks**
  - MPI+NCCL for CUDA-Aware CNTK
  - **OSU-Caffe**
- TensorFlow (Horovod)
- Scaling DNN Training on Multi-/Many-core CPUs
- PowerAI DDL

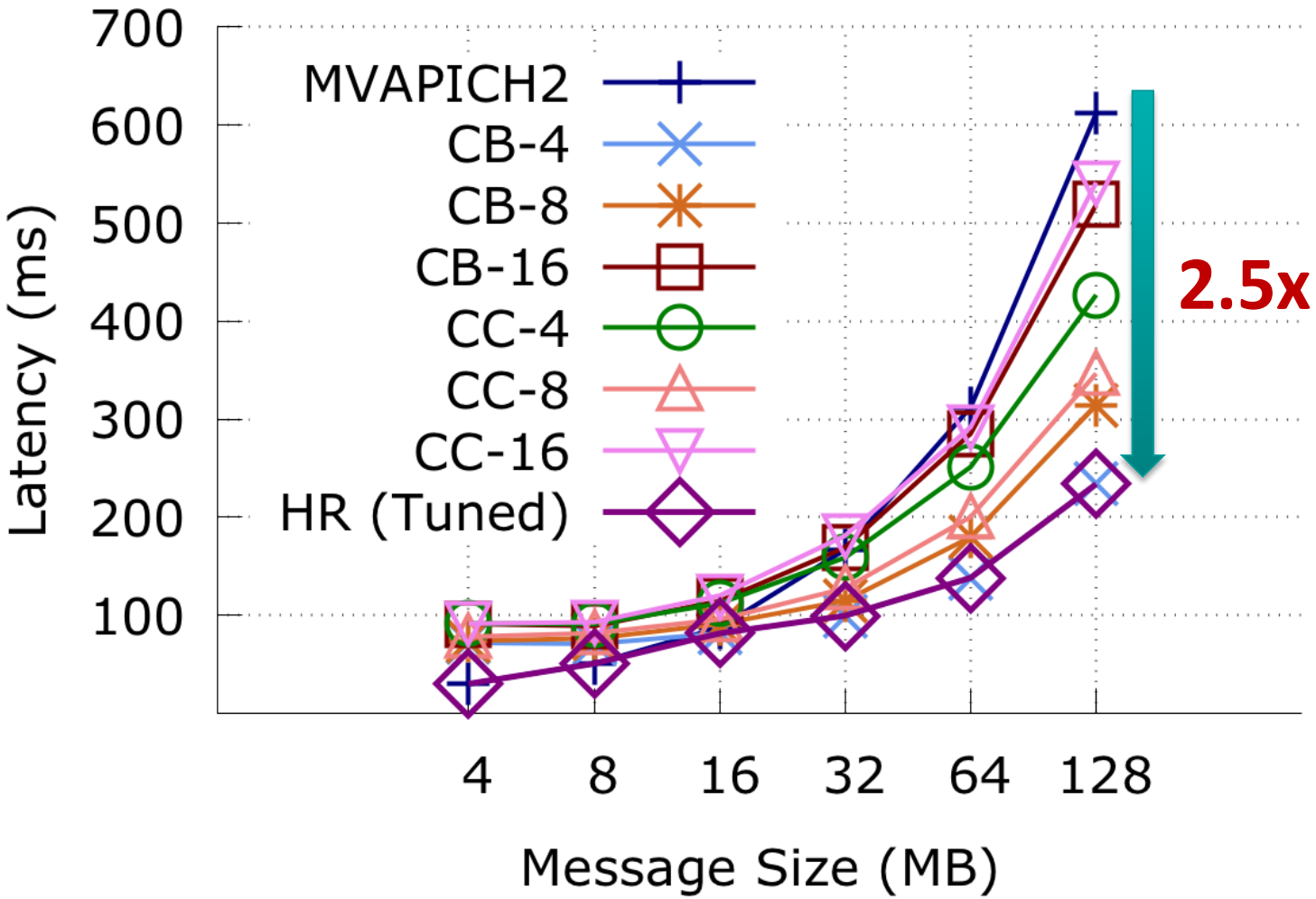


OSU-Caffe: Proposed Co-Design Overview

- To address the limitations of Caffe and existing MPI runtimes, we propose the **OSU-Caffe (S-Caffe)** framework
- At the application (DL framework) level
  - Develop a fine-grain workflow – i.e. layer-wise communication instead of communicating the entire model
- At the runtime (MPI) level
  - Develop support to perform reduction of very-large GPU buffers
  - Perform reduction using GPU kernels

OSU-Caffe is available from the HiDL project page (http://hidl.cse.ohio-state.edu)

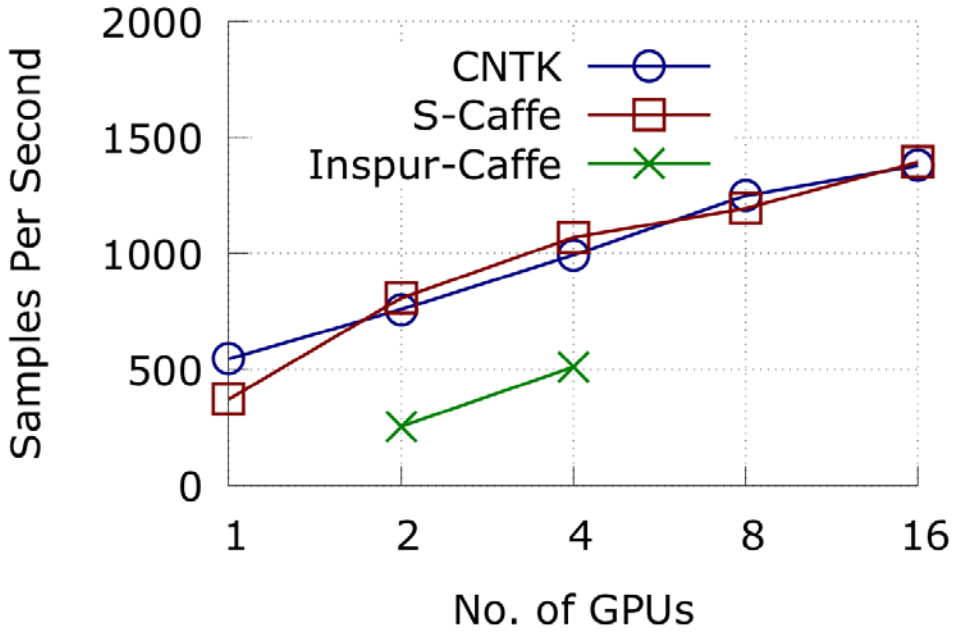
Hierarchical Reduce (HR) - 160 GPUs



- Various designs to achieve best performance across platforms
- Proposed HR-tuned provides the best performance (up to 2.5X)

