Deploying AI Frameworks on Secure HPC Systems with Containers.

24.09.2019| D. Brayford (LRZ)
A. Atanasov, F. Baruffa, W. Riviera (Intel)
S.Vallecorsa (CERN)

SuperMUC-NG (Next Generation)

Specs

- Peak Performance 26.7 Pflop/s
- 719 Tbyte main memory and
- 70 Pbyte disk storage
- 6,480 Lenovo ThinkSystem nodes with Intel Xeon processors (Skylake)
- 311,040 compute cores
- Intel Omni-Path interconnects
- Direct hot water cooled + Adsorption coolers (47 C)

HPC + Cloud

- Usage of own and individual virtual machines (integrated cloud)
- Pre- and post- processing with user's individual software
- Integrated development, ability to use familiar software and tools
- Remote visualization and integration to V2C

A New World is Emerging: High Performance AI (HPAI)

New User Communities with New Workflows

Ability and Expertise to Target Large Scale Problems

HPAI =

M&S

- Equation based on model
- Computing driven
- Numerically intensive
- Creates simulations
- Monte Carlo
- Larger problems
- Iterative methods
- PDE

+

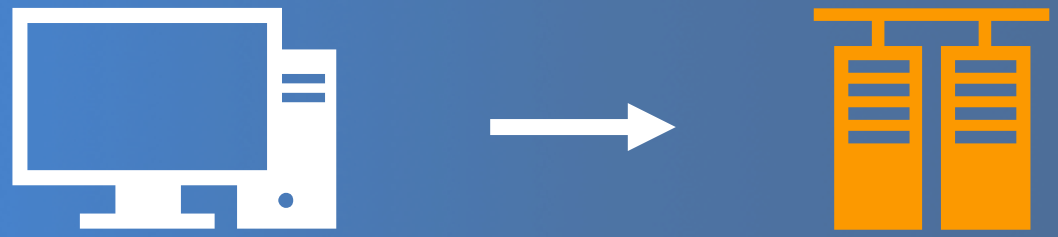
- Linear algebra
- Matrix operations
- Iterative methods
- Compute intensive
- Data transfer
- Predictive
- Probabilities
- Stencil codes
- Calculus
- Pattern recognition
- Graphs

Analytics

- Finds patterns
- Correlations in data
- Logic driven
- Creates inferences
- Knowledge discovery
- Graphs
- Data-driven science
- Predictions
- CNN
- RNN

High Performance AI (HPAI) in a **Container**

Transition AI algorithms from the
laptop to supercomputer
with minimal effort



“It just works”

Differences Between AI & HPC

HPC

- Small number of large files
- Memory per node (32/64GB)
- Multiple nodes
- Distribute compute over many nodes
- Typically diskless systems (no local node storage)
- Data transfer between multiple nodes
- Medium to large matrices
- User privileges

AI

- Large number of small files
- Large memory nodes (+1TB)
- Single node
- Single GPU/accelerator node
- Local node storage
- Data transfer within a single node. (PCI bus)
- Matrices are typically small
- Root privileges

Requirements for AI on HPC

Compute intensive hardware

Optimized AI frameworks
TensorFlow, Caffe

Optimized software
numerical libraries, Python

HPC specific software
distributed computing, workload manager

Method of deploying the AI software
in a simple, straight-forward and flexible way

Need to get to: “It just works”

