OAK RIDGE National Laboratory

HPC and AI Together for Science Campaigns

Wes Brewer
Arjun Shankar
Feiyi Wang
Junqi Yin

Advanced Technologies Section

National Center for Computational Sciences (NCCS)/
Oak Ridge Leadership Computing Facility (OLCF)

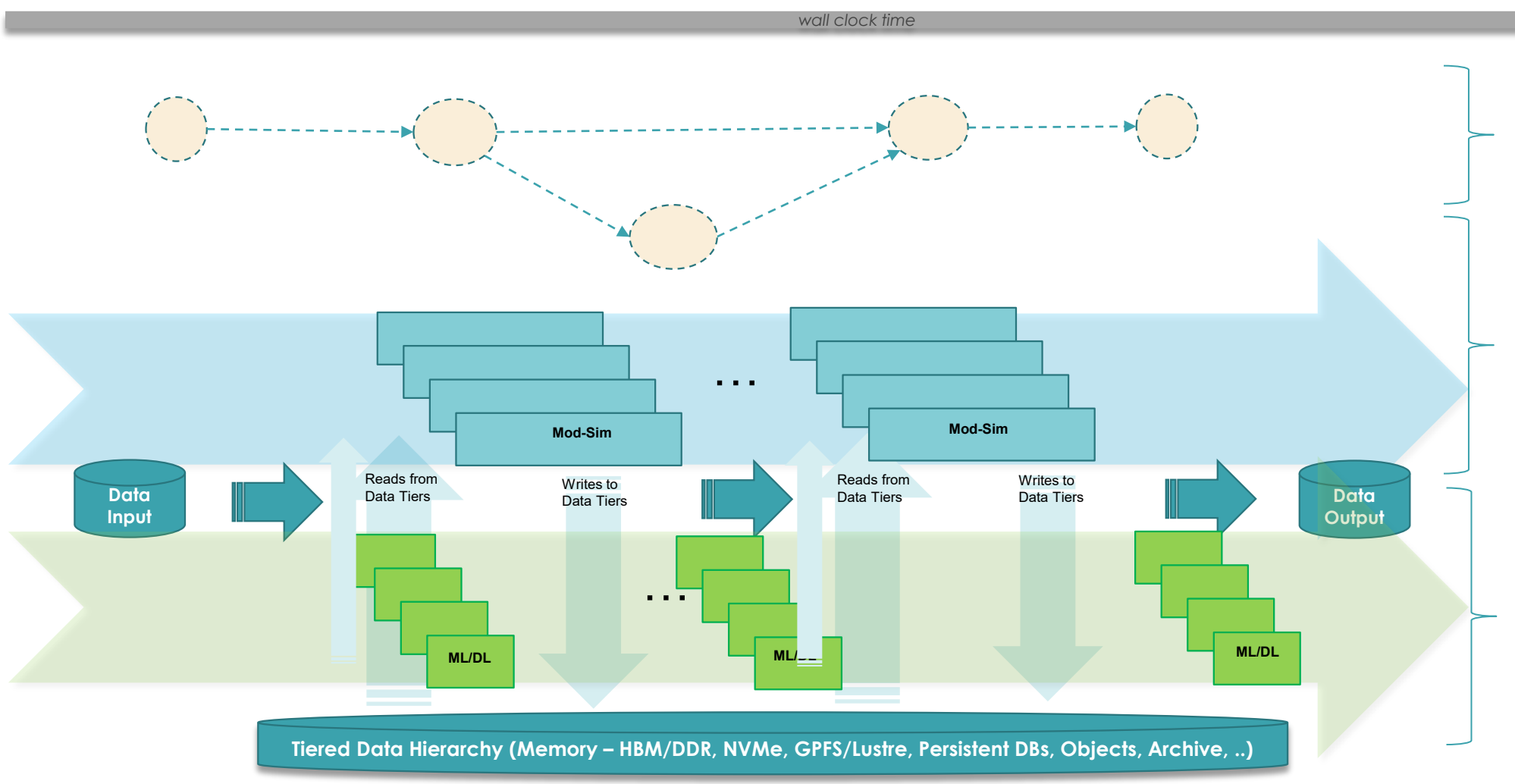
ORNL is managed by UT-Battelle, LLC for the US Department of Energy

U.S. DEPARTMENT OF ENERGY

Agenda

- Opportunities for AI integration with HPC
  - Interleaved HPC and AI
  - In Situ: Co-Operating Mod-Sim and AI Models
  - Ex Situ: Edge-to-Exascale AI workflow vignette
- Specific techniques integrating scalable operations of HPC Mod-Sim and AI
  - Client-Server for AI inference supported campaigns
  - Scaling performance considerations: tight-to-loose coupling performance characteristics

Design Pattern: Interleaved Mod-Sim + ML/DL/AI at Scale



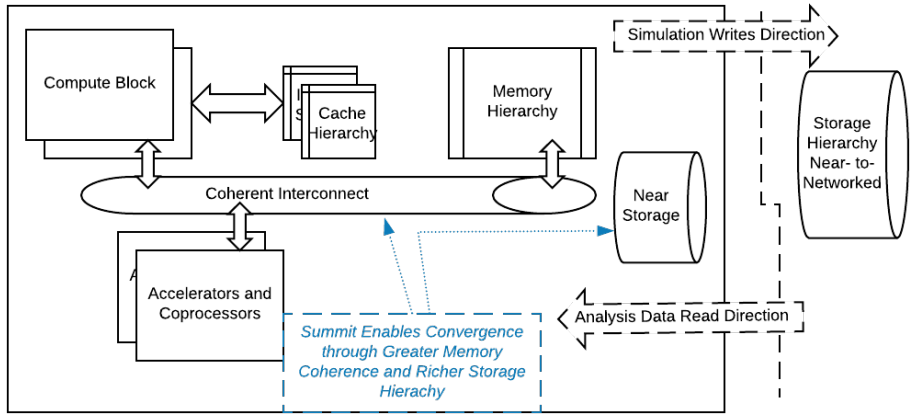
Workflow: Overlay of Intra/Inter-Facility System workflows help connect various phases of the data lifecycle.

Forward Problems: HPC Simulations, sparse, dense solvers, partitioners, mixed precision, etc.

Analysis and Inverse Problems: ML/DL – proxy/surrogates, un/semi/supervised, steering, in-situ state change and viz, repartitioning, mixed precision, etc.

Converged Platform for HPC and AI with Analogs in Learning and HPC Simulation Stacks

Node Layout View



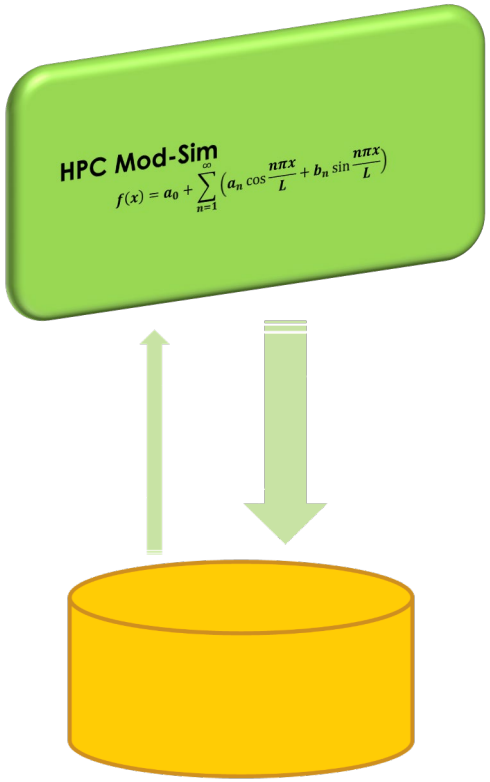
D. E. Womble, M. Shankar, W. Joubert, J. T. Johnston, J. C. Wells and J. A. Nichols, "Early experiences on Summit: Data analytics and AI applications," IBM Journal of Research and Development, vol. 63, no. 6, pp. 2:1-2:9, 1 Nov.-Dec. 2019