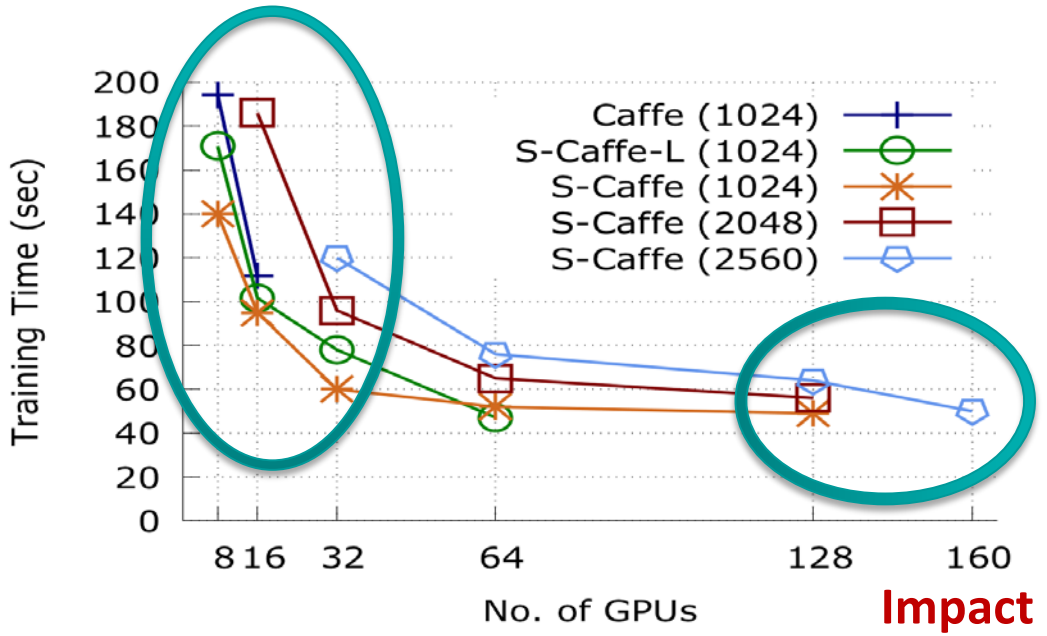
S-Caffe, Inspur-Caffe, and CNTK

- AlexNet: Notoriously hard to scale-out on multiple nodes due to comm. overhead!
- Large number of parameters ~ 64 Million (comm. buffer size = 256 MB)



- GoogLeNet is a popular DNN
- 13 million parameters (comm. buffer size = ~50 MB)

Up to 14% improvement (Scale-up)

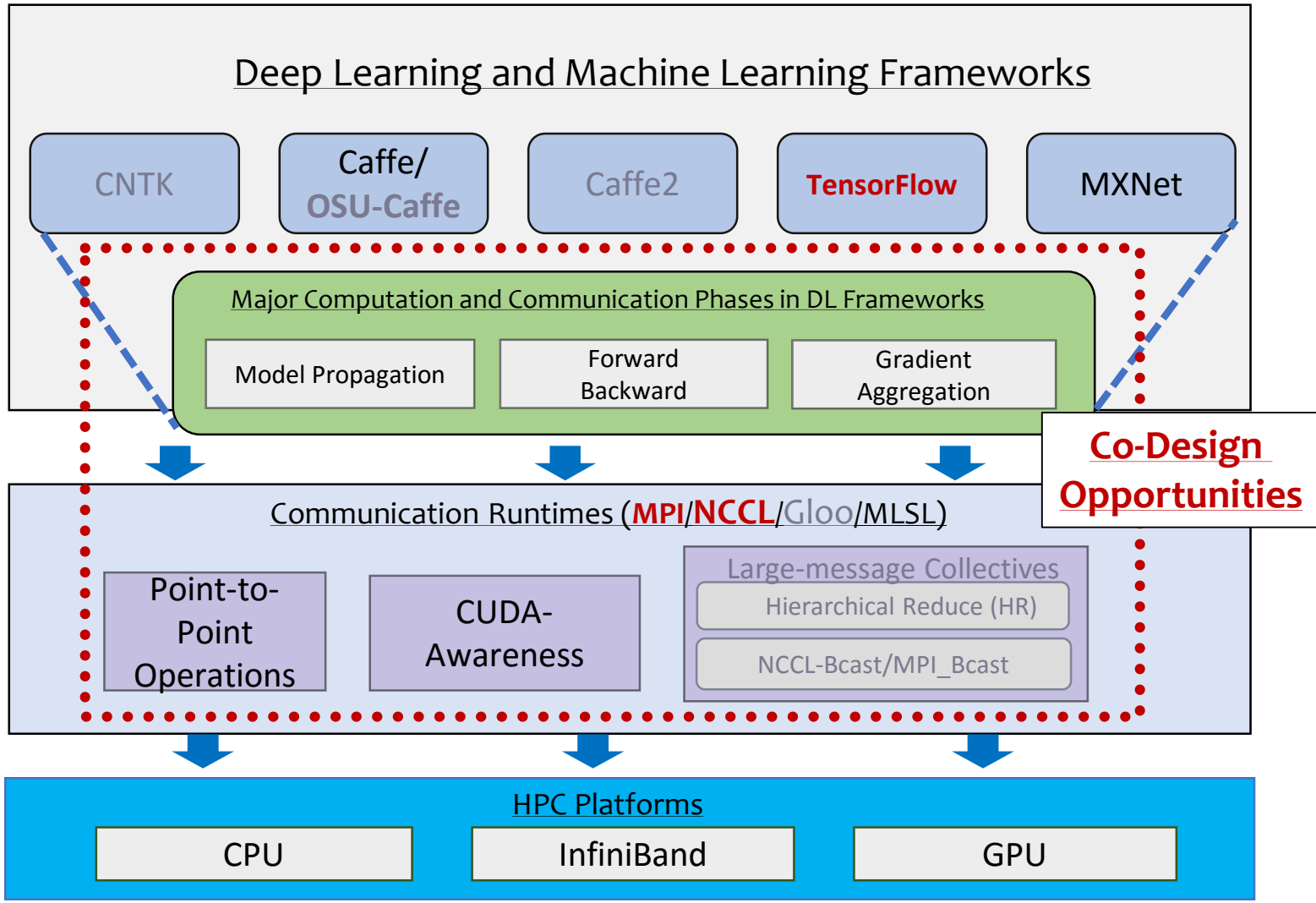


Impact of HR

S-Caffe delivers better or comparable performance with other multi-node capable DL frameworks

Solutions and Case Studies: Exploiting HPC for DL

- NVIDIA NCCL
- Baidu-allreduce
- Facebook Gloo
- Co-design MPI runtimes and DL Frameworks
  - MPI+NCCL for CUDA-Aware CNTK
  - OSU-Caffe
- **TensorFlow (Horovod)**
- Scaling DNN Training on Multi-/Many-core CPUs
- PowerAI DDL

