Vlaams Supercomputer Centrum User Day

17th December 2025

Supercharging AI with Hyperparameter Optimization and HPC

Eric Wulff
CTO for AI on HPC, CERN openlab
CERN IT Department

Outline

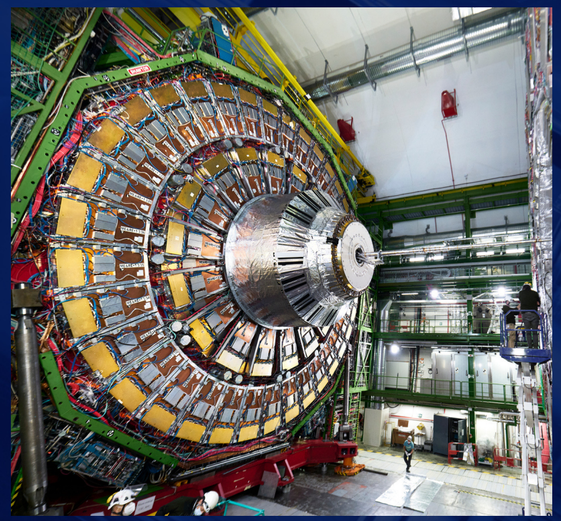
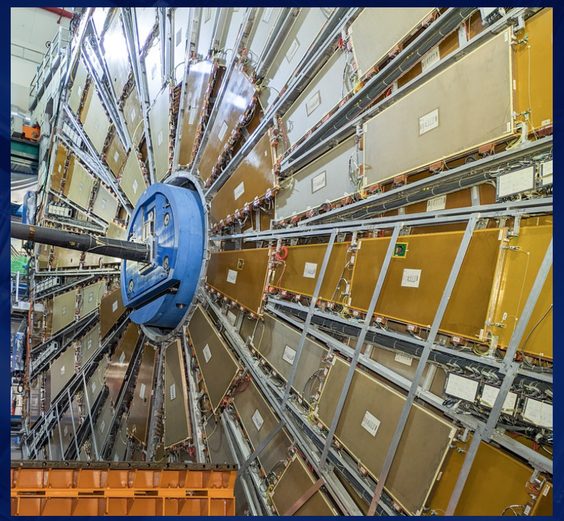
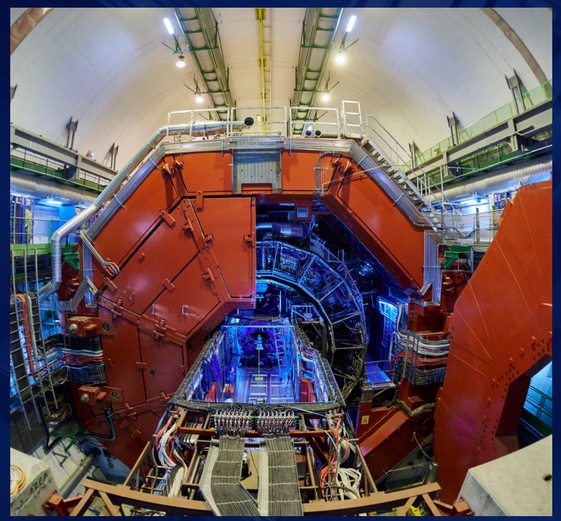
- CERN and its upcoming computing challenges
- A brief intro to distributed AI training and hyperparameter optimization
- Hyperparameter optimization on HPC in practice
- Example application from High Energy Physics: ML-based Particle Flow Reconstruction

CERN is the world's largest laboratory for particle physics

Our goal is to understand the most fundamental particles and laws of the universe

Large Hadron Collider (LHC)

- 27 km in circumference
- Around 100 m underground
- Superconducting magnets steer the particles around the ring
- Particles are accelerated to close to the speed of light

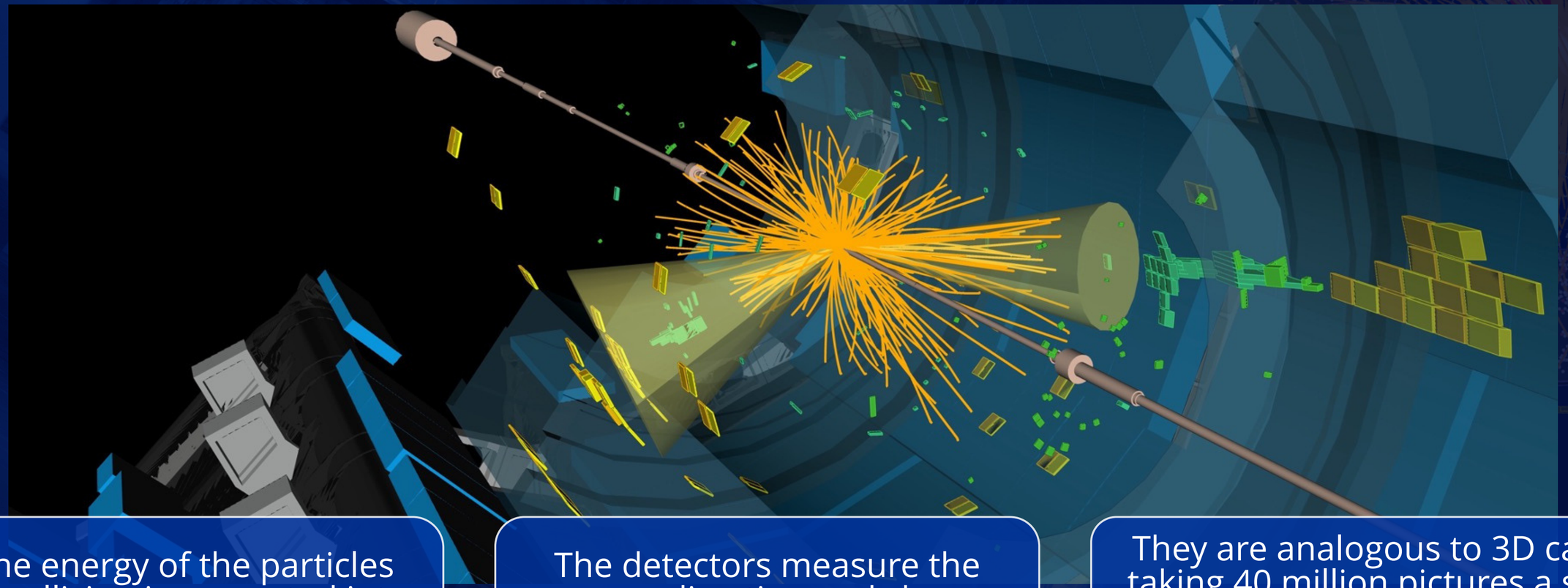





Giant detectors record the particles formed at the four collision points

We use them to answer fundamental scientific questions!

- Why is the universe made only of matter, with hardly any antimatter?
- Why is gravity so weak compared to other forces?
- Is there only one Higgs boson, and does it behave exactly as expected?

The LHC produces more than 1 billion particle collisions per second, resulting in 1TB/minute stored in our Data Centre



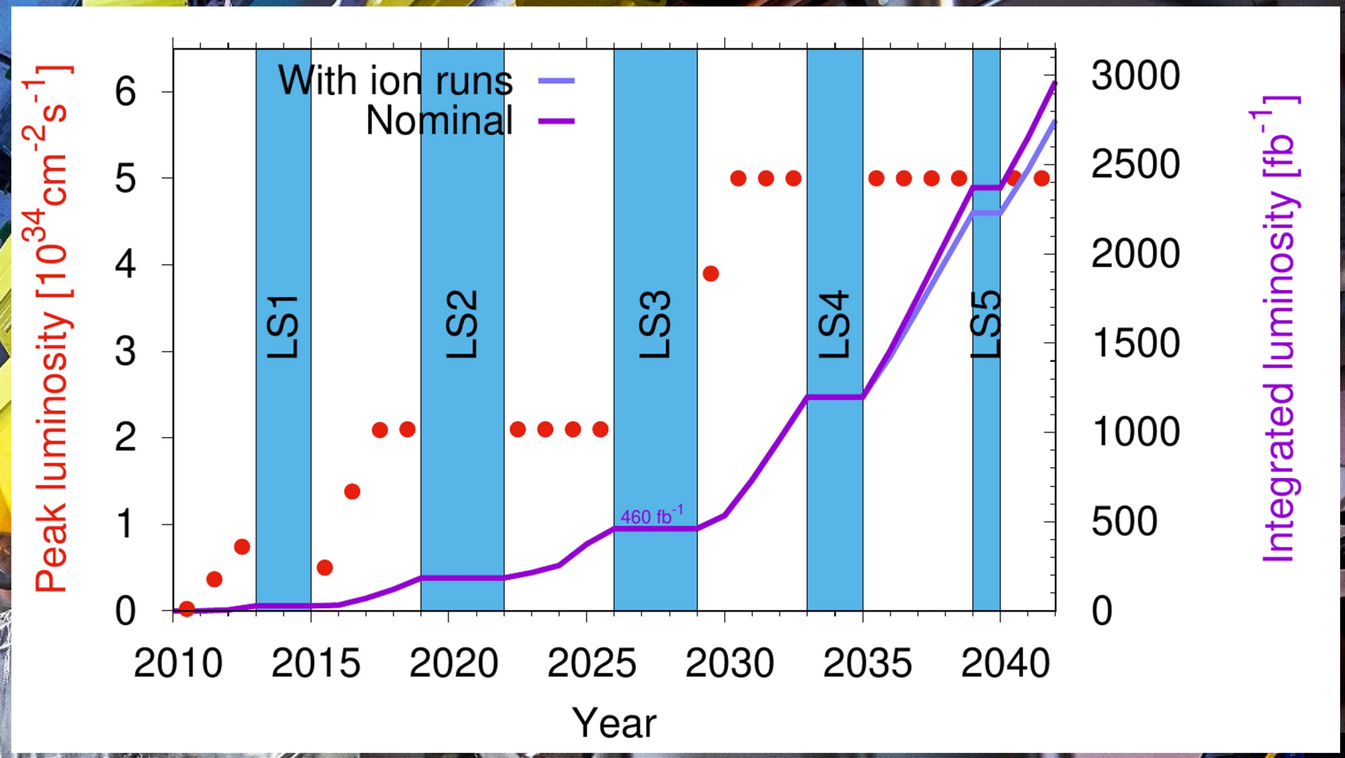
The energy of the particles in collision is converted into new particles