Key Challenges

Package Management

Frameworks have conflicting dependencies

The frameworks & their dependencies need to be combined in a single module

Rapid update cycles

Provide a mechanism for users to build there own frameworks

Dynamic Programming Environment

Python dependencies

Each unique framework needs its own Python instance

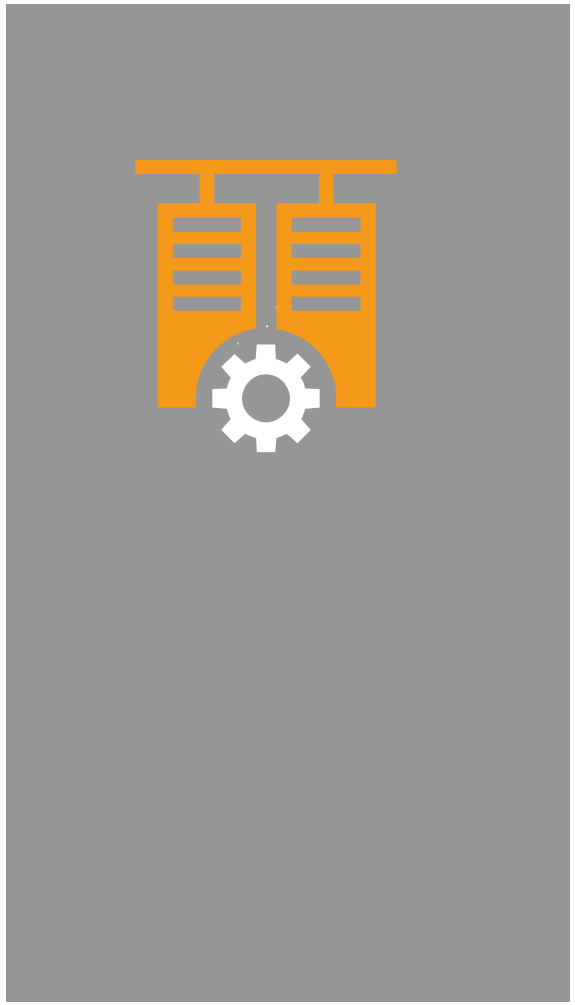
Connecting to external servers

Build frameworks on systems with internet access

Charliecloud containers in HPC

- Easy to install
- Charliecloud was developed to be run on highly secure HPC systems at US government labs
- Charliecloud runs entirely under the User ID
- Ability to run legacy design flows in containers
- Low overhead and ~ 800 lines of code
- LRZ deploys Charliecloud via Spack
- Charliecloud is available in the module system at LRZ

Achieving High Performance AI on Secure HPC Systems

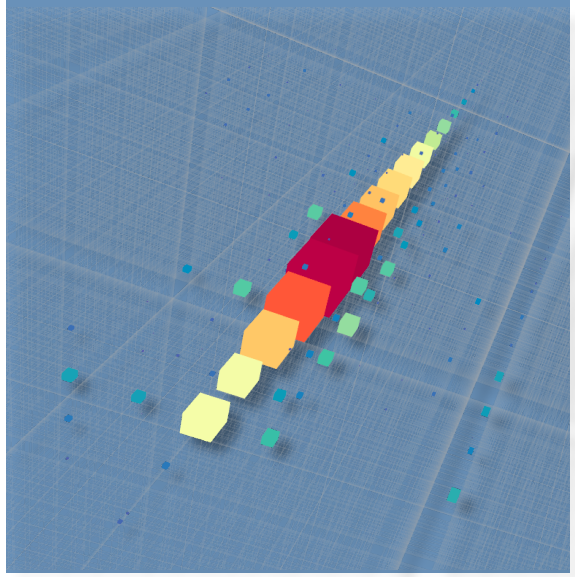


Mechanism for deploying AI at LRZ

- Download the Intel optimized TensorFlow Docker Image (intelaipg Dockerhub)
- Modify the Linux Docker image for HPC
- Modify Python to enable distributed TensorFlow execution
- Copy the training data and execution scripts to the modified Docker image
- Convert to a Charliecloud UDSS and copy the file to the HPC system
- Load the Charlicloud module
- Execute on SuperMUC-NG via Slurm