Software Stack View

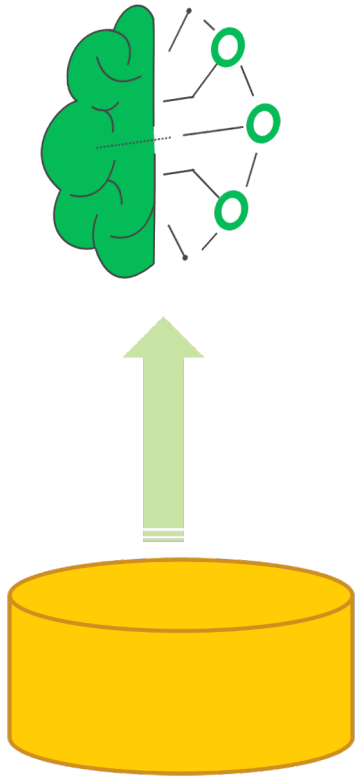
	Learning				Simulation			
Application	MLPerf, DAWNBench, etc	Vision	Language	etc	Climate	Cosmology	etc	HACC, LAMMPS, etc
Framework	TF-CNN-Benchmark, etc	TensorFlow PyTorch	etc	Functionality Scalability	RAJA, Kokkos	etc		Raja-perf-suite,etc
Kernel	DeepBench, HPL-AI, etc	GEMM Conv Comm, etc		Throughput	GEMM LU Comm, etc			SPEC, HPL, etc

Junqi Yin, et al., Comparative evaluation of deep learning workloads for leadership-class systems, Bench Council Transactions on Benchmarks, Standards and Evaluations, 2021, https://doi.org/10.1016/j.tbench.2021.100005.

HPC and AI are often treated separately



Non-AI Campaigns



AI Campaigns

Communication
- Forward/Backward Propagation
- Syncing updates across distributed nodes

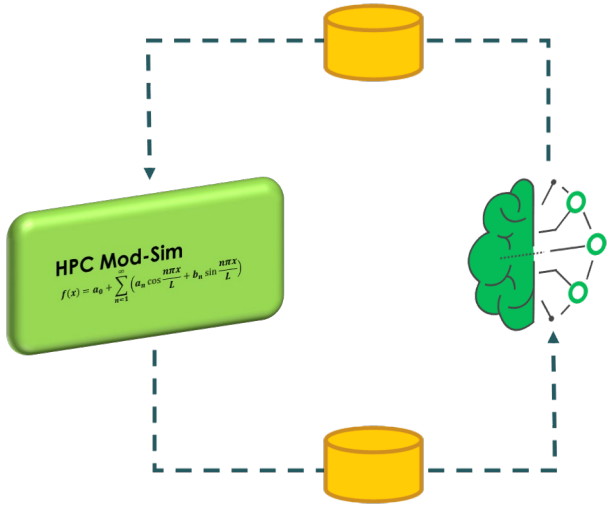
Computation
- Preprocessing, Tokenizing,
- Normalization, Augmentation

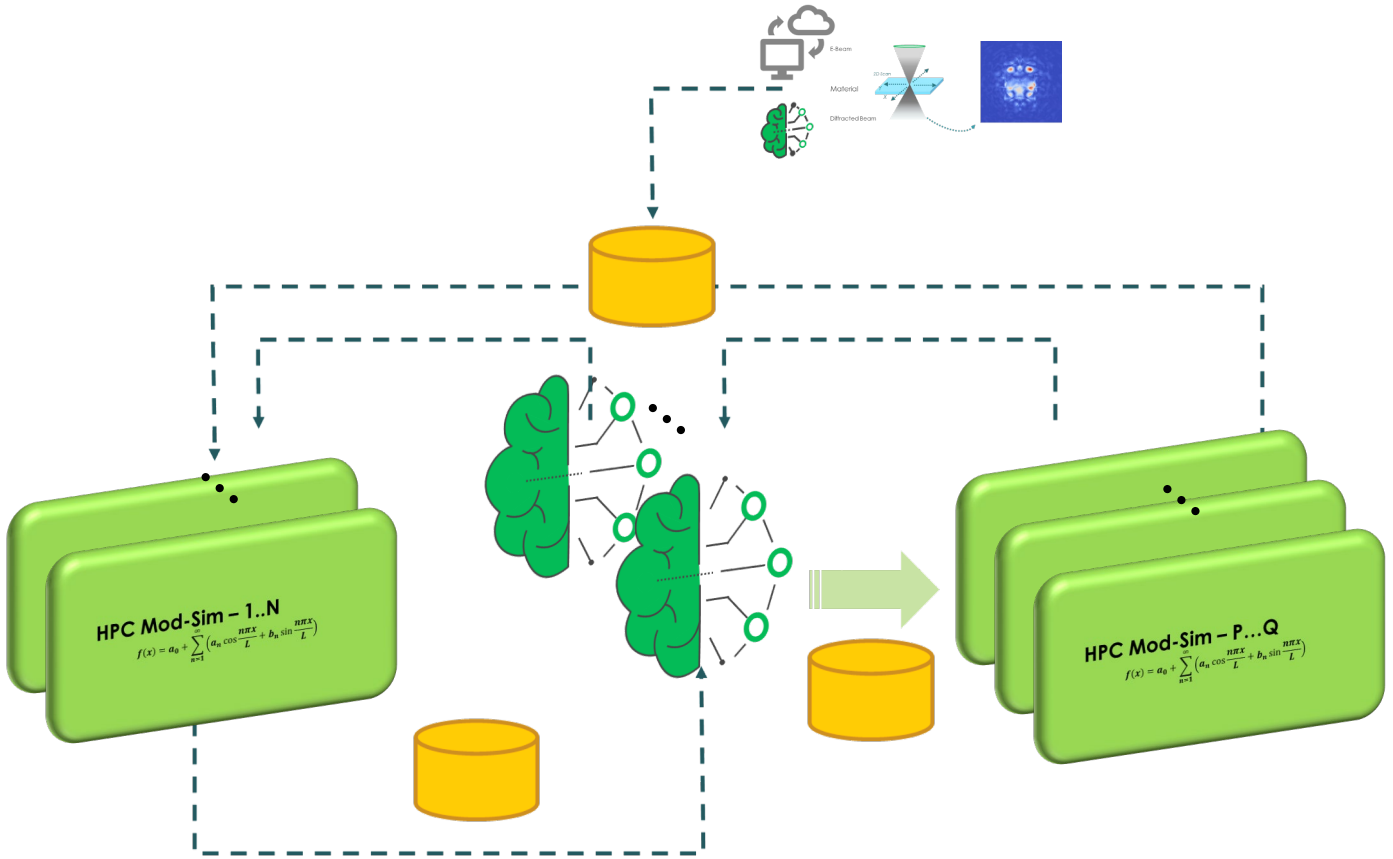
I/O (67% - 85%)
- Loading/Prefecting datasets
- Labels for training, shuffling

HVAC: Removing I/O bottleneck for Large-Scale Deep Learning Applications: A. Khan, et al., IEEE Cluster 2021: https://www.osti.gov/servlets/purl/1902810

Helpful Tips:
- Strategies for Scaling DL: https://doi.org/10.1109/DLS49591.2019.00016
- AI on Frontier: https://olcf.ornl.gov/wp-content/uploads/2-16-23_AIonFrontier.pdf

HPC-AI Usage In Situ





A Survey of AI4Science Workflow Applications, Middleware, and Performance, In preparation, A. Gainaru, S. Jha, F. Wang, F. Suter, W. Brewer, M. Emani, A. Shankar, et al.