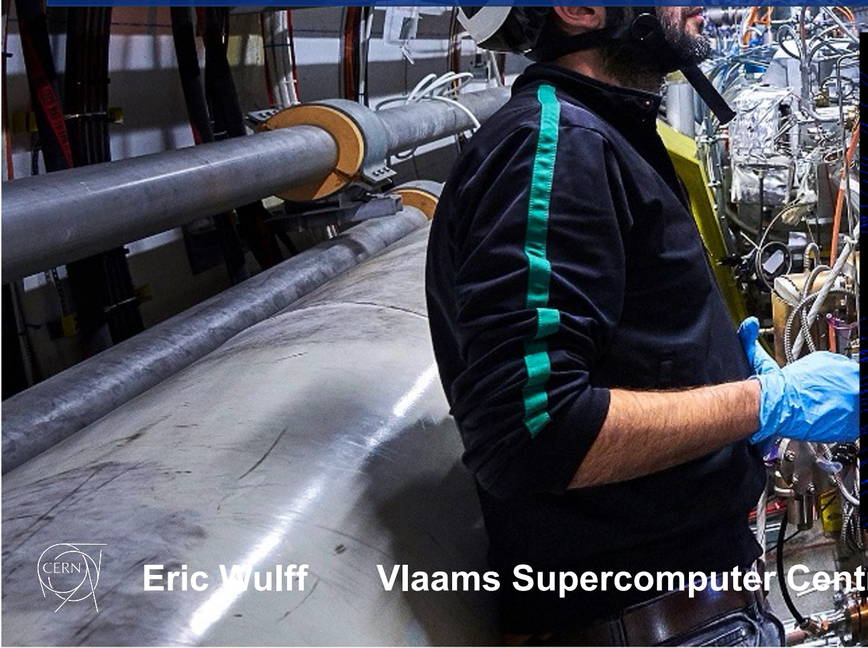
The detectors measure the energy, direction and charge of new particles formed

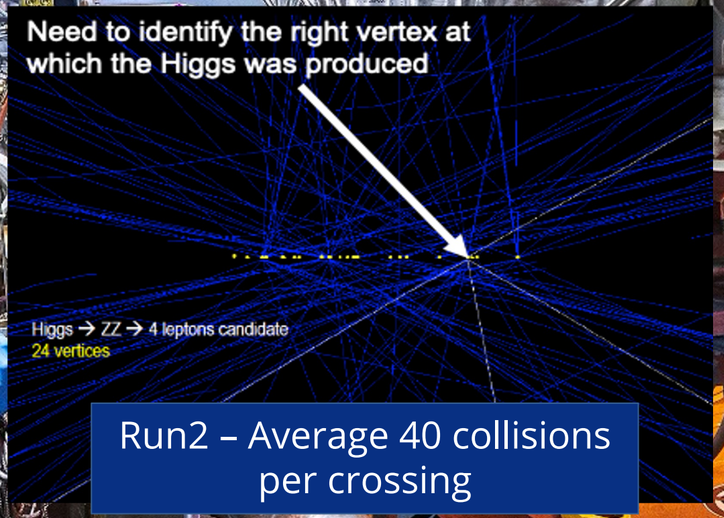
They are analogous to 3D cameras taking 40 million pictures a second, of which 1000 are selected and saved

LHC Upgrade

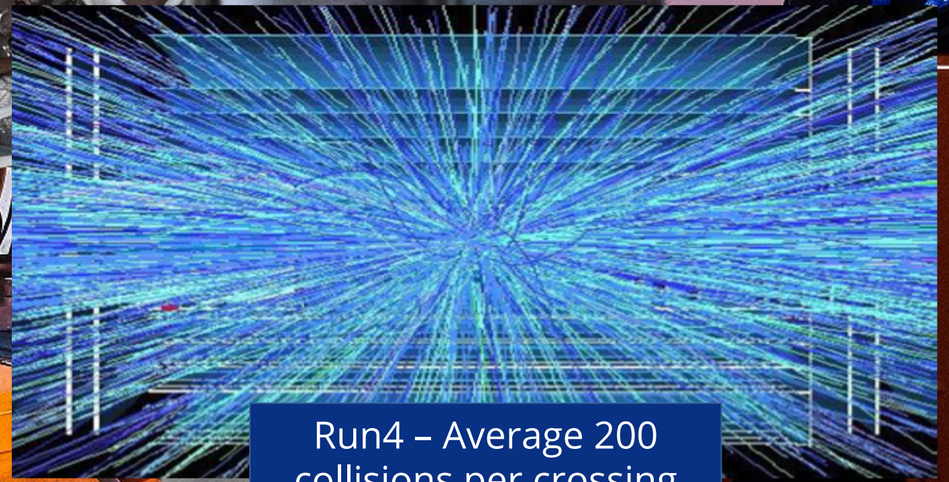
- Although the LHC have been running for a long time, we are only at 10% of its exploitation. There are still many things to do, both on the scientific and technical level.
- The HL-LHC will use new technologies to provide 10 times more collisions than the LHC
- It will provide greater precision and discovery potential
- It will start operating in 2029 and run until 2040







Run2 – Average 40 collisions per crossing

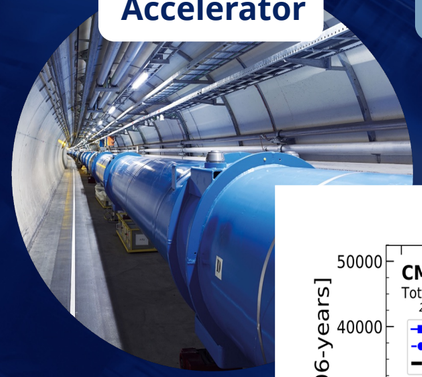


Run4 – Average 200 collisions per crossing

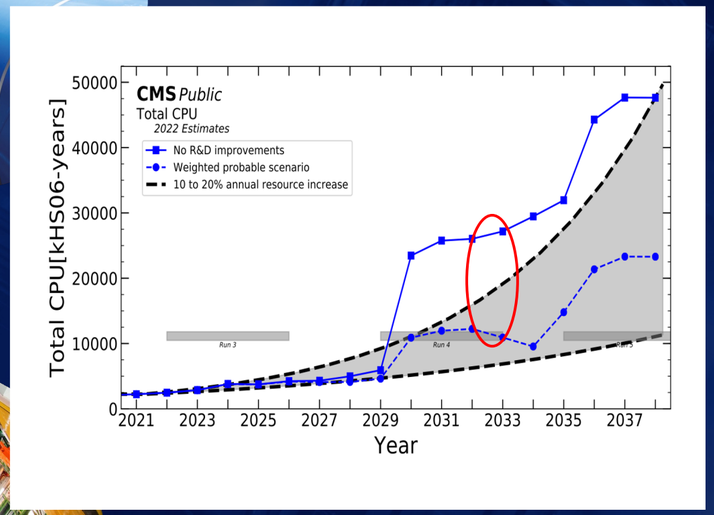
Challenges at HL-LHC

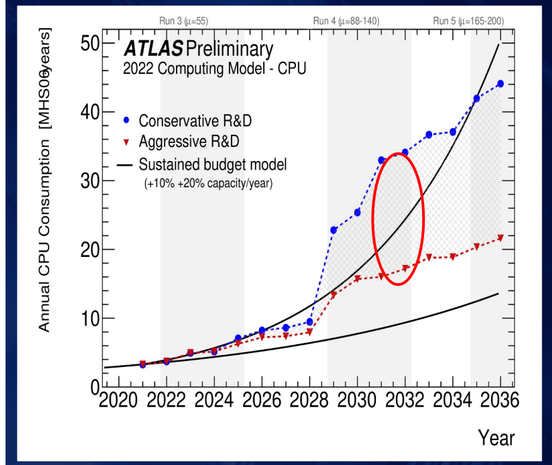
Upgraded Accelerator

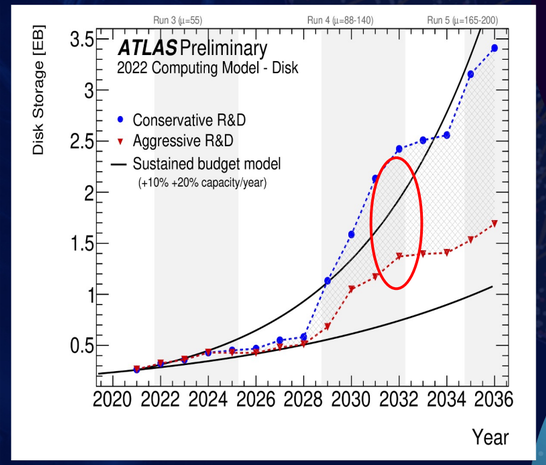
Higher luminosity



New Computing Challenges

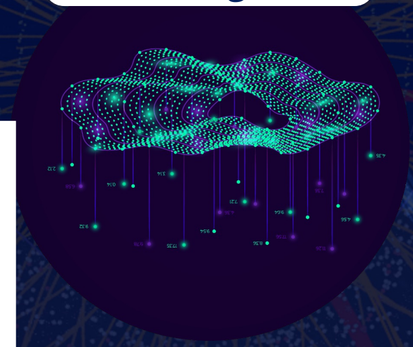






Higher data rates for Higher sensitivity

Changing Filtering Paradigms



Upgraded Detectors

Higher granularity for Higher occupancy

The resource gap motivates investment in:

Code modernization
HPC and hardware accelerators
New techniques, from **AI** to QC