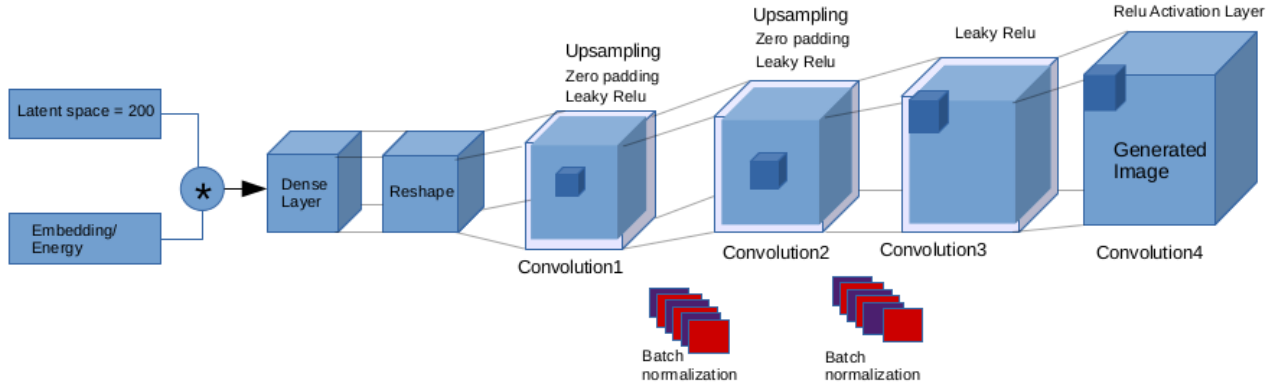
3D Convolutional Generative Adversarial Networks