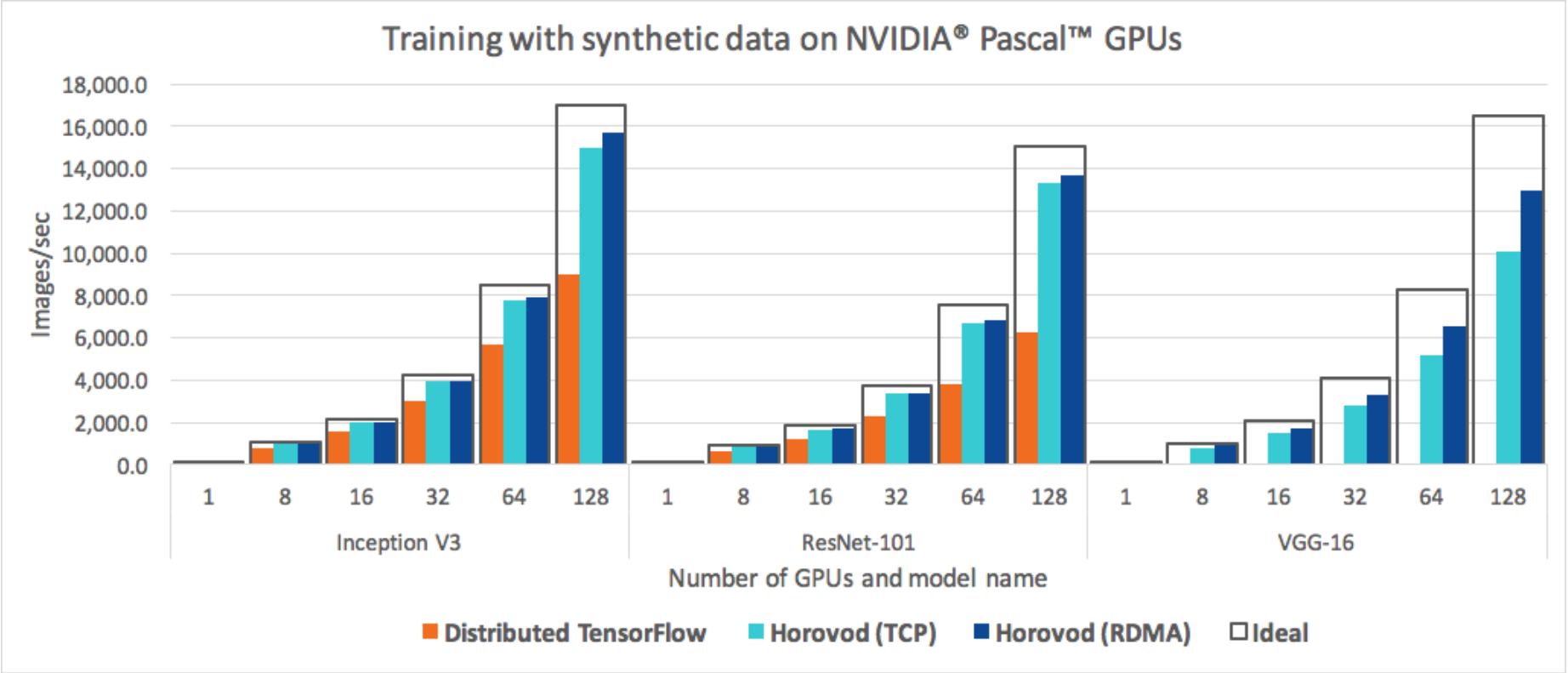


TensorFlow (Horovod)

- Baidu had a TensorFlow design that utilizes MPI library for gradient aggregation via a custom Allreduce design
  - Part of TensorFlow/contrib/
- Uber has built **Horovod** inspired by Baidu's approach but it provides a separate and easier installation process via pip
- Horovod uses **MPI_Allreduce** or **ncclAllreduce** depending on the build process a user follows
- TensorFusion optimization in Horovod to exploit efficient large message exchange
- More details available from: https://github.com/uber/horovod

Horovod Training (Synthetic Data)

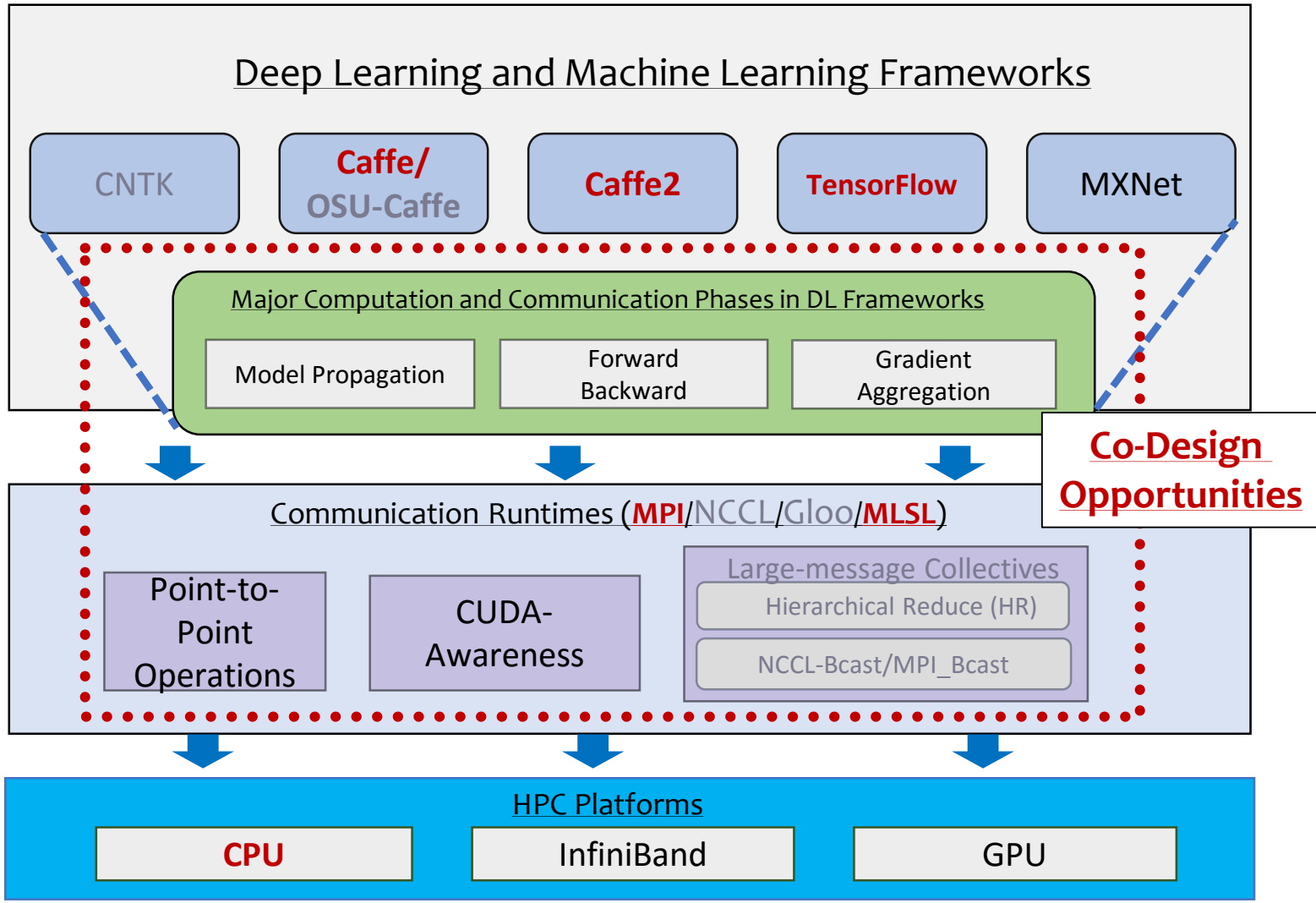
- Official distributed TensorFlow uses gRPC – which can use TCP or RDMA interface
- Horovod can also use TCP or RDMA.
  - RDMA has much better performance, as expected



