Distributed training and hypertuning of deep-learning based algorithms using HPC in CoE RAISE

Workshop for the USATLAS-USCMS HPC/Cloud Blueprint

27th of September 2022

Eric Wulff
CERN

Hyperparameter Optimization

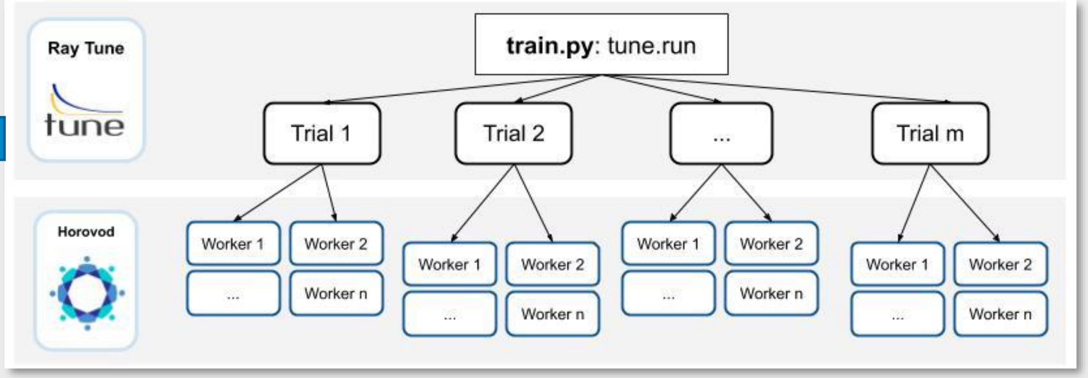
- Sometimes referred to as Hyperparameter Tuning or *Hypertuning*

- Hyperparameters stay constant during the learning process
  - Defines the model architecture (e.g., #layers, #nodes per layer, etc.)
  - Defines the optimization algorithm (e.g., learning rate, batch size, k in KNN, etc.)

- Hypertuning complex models and/or large datasets is compute-resource intensive
  - Benefits greatly from HPC resources
  - In need of smart, efficient search algorithms

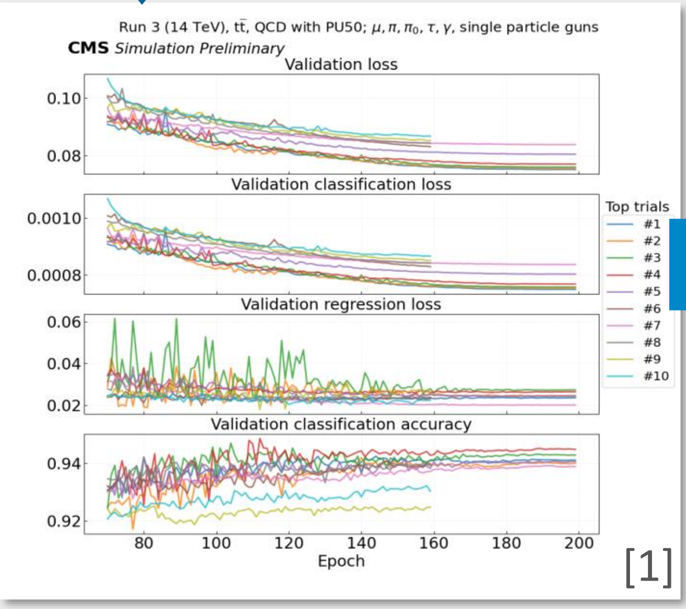
Task 4.1 – Large-Scale Distributed Hyperparameter Optimization