Examples:

- High-throughput virtual laboratory for drug discovery: http://dx.doi.org/10.1177/10943420211001565
- Intelligent Resolution: Integrating Cryo-EM with AI-driven Multi-resolution Simulations to Observe the SARS-CoV-2 Replication-Transcription Machinery in Action; https://doi.org/10.1101/2021.10.09.463779

AI Opportunities to Accelerate Campaigns

- Simulation steering

  Example: L. Ward et al., "Colmena: Scalable Machine-Learning-Based Steering of Ensemble Simulations for High Performance Computing," 2021 IEEE/ACM Workshop on Machine Learning in High Performance Computing Environments (MLHPC), St. Louis, MO, USA, 2021, pp. 9-20

- Physics Integrated Deep Learning

  Example: A predictor-corrector deep learning algorithm for high dimensional stochastic partial differential equations; Zhang et al., https://arxiv.org/abs/2208.09883

  Fourier Neural Operator for Parametric Partial Differential Equations, Li et al., https://arxiv.org/abs/2010.08895

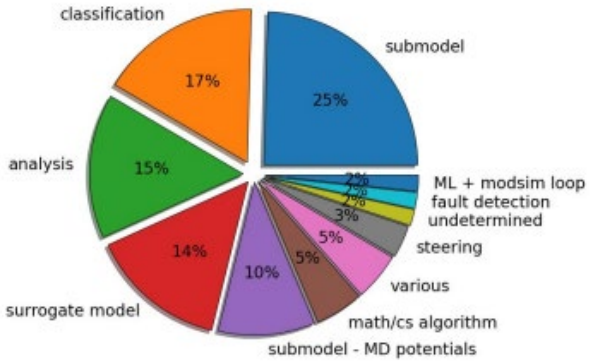
  Commerical tools: NVIDIA SimNet

- Domain-specific surrogates, data reduction

  Example: DeePMD: DeePMD-kit: A deep learning package for many-body potential energy representation and molecular dynamics, https://arxiv.org/abs/1712.03641

SCIENCE APPLICATION AI MOTIFS

Motif	Definition	Example
fault detection	detect algorithmic or other failure in execution, send signal for automatic or manual remediation	detect simulation defect caused by execution error
math/cs algorithm	ML is used to enhance some mathematical (non-science-proper) computation	solver's linear system dimension is reduced based on machine-learned parameter
submodel	a (proper) subset of a science computation is replaced by an ML model. molecular dynamics (MD) potentials as special case	physics-based radiation model in a climate code replaced by ML model
steering	automatic steering of the direction of a computation for some internal process	ML method to guide Monte Carlo sampling to include undersampled regions
surrogate model	full science model replaced by ML approximation that captures important aspects, used for speed or science understanding	data from tokamak simulation runs used to train surrogate model
analysis	results from modeling and simulation (modsim) runs are analyzed by a human using ML methods	use graph neural networks to analyze results of MD simulation
ML + modsim loop	both ML and traditional modsim, coupled	MD in loop used to refine deep learning model via active learning
classification	"pure" ML with little or no modsim used to classify some phenomenon; includes some other methods like reinforcement learning	deep neural network inference to detect rare astrophysical event
various	umbrella project with multiple unrelated subprojects using possibly different kinds of AI/ML	CAAR/ESP/NESAP application readiness
undetermined	manner of AI/ML use is undetermined	project is exploring AI/ML use but gives no details



Learning to Scale the Summit: AI for Science on a Leadership Supercomputer, Joubert W., et al., 2022, IPDPSW; https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=9835678

Example: AI Surrogates Reducing Costly Calculations

MC-based Exploration of High-Entropy Alloy System (MoNbTaW)

- Probability of *N* atoms in configuration *X* at temperature *T* follows Boltzmann's distribution $exp(-E(X)/k_BT)$ where *E* is the total configuration energy and k_B is the Boltzmann constant.
- Replica of the alloy systems at various *T* are simulated via replica exchange Monte Carlo simulations with transition probability between replica *m* and *n*:

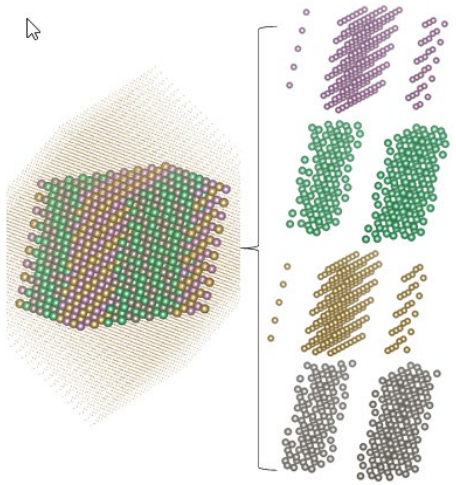
$$W(\{X_m, T_m\}|\{X_n, T_n\}) = \min\left[1, \exp(-\triangle)\right],$$

where

$$\triangle = (1/k_BT_n - 1/k_BT_m)(E(X_m) - E(X_n))$$

- Atoms *(i,j)* are exchanged with acceptance probability $P_{i,j}$ proportional to:

$$\min\left[1, \exp(-(E(\{x_i, x_j\}) - E(\{x_j, x_i\}))/k_BT)\right]$$



https://doi.org/10.1038/s43588-021-00139-3

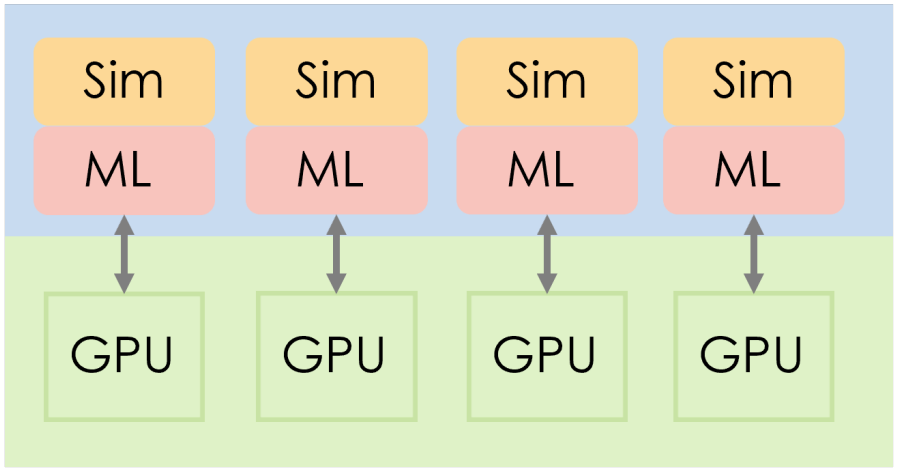
DEEP LEARNING SURROGATE MODELS FOR THE ENERGY EVALUATION OF MONBTAW ALLOY. THE ARCHITECTURE IS GIVEN BY THE NUMBER OF NODES IN EACH HIDDEN LAYER, AND IT IS THE SAME FOR EACH OF THE FOUR ELEMENT. THE R^2 SCORE IS THE AVERAGED MODEL PERFORMANCE OF ALL ELEMENTS.

name	architecture	# parameters	R^2 score	
			FP32	mixed
tiny	24-24	5,257	0.991	0.988
small	128x2-64x2 -24x2	55,817	0.991	0.992
mdedium	200x11	440,801	0.994	0.992
large	512x6-256x6 -128x6-64x6-32	2,019,009	0.994	0.993

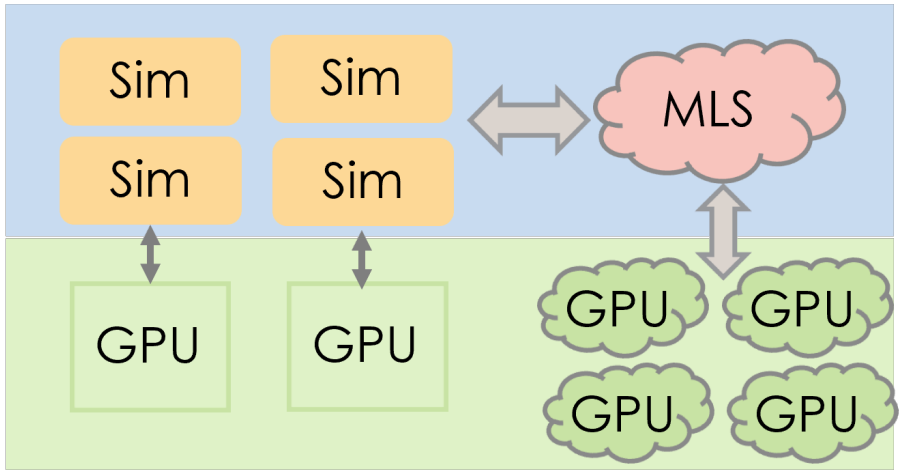
J. Yin, F. Wang, A. Shankar, Strategies for Integrating Deep Learning Surrogate Models with HPC Simulation Applications, IPDPSW 2022. https://ieeexplore.ieee.org/document/9835386