R&D Investments

We leverage HPC resources in HEP and beyond

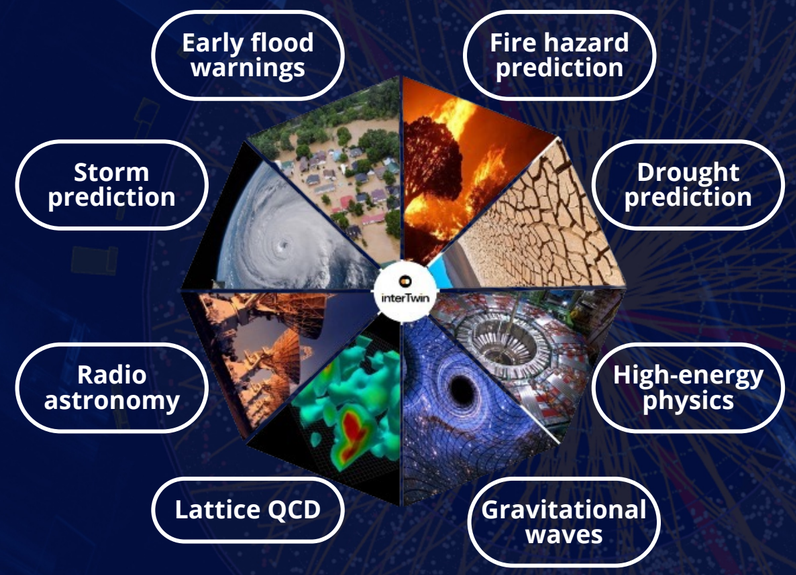
RAISE Center of Excellence → Research and development on AI- and simulation-based engineering at Exascale

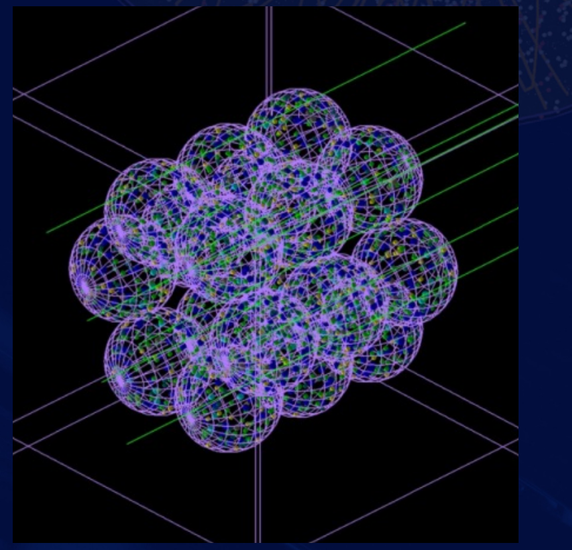
interTwin → To co-design and implement the prototype of an interdisciplinary Digital Twin Engine

SPECTRUM → CERN leverages **common visions and challenges** to help deliver a Strategic Research, Innovation and Deployment Agenda (SRIDA) and a Technical Blueprint

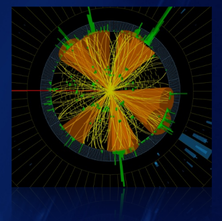
BioDynaMo.org → To build a software that supports scientists to easily create, run, and visualize multi-dimensional agent-based simulations

Responding to EuroHPC Community CoE (EuroHPC Joint Undertaking) → Focuses on the needs of a given community (would be HEP in our case), to elevate their codes to a better / more efficient / possible use on EuroHPC JU systems.

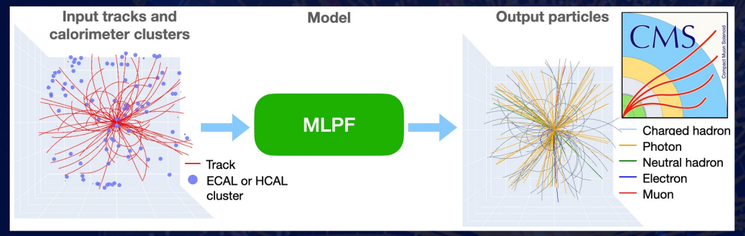




Artificial intelligence thrives at CERN



ML-particle flow



Accelerator systems

- Beam dynamics and control
- Enhanced diagnostics & predictions

Infrastructure

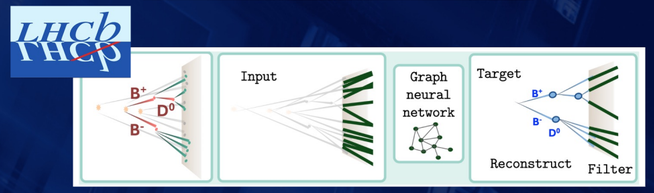
- Network saturation prediction
- Data quality monitoring

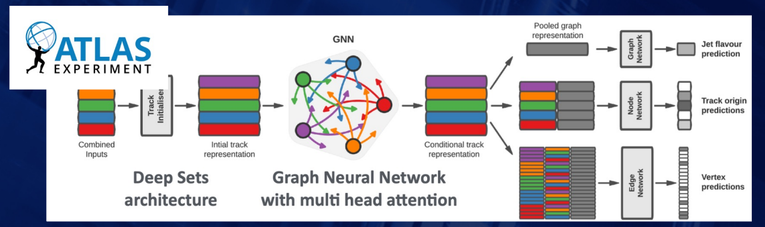
Real-time data selection & filtering

- O(nanos -micros) event reconstruction in FPGAs
- Fake reduction
- Anomaly detection w/ VAEs & CNNs
- Muon tracking
- On-detector data compression (AE)

Simulation

- Generative models for event generation and fast simulation e.g ATLAS FastCalo GAN





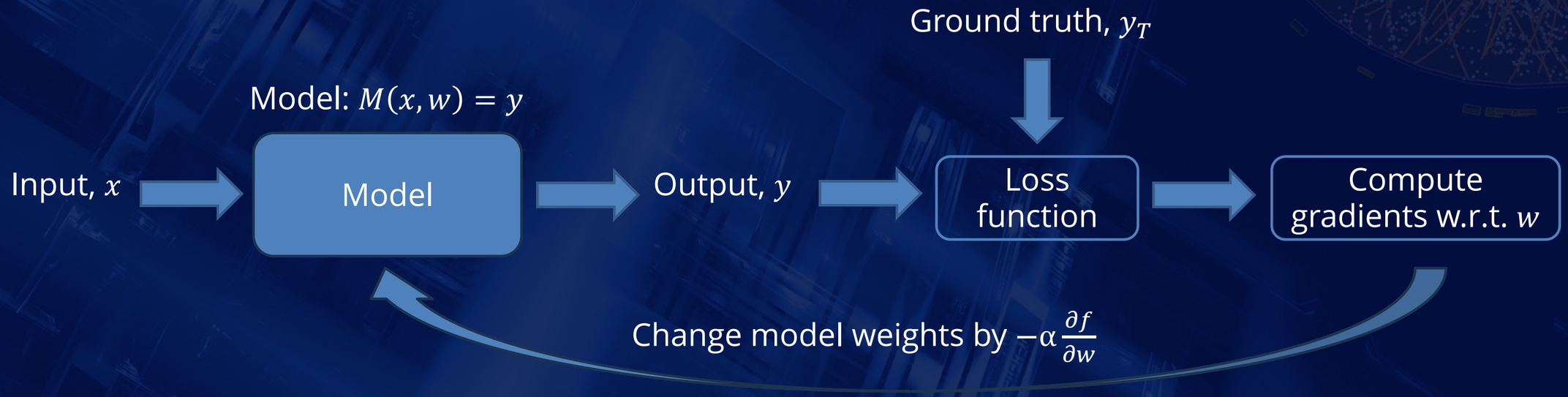
Analysis

- Clustering and pattern recognition
- Signal/background discriminations e.g *H*-> γγ
- ML-particle flow
- Particle classification & Jet tagging
- Energy calibration

A brief introduction to distributed AI training and hyperparameter optimization

Training deep learning models

- In DL, model parameters w are learned using backpropagation and gradient descent to minimize some objective, or loss, $f(w, \theta)$
- Training:
  - For each x in training data, compute the gradients of the loss and change the model's weights by subtracting from them the gradients multiplied by some learning rate, α
  - Repeat until convergence or reaching some other stopping criterion

