Scalable and Distributed Deep Learning (DL): Co-Design MPI Runtimes and DL Frameworks

OSU Booth Talk (SC '19)

Ammar Ahmad Awan

awan.10@osu.edu

Network Based Computing Laboratory

Dept. of Computer Science and Engineering

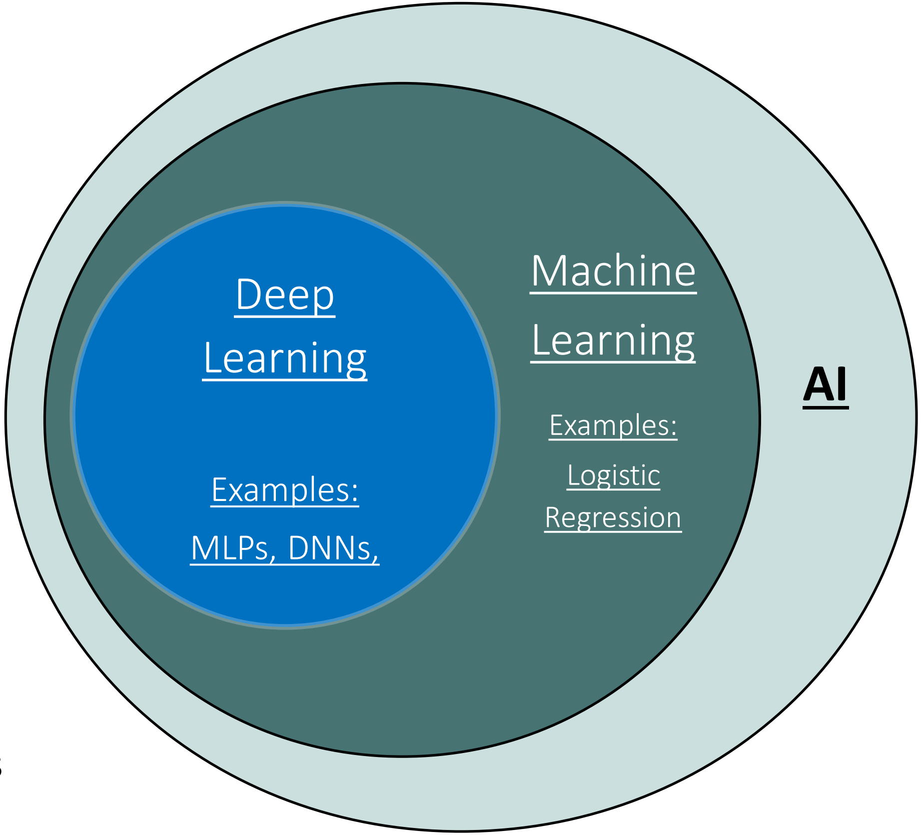
The Ohio State University

Agenda

- **Introduction**
  - **Deep Learning Trends**
  - **CPUs and GPUs for Deep Learning**
  - **Message Passing Interface (MPI)**
- Research Challenges: Exploiting HPC for Deep Learning
- Proposed Solutions
- Conclusion

Understanding the Deep Learning Resurgence

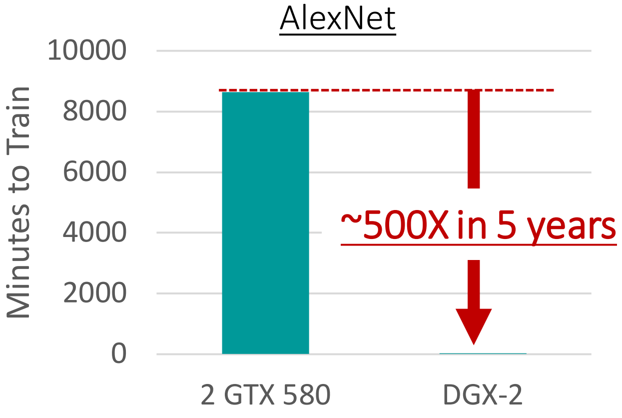
- Deep Learning (DL) is a sub-set of Machine Learning (ML)
  - Perhaps, the most revolutionary subset!
  - **Feature extraction** vs. **hand-crafted features**
- Deep Learning
  - A renewed interest and a lot of hype!
  - Key success: Deep Neural Networks (DNNs)
  - Everything was there since the late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

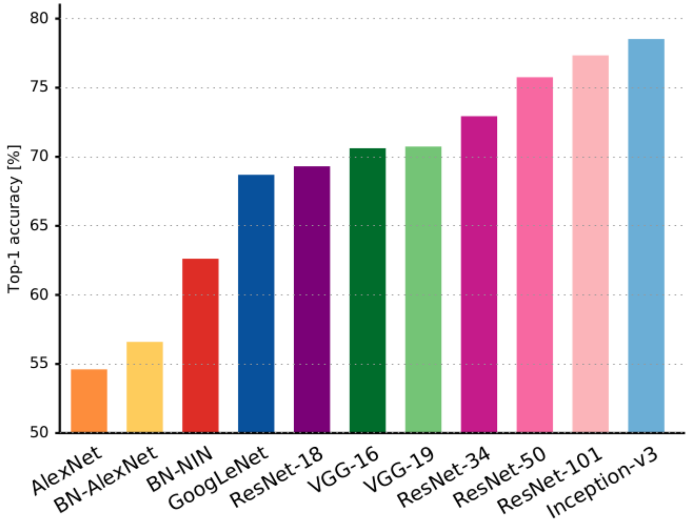


Adopted from: http://www.deeplearningbook.org/contents/intro.html

Deep Learning in the Many-core Era

- Modern and efficient hardware enabled
  - **Computability of DNNs – impossible in the past!**
  - GPUs – at the core of DNN training
  - CPUs – catching up fast
- Availability of **Datasets**
  - MNIST, CIFAR10, ImageNet, and more…
- Excellent **Accuracy** for many application areas
  - Vision, Machine Translation, and several others...

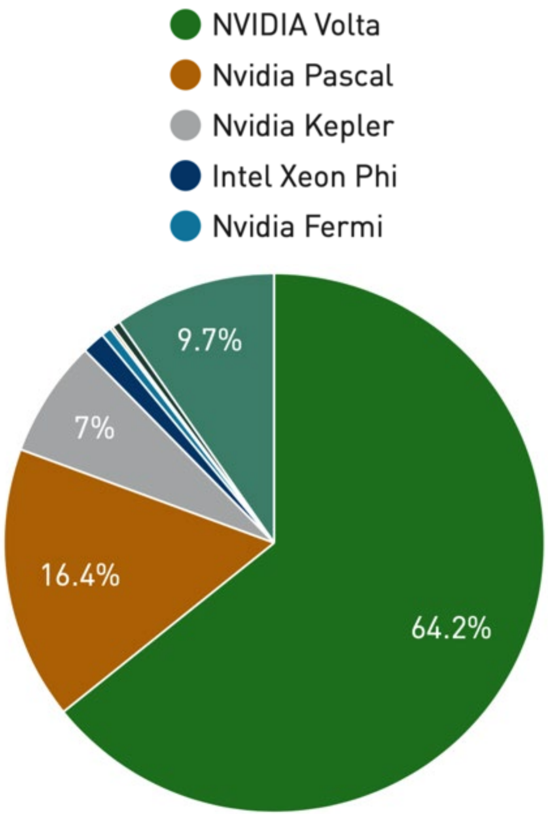




Courtesy: A. Canziani et al., "An Analysis of Deep Neural Network Models for Practical Applications", *CoRR*, 2016.

Deep Learning and HPC

- NVIDIA GPUs - main driving force for faster training of DL models
  - The ImageNet Challenge - (ILSVRC)
  - 90% of the ImageNet teams used GPUs in 2014
  - DNNs like Inception, ResNet(s), NASNets, and Amoeba
  - Natural fit for DL workloads – throughput-oriented
- In the High Performance Computing (HPC) arena
  - 124/500 Top HPC systems use NVIDIA GPUs (Jun '19)
  - CUDA-Aware Message Passing Interface (MPI)
  - NVIDIA Fermi, Kepler, Pascal, and Volta GPUs
  - DGX-1 (Pascal) and DGX-2 (Volta) - Dedicated DL supercomputers

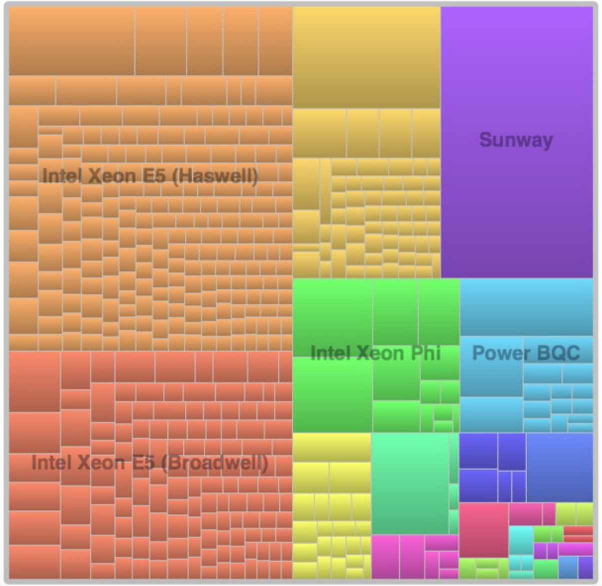


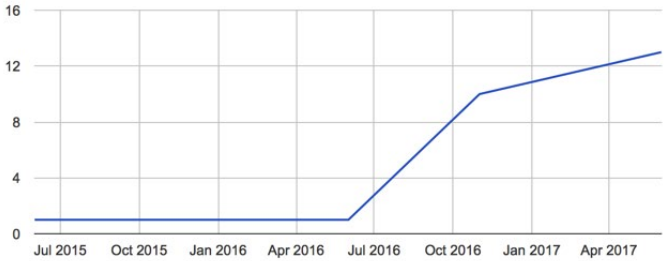
Accelerator/CP Performance Share
www.top500.org

And CPUs are catching up fast

https://www.top500.org/statistics/list/

- Intel CPUs are everywhere and many-core CPUs are emerging according to Top500.org
- Host CPUs exist even on the GPU nodes
  - Many-core Xeon(s) and EPYC(s) are increasing
- Usually, we hear CPUs are ***10x – 100x*** slower than GPUs? [1-3]
  - But, CPU-based ML/DL is getting attention and performance has significantly improved now





System Count for Xeon Phi

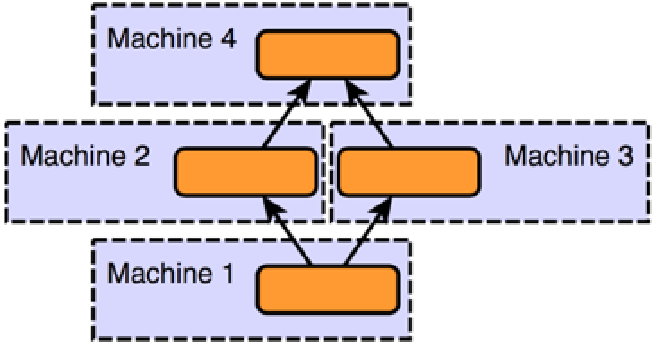
1- https://dl.acm.org/citation.cfm?id=1993516
2- http://ieeexplore.ieee.org/abstract/document/5762730/
3- https://dspace.mit.edu/bitstream/handle/1721.1/51839/MIT-CSAIL-TR-2010-013.pdf?sequence=1

Deep Learning Frameworks – CPUs or GPUs?