High Energy Physics detector simulation

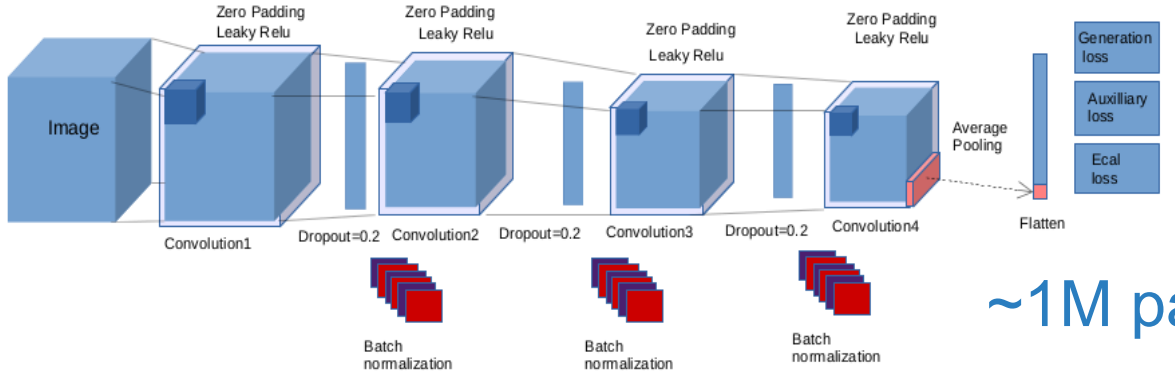


- 3DGAN can be used to generate detector, represented as 3D image

Generator



Discriminator



CERN openlab

~1M parameters

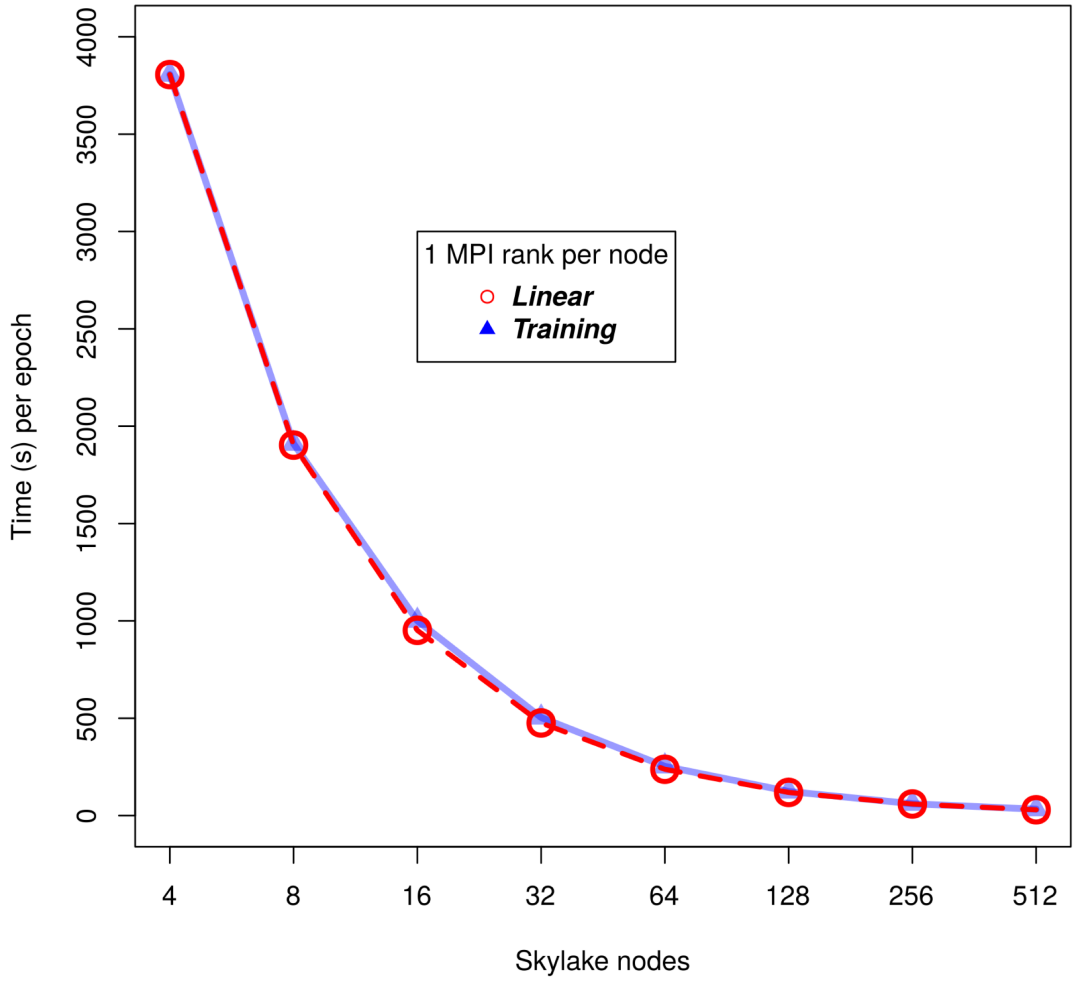
Total model Size: 3.8MB

Distributed TensorFlow Results SNG 1 MPI Rank