Distributed hypertuning

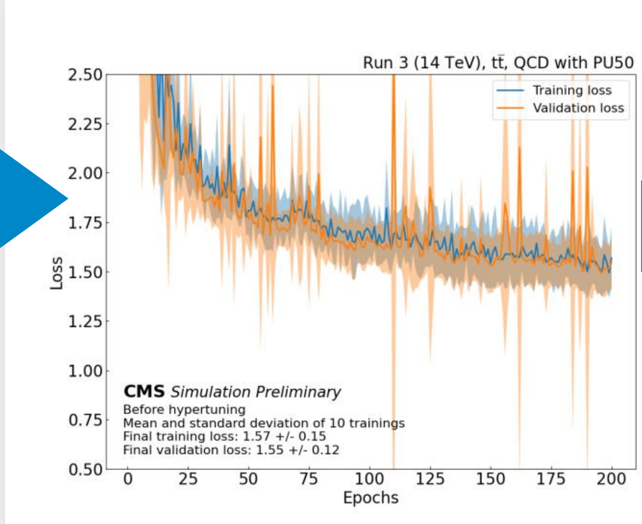


Model selection

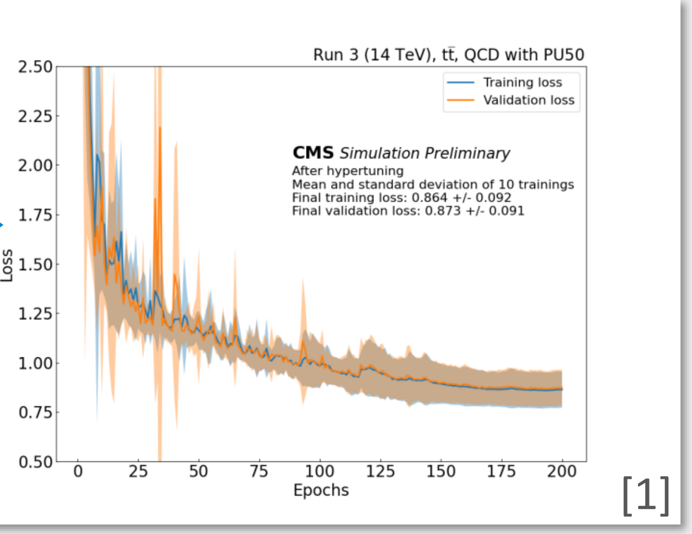
- Scalable up to hundreds of GPUs
- Mean validation loss decreased by ~44% giving a significant performance improvement