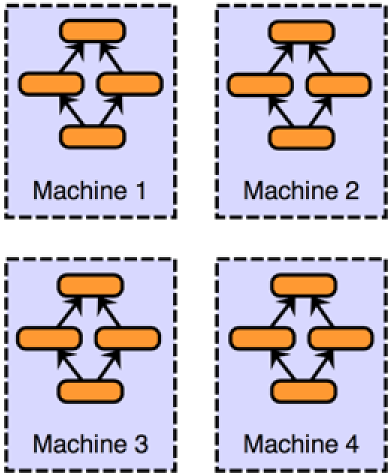
- There are several Deep Learning (DL) or DNN Training frameworks
- Every (almost every) framework has been optimized for NVIDIA GPUs
  - cuBLAS and cuDNN have led to significant performance gains!
- But every framework is able to execute on a CPU as well
  - So why are we not using them?
  - Performance has been "terrible" and several studies have reported significant degradation when using CPUs (see nvidia.qwiklab.com)
- But there is hope, a lot of great progress here!
  - And MKL-DNN, just like cuDNN, has definitely rekindled this!!
  - The landscape for CPU-based DL looks promising..

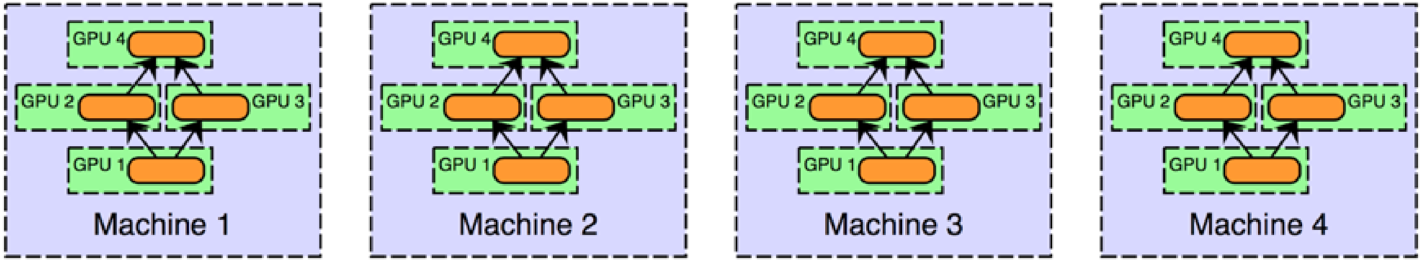
Parallelization Strategies for DL

- Some parallelization strategies..
  - Data Parallelism or Model Parallelism
  - Hybrid Parallelism



Model Parallelism





Hybrid (Model and Data) Parallelism

Data Parallelism

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

What to use for Deep Learning scale-out?

- What is Message Passing Interface (**MPI**)?
  - a de-facto standard for expressing distributed-memory parallel programming
  - used for communication between processes in multi-process applications
- ***MVAPICH2 is a high-performance implementation of the MPI standard***
- **What can MPI do for Deep Learning?**
  - MPI has been used for large scale scientific applications
  - Deep Learning can also exploit MPI to perform high-performance communication
- **Why do I need communication in Deep Learning?**
  - If you use one GPU or one CPU, you do not need communication
  - But, one GPU or CPU is not enough! DL needs as many compute elements as it can get!
  - ***MPI is a great fit – Point to Point and Collectives (Broadcast, Reduce, and Allreduce) are all you need for many types of parallel DNN training (data-parallel, model-parallel, and hybrid-parallel)***

MVAPICH2: The best MPI Library for Deep Learning!

- 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 3,050 organizations in 89 countries**
  - **More than 615,000 (> 0.6 million) downloads from the OSU site directly**
  - Empowering many TOP500 clusters (June '19 ranking)
    - 3rd ranked 10,649,640-core cluster (Sunway TaihuLight) at NSC, Wuxi, China
    - 8th, 391,680 cores (ABCI) in Japan
    - 16th, 556,104 cores (Oakforest-PACS) in Japan
    - 19th, 367,024 cores (Stampede2) at TACC
    - 31st, 241,108-core (Pleiades) at NASA and many others
  - Available with software stacks of many vendors and Linux Distros (RedHat, SuSE, OpenHPC, and Spack)
  - **http://mvapich.cse.ohio-state.edu**
- Empowering Top500 systems for over a decade

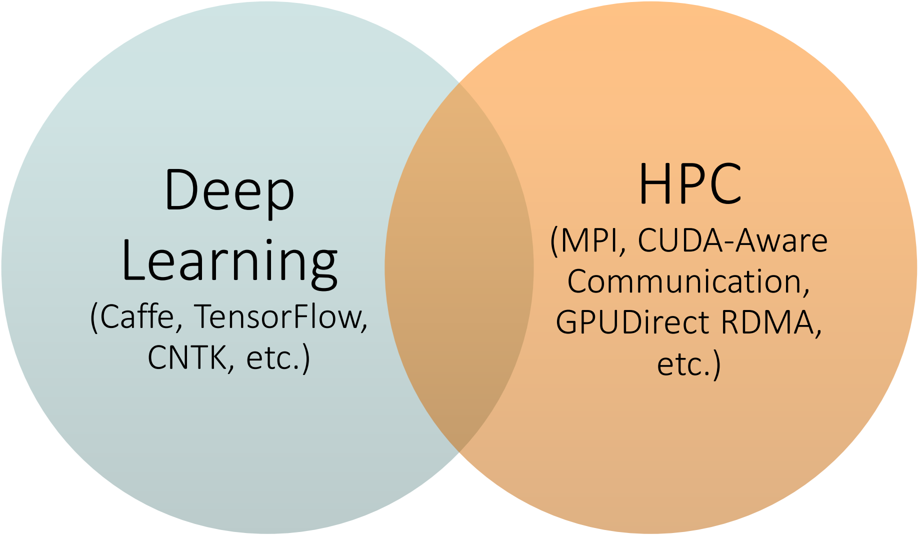
Partner in the 5th ranked TACC Frontera System

Agenda

- Introduction
- **Research Challenges: Exploiting HPC for Deep Learning**
- Proposed Solutions
- Conclusion

Research Area: Requirements and Trends

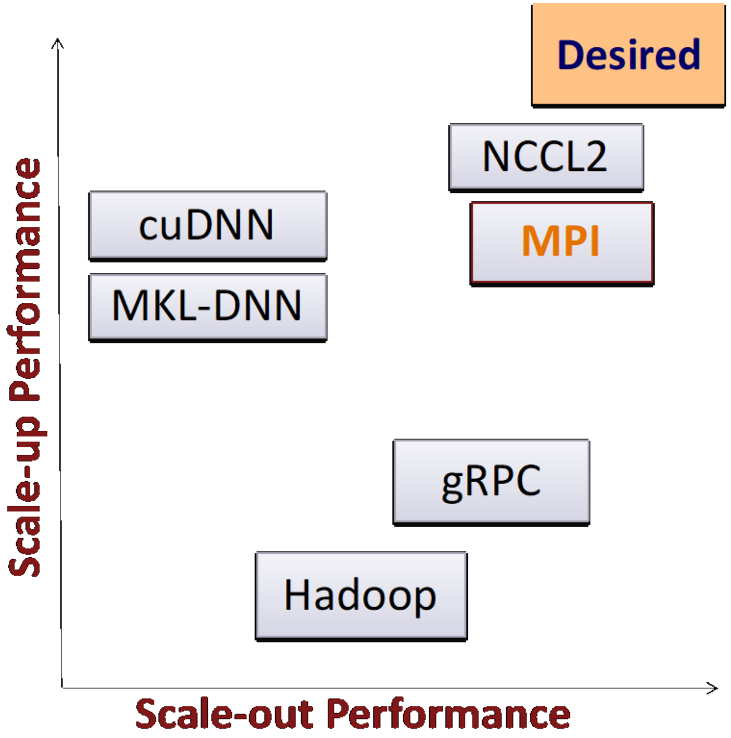
- Intersection of HPC and Deep Learning
  - DL Frameworks
  - Communication Runtimes
  - GPUs and Multi-/Many-core CPUs
  - High-Performance Interconnects



- Large DNNs – very-large messages, GPU buffers, and out-of-core workloads!
- HPC-oriented Communication Middleware – under-optimized for such workloads!
- DL Frameworks – mostly optimized for single-node
  - Distributed/Parallel Training – an emerging trend!
  - Scale-up (Intra-node) and Scale-out (Inter-node) options need to be explored

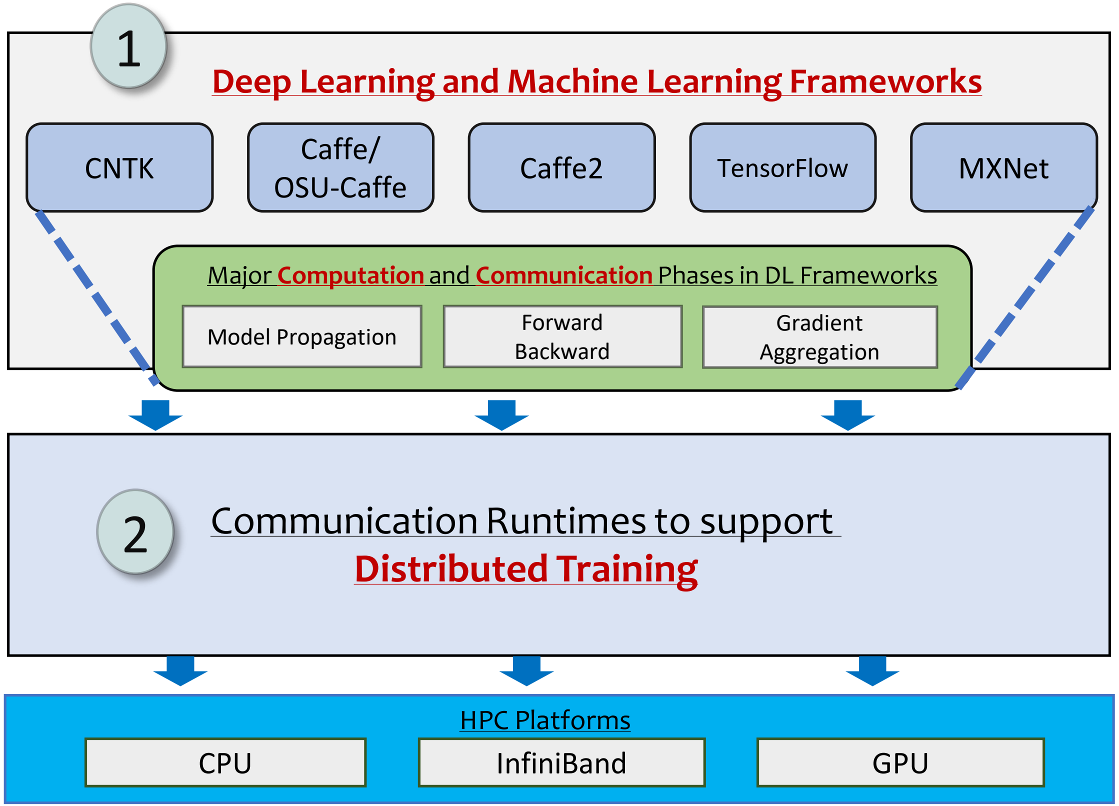
Broad Challenge

How to efficiently Scale-up and Scale-out Deep Learning (DL) workloads by exploiting diverse High Performance Computing (HPC) technologies and co-designing Communication Middleware like MPI and DL Frameworks?