1 MPI rank & 48 OpenMP threads per node

Intel Skylake Platinum Xeon 8174

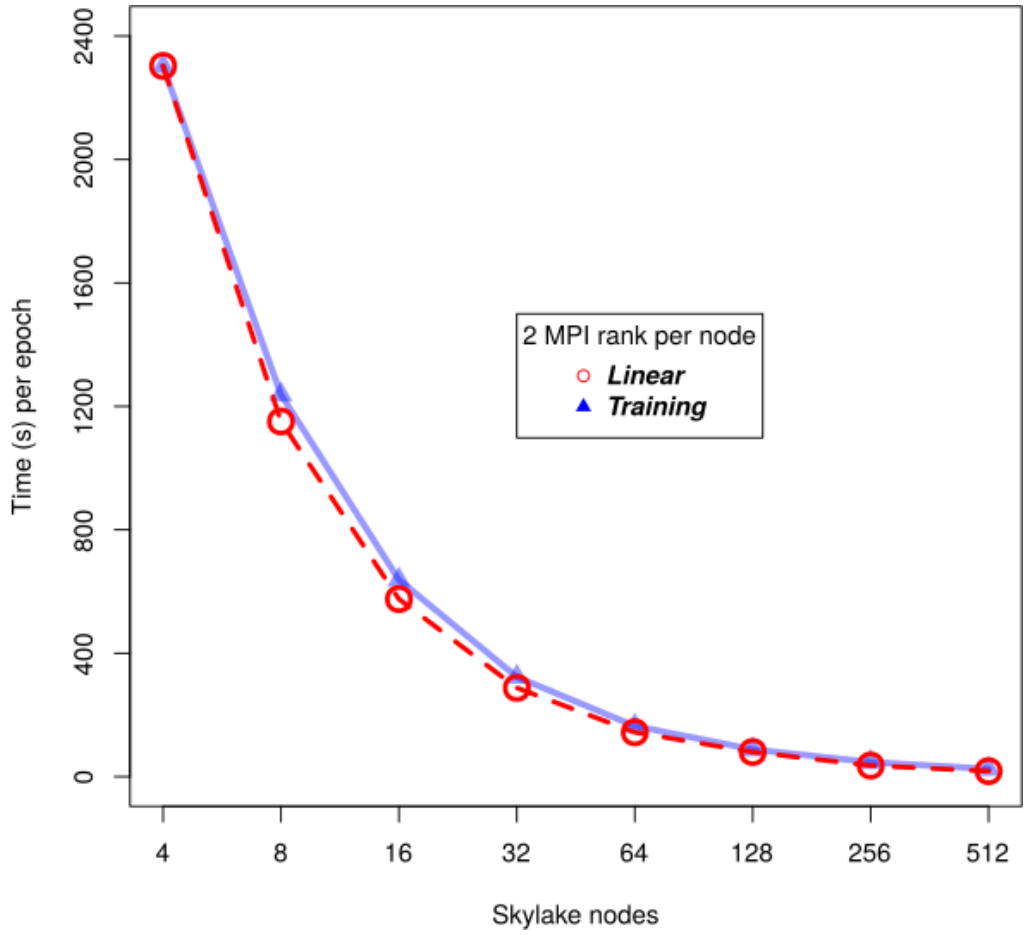
Nodes	Training Time(S) per Epoch	Linear Time(S) per Epoch	Scaling Efficiency
4	3806	3806	-
8	1910	1903	99.6%
16	1001	951.5	95.1%
32	504	475.75	94.4%
64	253	237.87	94%
128	124	118.93	95.9%
256	61	59.46	97.5%
512	33	29.73	90.1%