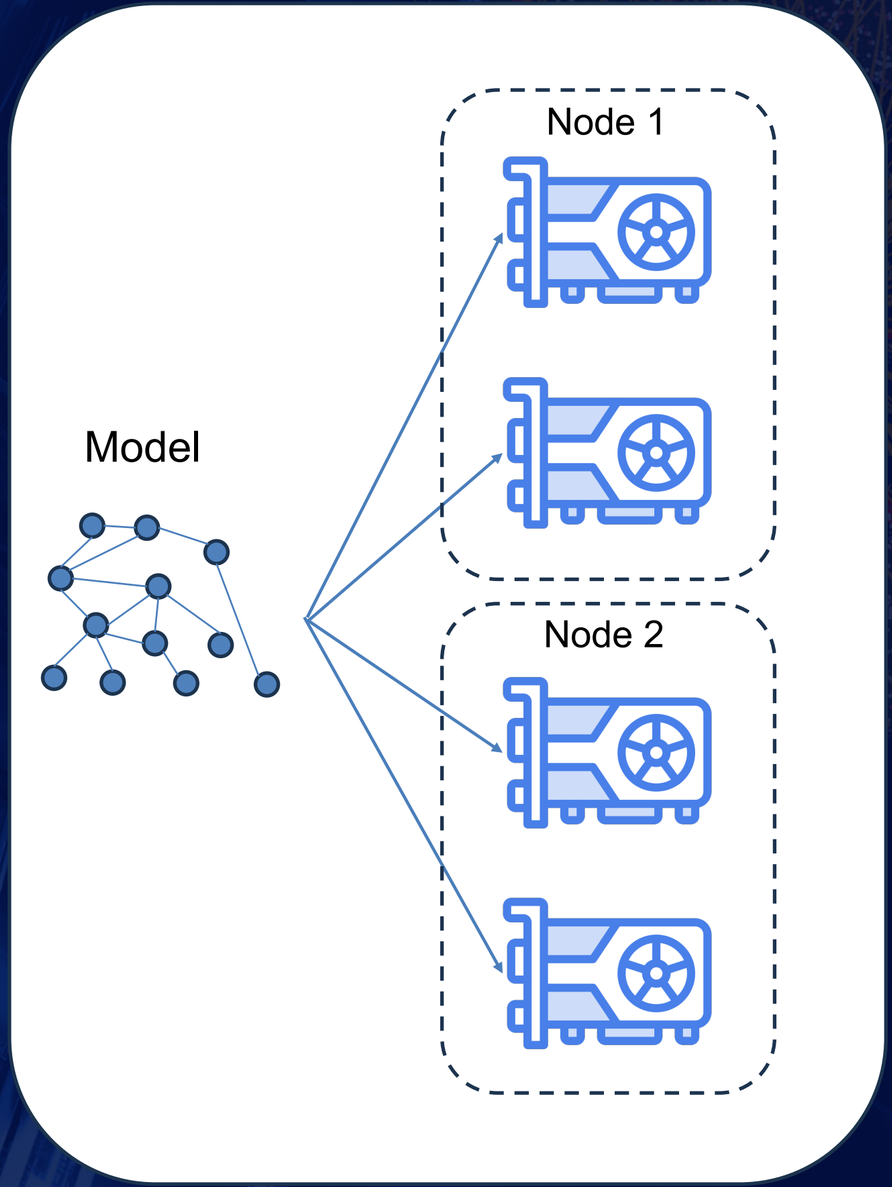


Distributed training

- What is Distributed Training?
  - Training models across multiple devices/nodes
  - Enables scaling to larger models and datasets

- Why It Matters:
  - Faster training times
  - Faster development cycles
  - Overcome memory constraints

- Examples of distributed strategies:
  - Data parallel
  - Fully sharded data parallel, ZeRO-3, ZeRO-2, ZeRO-1
  - Tensor parallel
  - Pipeline parallel
  - Etc.

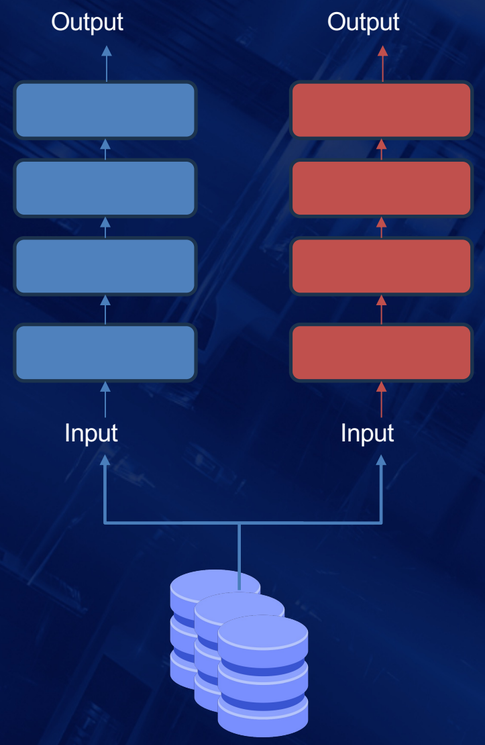


Distributed training strategies

Data parallelism

Replicates same model across multiple devices

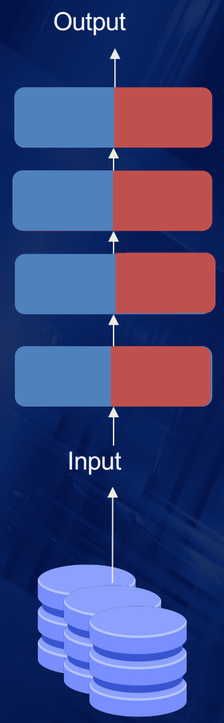
- Copy entire model on each device
- Each copy trains on different batches
- Gradients are averaged and synchronized across devices in each optimization step



Tensor parallelism

Splits individual layers across GPUs

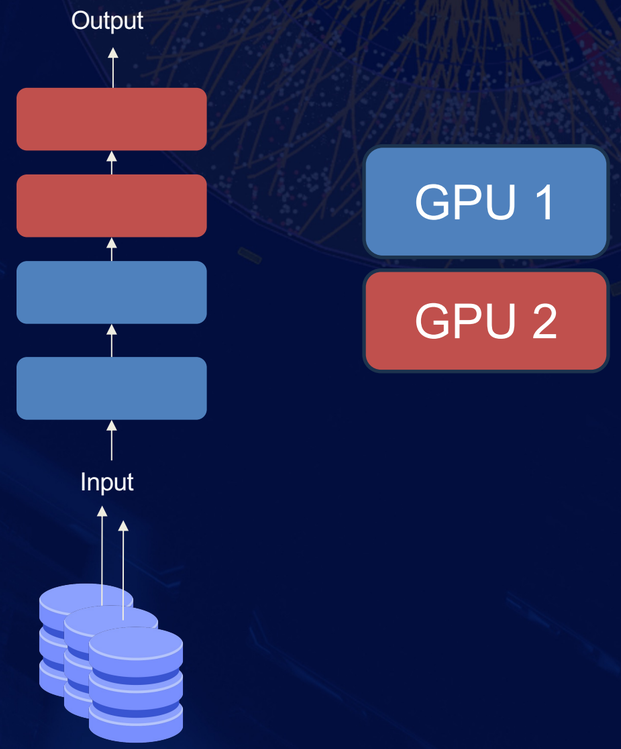
- Divides the computation of a single tensor operation among devices
- Each device handles a fraction of the overall computation



Pipeline parallelism

Breaks model into sequential stages that run on separate GPUs

- Input data flows through the pipeline stages sequentially
- Overlapping execution: while one batch is processed in later stages, the next batch can enter the pipeline

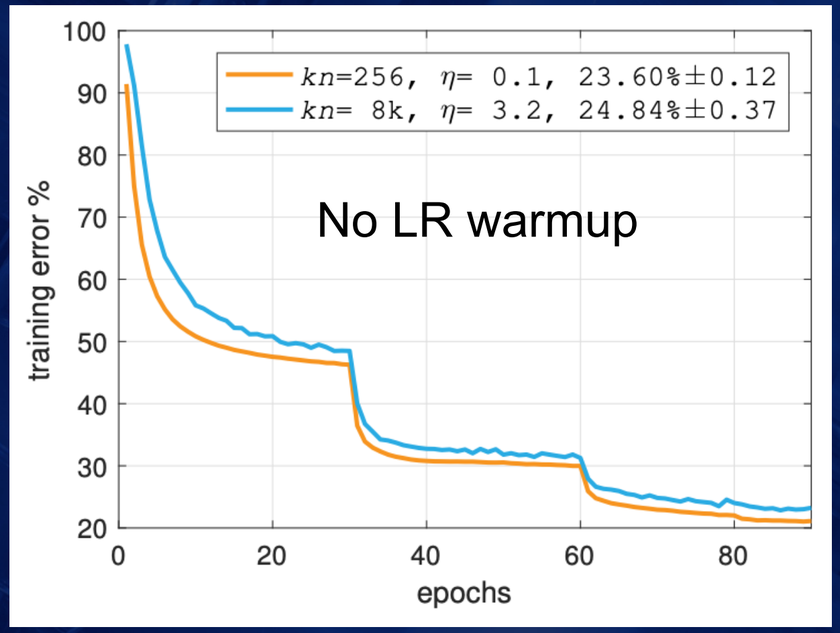


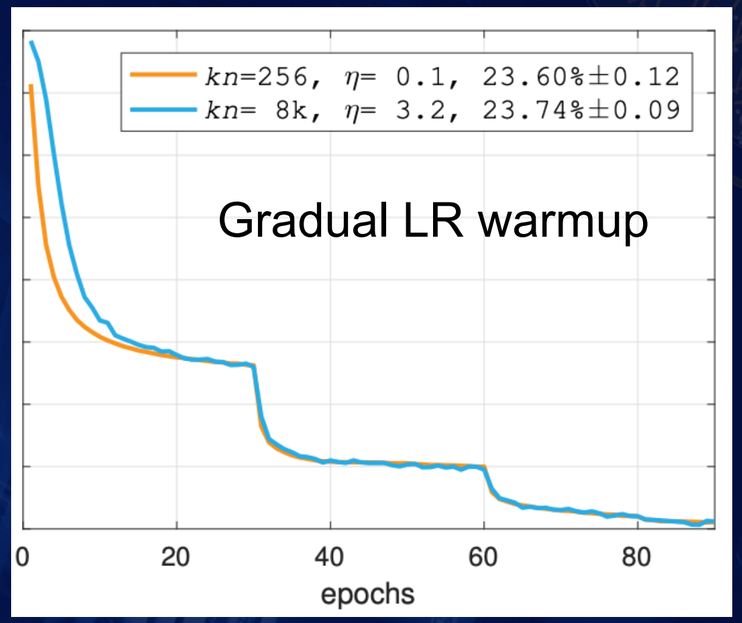
Challenges in distributed training

- **Communication overhead:** Synchronization between GPUs
- **Memory/data loading bottlenecks:** Data loading not fast enough to saturate all GPUs
- **Debugging complexity:** Harder to troubleshoot than single-GPU training
- **Generalization gap:** Large global batch size can sometimes result in a generalization gap

The generalization gap

- Even with LR-scaling a generalization gap may appear when training with large batch sizes (*Accurate, Large Minibatch SGD: Training ImageNet in 1 Hour* arxiv:1706.02677)
- This may be due to several causes and will differ in different models and datasets
  - Larger batch size leads to fewer optimization steps (if number of epochs is kept constant) → train for longer
  - Large LR leads to training instabilities → LR warmup or scale LR less aggressively
  - Insufficient tuning of other HPs when LR is scaled → Hyperparameter optimization
  - Try using a specialized optimizer such as LAMB (arxiv:1904.00962)