Assess learning variability



Better learning

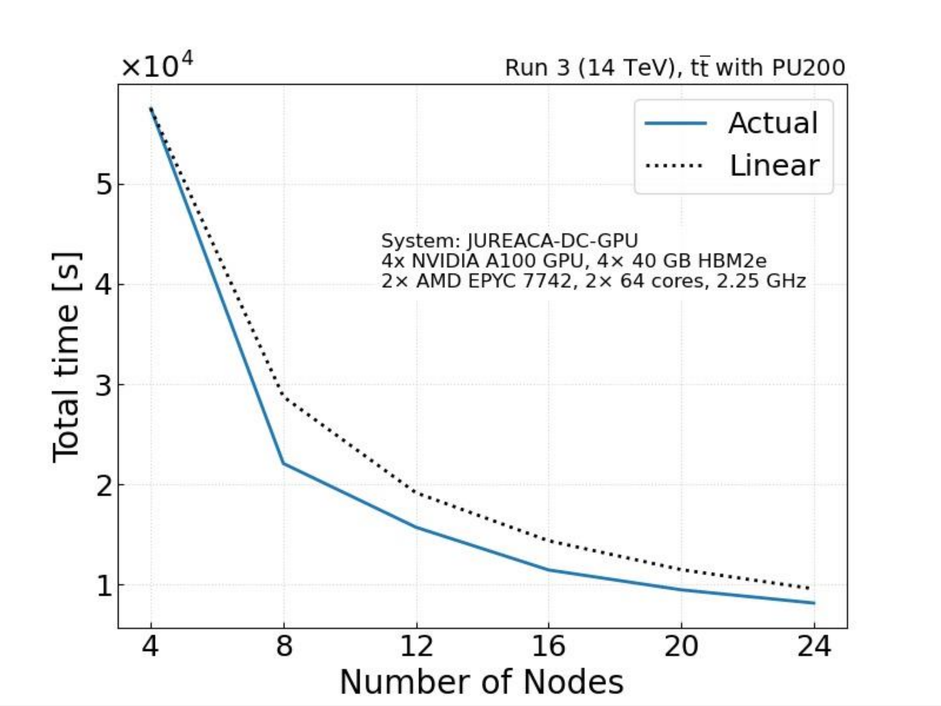


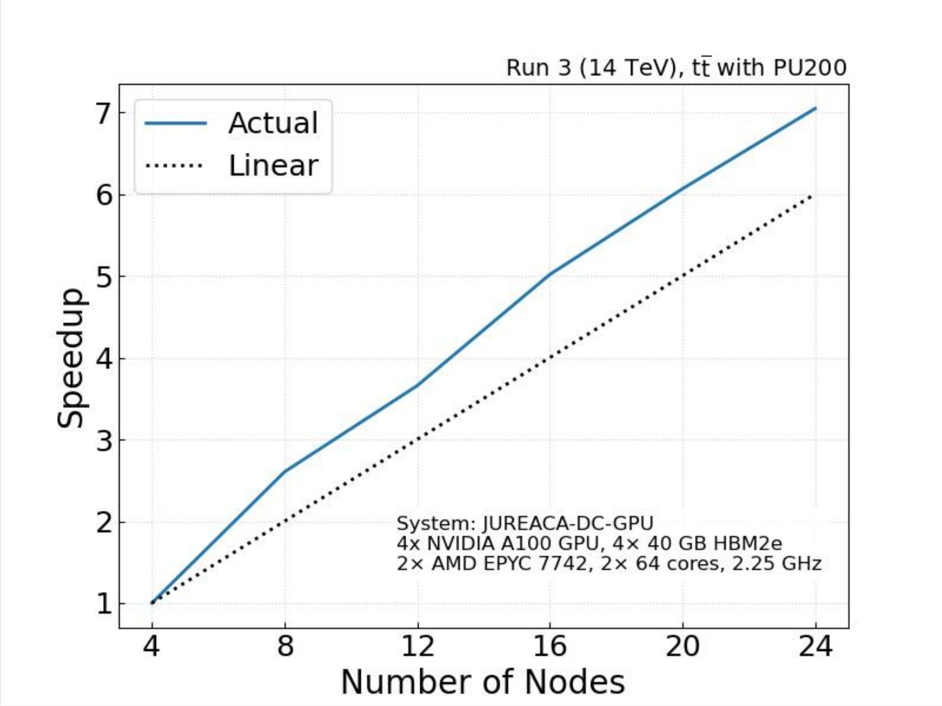
[1] E. Wulff, M. Girone, J. Pata https://arxiv.org/abs/2203.01112

Scaling of MLPF hypertuning on multiple compute nodes

RAISE Center of Excellence

- Scaling of a hypertuning run of MLPF on the JURECA-DC-GPU system at the Jülich Supercomputer Centre (JSC), 4 NVIDIA A100 and 2× 64 cores AMD EPYC 7742 per node
- **Better than linear** due to excessive re-loading of models when using fewer nodes





Data used: *Simulated particle-level events of ttbar and QCD with PU200 using Pythia8+Delphes3 for machine learned particle flow (MLPF)* [1]

[1] https://doi.org/10.5281/zenodo.4452282

drive. enable. innovate.

RAISE Center of Excellence

The CoE RAISE project has received funding from the European Union's Horizon 2020 – Research and Innovation Framework Programme H2020-INFRAEDI-2019-1 under grant agreement no. 951733

Follow us:

Backup

Using Ray Tune on SLURM clusters

RAISE Center of Excellence

- Can be unintuitive when first setting up
- Ray expects a head-worker architecture with a single point of entry
  - We must start a head node and multiple worker nodes before running the Ray Tune training script on the head node
- Once properly set-up, works great