Deployment Architectures

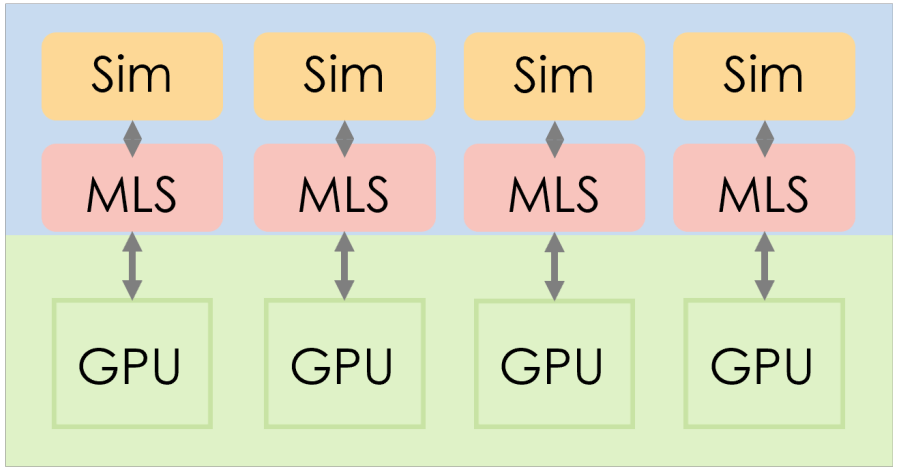
Tightly coupled

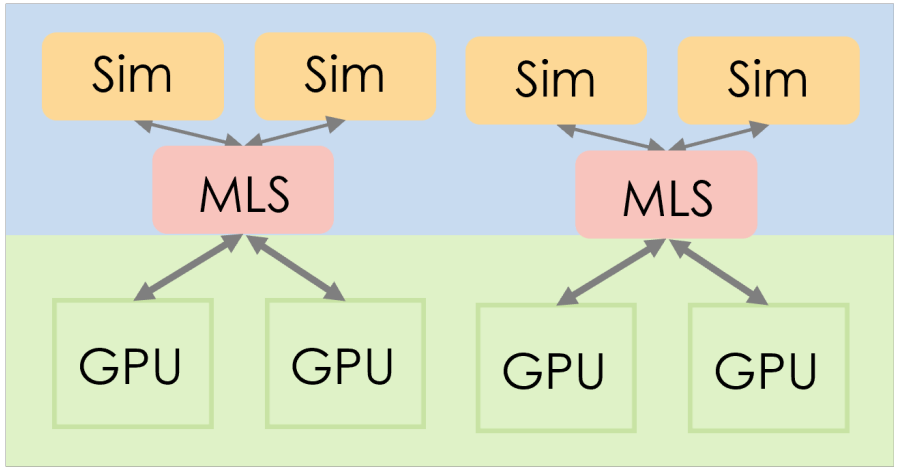


Loosely coupled

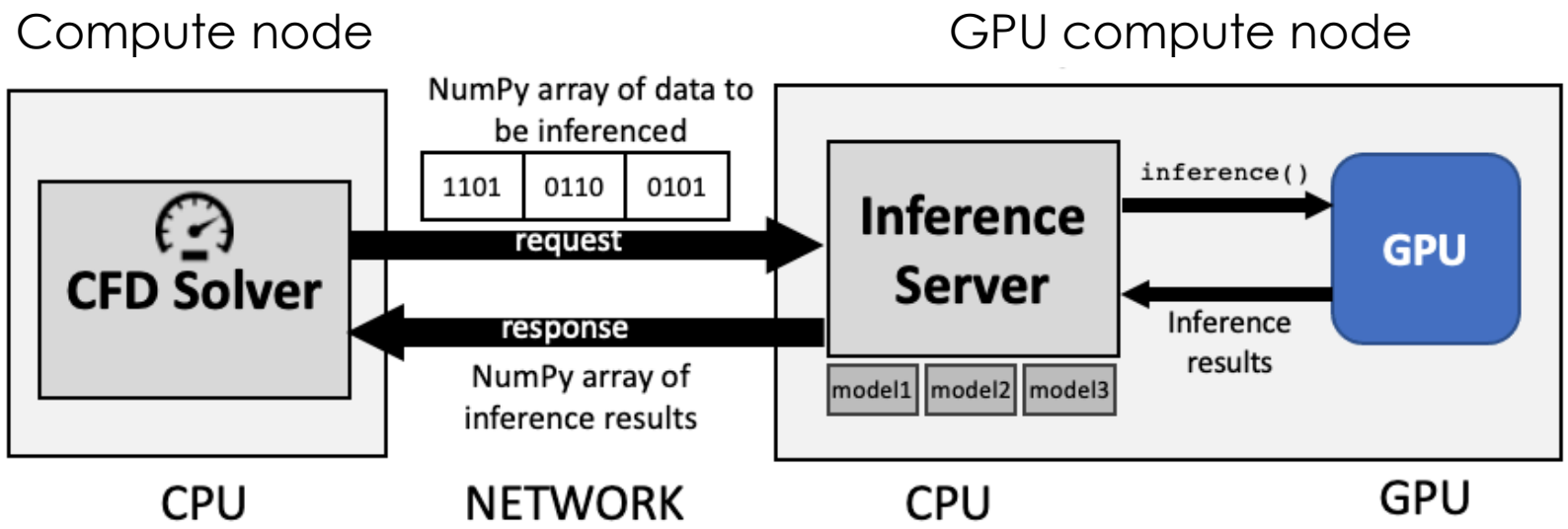


Semi tightly coupled





HPC-AI for Inference on Summit



TensorFlow serving ↔ gRPC

redisAI ↔ RESP

Brewer, Wesley, et al. "Production deployment of machine-learned rotorcraft surrogate models on HPC." *MLHPC'21*.

https://code.ornl.gov/whb/osmi-bench

TF Serving on Summit

Export TF Model

```
from tensorflow.keras.models import Model
model = Model(…)
model.save("/path/to/mymodel/1")
```

Start TF Serving

- module load open-ce/1.1.3-py38-0
- module load cuda/11.0.2
- tensorflow_model_server [--port 8500] --model_base_path=/path/to/mymodel --model_name="mymodel"
- lsof -i :8500 # check that TF Serving is running and listening to port 8500

Test implementation

- saved_model_cli show --all --dir 1
- python client.py

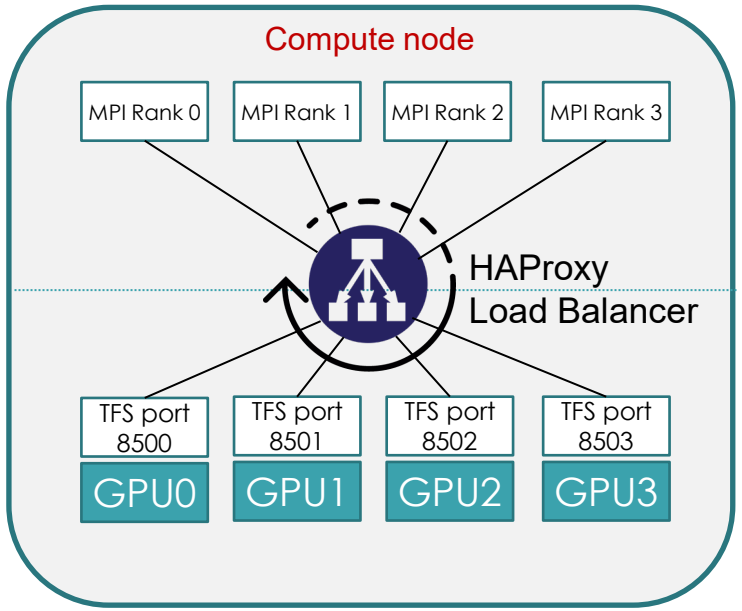
client.py

https://code.ornl.gov/whb/osmi-bench

```
import grpc
import tensorflow as tf
from tensorflow_serving.apis import predict_pb2
from tensorflow_serving.apis import prediction_service_pb2_grpc
# set parameters
batch_size = 32
input_shape, output_shape = (batch_size, 8, 48), (batch_size, 2, 12)
mname, sname = 'mymodel', 'serving_default' # names of model and signature
iname, oname = 'inputs', 'dense_3' # names of input and output layers
# create some random data and perform inference
data = np.array(np.random.random(input_shape, dtype=np.float32])
channel = grpc.insecure_channel('localhost:8500')
stub = prediction_service_pb2_grpc.PredictionServiceStub(channel)
request = predict_pb2.PredictRequest() # request proto data structure
request.model_spec.name = mname
request.model_spec.signature_name = sname
request.inputs[iname].CopyFrom(tf.make_tensor_proto(data, shape=input_shape)
result = stub.Predict(request) # send request
# extract results
result_array = np.reshape(result.outputs[oname].float_val, output_shape)
```