Accurate, Large Minibatch SGD: Training ImageNet in 1 Hour (arxiv:1706.02677)

Generalization gap in MLPF training

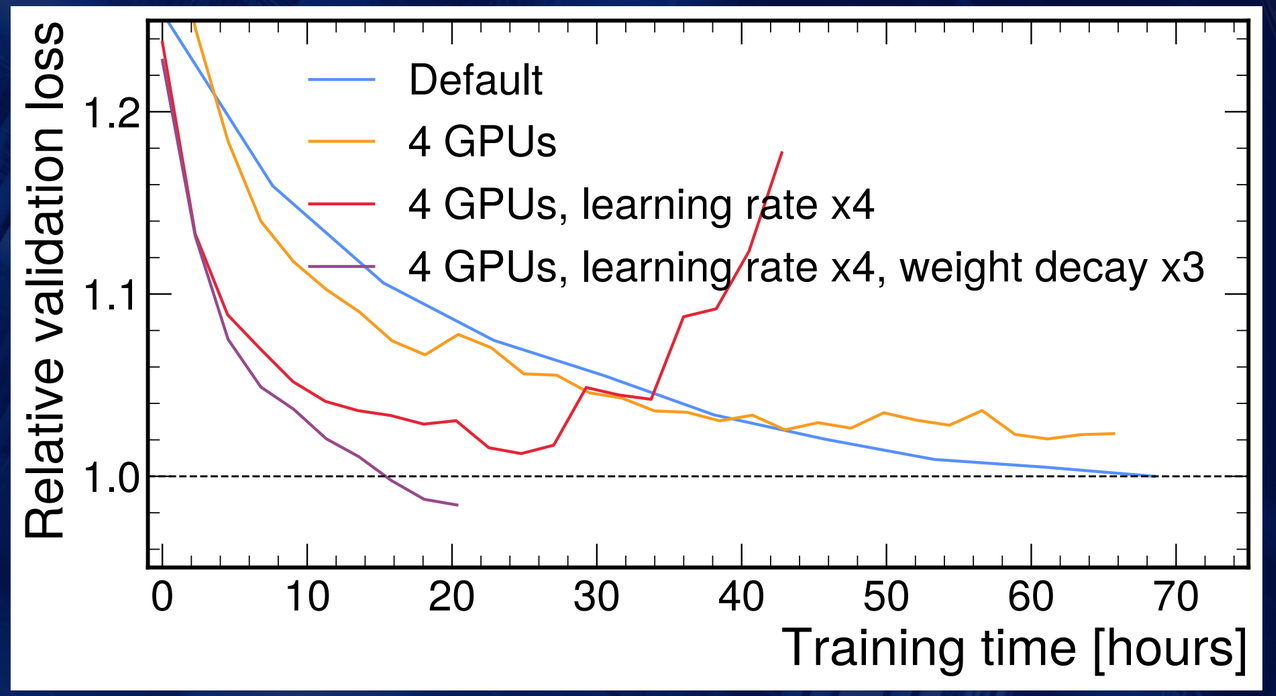
- Generalization gap appeared when moving to multi-GPU training
- LR-scaling alone proved insufficient
- LR-scaling + additional hyperparameter tuning did the trick!

Fine-tuning machine-learned particle-flow reconstruction for new detector geometries in future colliders

Farouk Mokhtar,[1,*] Joosep Pata,[2,†] Dolores Garcia,[3] Eric Wulff,[3] Mengke Zhang,[1] Michael Kagan,[4] and Javier Duarte[1]

[1] *University of California San Diego, La Jolla, USA*
[2] *National Institute of Chemical Physics and Biophysics, Tallinn, Estonia*
[3] *European Center for Nuclear Research (CERN), Geneva, Switzerland*
[4] *SLAC National Accelerator Laboratory, Stanford, USA*

(Dated: March 25, 2025)



Farouk Mokhtar, Joosep Pata, Dolores Garcia, Eric Wulff, Mengke Zhang, Michael Kagan, Javier Duarte, Fine-tuning machine-learned particle-flow for new detector geometries in future colliders https://arxiv.org/abs/2503.00131

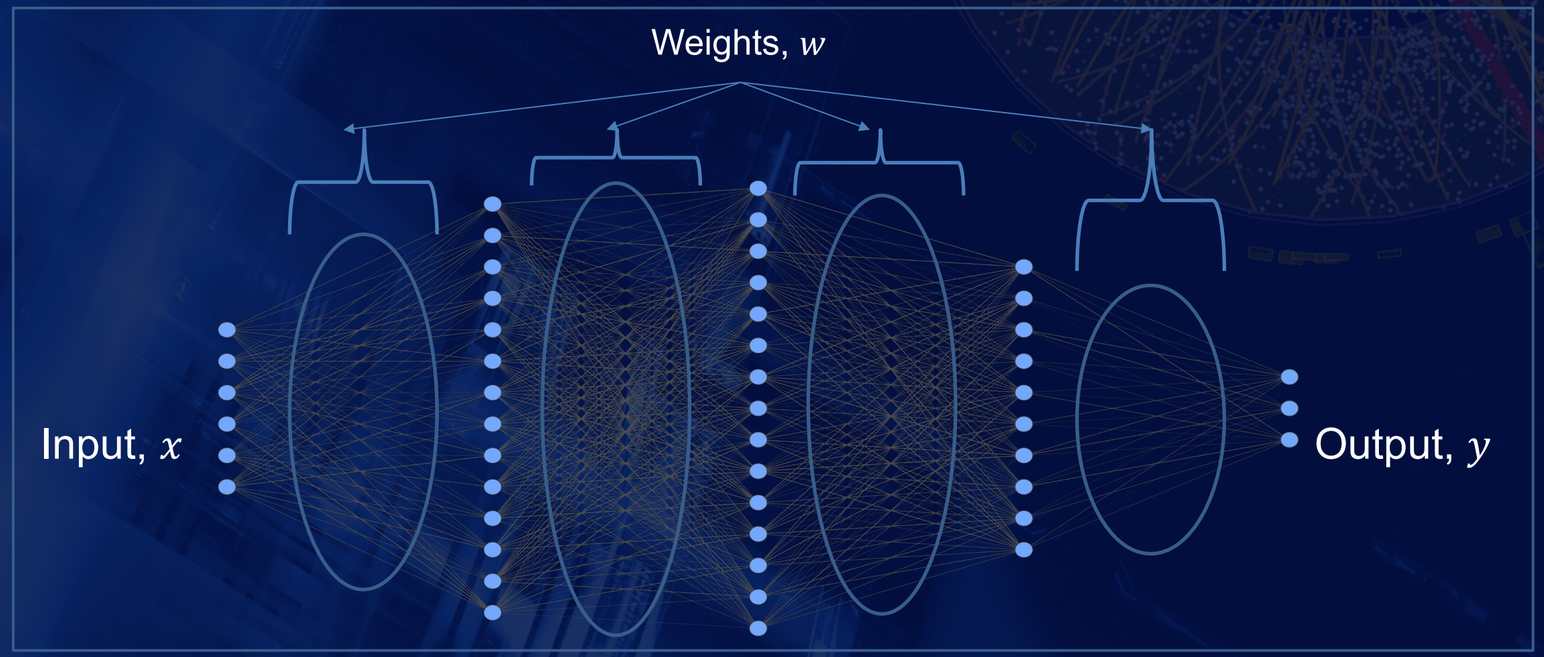
Hyperparameter optimization (HPO) (1)

Hyperparameter optimization (HPO) (1)

- $f(w, \theta)$, depends not only on w, but also on θ
  - w: Model parameters
  - θ: Hyperparameters
    - Number of layers or nodes
    - Choice of optimizer, learning rate, batch size, etc.
- Hyperparameter optimization (HPO) is the process of tuning θ to improve performance

f is the final validation loss after completed training, f is not the model itself

Model: $M(x, w) = y$

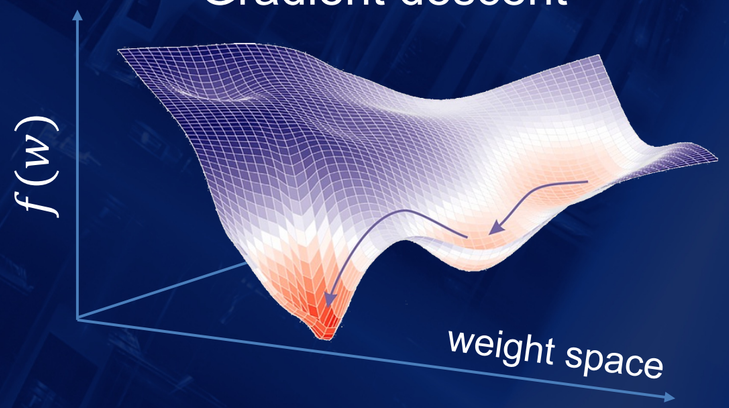


Hyperparameter optimization (HPO) (2)

- Optimizing the objective $f(w, \theta)$ is done in two different ways
  1. Training: Optimize f w.r.t. w by gradient descent $\Longrightarrow$ search for $w^* = \arg\min_w f(w, \theta)$
  2. HPO: Optimize f w.r.t. $\theta \Longrightarrow$ search for $\theta^* = \arg\min_\theta f(w, \theta)$

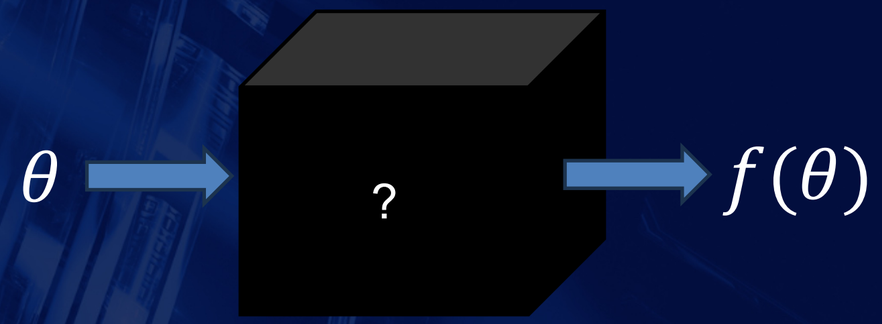
$f(w, \theta)$ is differentiable w.r.t. w

