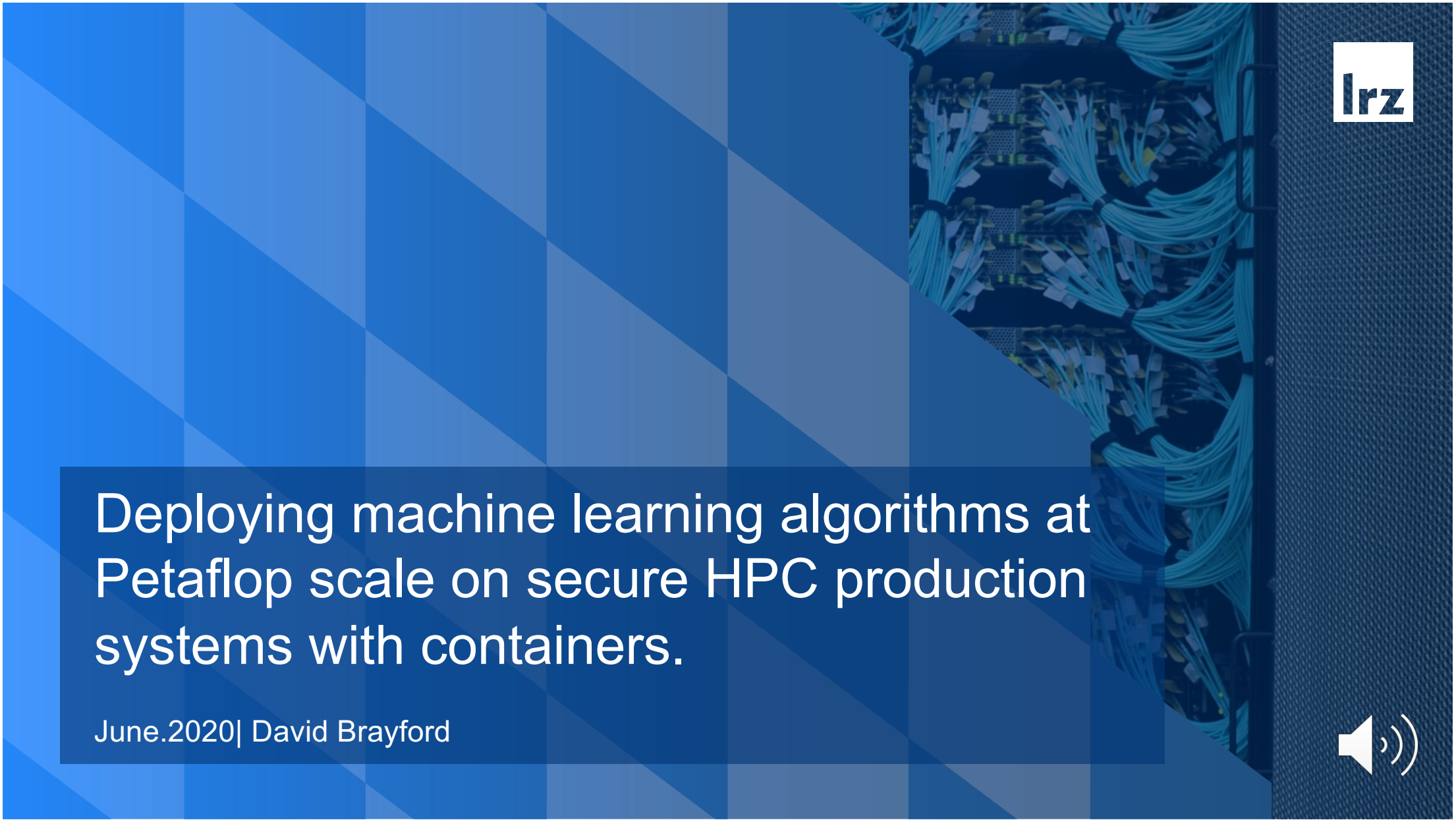
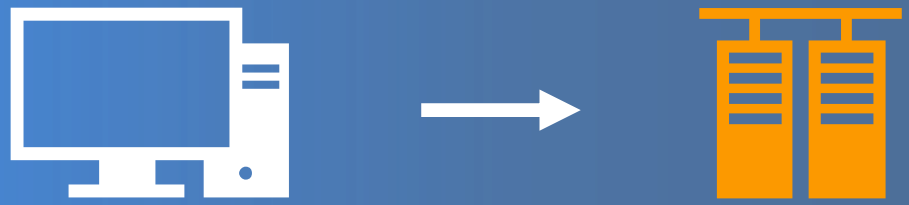


Deploying machine learning algorithms at Petaflop scale on secure HPC production systems with containers.