Approaches for Scaling Up

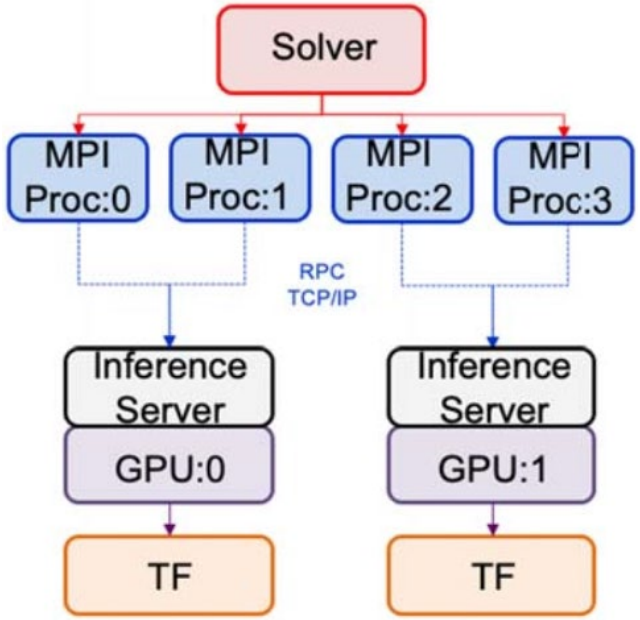
HAProxy Load Balancer



> singularity pull docker://haproxy

Brewer, Wesley, et al. "Production deployment of machine-learned rotorcraft surrogate models on HPC." *MLHPC'21*.

MPI

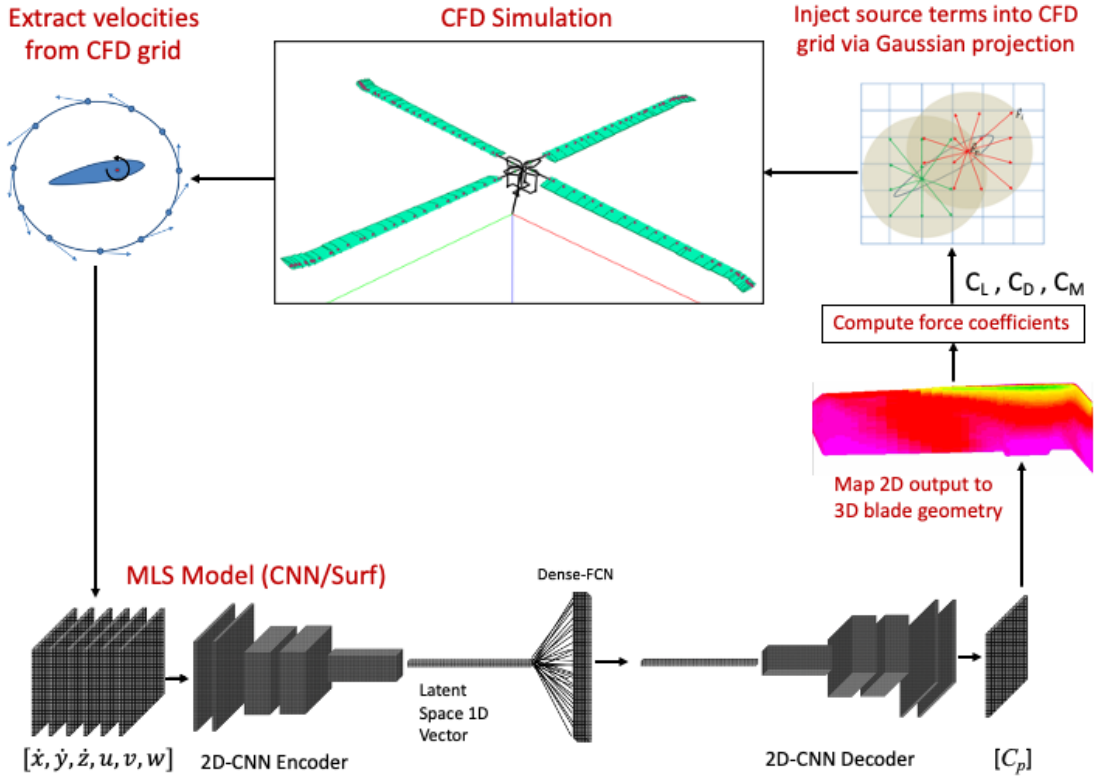


Boyer, Mathew, et al. "Scalable Integration of Computational Physics Simulations with Machine Learning." *AI4S'22.*

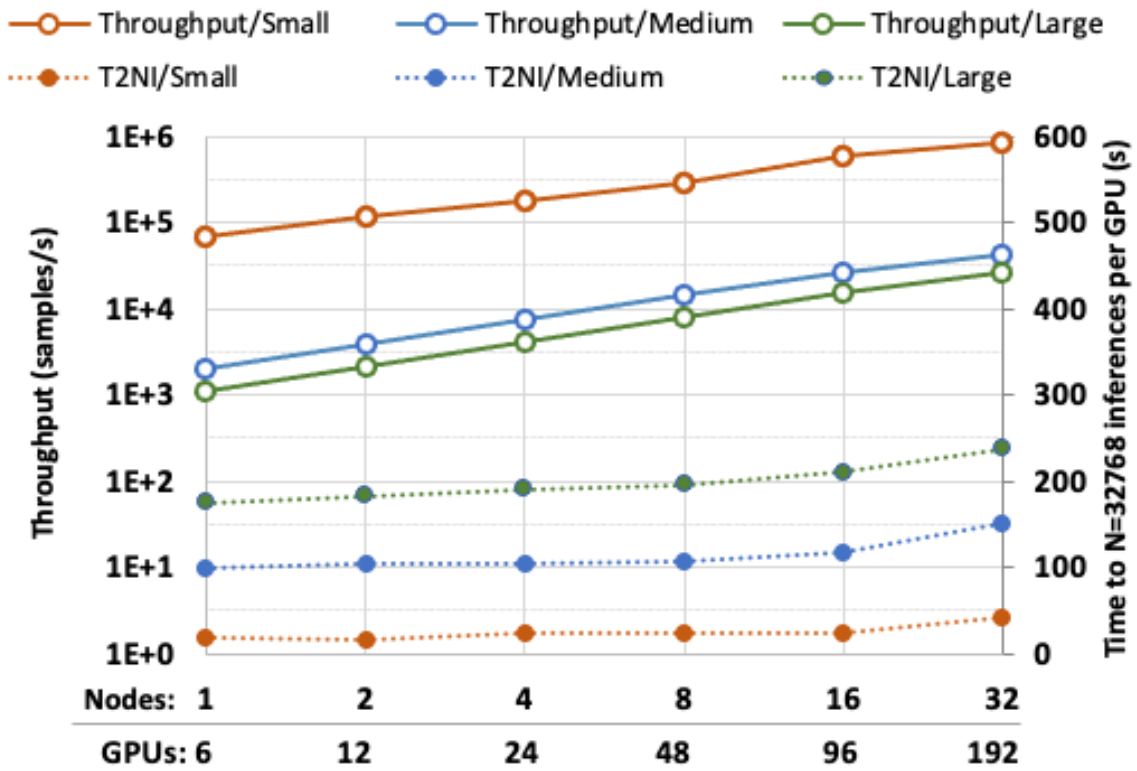
Example: Rotorcraft Aerodynamics

1E6 inferences per second

Workflow



Weak scaling performance on Summit



Brewer, Wesley, et al. "Production deployment of machine-learned rotorcraft surrogate models on HPC." *MLHPC'21*.