Distributed TensorFlow Results SNG 2 MPI Ranks & Hyperthreading

2 MPI rank & 48 OpenMP threads per node

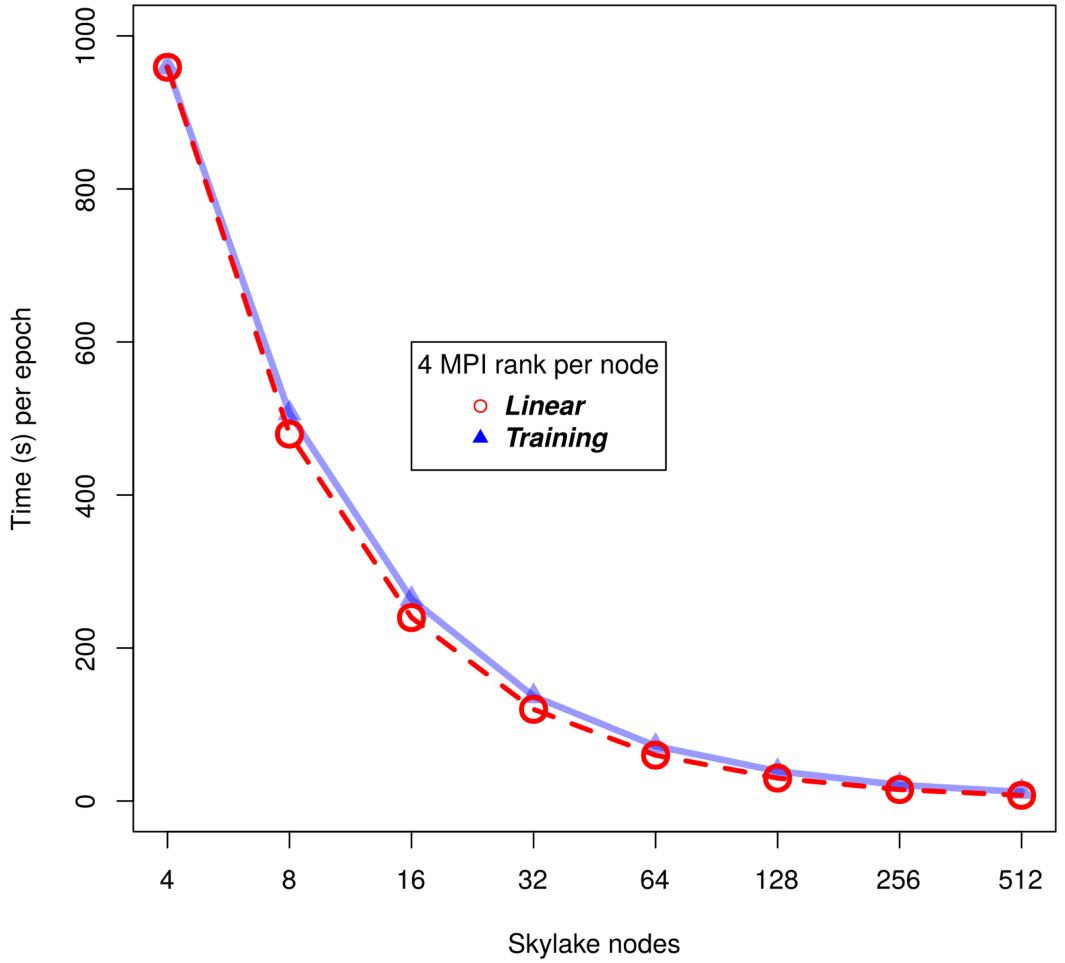
Nodes	Training Time(S) per Epoch	Linear Time(S) per Epoch	Scaling Efficiency
4	2302	2302	-
8	1238	1151	93%
16	638	575.5	90.2%
32	323	287.75	89.1%
64	164	143.87	87.7%
128	88	79.93	81.8%
256	47	35.96	76.6%
512	25	17.98	71.9%



Distributed TensorFlow Results SNG 4 MPI Ranks

4 MPI rank & 12 OpenMP threads per MPI task

Nodes	Training Time(S) per Epoch	Linear Time(S) per Epoch	Scaling Efficiency
4	959	959	-
8	507	479.5	94.6%
16	264	239.75	90.8%
32	137	119.87	87.5%
64	72	59.93	83.3%
128	39	29.96	76.8%
256	21	14.98	71.4%
512	12	7.49	62.5%