June.2020| David Brayford

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

Transition AI algorithms from the
laptop to supercomputer
with minimal effort



“It just works”

High Performance AI =

Modeling & Simulation

- 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

Requirements for AI on HPC

Compute intensive hardware

Optimized AI frameworks
TensorFlow, PyTorch, 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”

Distributed Computing

- Data parallelism
  - Same model/network executing, different data
  - Might involve decomposing and distrubting data objects
- Task parallelism
  - Decomposition of the model/network
  - Domain decomposition for the data
- MPI
- OpenMP

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 their own frameworks

Dynamic Programming Environment

Python dependencies

Each unique framework needs its own Python instance

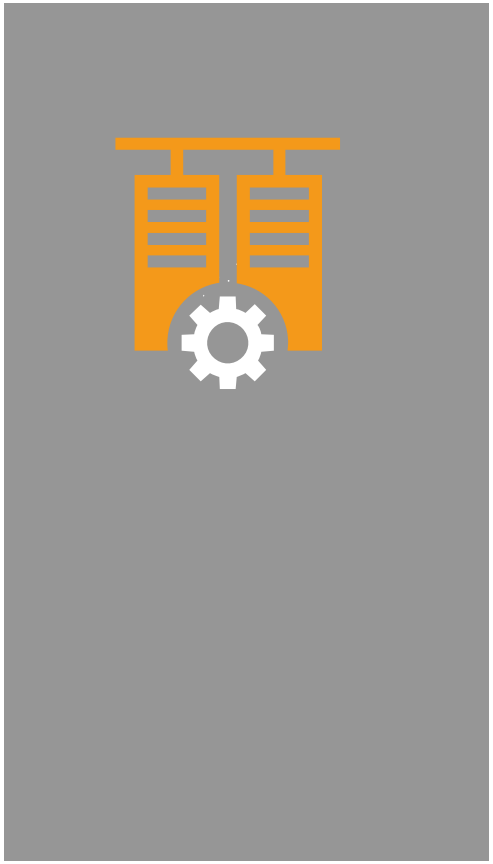
Connecting to external servers

Build frameworks on systems without 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 small number of lines of code
- Easy to install and is available via Spack
- Charliecloud is available in the HPC module system at LRZ

Deploying Containers on Secure HPC Systems at LRZ



Mechanism for deploying Containers at LRZ

- Download or create a Docker Image from a Dockerfile
- If required modify the Docker image on your local system
- Convert to a Charliecloud UDSS and copy the file to the HPC system
- Load the Charliecloud module
- Mount recursively the desired directories you need to access data, libraries and applications
- Execute on SuperMUC-NG or Linux Cluster via Slurm