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



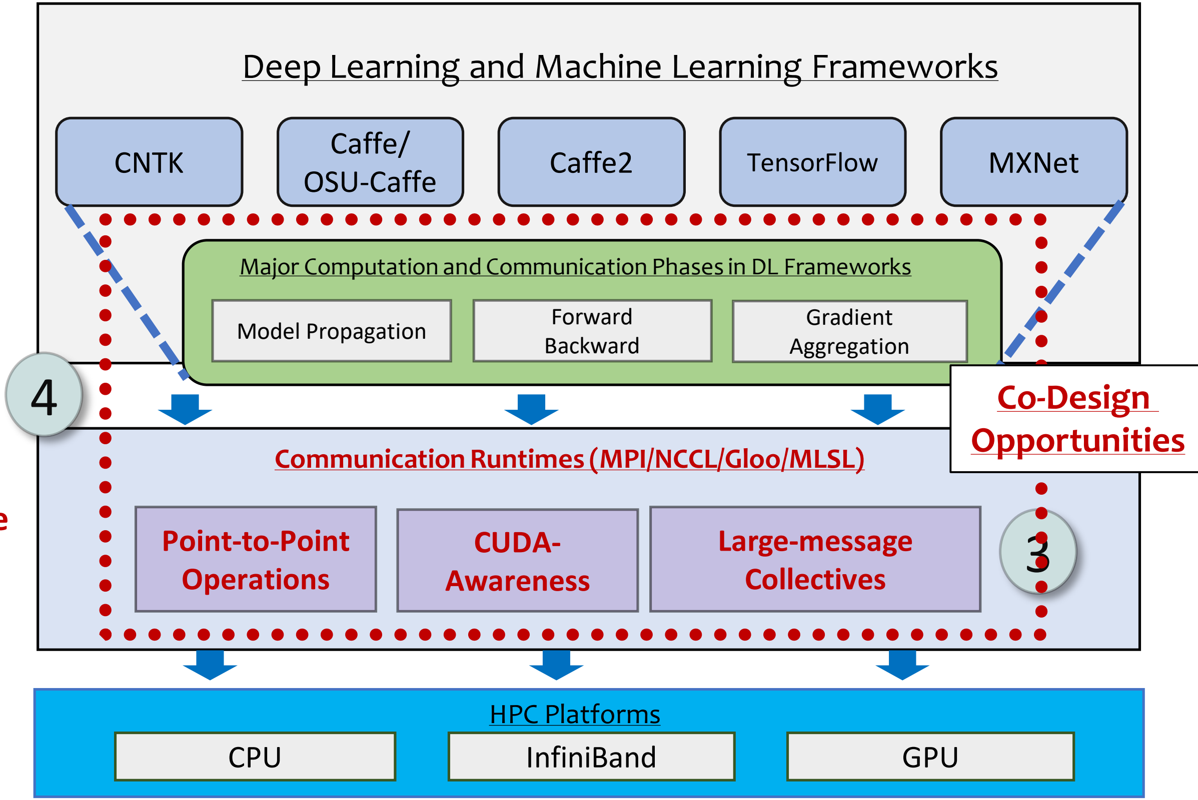
Research Challenges to Exploit HPC Technologies (Cont'd)

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?

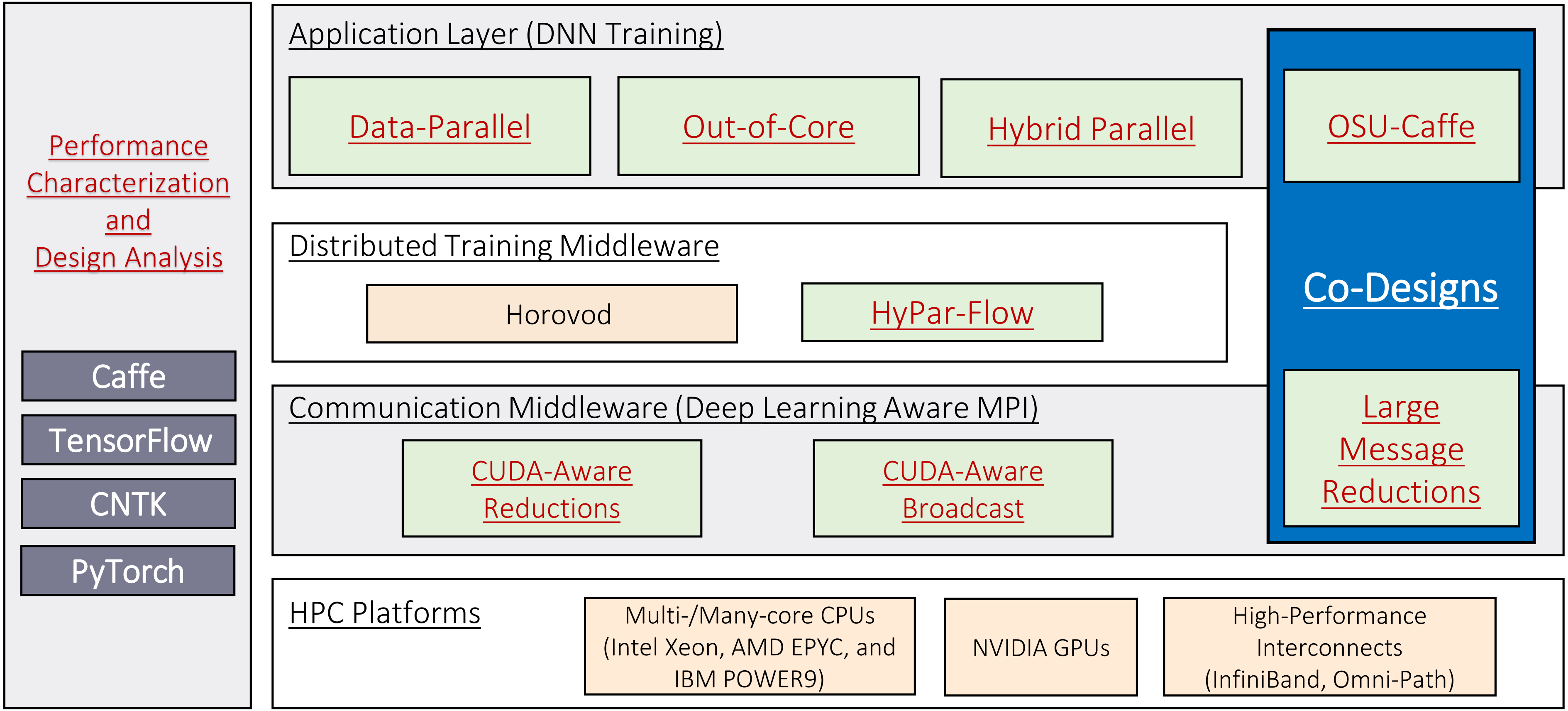
- **Co-Design** the support at **Runtime level** and Exploit it at the **DL Framework level**
- What performance benefits can be observed?
- What needs to be fixed at the **communication runtime** layer?



Agenda

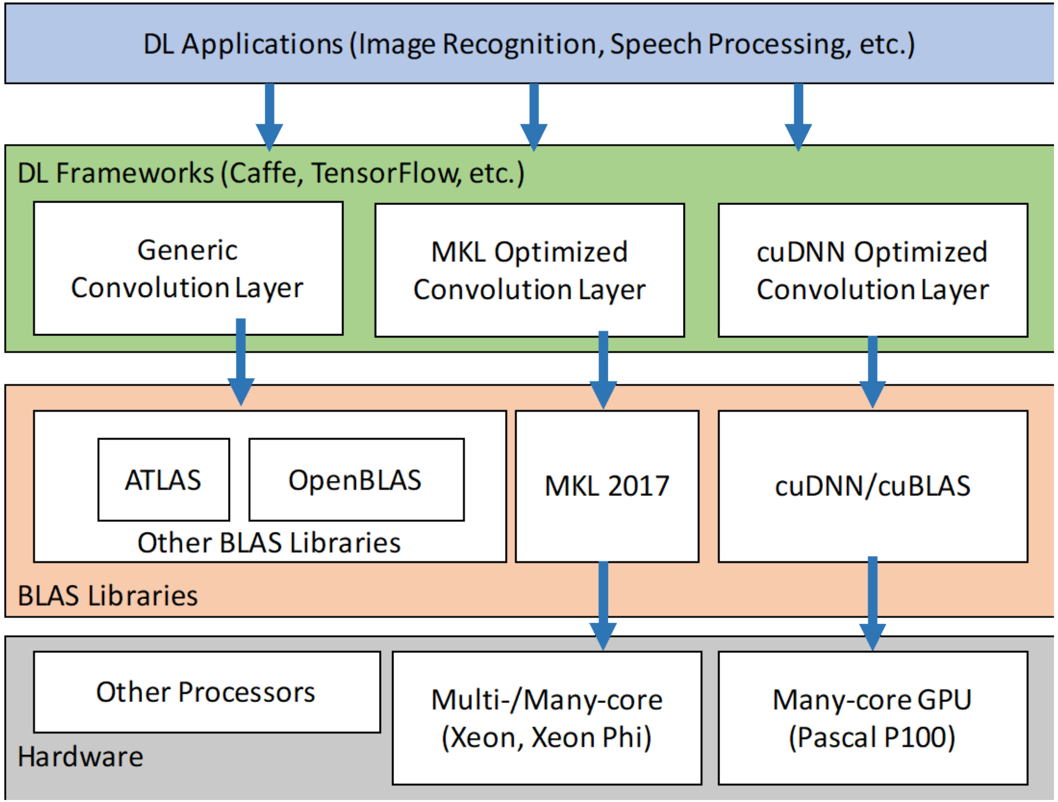
- Introduction
- Research Challenges: Exploiting HPC for Deep Learning
- **Proposed Solutions**
- Conclusion