Strategy:

Maximize batch size,

Concurrency = 2

Example: Machine-learned Boundary Conditions

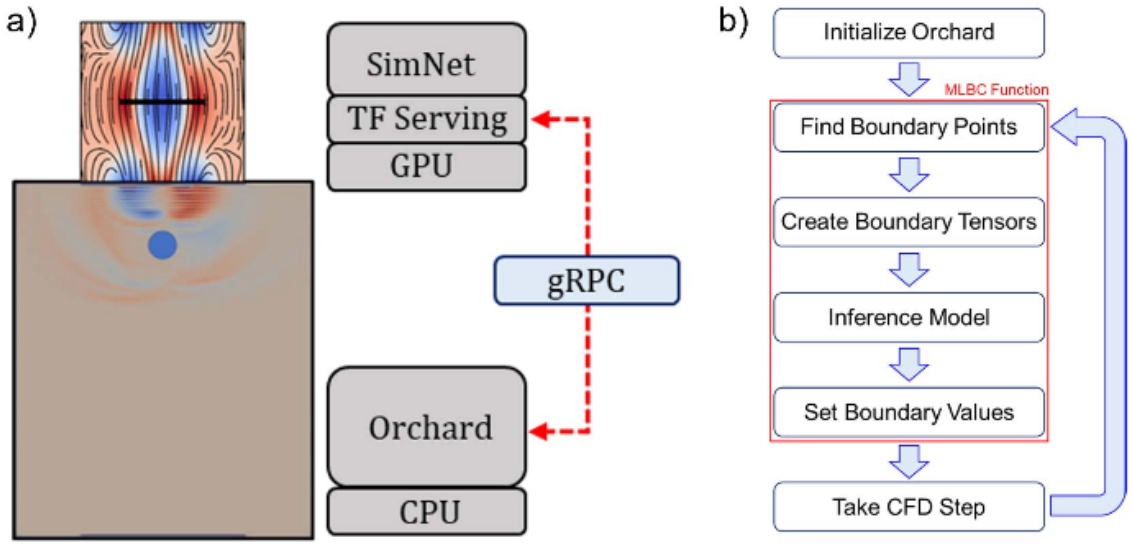


Fig. 4. a) Machine-learned boundary condition implementation design. b) Flowchart for machine-learned boundary condition implementation in Orchard.

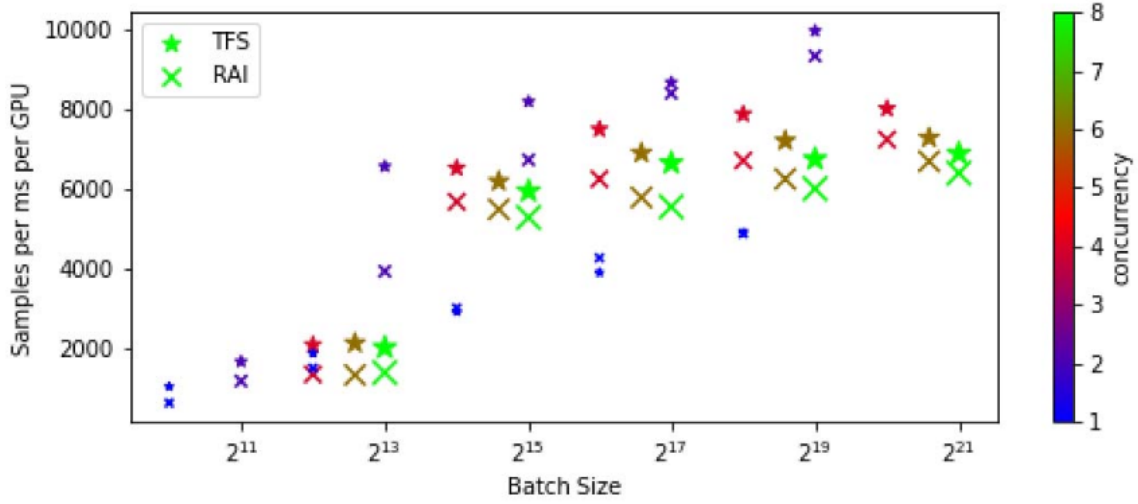


Fig. 5. Per-GPU throughput of TF Serving (TFS) and RedisAI (RAI) C++ clients for a SimNet fully connected neural network with about 100k parameters.

Boyer, Mathew, et al. "Scalable Integration of Computational Physics Simulations with Machine Learning." *AI4S'22.*

Rankings for RedisAI vs. TF Serving

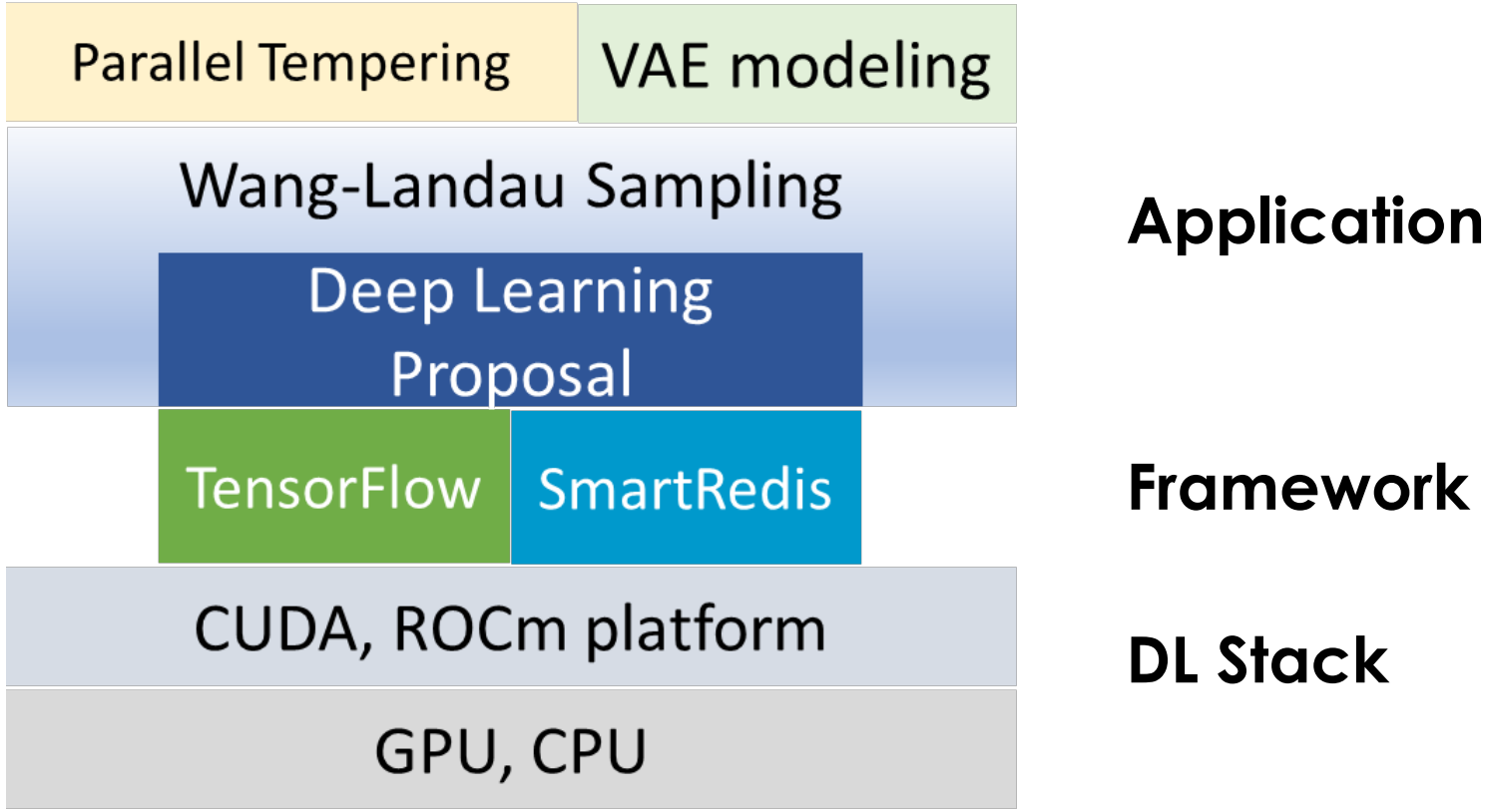
TF SERVING VS. REDISAI VS. EMBEDDED FOR BOTH C++ AND PYTHON CLIENTS ON T4 GPU (BATCH SIZE 1024).