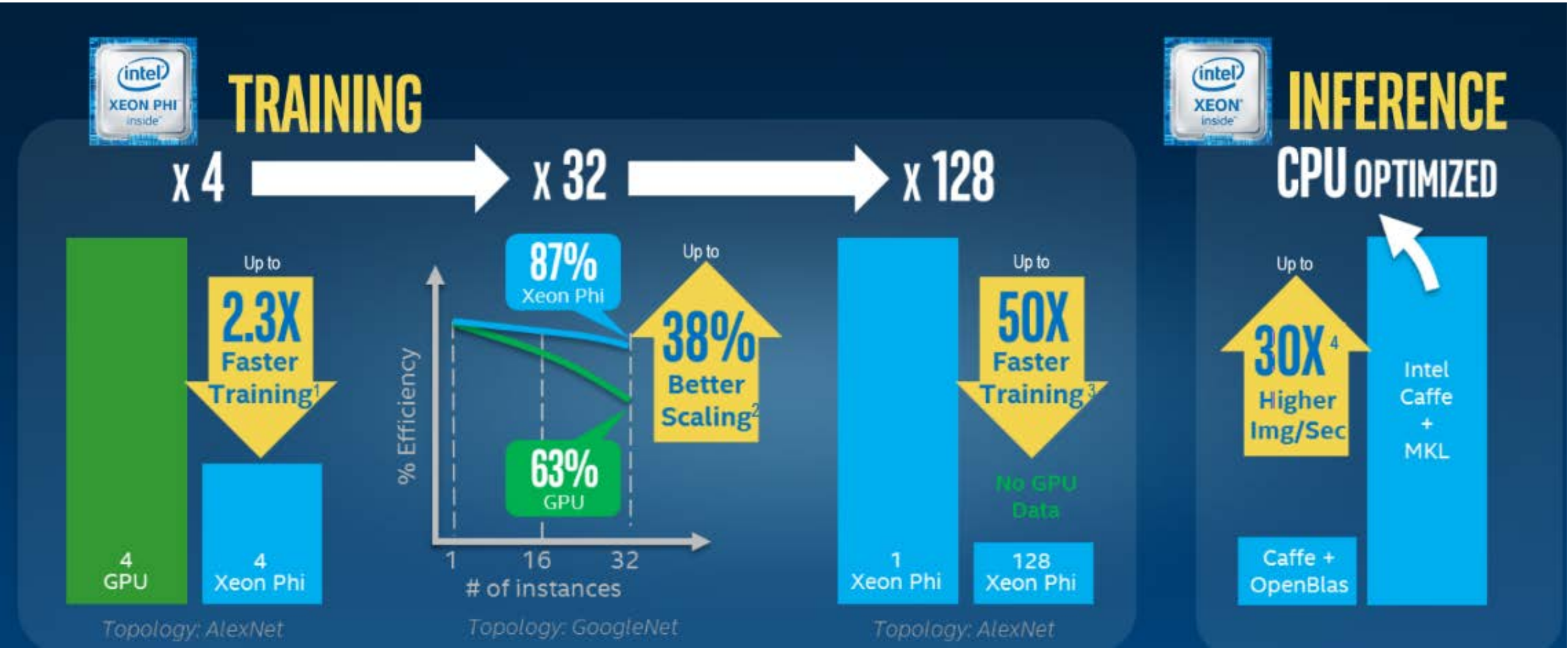
Courtesy: https://github.com/uber/horovod

Solutions and Case Studies: Exploiting HPC for DL

- NVIDIA NCCL
- Baidu-allreduce
- Facebook Gloo
- Co-design MPI runtimes and DL Frameworks
  - MPI+NCCL for CUDA-Aware CNTK
  - OSU-Caffe
- TensorFlow (Horovod)
- **Scaling DNN Training on Multi-/Many-core CPUs**
- PowerAI DDL

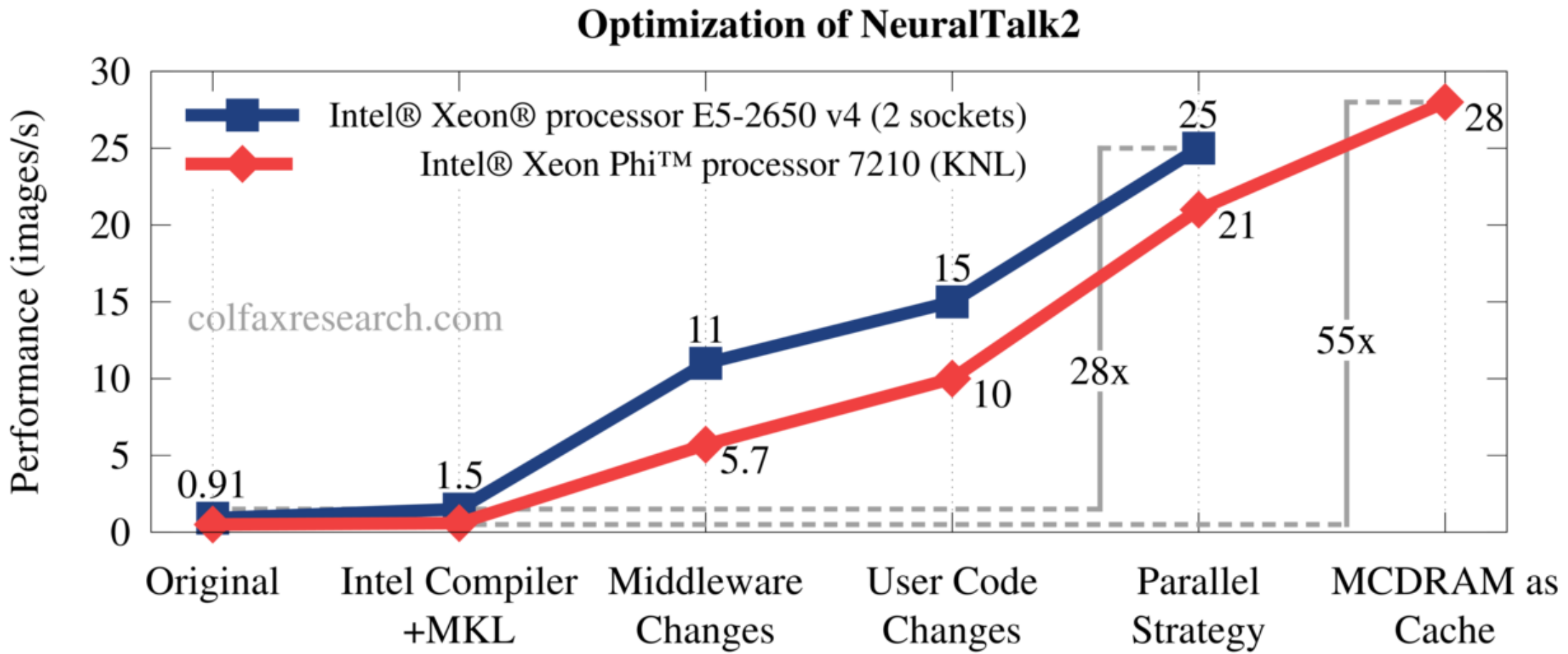


Optimizing and Scaling DL on Intel CPUs



Courtesy: https://www.nextplatform.com/2016/06/21/knights-landing-solid-ground-intels-stake-deep-learning/