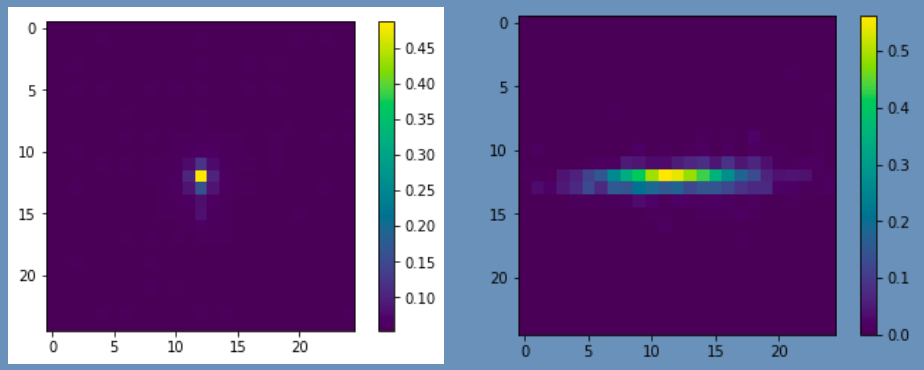
Charliecloud Commands

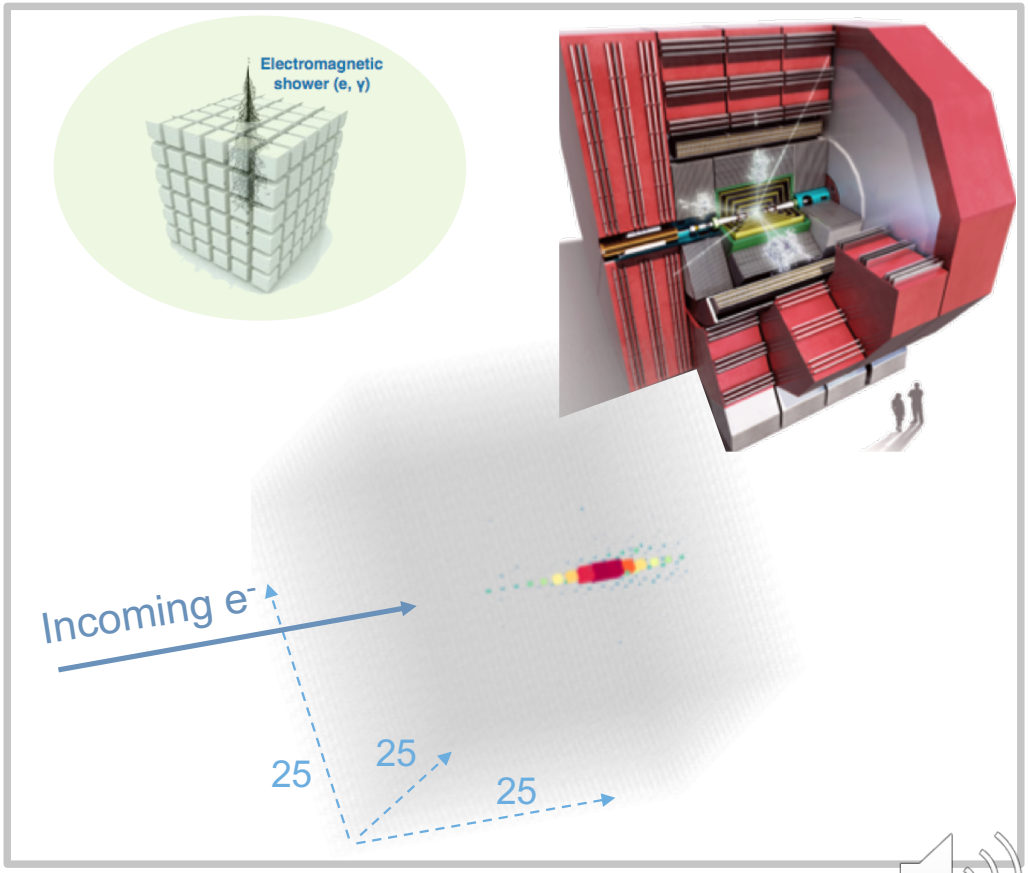
- Load the Charliecloud module:
- Execute the Charliecloud containerized command:
  - ch-run –w <container_name> -- bash
  - ch-run -w <container_name> -- python /model/train.py
  - ch-run -b /lrz/sys/.:/lrz/sys/ -w <container_name> -- bash
- Distributed execution line in a Slurm script:
  - mpiexec -n $SLURM_NTASKS -ppn $SLURM_NTASKS_PER_NODE ch-run -b /lrz/sys/.:/lrz/sys/ -w ./container_name -- python /model/train.py

CERN Compact Linear Collider (CLIC)

Detecting and Identifying High Energy Physic Particles

- CLIC Electromagnetic calorimeter
  - Sparse images
  - Highly segmented (pixelized)
  - Large dynamic range
- Segmentation is critical for particle identification and energy determination