Gradient descent



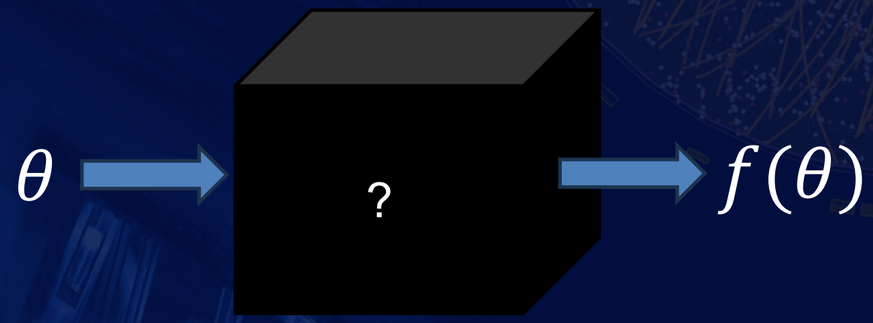
$f(w, \theta)$ non-differentiable w.r.t. θ

- Black-box optimization
- No straightforward update rule for θ



Hyperparameter optimization (HPO) (3)

- We want to find $\theta^* = \arg\min_\theta f(w, \theta)$ but only get to query values of f, not compute its gradient w.r.t. θ
  - w: Model parameters (learned by gradient descent)
  - θ: Hyperparameters
  - $f(w, \theta)$: What we're trying to minimize, e.g., loss
  - f is non-differentiable w.r.t. θ
- f is often expensive to evaluate
- HPO is compute-resource intensive
  - Benefits greatly from HPC resources
  - In need of smart, efficient search algorithms



Some popular HPO algorithms

- Search algorithms
  - Model free:
    - Grid search
    - Random search
    - Evolutionary search
  - Model-based:
    - Bayesian optimization
- Scheduling algorithms
  - Successive Halving (SHA)
  - Hyperband
  - Asynchronous SHA (ASHA)
  - Resource Adaptive Successive Doubling (RASDA)

Static configuration ***selection***

Adaptive configuration ***selection***

Adaptive configuration ***evaluation***

Search and scheduling algorithms can easily be combined

Search algorithms for HPO

Grid and random search

- Grid search
  - Deterministic
  - Exhaustive search (on the grid)
  - Uses same value several times
- Random Search
  - Stochastic
  - Exhaustive search (on the random points)
  - Explores many more values of each HP

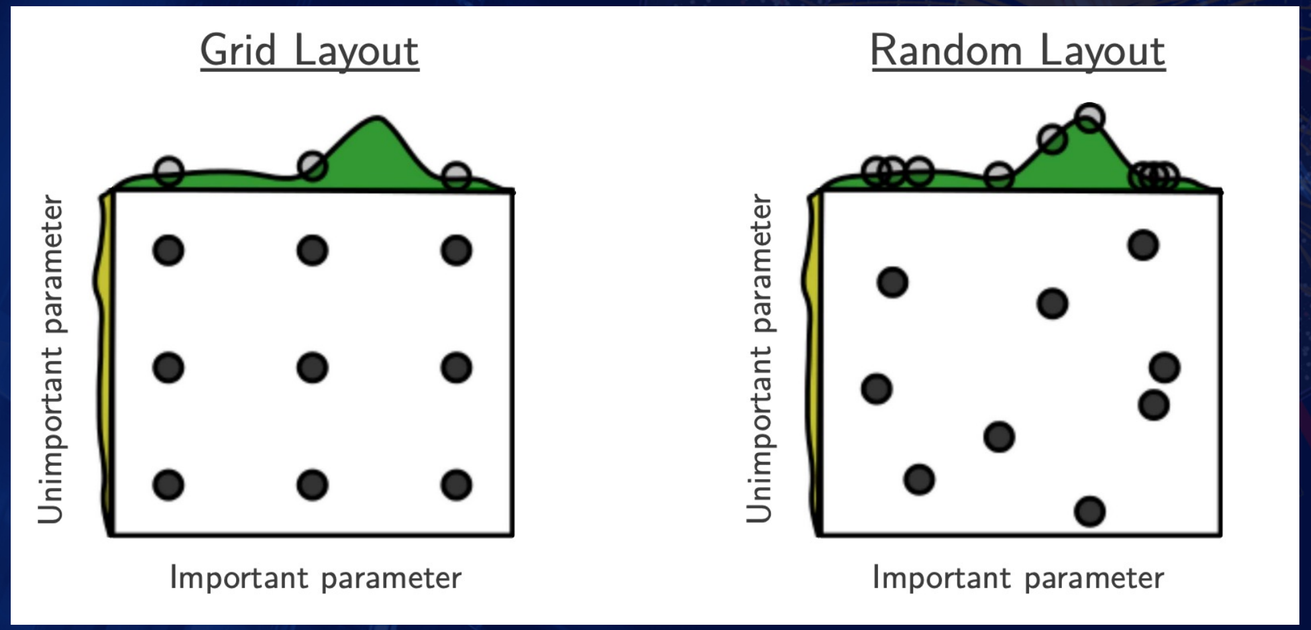


Figure from: James Bergstra and Yoshua Bengio: Random Search for Hyper-Parameter Optimization, https://www.jmlr.org/papers/volume13/bergstra12a/bergstra12a.pdf

Bayesian optimization (1)

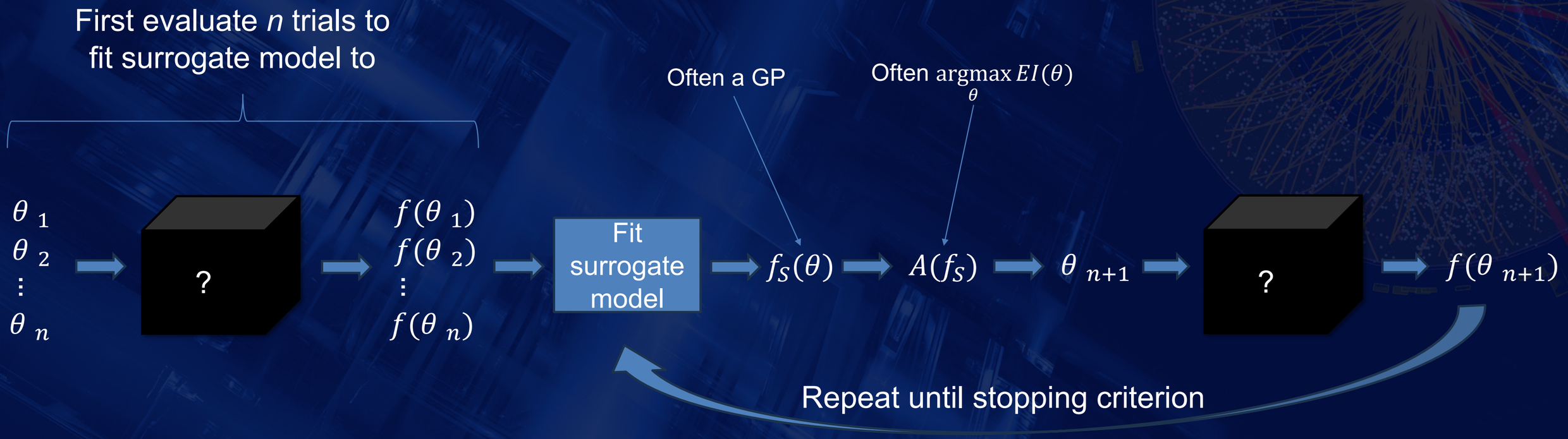
Bayesian optimization (1)

- Bayesian Optimization (BO) is a black-box optimization technique for expensive and/or noisy objectives

- Surrogate model
  - Estimates $f(\theta)$, given some HPs θ
  - Estimates uncertainty of the objective function estimate
  - Must be much faster than evaluating f

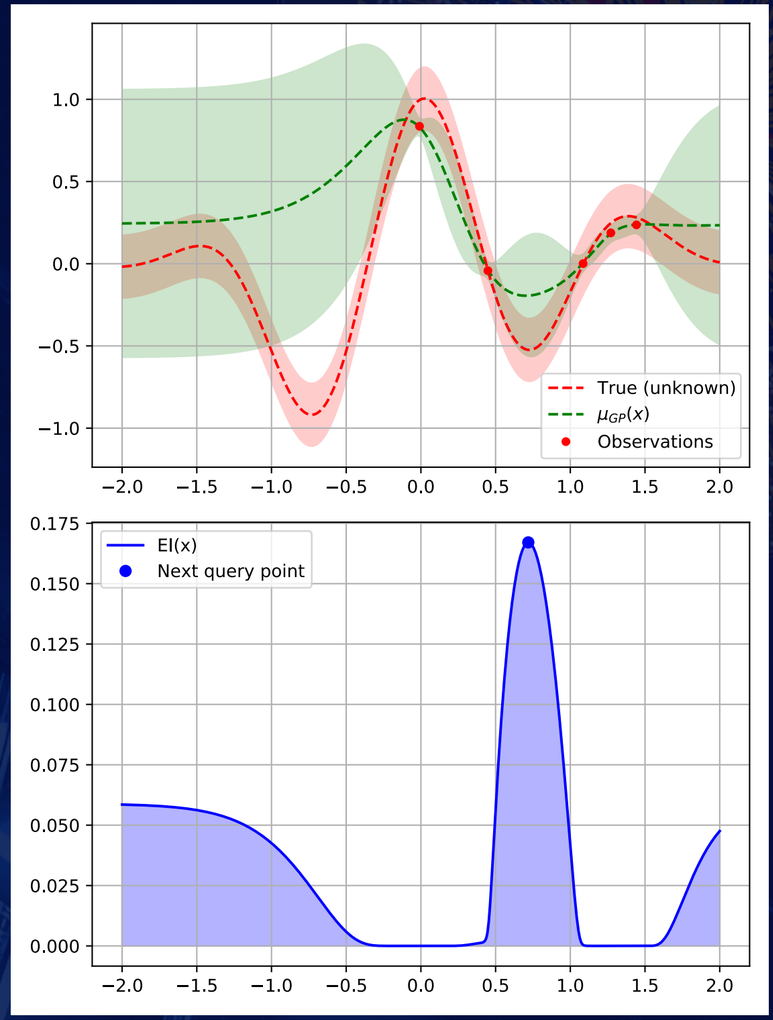
- Acquisition function
  - Selects next θ to evaluate
  - Makes exploitation/exploration trade-off
  - Popular choice: Expected Improvement (EI), i.e., how much better is the next observation going to be over our current best?