Rank	Approach	Avg. Latency (s)	99th Perc. Latency (s)	Throughput (samples/s)	Distributed
1	Embedded C++	0.90	0.90	1137	No
2	TF Serving with C++ client	1.11	1.14	922	Yes
3	Embedded Python	1.16	1.20	881	No
4	RedisAI with C++ client	1.25	1.26	809	Yes
5	RedisAI with Python client	1.31	1.34	768	Yes
6	TF Serving with Python client	2.50	2.53	410	Yes

Brewer, Wesley, et al. "Production deployment of machine-learned rotorcraft surrogate models on HPC." *MLHPC'21*.

Application Example: DL-accelerated Monte Carlo

- Architecture
  - Application
  - Framework
  - DL stack
- Application layer is independent of DL stack, and built upon
  - TF C++
  - SmartRedis



https://doi.org/10.5281/zenodo.6612174, IPDPS23

SmartRedis Implementation

- Launch server:

redis-server --port --loadmodule

- Model can be dynamically added
- Data save to in-memory DB

Implementation 3 Use SmartRedis/RedisAI

```
1:  function ENERGY
2:      SSDB ← redis_host_ip : port
3:      SmartRedis::Client* clients[Nelements]
4:      for i ← 0, Nelements do
5:          Client_i.set_model_from_file(model_i, path_i, "TF", "GPU", batch_size,
    mini_batch_size, "hea", {"x"}, {"Identity"})
6:      end for

7:      for i ← 0, Nelements do                ▷ Add energy by element species
8:          uint8 Data[]
9:          data_key ← "data_i"
10:         outputs_key ← "energy_i"
11:         for j ← 0, Nneighbors do
12:             Data ← Atom[i][j]
13:         end for
14:         Client_i.put_tensor(data_key, Data, NT[t], Nneighbors*(NE-1), SmartRe-
    dis::TensorType::flt, SmartRedis::MemoryLayout::contiguous))
15:         Client_i.run_model(model_i, {data_key}, {outputs_key})
16:         Client_i.unpack_tensor(outputs_key, Outputs.data(), NT[t], SmartRe-
    dis::TensorType::flt, SmartRedis::MemoryLayout::contiguous)
17:         Energy ← Outputs
18:     end for
19:     return Energy
20: end function
```

TF C++ Implementation

- Assume model in TF *SavedModel* format
- Link with *libtensorflow_cc.so*
- Support *half, uint8, …*

Implementation 1 Use TensorFlow C++ API

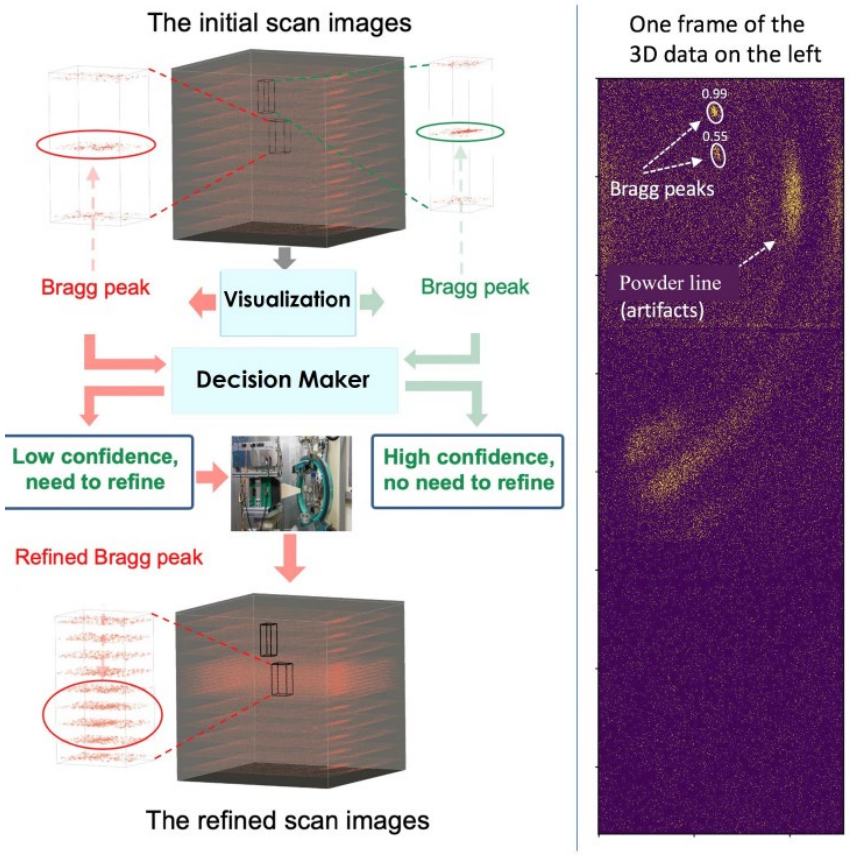
```
1:  function LOADMODEL(string ModelPath)
2:      tensorflow::SessionOptions SessOpt
3:      tensorflow::RunOptions RunOpt
4:      tensorflow::SavedModelBundle Model          ▷ Get model path, and setup session and run options
5:      tensorflow::LoadSavedModel(SessOpt, RunOpt,
            ModelPath, {"serve"}, &Model)           ▷ load pre-trained model
6:      return Model
7:  end function

8:  function PREDICT(tensorflow::SavedModelBundle Model, tensorflow::Tensor Inputs)
9:      tensorflow::Tensor Inputs                   ▷ input tensor
10:     vector<tensorflow::Tensor> Outputs
11:     string InputNode
12:     string OutputNode                           ▷ input and output nodes

13:     Model.GetSession().Run(data, {OutputNode}, {}, &Outputs)
14:     return Outputs
15: end function

16: function ENERGY
17:     for i ← 0, Nelements do
18:         Model_i ← LOADMODEL(Path_i)
19:     end for

20:     for i ← 0, Nelements do                     ▷ Add energy by element species
21:         tensorflow::TensorShape DataShape
22:         tensorflow::Tensor Inputs(tensorflow::DT_UINT8, DataShape)
23:         for j ← 0, Nneighbors do
24:             Inputs ← Atom[i][j]
25:         end for
26:         Outputs = PREDICT(Model_i,Inputs)
27:         Energy ← Outputs
28:     end for
29:     return Energy
30: end function
```