Optimizing NeuralTalk on Intel CPUs with Intel MKL



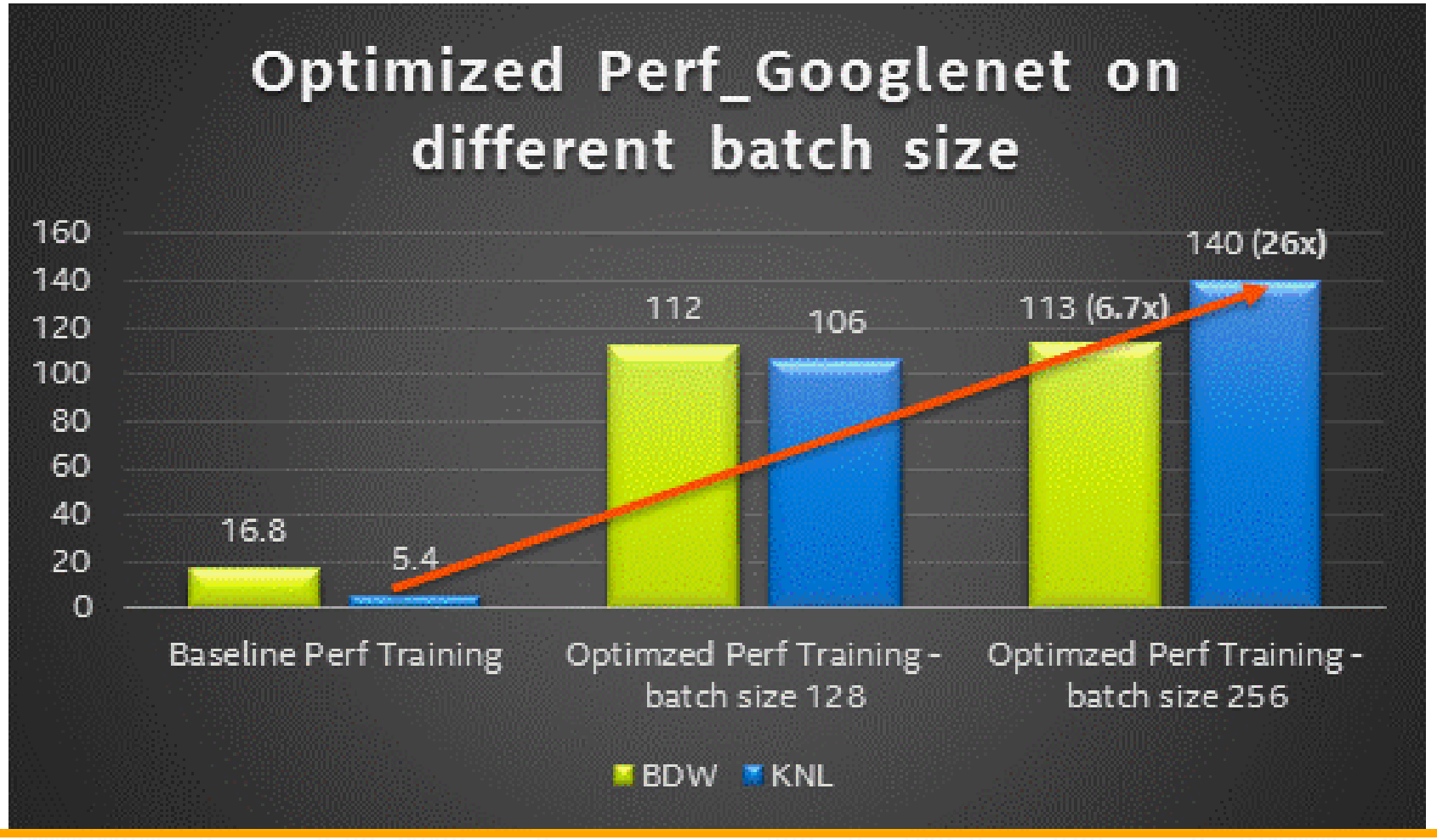
Courtesy: https://colfaxresearch.com/isc16-neuraltalk/

Caffe2 Performance Optimization with Intel MKL

OMP_NUM_THREADS=44		OMP_NUM_THREADS=1		
batch size	Intel® MKL (images/sec)	Eigen BLAS (images/sec)	Intel® MKL (images/sec)	Eigen BLAS (images/sec)
1	173.4	5.2	28.6	5.1
32	1500.2	29.3	64.6	15.4
64	1596.3	35.3	66.0	15.5
256	1735.2	44.9	67.3	16.2

Courtesy: https://software.intel.com/en-us/blogs/2017/04/18/intel-and-facebook-collaborate-to-boost-caffe2-performance-on-intel-cpu-s

TensorFlow Optimization for Intel CPUs



26x Speedup From New Optimizations – available through Google's TensorFlow Git

Courtesy: https://software.intel.com/en-us/articles/tensorflow-optimizations-on-modern-intel-architecture

Intel Machine Learning Scaling Library (MLSL)

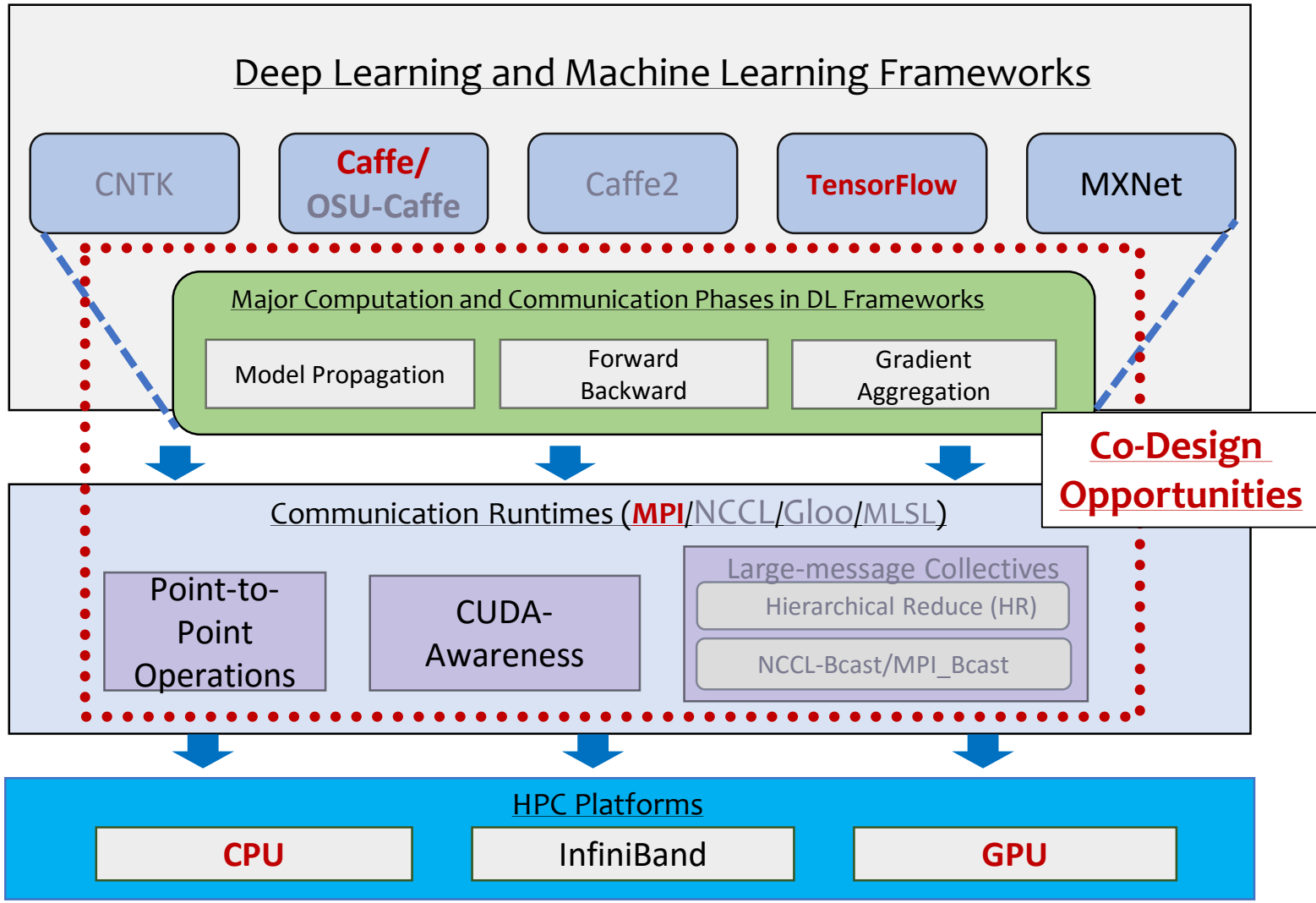
- Intel MLSL is built on top of MPI primitives
  - https://github.com/01org/MLSL
- Works across various interconnects: Intel(R) Omni-Path Architecture, InfiniBand*, and Ethernet
- Common API to support Deep Learning frameworks (Caffe*, Theano*, Torch*, etc.)