Overview of the Proposed Solutions



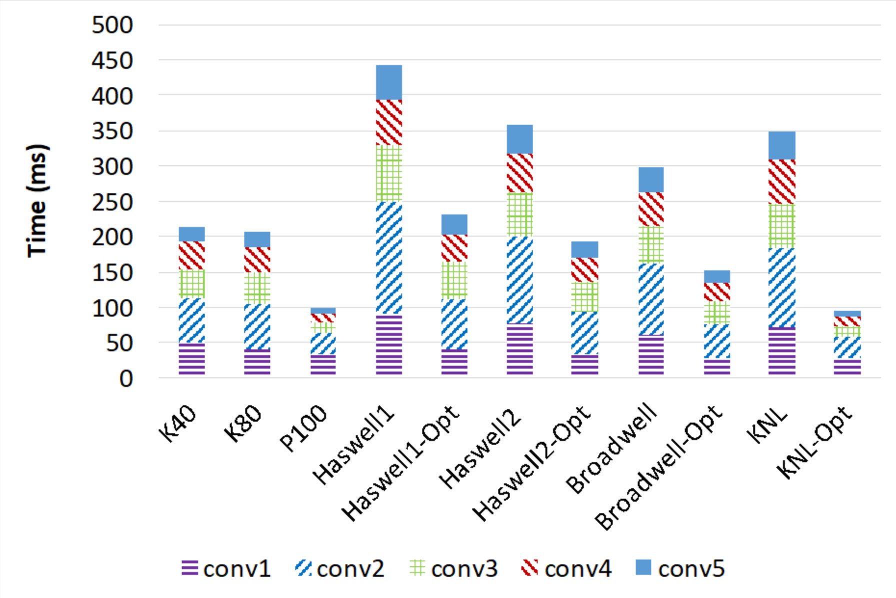
Understanding the Impact of Execution Environments

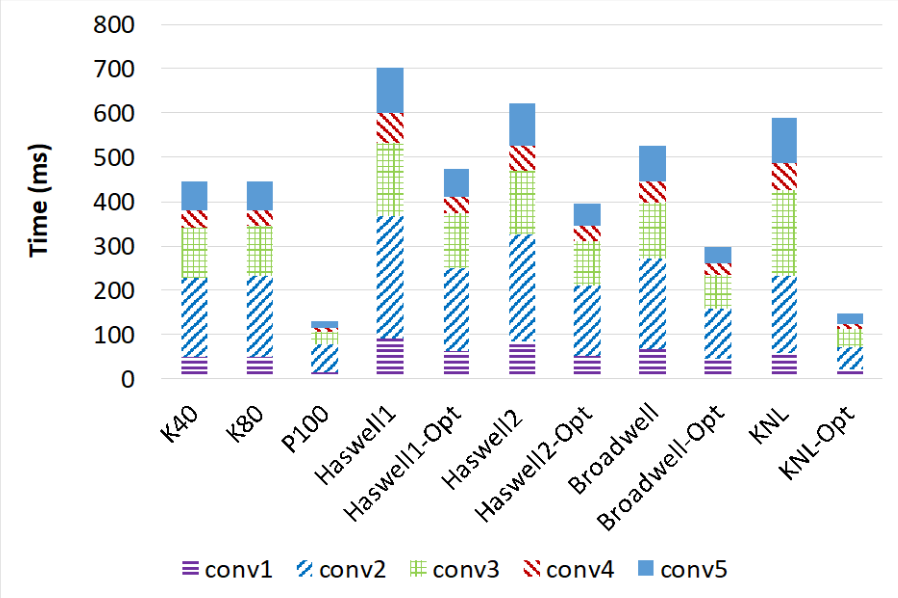
- Performance depends on many factors
- Hardware Architectures
  - GPUs
  - Multi-/Many-core CPUs
  - Software Libraries: cuDNN (for GPUs), MKL-DNN/MKL 2017 (for CPUs)
- Hardware and Software co-design
  - Software libraries optimized for one platform will not help the other!
  - cuDNN vs. MKL-DNN



A. A. Awan, H. Subramoni, D. Panda, "An In-depth Performance Characterization of CPU- and GPU-based DNN Training on Modern Architectures" 3rd Workshop on Machine Learning in High Performance Computing Environments, held in conjunction with SC17, Nov 2017.

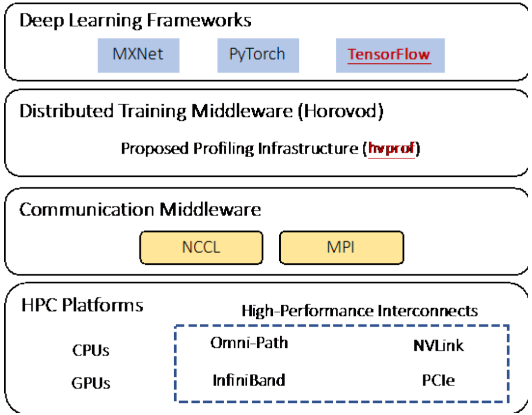
The Full Landscape for AlexNet Training on CPU/GPU





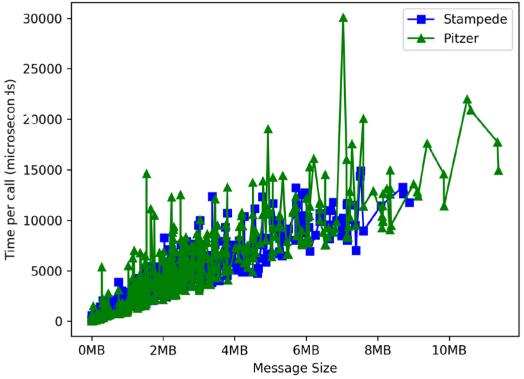
- Convolutions in the Forward and Backward Pass
- ***Faster Convolutions → Faster Training***
- Most performance gains are based on ***conv2*** and ***conv3***.

Communication Profiling of Distributed TF

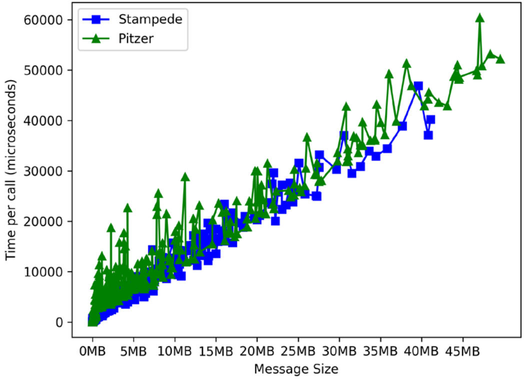


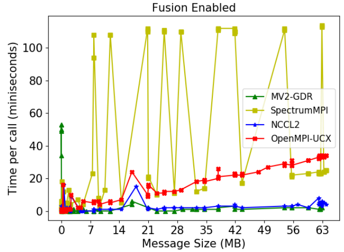
- White-box profiling is needed for complex DL frameworks
- hvprof provides multiple types of valuable metrics for
  - 1) ML/DL developers and 2) Designers of MPI libraries
- Profile of Latency for Allreduce (NVLink, PCIe, IB, Omni-Path)
- *Summary: Non-power of 2 is under-optimized for all libraries!*

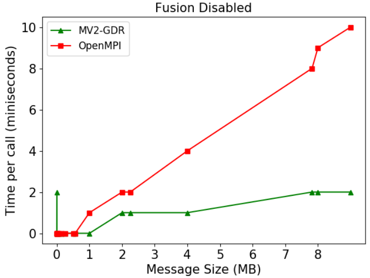
Inception-v4– Intel MPI



ResNet-101– MVAPICH2

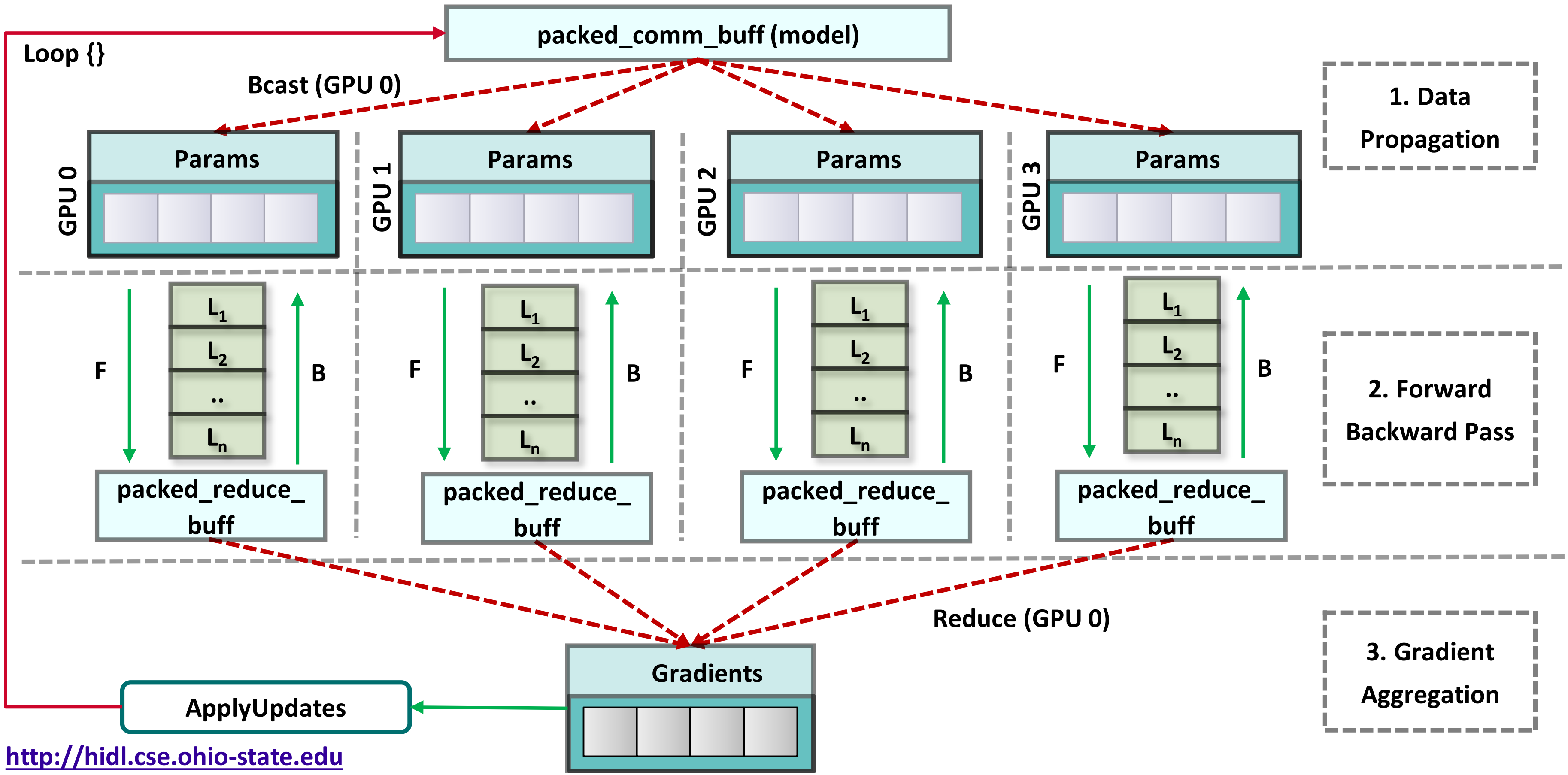






A. A. Awan et al., "Communication Profiling and Characterization of Deep Learning Workloads on Clusters with High-Performance Interconnects", IEEE Hot Interconnects '19.