RAY

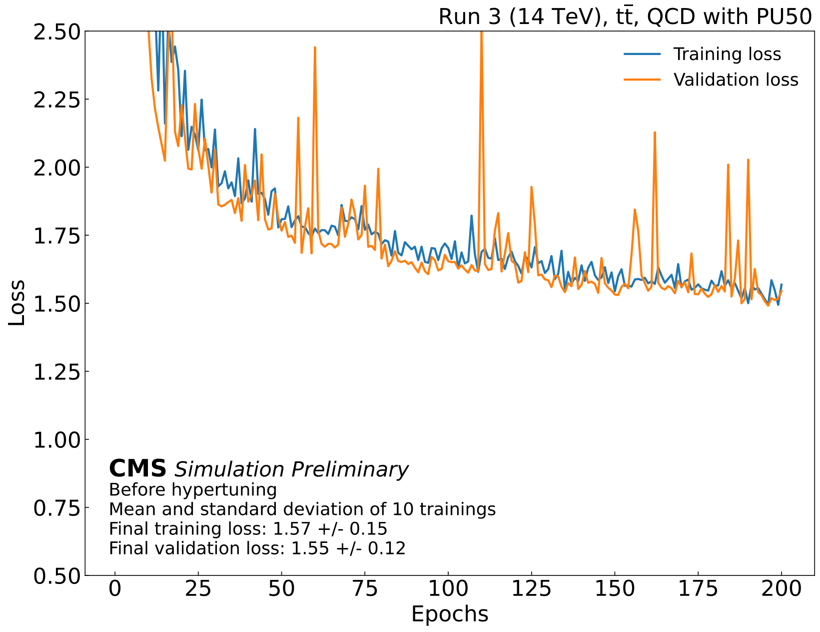
slurm workload manager

```
#!/bin/sh

#SBATCH ...
#SBATCH ...


# Get the node names
nodes=$(scontrol show hostnames $SLURM_JOB_NODELIST)
nodes_array=( $nodes )

# Get the head node
node_1=${nodes_array[0]}
ip=$(srun --nodes=1 --ntasks=1 -w $node_1 host ${node_1}i | awk '{ print $4 }') port=6379
ip_head=$ip:$port
export ip_head
echo "IP Head: $ip_head"

echo "STARTING HEAD at $node_1"
srun --nodes=1 --ntasks=1 -w $node_1 mlpf/raytune/start-head.sh $ip &
sleep 30

worker_num=$(($SLURM_JOB_NUM_NODES - 1)) #number of nodes other than the head node
for (( i=1; i<=$worker_num; i++ ))
do
  node_i=${nodes_array[$i]}
  echo "STARTING WORKER $i at $node_i"
  srun --nodes=1 --ntasks=1 -w ${node_i} mlpf/raytune/start-worker.sh $ip_head &
  sleep 5
done

# Run the Ray Tune script
python3 tune_script.py --cpus "${SLURM_CPUS_PER_TASK}" --gpus "${SLURM_GPUS_PER_TASK}"
exit
```

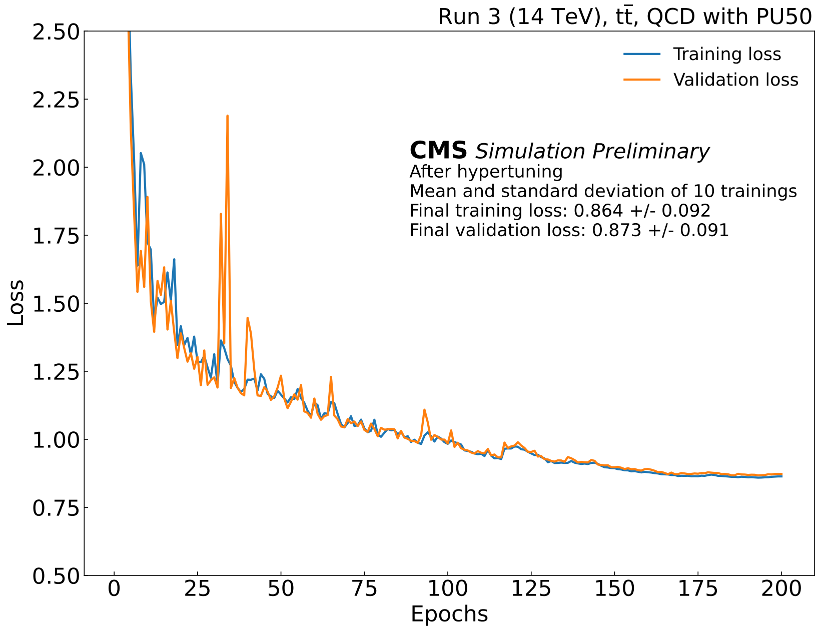
Code available at [1]. Cluster launcher adapted from [2].