MLSL::Activation	A wrapper class for operation input and output activations
MLSL::CommBlockInfo	A class to hold block information for activations packing/unpacking
MLSL::Distribution	A class to hold the information about the parallelism scheme being used
MLSL::Environment	A singleton object that holds global Intel MLSL functions
MLSL::Operation	A class to hold information about learnable parameters (parameter sets) and activations corresponding to a certain operation of the computational graph
MLSL::OperationRegInfo	A class to hold Operation registration information
MLSL::ParameterSet	A wrapper class for operation parameters
MLSL::Session	A class to represent a collection of Operation objects with the same global mini-batch size
MLSL::Statistics	A class to measure and store performance statistics of communication among processes that perform computation in the computational graph

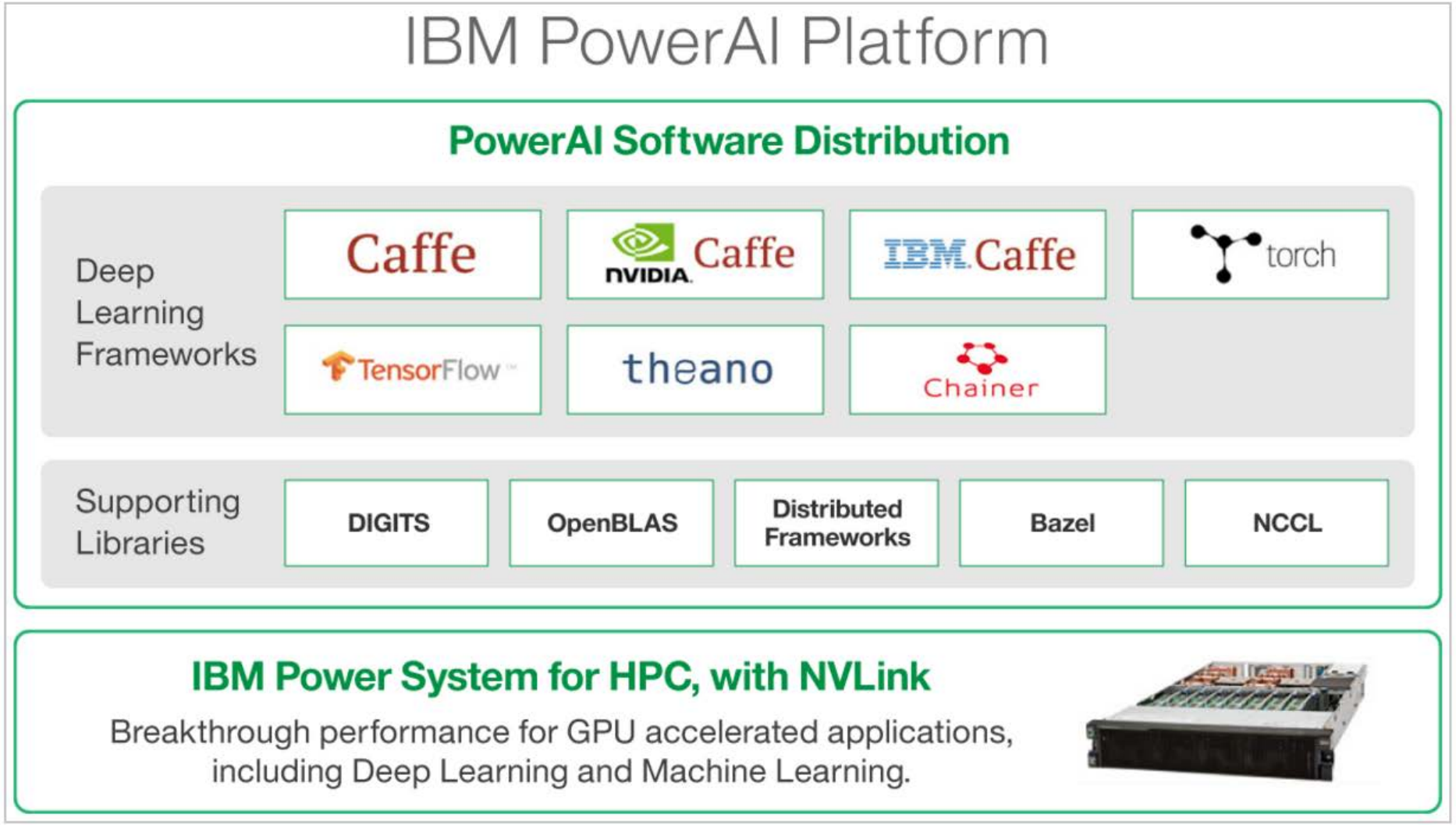
Courtesy: https://github.com/01org/MLSL

Solutions and Case Studies: Exploiting HPC for DL

- NVIDIA NCCL
- Baidu-allreduce
- Facebook Gloo
- Co-design MPI runtimes and DL Frameworks
  - MPI+NCCL for CUDA-Aware CNTK
  - OSU-Caffe
- TensorFlow (Horovod)
- Scaling DNN Training on Multi-/Many-core CPUs
- **PowerAI DDL**