OSU-Caffe Architecture

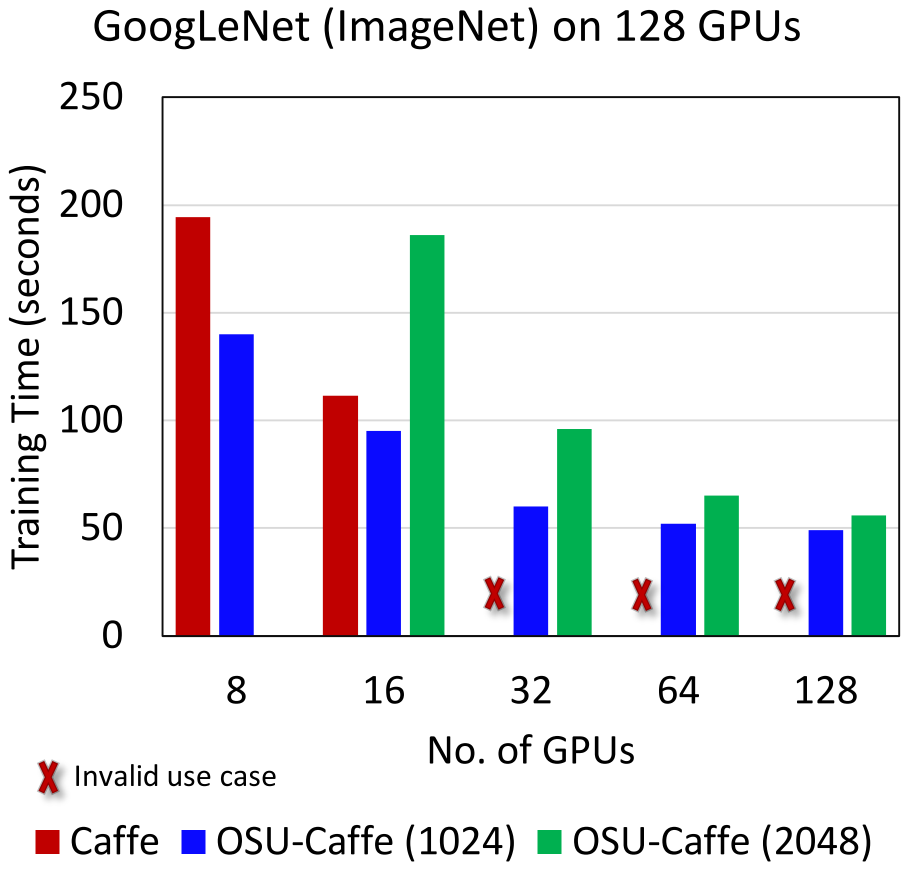


http://hidl.cse.ohio-state.edu

OSU-Caffe 0.9: Scalable Deep Learning on GPU Clusters

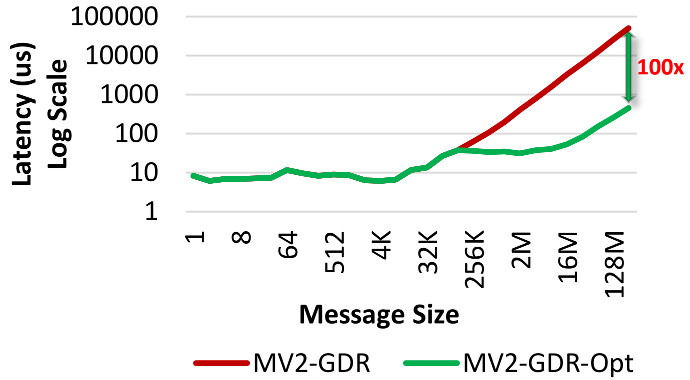
- Caffe : A flexible and layered Deep Learning framework.
- Benefits and Weaknesses
  - Multi-GPU Training within a single node
  - Performance degradation for GPUs across different sockets
  - Limited Scale-out
- OSU-Caffe: MPI-based Parallel Training
  - Enable Scale-up (within a node) and Scale-out (across multi-GPU nodes)
  - Scale-out on 64 GPUs for training CIFAR-10 network on CIFAR-10 dataset
  - Scale-out on 128 GPUs for training GoogLeNet network on ImageNet dataset

OSU-Caffe 0.9 available from HiDL site

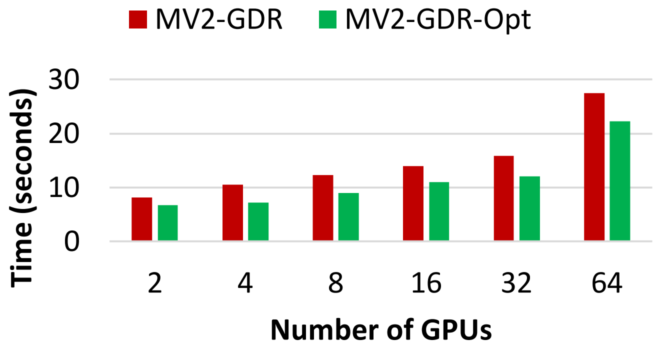


Efficient Broadcast for MVAPICH2-GDR using NVIDIA NCCL

- NCCL has some limitations
  - Only works for a single node, thus, no scale-out on multiple nodes
  - Degradation across IOH (socket) for scale-up (within a node)
- We propose optimized MPI_Bcast
  - Communication of very large GPU buffers (order of megabytes)
  - Scale-out on large number of dense multi-GPU nodes
- Hierarchical Communication that efficiently exploits:
  - CUDA-Aware MPI_Bcast in MV2-GDR
  - NCCL Broadcast primitive



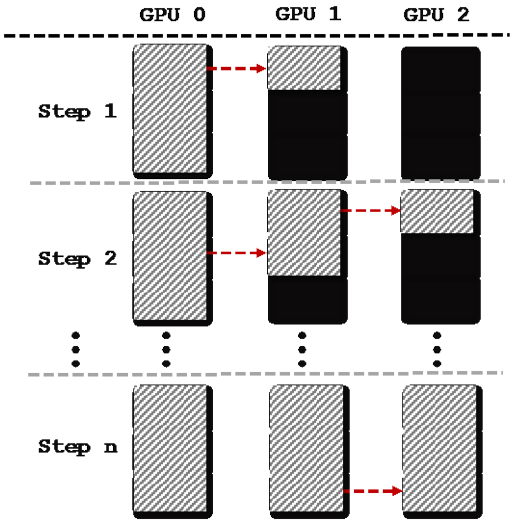
Performance Benefits: OSU Micro-benchmarks



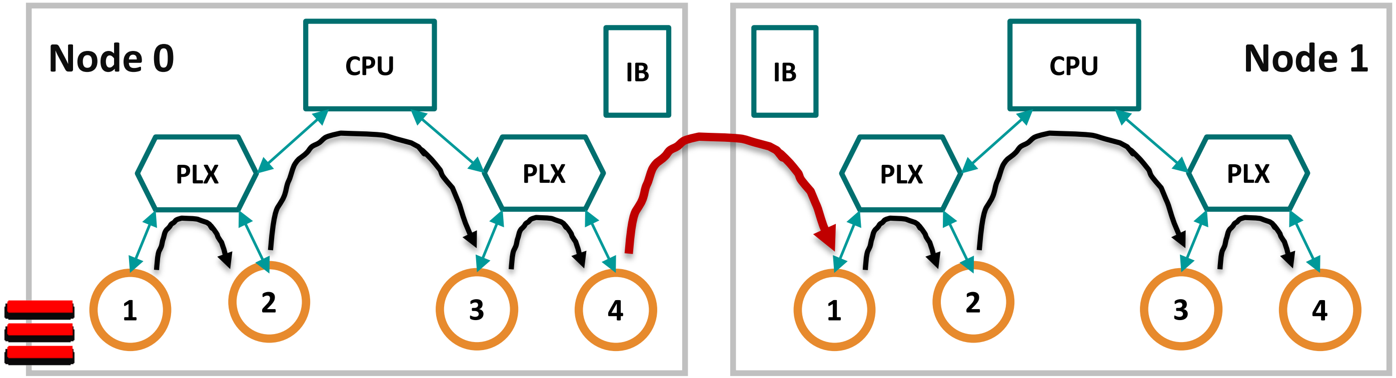
Performance Benefits: Microsoft CNTK DL framework
(25% avg. improvement)

Efficient Large Message Broadcast using NCCL and CUDA-Aware MPI for Deep Learning, A. Awan , K. Hamidouche , A. Venkatesh , and D. K. Panda, EuroMPI 16
[Best Paper Runner-Up]

Pure MPI Large Message Bcast (w/out NCCL)



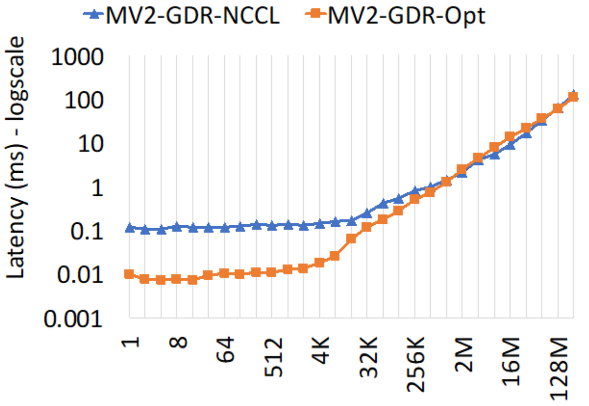
- Efficient Intra-node communication on PCIe-based dense-GPU systems
  - Pipeline multiple chunks in a ***uni-directional*** ring fashion
  - Take advantage of the PCIe and IB topology to utilize all ***bi-directional*** links to saturate the maximum available bandwidth between GPUs



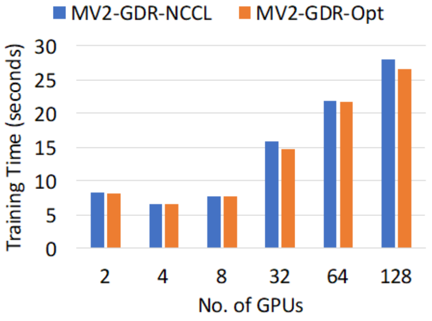
A. A. Awan et al., "Optimized Large-Message Broadcast for Deep Learning Workloads: MPI, MPI+NCCL, or NCCL2?", J. Parallel Computing (2019)