Bayesian optimization (2)



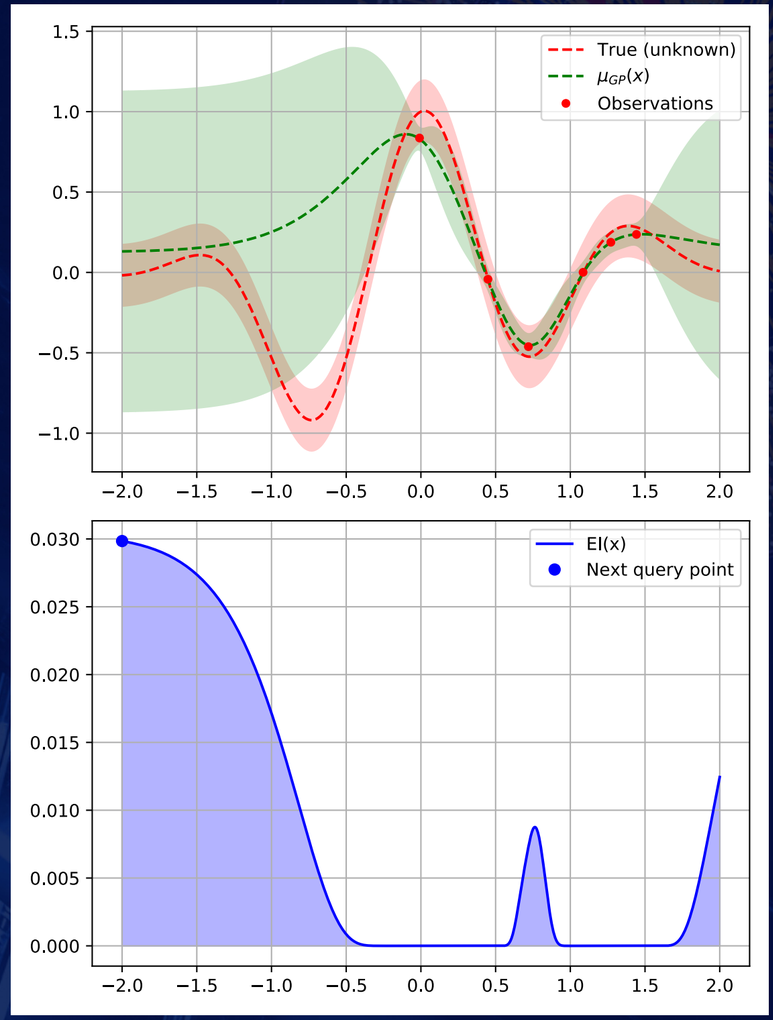
Bayesian optimization (3)

- Let’s visualize the BO process
- In this example we have
  - a Gaussian Process as the surrogate model and
  - use EI as the acquisition function



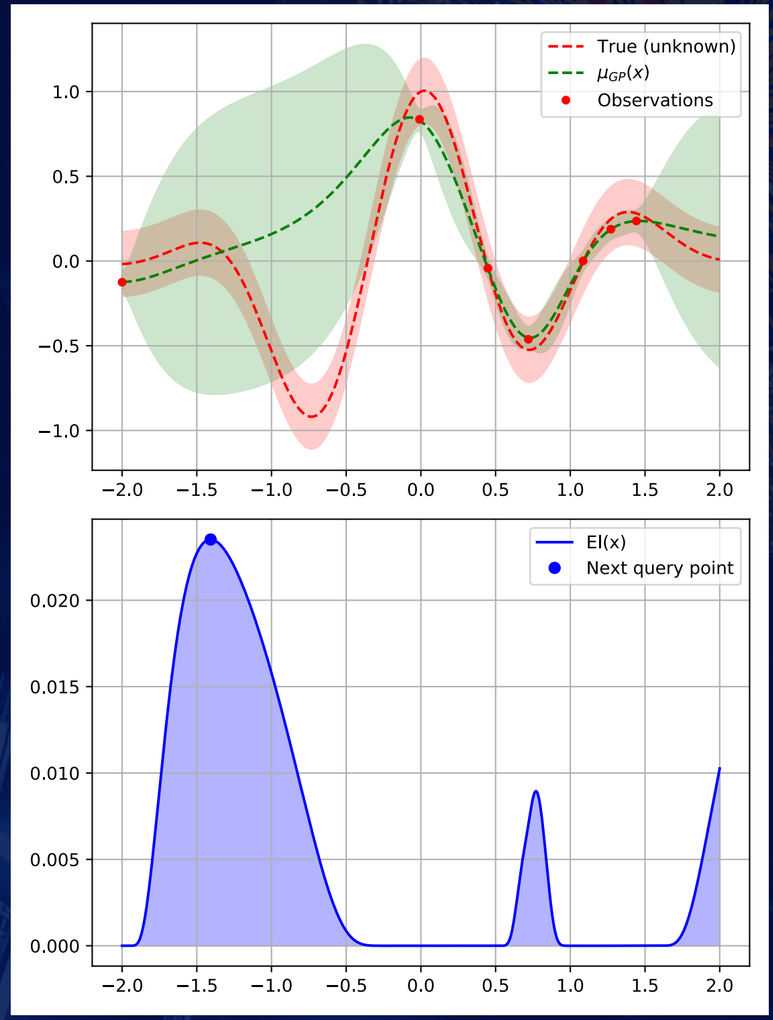
Bayesian optimization (3)

- Let’s visualize the BO process
- In this example we have
  - a Gaussian Process as the surrogate model and
  - use EI as the acquisition function



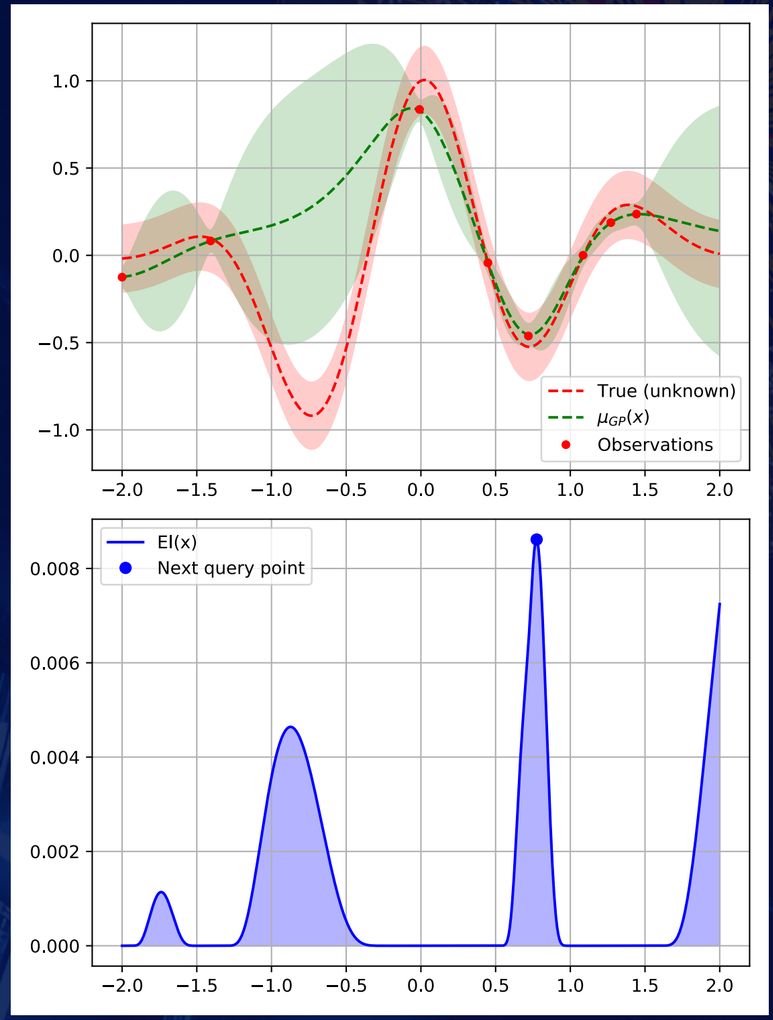
Bayesian optimization (3)

- Let’s visualize the BO process
- In this example we have
  - a Gaussian Process as the surrogate model and
  - use EI as the acquisition function