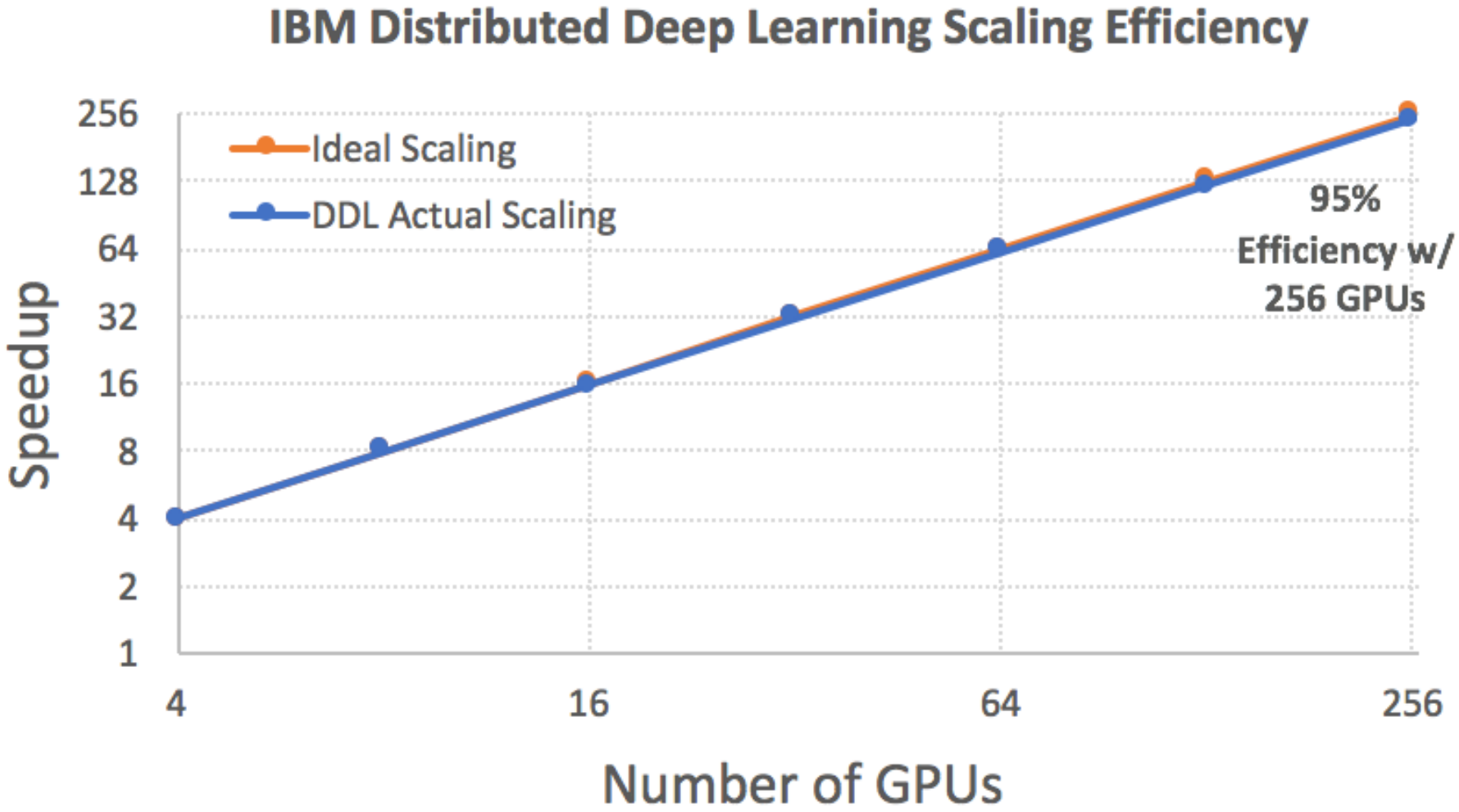


IBM PowerAI DDL



Courtesy: https://www.hpcwire.com/2017/08/08/ibm-raises-bar-distributed-deep-learning/

PowerAI DDL Performance



Caffe with PowerAI DDL on ResNet-50 model using the ImageNet-1K data set on 64 Power8 servers

Courtesy:
https://www.ibm.com/blogs/research/2017/08/distributed-deep-learning/
https://arxiv.org/pdf/1708.02188.pdf

Outline

- Introduction
- Overview of Execution Environments
- Parallel and Distributed DNN Training
- Latest Trends in HPC Technologies
- Challenges in Exploiting HPC Technologies for Deep Learning
- Solutions and Case Studies
- **Open Issues and Challenges**
- Conclusion

Open Issues and Challenges

- Which Framework should I use?
- Convergence of DL and HPC
- Scalability and Large batch-size training?
- DL Benchmarks and Thoughts on Standardization
- Open Exchange and Making AI accessible?

Which Framework should I use?

- Depends on the higher-level Application needs
  - Image, Speech, Sequences, etc.
- Depends on the hardware availability
  - GPUs are good in general
  - If you have Intel CPUs, Intel-Caffe and Intel-optimized TensorFlow are a good start
- Also depends upon your programming knowledge and requirements
  - Python frontend or C++ frontend?
  - Model designer tools needed?
    - Keras can use other DL frameworks as a back-end and provides a high-level interface.

Convergence of DL and HPC

- Is Deep Learning an HPC Problem?
  - Distributed DNN Training is definitely an HPC problem
  - Inference – not yet an HPC problem
- Why HPC can help?
  - Decades of research for communication models and performance optimizations
  - MPI, PGAS, and other upcoming programming models and communication runtimes can help for “data-parallel” training
- Some of the needs for DNN training are an exact match
  - Compute intensive problem
- Some needs are new for distributed/parallel communication runtimes
  - Large Message Communication
  - CUDA-Aware Communication

Scalability and Large batch-size training?

- Large batch-size helps improve the scalability
  - Lesser communication and more compute before synchronization
  - Limits to large batch-size
    - DL community is actively exploring this area
    - HPC community can also investigate overlap and latency-hiding techniques
- Is there a limit to DNN size?
  - Noam Shazeer's Outrageously Large Model (137 Billion Parameters)
  - https://arxiv.org/pdf/1701.06538.pdf
- Out-of-core Training for GPUs?
  - NVIDIA's vDNN - https://arxiv.org/pdf/1602.08124.pdf
  - Prune the network or selectively allocate/de-allocate memory on GPUs

DL Benchmarks and Thoughts on Standardization

- Can we have a standardized interface?
  - Are we there yet?
  - Deep Learning Interface (DLI)? Inspired by Message Passing Interface (MPI)
    - What can be a good starting point?
    - Will it come from the HPC community or the DL community?
    - Can there be a collaboration across communities?
- What about standard benchmarks?
  - Is there a need?
  - State-of-the-art
    - HKBU benchmarks - http://dlbench.comp.hkbu.edu.hk
    - Soumith Chintala's benchmarks - https://github.com/soumith/convnet-benchmarks

Open Exchange and Making AI accessible?

- OpenAI – a company focused towards making AI accessible and open
  - Backed up by several industry partners
    - Amazon, Microsoft, Infosys, etc.
  - And individuals
    - Elon Musk, Peter Thiel, others.

- ONNX format
  - An open format to exchange trained models
  - Cross-framework compatibility
  - Created by Facebook and Microsoft
  - TensorFlow and CoreML (Apple) are also supported (Convertor only)

Outline

- Introduction
- Overview of Execution Environments
- Parallel and Distributed DNN Training
- Latest Trends in HPC Technologies
- Challenges in Exploiting HPC Technologies for Deep Learning
- Solutions and Case Studies
- Open Issues and Challenges
- **Conclusion**

Conclusion

- Exponential growth in Deep Learning frameworks
- Provided an overview of issues, challenges, and opportunities for communication runtimes
  - Efficient, scalable, and hierarchical designs are crucial for DL frameworks
  - Co-design of communication runtimes and DL frameworks will be essential
    - OSU-Caffe
    - TensorFLow (MATEX, Baidu, Uber, etc.)
    - Intel-Caffe and Intel-MLSL
    - Neon and Nervana Graph
- Need collaborative efforts to achieve the full potential
- Standardization may help remove fragmentation in DL frameworks

Funding Acknowledgments

Funding Support by

Equipment Support by

Personnel Acknowledgments

Current Students (Graduate)

- A. Awan (Ph.D.)
- R. Biswas (M.S.)
- M. Bayatpour (Ph.D.)
- S. Chakraborthy (Ph.D.)
- C.-H. Chu (Ph.D.)
- S. Guganani (Ph.D.)
- J. Hashmi (Ph.D.)
- H. Javed (Ph.D.)
- P. Kousha (Ph.D.)
- D. Shankar (Ph.D.)
- H. Shi (Ph.D.)
- J. Zhang (Ph.D.)