Improvements from hypertuning

- Loss curves before (left) and after (right) hypertuning
- Only the physical datasets, no single particle gun samples
- Mean and standard deviation of 10 trainings with identical hyperparameters
- Mean validation loss decreased by ~44%



[1]

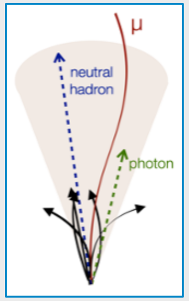


[1]

[1] E. Wulff, M. Girone, J. Pata https://arxiv.org/abs/2203.01112

Task 4.1 – Machine-Learned Particle Flow

Physics simulation

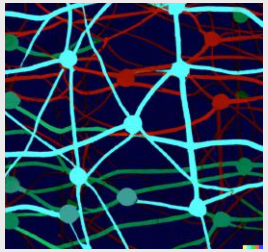


Data selection

Dataset creation

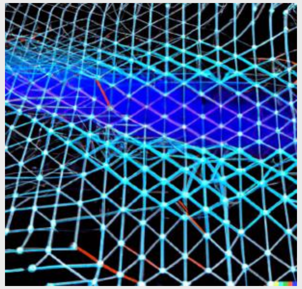
Data pre-processing

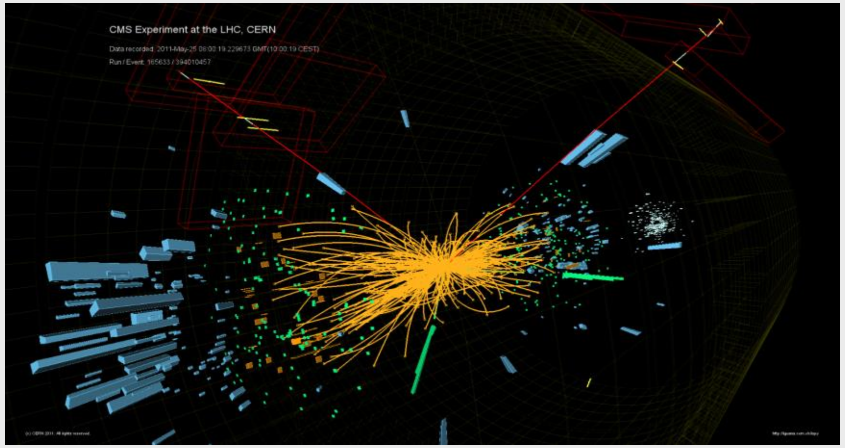
GNN training



Model export

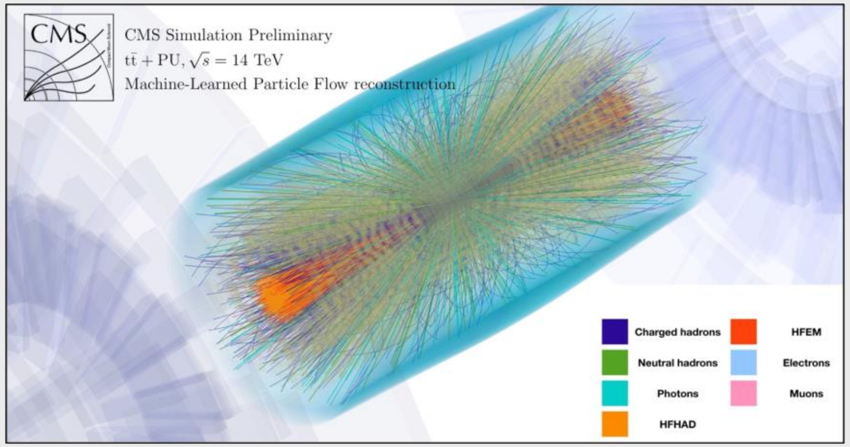
Trained model





CMS Collision event

Event reconstruction



MLPF event reconstruction [1]