Pure MPI Large Message Bcast (w/out NCCL)

- MPI_Bcast: Design and Performance Tuning for DL Workloads
  - Design ring-based algorithms for large messages
  - Harness a multitude of algorithms and techniques for best performance across the full range of message size and process/GPU count
- Performance Benefits
  - Performance comparable or better than NCCL-augmented approaches for large messages
  - Up to 10X improvement for small/medium message sizes with micro-benchmarks and up to 7% improvement for VGG training



MPI Bcast Benchmark: 128 GPUs (8 nodes)

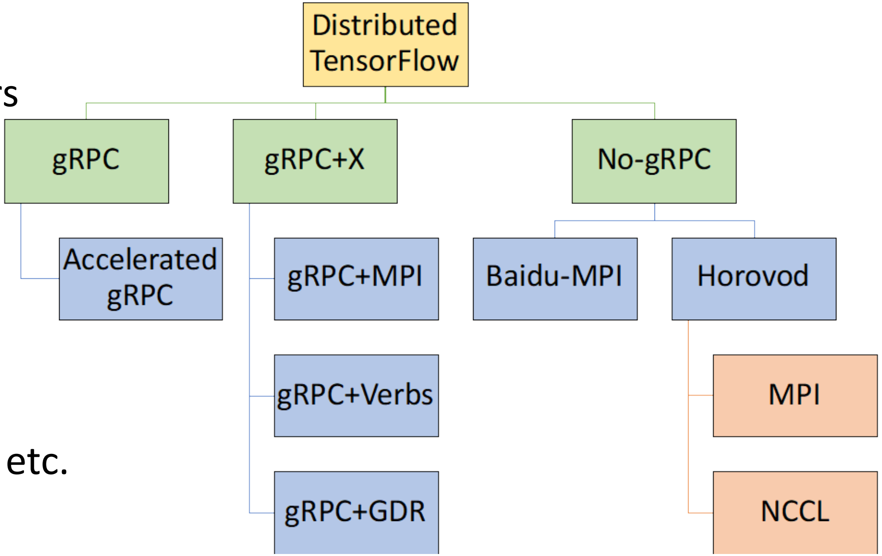


VGG Training with CNTK

A. A. Awan et al., "Optimized Large-Message Broadcast for Deep Learning Workloads: MPI, MPI+NCCL, or NCCL2?", J. Parallel Computing (2019)

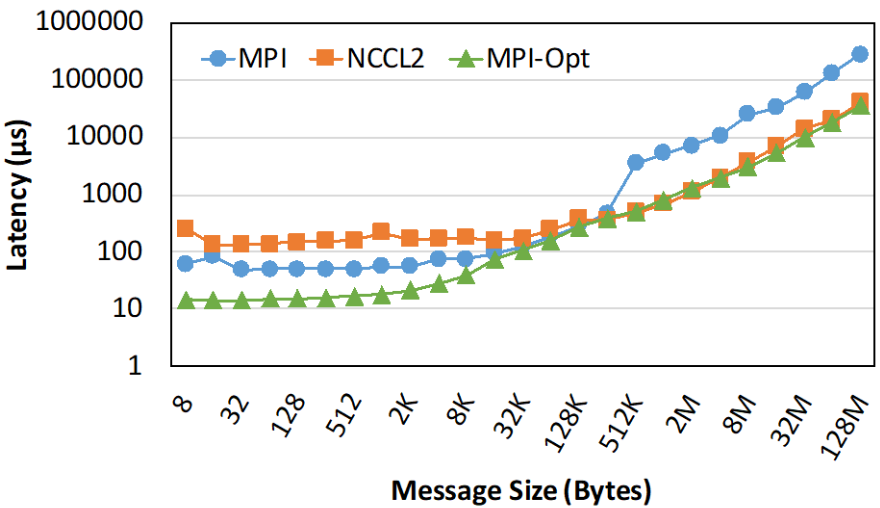
Data Parallel Training with TensorFlow (TF)

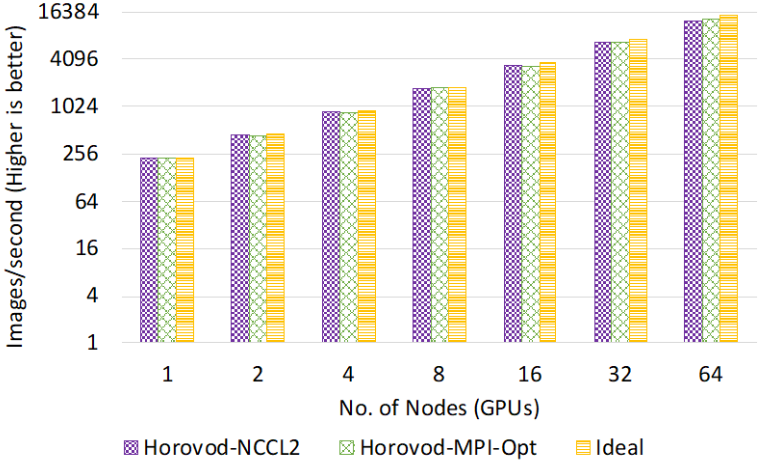
- Need to understand several options currently available
- gRPC (official support)
  - Open-source – can be enhanced by others
  - Accelerated gRPC (add RDMA to gRPC)
- gRPC+X
  - Use gRPC for bootstrap and rendezvous
  - ***Actual communication is in "X"***
  - X→ MPI, Verbs, GPUDirect RDMA (GDR), etc.
- No-gRPC
  - Baidu – the first one to use MPI Collectives for TF
  - Horovod – Use NCCL, or MPI, or any other future library (e.g. IBM DDL recently added)



A. A. Awan, J. Bedorf, C.-H. Chu, H. Subramoni and D. K. Panda, "Scalable Distributed DNN Training using TensorFlow and CUDA-Aware MPI: Characterization, Designs, and Performance Evaluation", CCGrid '19. https://arxiv.org/abs/1810.11112

Data Parallel Training with TF: NCCL vs. MVAPICH2-GDR





Faster Allreduce in the proposed MPI-Opt implemented in MVAPICH2-GDR

–>

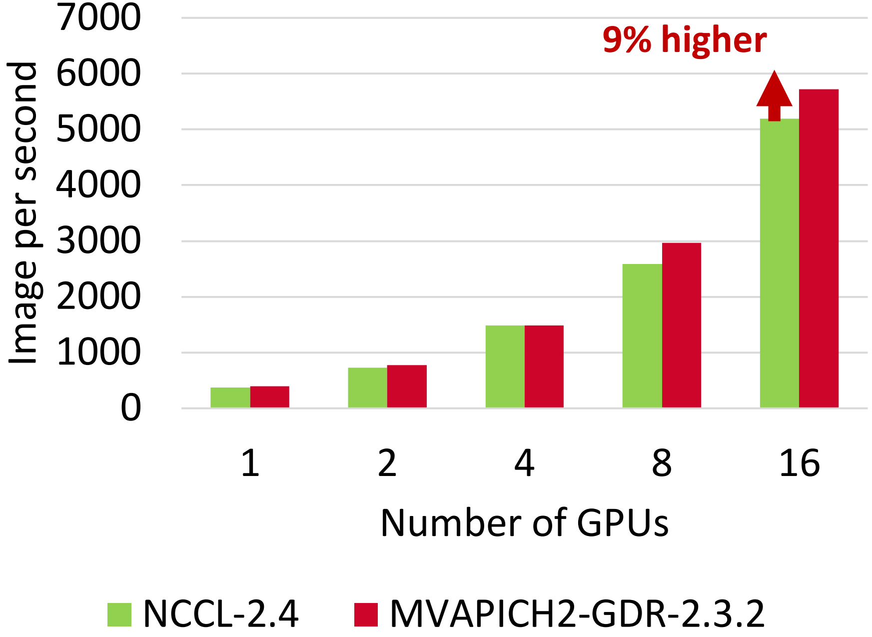
Faster (near-ideal) DNN Training speed-ups in TensorFlow-Horovod

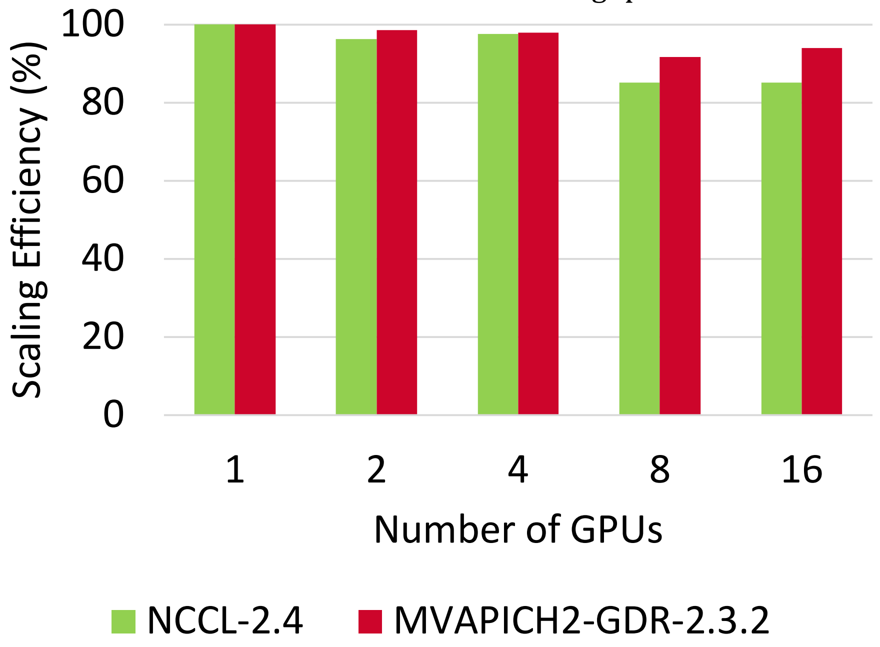
A. A. Awan, J. Bedorf, C.-H. Chu, H. Subramoni and D. K. Panda, "Scalable Distributed DNN Training using TensorFlow and CUDA-Aware MPI: Characterization, Designs, and Performance Evaluation", CCGrid '19. https://arxiv.org/abs/1810.11112

Data Parallel Training with TF and MVAPICH2 on DGX-2

- **ResNet-50 Training using TensorFlow benchmark on 1 DGX-2 node (16 Volta GPUs)**

$$\text{Scaling Efficiency} = \frac{\text{Actual throughput}}{\text{Ideal throughput at scale}} \times 100\%$$