Current Students (Undergraduate)

- N. Sarkauskas (B.S.)

Current Research Scientists

- X. Lu
- H. Subramoni

Current Research Specialist

- J. Smith
- M. Arnold

Current Post-doc

- A. Ruhela
- K. Manian

Past Students

- A. Augustine (M.S.)
- P. Balaji (Ph.D.)
- S. Bhagvat (M.S.)
- A. Bhat (M.S.)
- D. Buntinas (Ph.D.)
- L. Chai (Ph.D.)
- B. Chandrasekharan (M.S.)
- N. Dandapanthula (M.S.)
- V. Dhanraj (M.S.)
- T. Gangadharappa (M.S.)
- K. Gopalakrishnan (M.S.)
- W. Huang (Ph.D.)
- W. Jiang (M.S.)
- J. Jose (Ph.D.)
- S. Kini (M.S.)
- M. Koop (Ph.D.)
- K. Kulkarni (M.S.)
- R. Kumar (M.S.)
- S. Krishnamoorthy (M.S.)
- K. Kandalla (Ph.D.)
- M. Li (Ph.D.)
- P. Lai (M.S.)
- J. Liu (Ph.D.)
- M. Luo (Ph.D.)
- A. Mamidala (Ph.D.)
- G. Marsh (M.S.)
- V. Meshram (M.S.)
- A. Moody (M.S.)
- S. Naravula (Ph.D.)
- R. Noronha (Ph.D.)
- X. Ouyang (Ph.D.)
- S. Pai (M.S.)
- S. Potluri (Ph.D.)
- R. Rajachandrasekar (Ph.D.)
- G. Santhanaraman (Ph.D.)
- A. Singh (Ph.D.)
- J. Sridhar (M.S.)
- S. Sur (Ph.D.)
- H. Subramoni (Ph.D.)
- K. Vaidyanathan (Ph.D.)
- A. Vishnu (Ph.D.)
- J. Wu (Ph.D.)
- W. Yu (Ph.D.)

Past Research Scientist

- K. Hamidouche
- S. Sur

Past Programmers

- D. Bureddy
- J. Perkins

Past Post-Docs

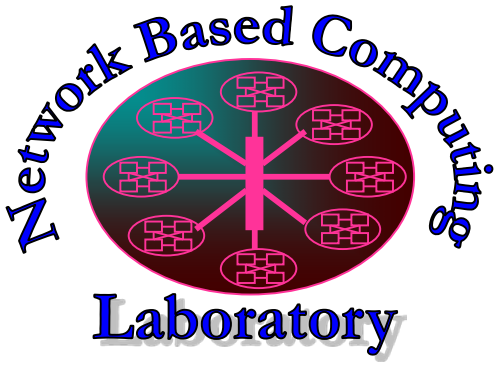
- D. Banerjee
- X. Besseron
- H.-W. Jin
- J. Lin
- M. Luo
- E. Mancini
- S. Marcarelli
- J. Vienne
- H. Wang

Thank You!

panda@cse.ohio-state.edu

awan.10@osu.edu

subramon@cse.ohio-state.edu



Network-Based Computing Laboratory
http://nowlab.cse.ohio-state.edu/

The MVAPICH2 Project
http://mvapich.cse.ohio-state.edu/

The High-Performance Deep Learning Project
http://hidl.cse.ohio-state.edu/

Appendix
(Details and Statistics related to DL Frameworks)

Berkeley (BVLC) Caffe

- Nearly 4,000 citations, usage by award papers at CVPR/ECCV/ICCV, and tutorials at ECCV'14 and CVPR'15
- Several efforts towards parallel/distributed training
  - OSU-Caffe - http://hidl.cse.ohio-state.edu/overview/
  - Intel-Caffe - https://github.com/intel/caffe
  - NVIDIA-Caffe - https://github.com/nvidia/caffe

GitHub Statistics	2017 (August)	2018 (Jan)
Stars	15,000	22,367
Contributors	250	256
Forks	10,000	13,689
Commits	N/A	4,078
Releases	14	14

Facebook Caffe2

- Official Parallel/Distributed Training support
- Modularity: Multiple communication back-ends supported
  - Facebook Gloo (Redis/MPI to bootstrap communication)
  - NVIDIA NCCL
  - Message Passing Interface (MPI)

GitHub Statistics	2017 (August)	2018 (Jan)
Stars	5,000	6,844
Contributors	112	150
Forks	N/A	1,547
Commits	2,300	3,120
Releases	4	4

Google TensorFlow

- Parallel/Distributed training
  - Official support through gRPC[1] library
- Several community efforts (TensorFlow/contrib)
  - MPI version by PNNL (MATEX) - https://github.com/matex-org/matex
  - MPI version by Baidu - https://github.com/baidu-research/tensorflow-allreduce
  - MPI+gRPC version by Minds.ai - https://www.minds.ai

[1] https://grpc.io/

GitHub Statistics	2017 (August)	2018 (Jan)
Stars	67,000	86,205
Contributors	1,012	1,244
Forks	33,000	42,027
Commits	21,000	27,152
Releases	38	45

Microsoft Cognitive Toolkit (CNTK)

- Parallel and Distributed Training (MPI and NCCL2 support)
- Community efforts
  - OSU's CUDA-Aware CNTK*

GitHub Statistics	2017 (August)	2018 (Jan)
Stars	12,000	13,614
Contributors	146	161
Forks	3,000	3,565
Commits	14,000	15,359
Releases	N/A	33

* Dip Sankar Banerjee, Khaled Hamidouche, and Dhabaleswar K. Panda. "Re-designing CNTK Deep Learning Framework on Modern GPU Enabled Clusters", 8th IEEE International Conference on Cloud Computing Technology and Science (CloudCom), Luxembourg 12-15, December 2016

Facebook Torch/PyTorch

- https://github.com/pytorch/pytorch
- Very active development
- Very recently got distributed training support
  - http://pytorch.org/docs/master/distributed.html

GitHub Statistics	2017 (August)	2018 (Jan)
Stars	7,000	11,384
Contributors	287	385
Forks	1,300	2,357
Commits	4,200	6,159
Releases	13	14

Preferred Networks Chainer/ChainerMN

- Preferred Networks (PN) is an NVIDIA Inception Program Startup
- Chainer is a very recent and emerging framework
- https://github.com/chainer/chainer

GitHub Statistics	2017 (August)	2018 (Jan)
Stars	2,853	3,383
Contributors	134	152
Forks	753	904
Commits	10,455	12,397
Releases	48	56

Intel Neon

- Neon is a Deep Learning framework by Intel/Nervana
  - Works on CPUs as well as GPUs!
  - Neon - https://github.com/NervanaSystems/neon
  - Claims to be very efficient in terms of performance
    - https://github.com/soumith/convnet-benchmarks

GitHub Statistics (Neon)	2018 (Jan)
Stars	3,389
Contributors	76
Forks	760
Commits	1,107
Releases	35

Courtesy: https://software.intel.com/en-us/ai-academy/frameworks/neon