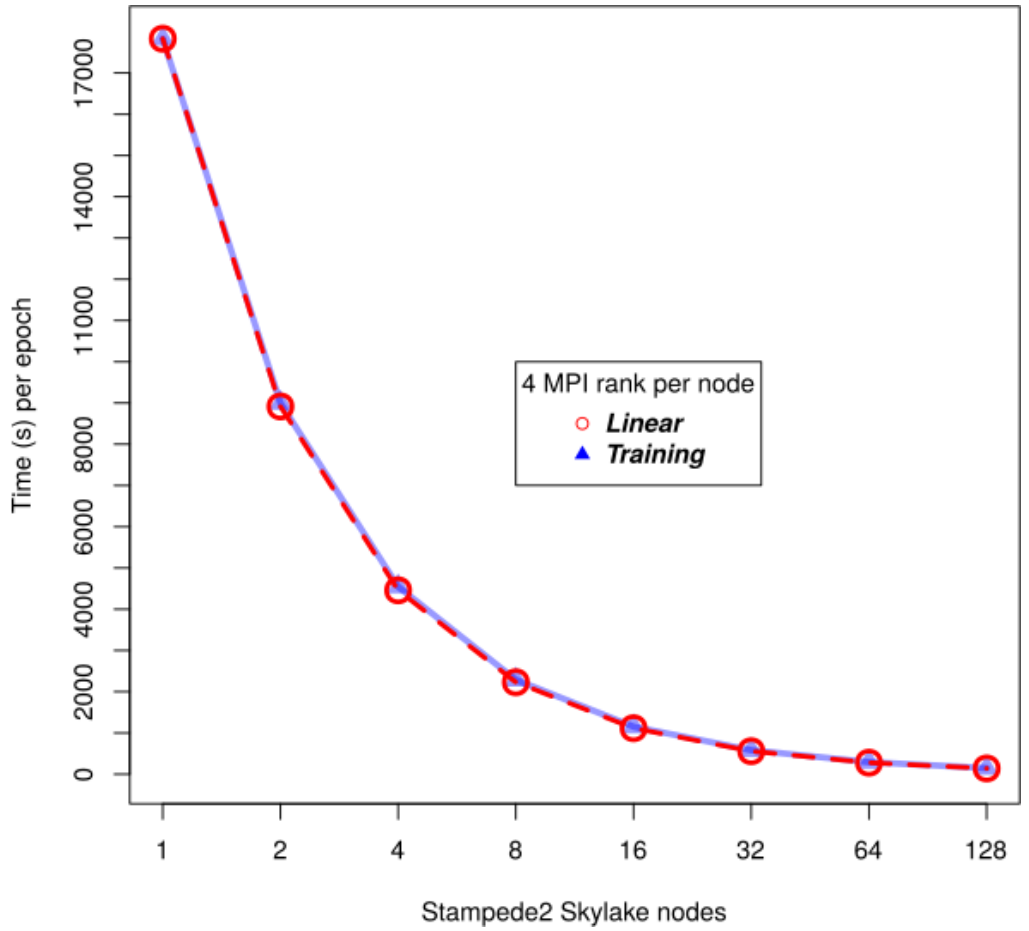
Deployment @ TACC

Distributed TensorFlow Results TACC Stampede2 No Charliecloud

4 MPI rank & 11 OpenMP threads per MPI task

Intel Skylake Platinum Xeon 8160

Nodes	Training Time(S) per Epoch	Linear Time(S) per Epoch	Scaling Efficiency
1	17831	17831	-
2	8998	8915.5	99.1%
4	4545	4457.75	98.08%
8	2288	2228.87	97.4%
16	1151	1114.44	96.8%
32	581	557.22	95.9%
64	293	278.61	95.1%
128	148	139.60	94.1%



Collaboration

Achieving High Performance AI for the community

Intel provides optimized components

- Intel® MPI Library
- Intel® Math Kernel Library
- Intel® Compilers
- Optimized third-party software
- Developer version containing Intel® VTune™ Amplifier and Intel® Advisor

LRZ turns them into vertically integrated containers

- System specific (drivers, runtimes, …)
- User-focused set of software modules
- Semi-automated conversion process
- Focused on latest versions of tool kits
- With packaging expertise for ease-of-use

Together

- Demonstrate vertically integrated solutions on showcase applications highlighting HPAI
- Conduct BoF, presentations, workshops and tutorials
- Create a community-focused HPC/AI benchmark suite
- Container recipes, best practices