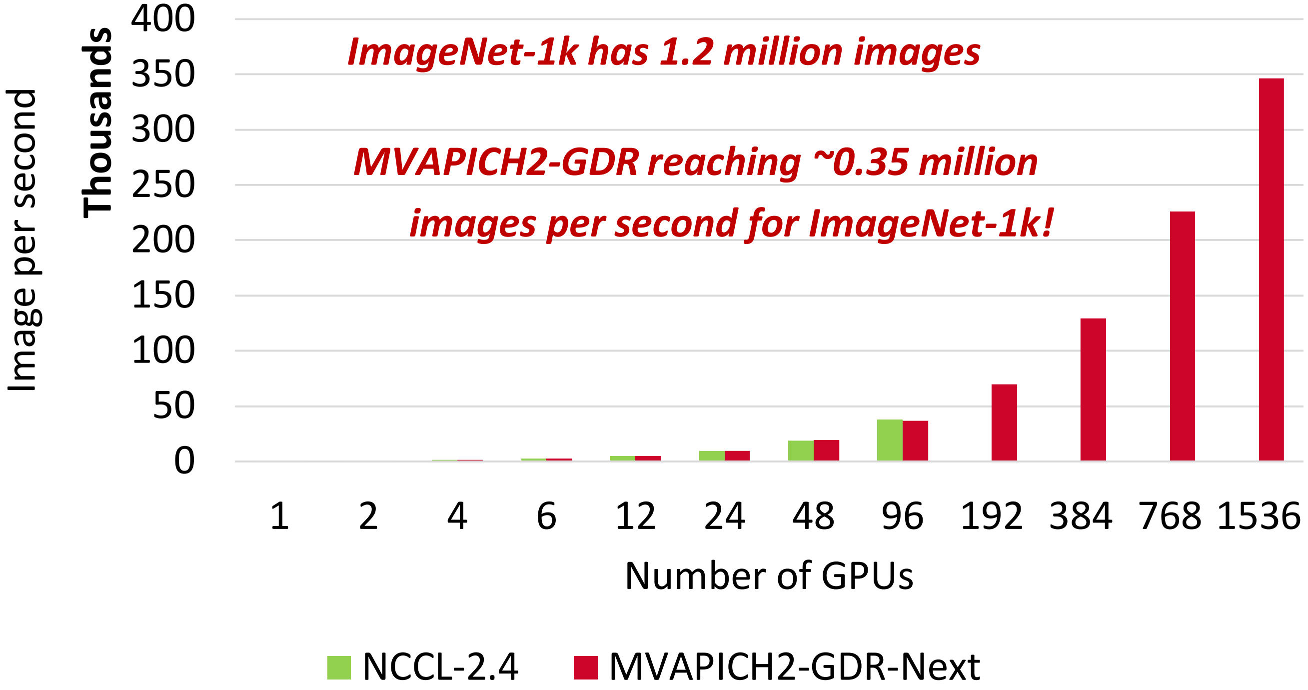




Platform: Nvidia DGX-2 system (16 Nvidia Volta GPUs connected with NVSwitch), CUDA 9.2

Data Parallel Training with TF and MVAPICH2 on Summit

- ResNet-50 Training using TensorFlow benchmark on SUMMIT -- 1536 Volta GPUs!
- 1,281,167 (1.2 mil.) images
- Time/epoch = 3.6 seconds
- Total Time (90 epochs) = 3.6 x 90 = 332 seconds = **5.5 minutes!**

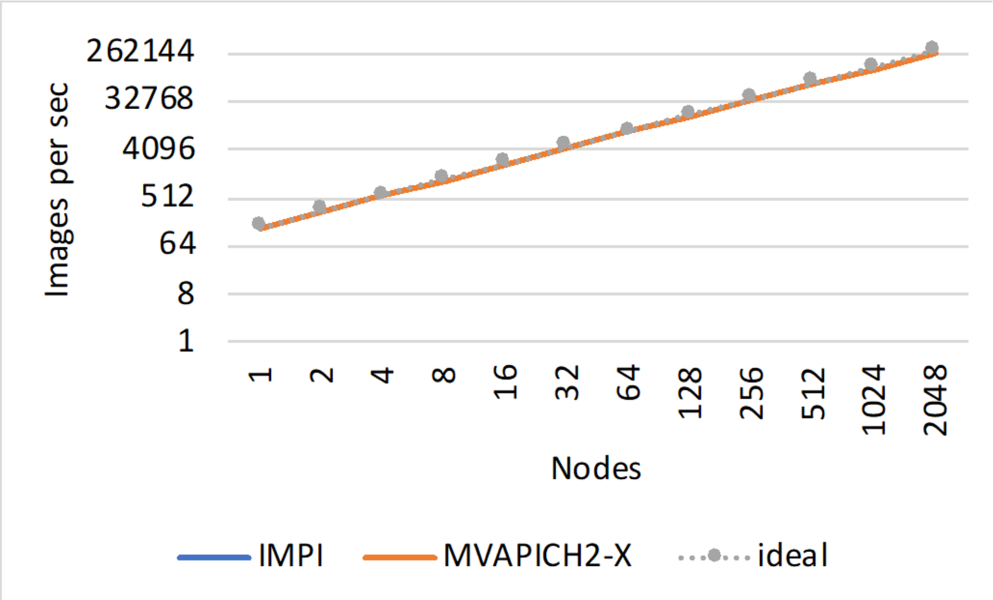


*We observed errors for NCCL2 beyond 96 GPUs

Platform: The Summit Supercomputer (#1 on Top500.org) – 6 NVIDIA Volta GPUs per node connected with NVLink, CUDA 9.2

Data Parallel Training with TF and MVAPICH2 on Frontera

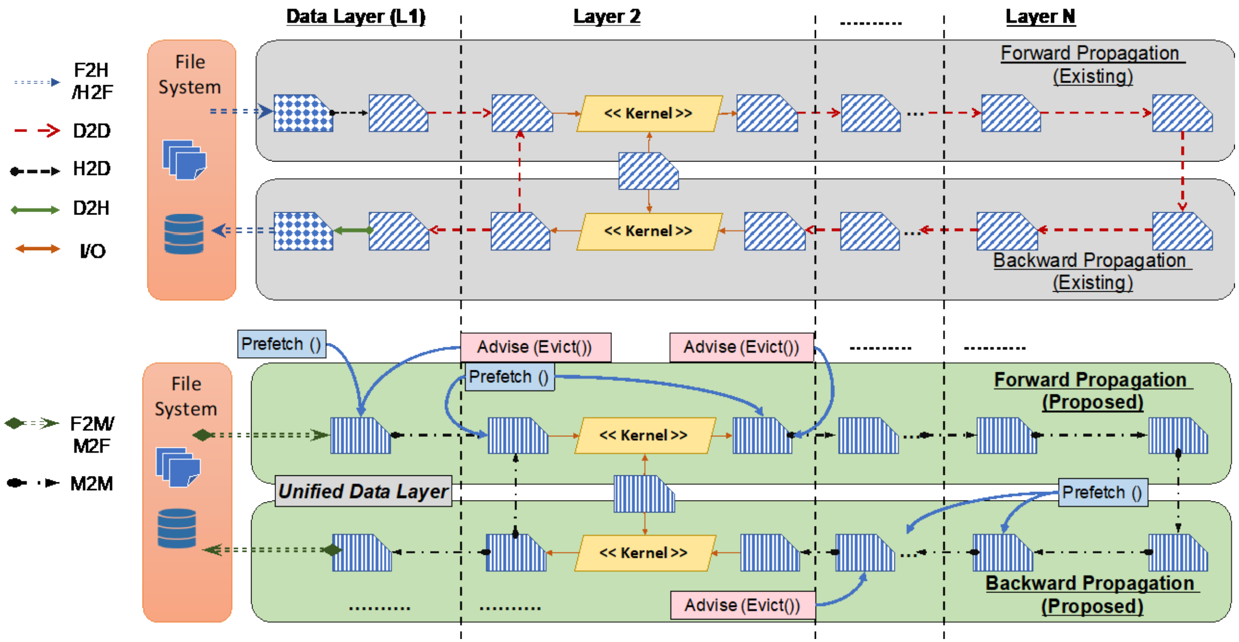
- Scaled TensorFlow to 2048 nodes on Frontera using MVAPICH2 and IntelMPI
- MVAPICH2 and IntelMPI give similar performance for DNN training
- Report a peak of 260,000 images/sec on 2048 nodes
- On 2048 nodes, ResNet-50 can be trained in 7 minutes!



*Jain et al., "Scaling TensorFlow, PyTorch, and MXNet using MVAPICH2 for High-Performance Deep Learning on Frontera", DLS '19 (in conjunction with SC '19).

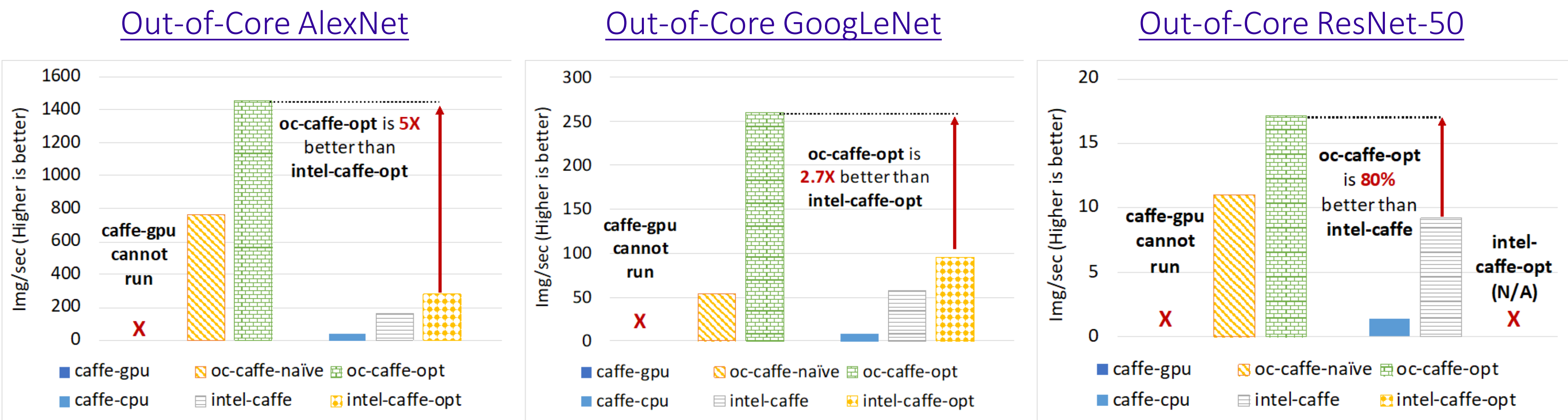
Out-of-core DNN Training

- What if your Neural Net is bigger than the GPU memory (out-of-core)?
  - Use our proposed Unified Memory solution called OC-DNN :-)



A. A. Awan et al., "OC-DNN: Exploiting Advanced Unified Memory Capabilities in CUDA 9 and Volta GPUs for Out-of-Core DNN Training", HiPC '18