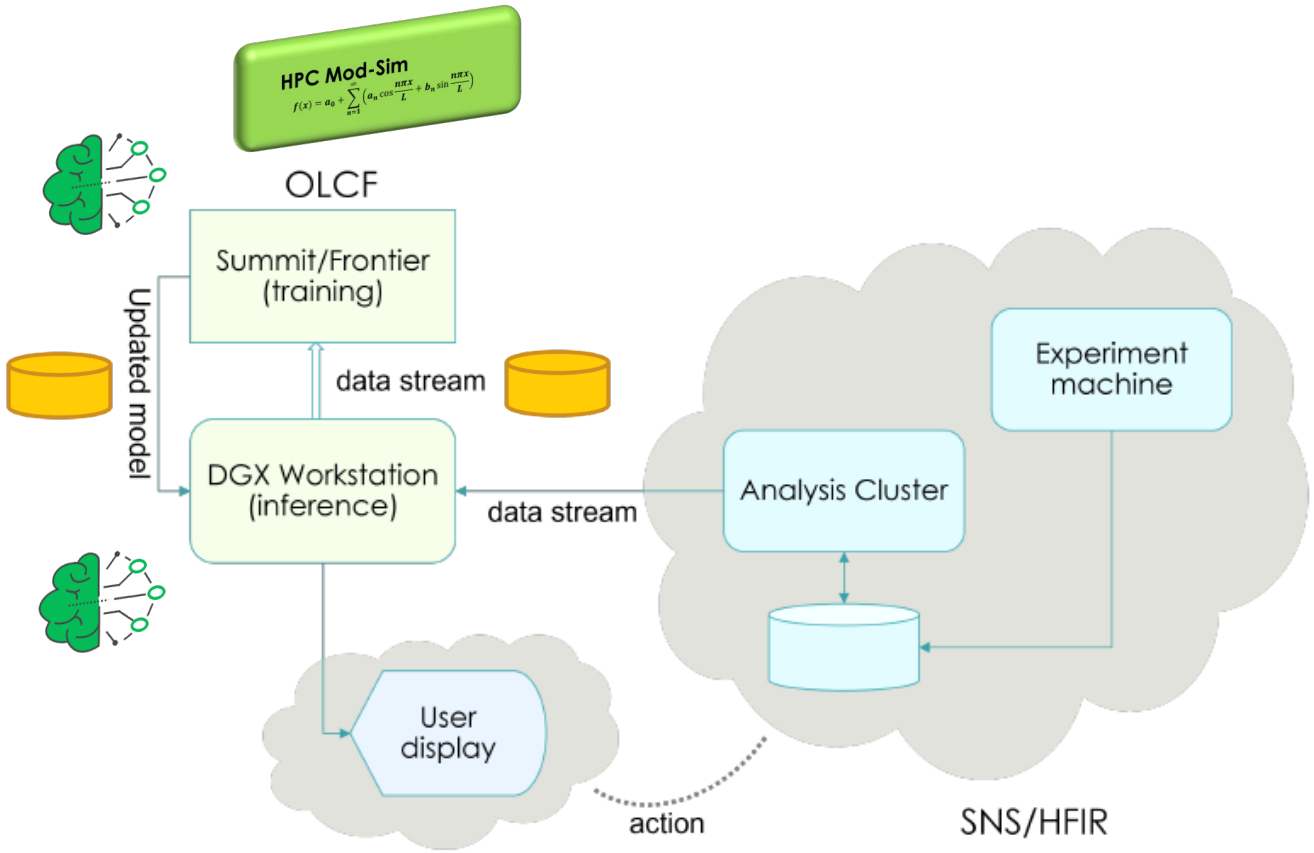
Implementation Considerations

OVERVIEW OF THE CHARACTERISTICS OF DIFFERENT DEPLOYMENT METHODS FOR SURROGATE MODELING IN SIMULATIONS.

deployment method	usability		performance		
	portability	flexibility	concurrency	latency	throughp
TensorFlow C++ API	●	◑	◑	●	◑
TensorFlow Serving	◑	◕	●	◑	◕
SmartRedis/RedisAI	◑	●	●	◕	●

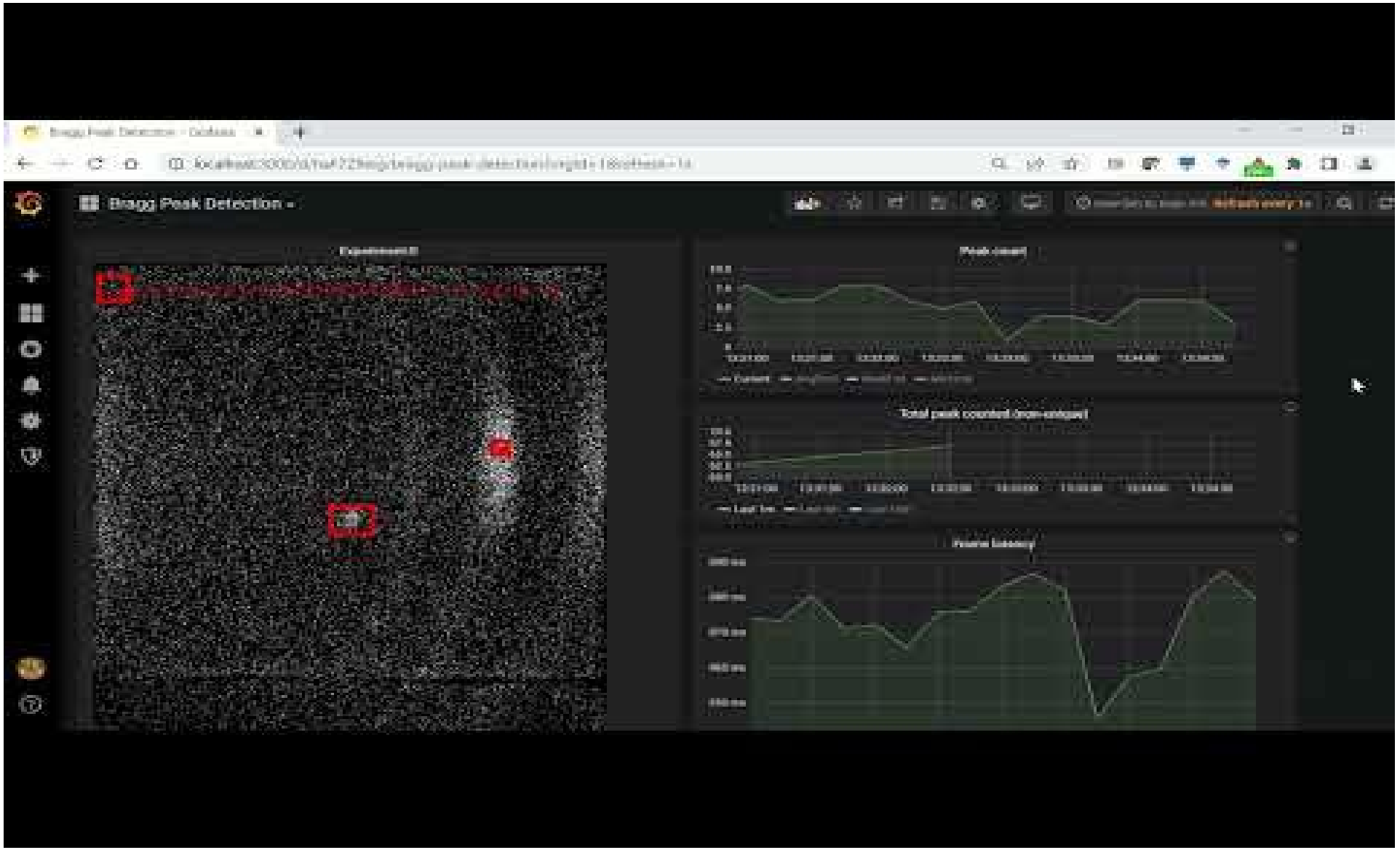
HPC-AI Usage Ex Situ: Distributed System





J. Yin, G. Zhang, H. Cao, S. Dash, B. Chakoumakos, and F. Wang, "Toward an Autonomous Workflow for Single Crystal Neutron Diffraction." SMC 2022

Demonstration



HPC + AI Training Resources

- Upcoming: AI and Model Deployment
  - https://www.olcf.ornl.gov/calendar/ai-for-science-at-scale-intro/ , June 15th, 2023
  - https://www.olcf.ornl.gov/calendar/smartsim-at-olcf/ - HPE Training, July 13th, 2023
- AI Model Training
  - *AI on Frontier: https://olcf.ornl.gov/wp-content/uploads/2-16-23_AIonFrontier.pdf*
  - *Data and Viz: https://www.olcf.ornl.gov/calendar/data-visualization-and-analytics-training-series-jupyter-workflow-at-olcf/*
- Hyperparameter exploration
  - *Link to OLCF training:* https://www.olcf.ornl.gov/wp-content/uploads/Jupyter_DL_Workflow.pdf (RayTune)

Thank you!

We are Hiring! Visit: https://jobs.ornl.gov/