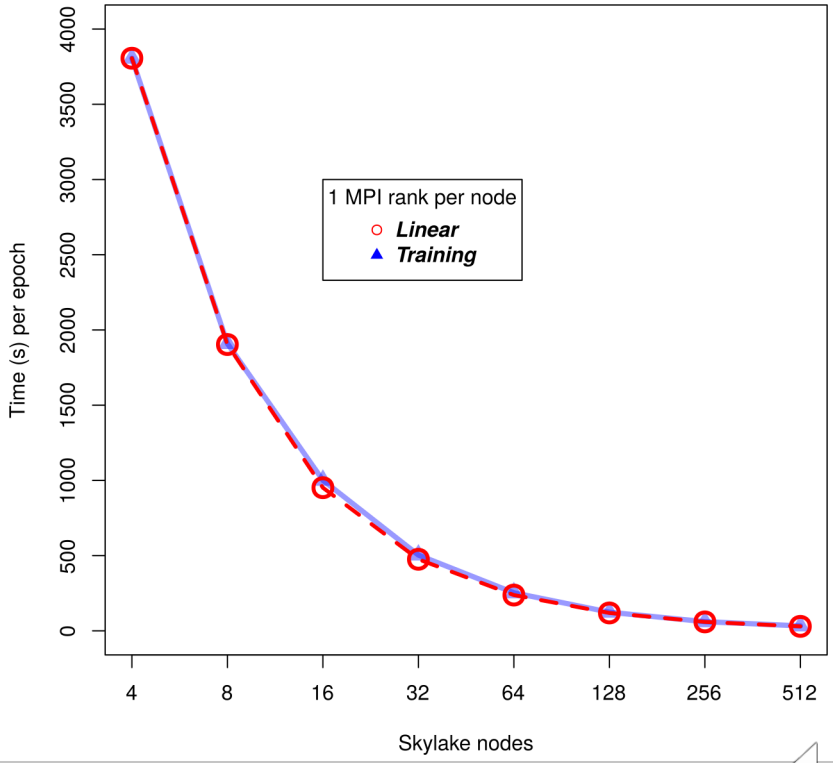


Distributed TensorFlow Results LRZ SNG 1 MPI Rank per Node

1 MPI rank & 48 OpenMP threads per node Intel Skylake Platinum Xeon 8174

- Container OS distribution version of MPICH MPI
- Horovod

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%



Execution on SNG with >= 2 MPI Ranks per Node

Hyperthreading, 48 OpenMP threads per MPI task & 2 MPI ranks per node, standard Horovod + MPICH MPI

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%

12 OpenMP threads per MPI task & 4 MPI ranks per node, standard Horovod + MPICH MPI

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%

Execution on SNG using Intel MPI and vendor network fabric software

Mounted the LRZ file system into the container and used the system version of Intel MPI at runtime.

ch-run -b /lrz/sys/.:/lrz/sys/ -w container_name – python /location/in/container/training_script.py

Nodes	Training Time(S) per Epoch	Linear Time(S) per Epoch	Scaling Efficiency
4	907.26	907.26	-
8	479.52	453.63	94.6%
16	244.42	226.82	92.8%
32	124.22	113.41	91.3%
64	62.24	56.70	91.1%
128	31.22	28.35	90.8%
256	15.63	14.18	90.7%
512	7.84	7.09	90.4%
768	3.94	3.54	89.9%

Nodes	Measured Performance petaflops	Percentage of Theoretical Peak
4	0.01099	66.17%
8	0.02199	66.21%
16	0.04450	67.01%
32	0.08386	63.14%
64	0.17313	65.17%
128	0.31878	67.60%
256	0.70547	66.39%
512	1.39412	65.60%
768	2.08143	65.29%

Beyond 768 nodes the constant set up costs become the dominant factor.

2020

General HPC Docker image
Verified recipes to enable the deployment of AI on HPC systems using secure containers

Current ML frameworks containers supported on SNG

TensorFlow, PyTorch

New Users & Infrastructure

More users; cloud providers; additional ML, AI & data analytics software; different operational modes.

Demonstration

Documentation & Contacts

- **High Performance AI (HPAI)**
- Github repository https://github.com/DavidBrayford/HPAI

- **Online Documentation**
- https://docs.docker.com/
- https://hpc.github.io/charliecloud/tutorial.html

- **Contacts**
- Email : brayford@lrz.de
- LinkedIn : https://www.linkedin.com/in/david-brayford-5900a817/
- Twitter : @david_brayford