Performance Benefits of OC-Caffe

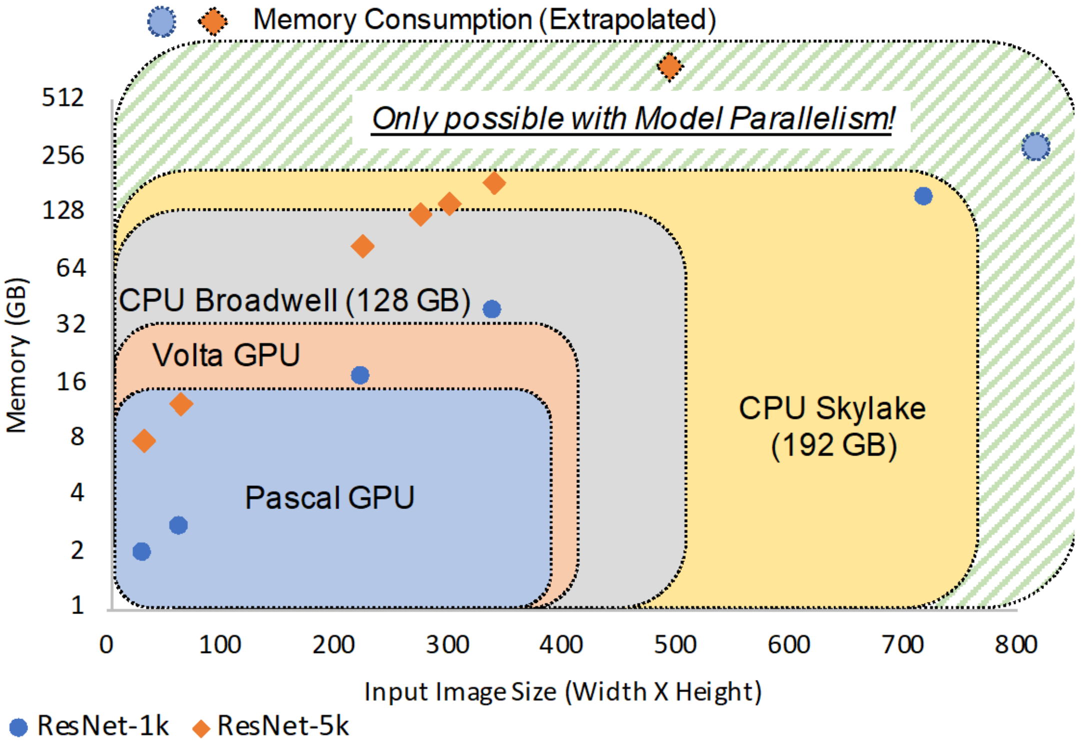


- Out-of-Core workloads – no good baseline to compare
  - Easiest fallback is to use CPU –> A lot more CPU memory available than GPU memory
- OC-Caffe-Optimized (Opt) designs provide much better than CPU/Optimized CPU designs!
  - DNN depth is the major cause for slow-downs → significantly more intra-GPU communication

A. A. Awan et al., "OC-DNN: Exploiting Advanced Unified Memory Capabilities in CUDA 9 and Volta GPUs for Out-of-Core DNN Training", HiPC'18

HyPar-Flow: Hybrid Parallelism for TensorFlow

- Why Hybrid parallelism?
  - Data Parallel training has limits! →
- We propose HyPar-Flow
  - An easy to use Hybrid parallel training framework
    - Hybrid = Data + Mode
  - Supports Keras models and exploits TF 2.0 Eager Execution
  - Exploits MPI for Point-to-point and Collectives

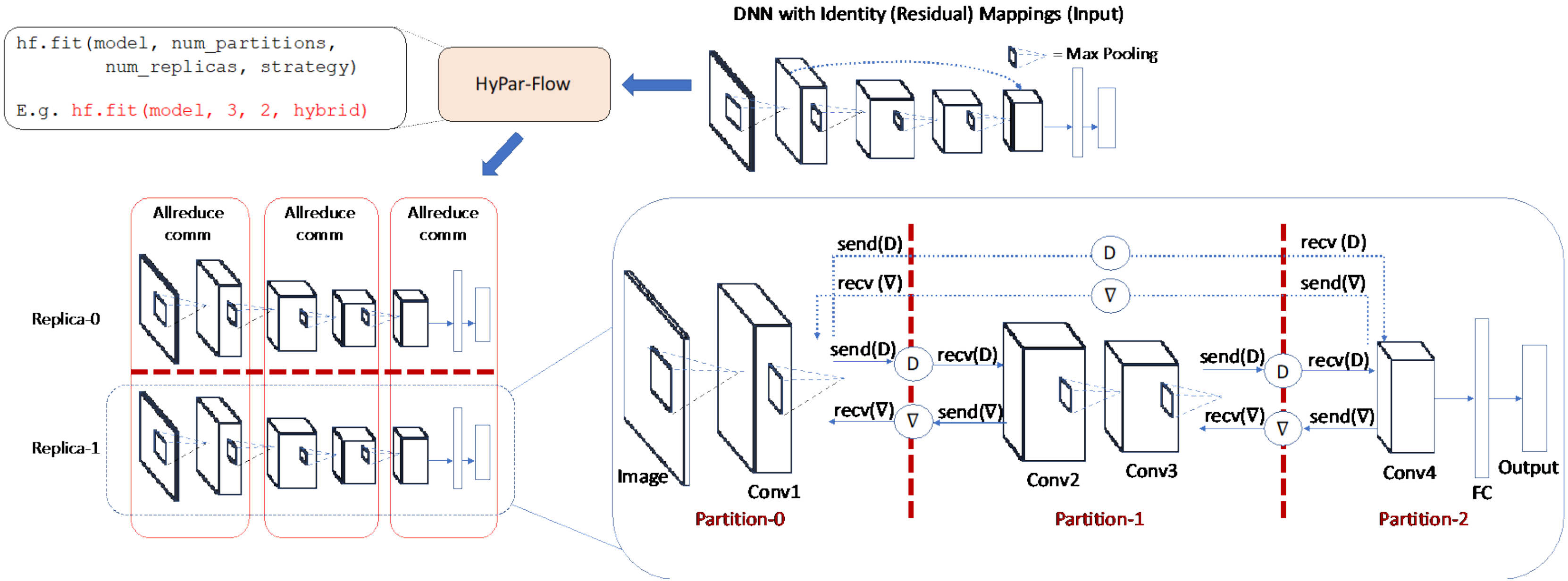


Benchmarking large-models lead to better insights and ability to develop new approaches!

*Awan et al., "HyPar-Flow: Exploiting MPI and Keras for Hybrid Parallel Training of TensorFlow models", arXiv '19. https://arxiv.org/pdf/1911.05146.pdf

HyPar-Flow: Design Overview

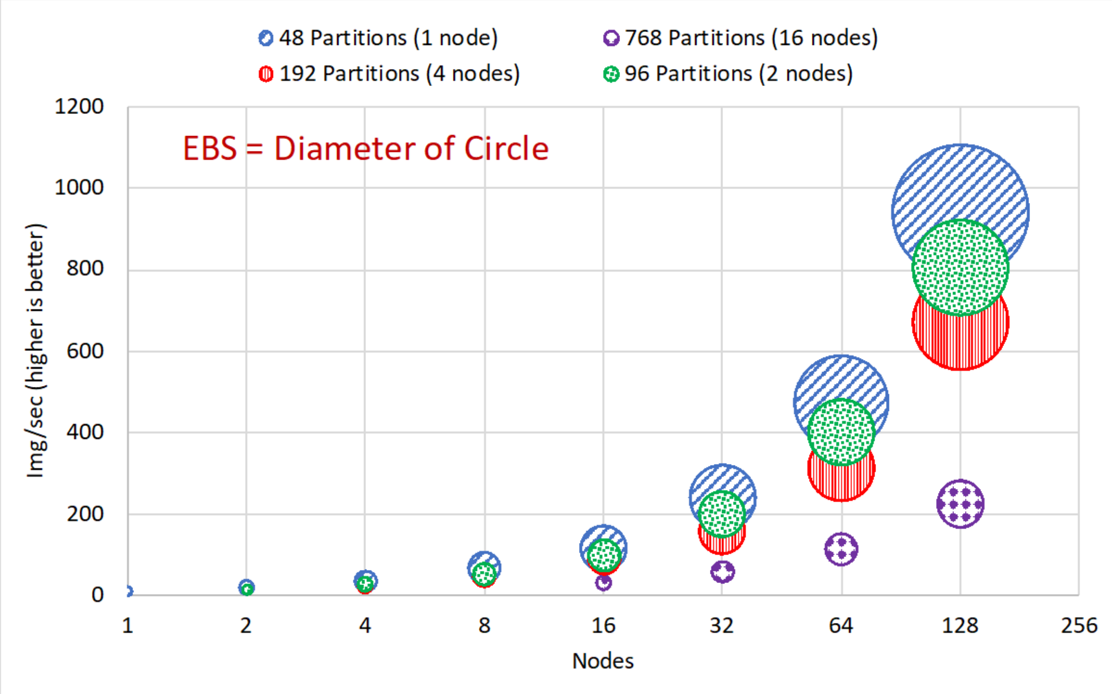
- HyPar-Flow: easy to use Hybrid parallel training framework
  - Supports Keras models and exploits TF 2.0 Eager Execution
  - Exploits MPI Pt-to-pt and Collectives for communication



*Awan et al., "HyPar-Flow: Exploiting MPI and Keras for Hybrid Parallel Training of TensorFlow models", arXiv '19. https://arxiv.org/pdf/1911.05146.pdf

HyPar-Flow (HF): Hybrid Parallelism for TensorFlow

- CPU based results
  - AMD EPYC
  - Intel Xeon
- Excellent speedups for
  - VGG-19
  - ResNet-110
  - ResNet-1000 (1k layers)
- Able to train “future” models
  - E.g. ResNet-5000 (a syntheti
    5000-layer model we
    benchmarked)



110x speedup on 128 Intel Xeon Skylake nodes (TACC Stampede2 Cluster)

*Awan et al., “HyPar-Flow: Exploiting MPI and Keras for Hybrid Parallel Training of TensorFlow models”, arXiv ’19. https://arxiv.org/pdf/1911.05146.pdf

Agenda

- Introduction
- Research Challenges: Exploiting HPC for Deep Learning
- Proposed Solutions
- **Conclusion**

Conclusion

- Deep Learning on the rise
- Single node is not enough
- **Focus on distributed Deep Learning - many open challenges!**
- MPI offers a great abstraction for communication in DNN Training
- A **co-design of DL frameworks and communication runtimes** will be required to make DNN Training highly scalable
- Various parallelization strategies like data, model, and hybrid to address diversity of DNN architectures and Hardware architectures

Thank You!

awan.10@osu.edu

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

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

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

HiDL
High-Performance
Deep Learning

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

MVAPICH
MPI, PGAS and Hybrid MPI+PGAS Library

The High-Performance MPI/PGAS Project
http://mvapich.cse.ohio-state.edu/