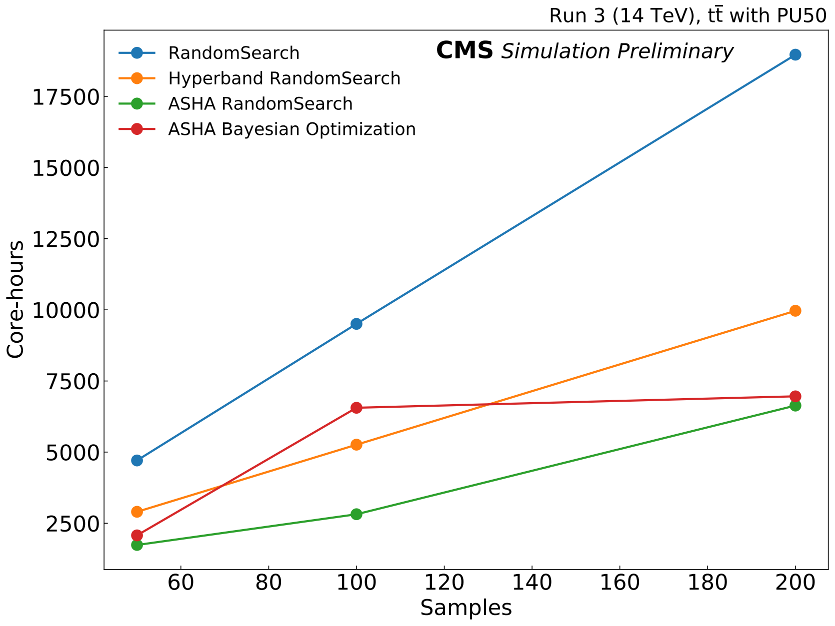
[1] Pata, J., Duarte, J., Mokhtar, F., Wulff, E., Yoo, J., Vlimant, J.-R., … Girone, M. (2022). *Machine Learning for Particle Flow Reconstruction at CMS*. Retrieved from http://arxiv.org/abs/2203.00330

Comparison of hypertuning algorithms in Ray Tune

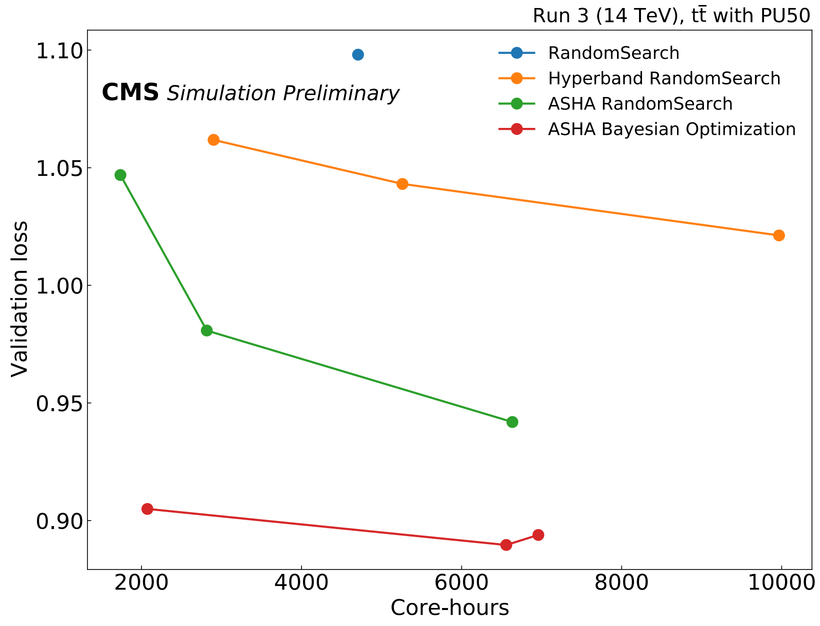
RAISE Center of Excellence

- Using MLPF on a subset of the training data
- Using 4 compute nodes with 4 GPUs per node
  - NVIDIA A100 SXM4 40GB
  - 64 core Intel Xeon Platinum 8358 CPU @ 2.60GHz

- Both Hyperband and ASHA much more efficient than random search
- ASHA beats Hyperband in efficiency due to its asynchronous nature
- ASHA + BO gives best performance per spent core-hour



[1]



[1]

[1] E. Wulff, M. Girone, J. Pata https://arxiv.org/abs/2203.01112