Future

Third Quarter 2019

Release SC'19 Denver

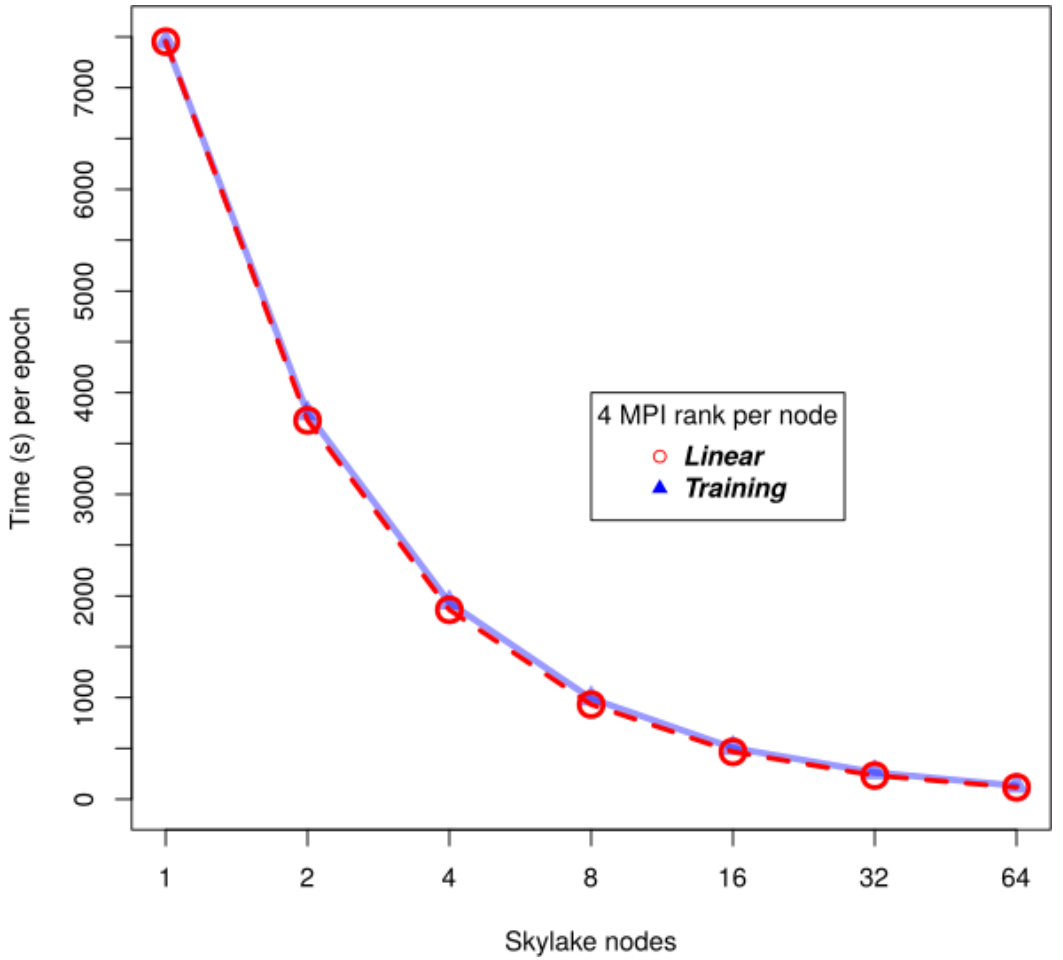
HPC suitable Intel optimized TensorFlow Docker image
Verified recipes to enable the deployment of AI on HPC systems using secure containers
Github repository https://github.com/DavidBrayford/HPAI

Distributed TensorFlow Results Intel Skylake Cluster No Charliecloud

4 MPI rank & 10 OpenMP threads per MPI task

Intel Skylake Gold Xeon 6148

Nodes	Training Time(S) per Epoch	Linear Time(S) per Epoch	Scaling Efficiency
1	7453	7453	-
2	3797	3726.5	98.14%
4	1934	1863.25	96.34%
8	990	931.63	94.1%
16	504	465.81	92.42%
32	263	232.91	88.55%
64	132	116.45	88.22%



Overheads

TABLE II.

Benchmark	TF Throughput with Charliecloud [img/s]	TF Throughput without Charliecloud [img/s]
AlexNet with cifar10	1968	1973
ResNet-50	75	74

Table 2. Achieved throughput [img/s] for AlexNet and ResNet-50 based on Tensorflow 1.11 with and without Charliecloud.

TABLE III.

Benchmark	Free System Memory with Charliecloud [GB]	Free System Memory without Charliecloud [GB]
AlexNet with cifar	331.29	331.33
ResNet50 with imagenet	324.47	324.89

Table 3. Free system memory for AlexNet and Resnet50 based on Tensorflow 1.11 with and without Charliecloud.