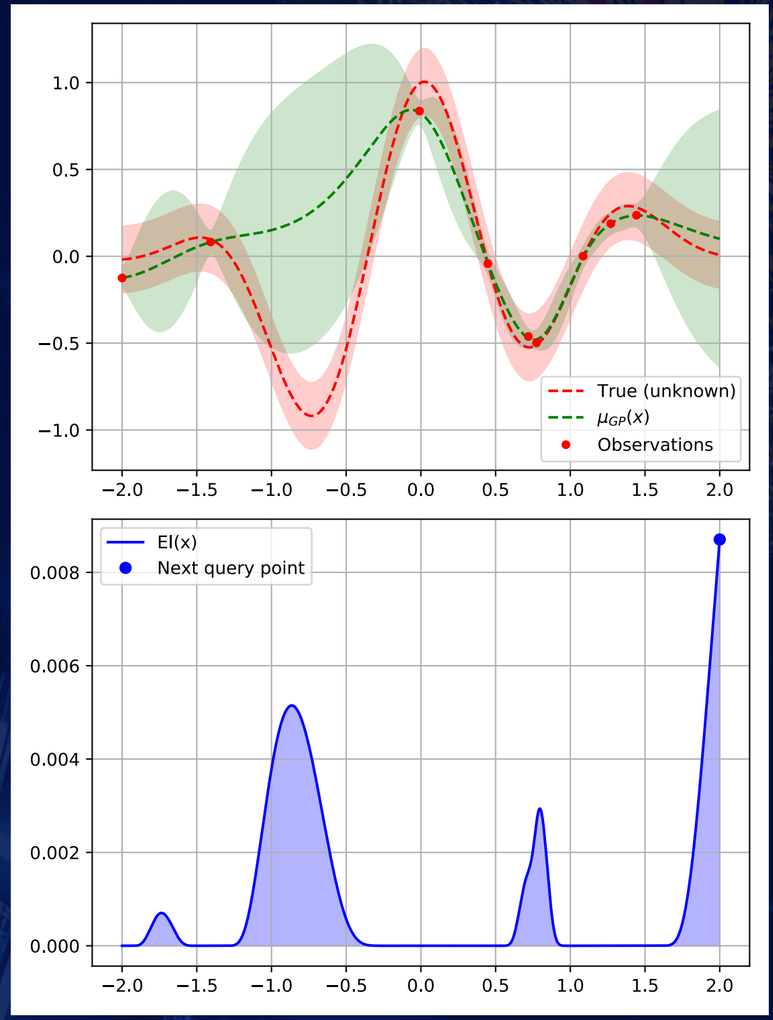
Bayesian optimization (3)

- Let’s visualize the BO process
- In this example we have
  - a Gaussian Process as the surrogate model and
  - use EI as the acquisition function



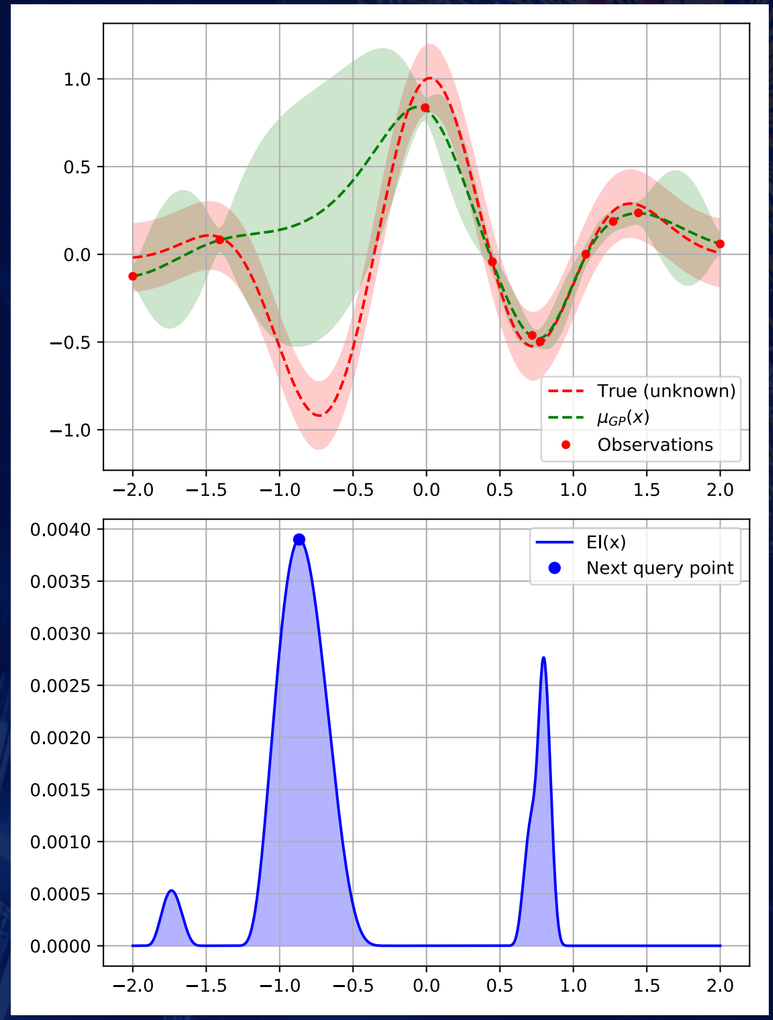
Bayesian optimization (3)

- Let’s visualize the BO process
- In this example we have
  - a Gaussian Process as the surrogate model and
  - use EI as the acquisition function



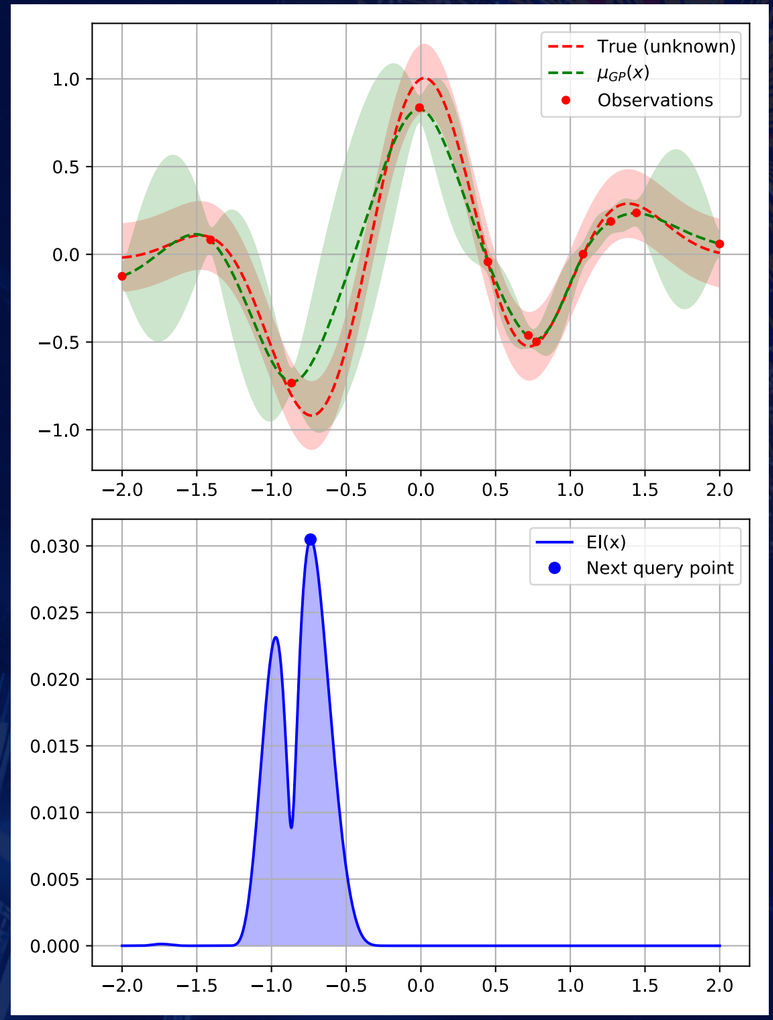
Bayesian optimization (3)

- Let’s visualize the BO process
- In this example we have
  - a Gaussian Process as the surrogate model and
  - use EI as the acquisition function



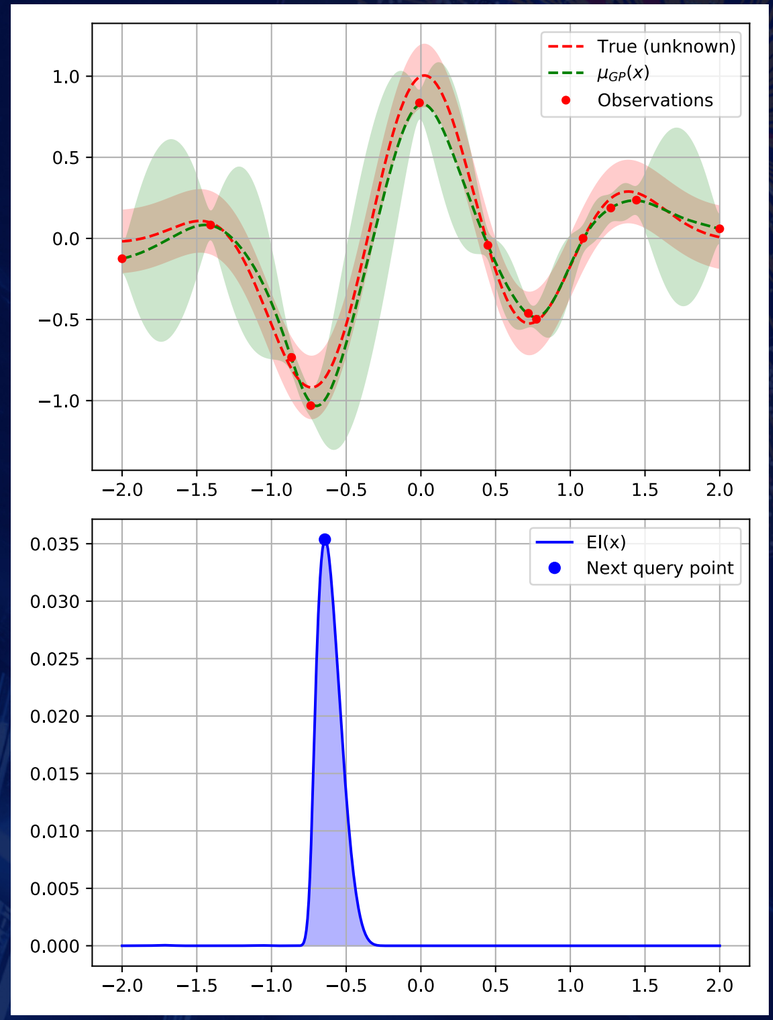
Bayesian optimization (3)

- Let’s visualize the BO process
- In this example we have
  - a Gaussian Process as the surrogate model and
  - use EI as the acquisition function



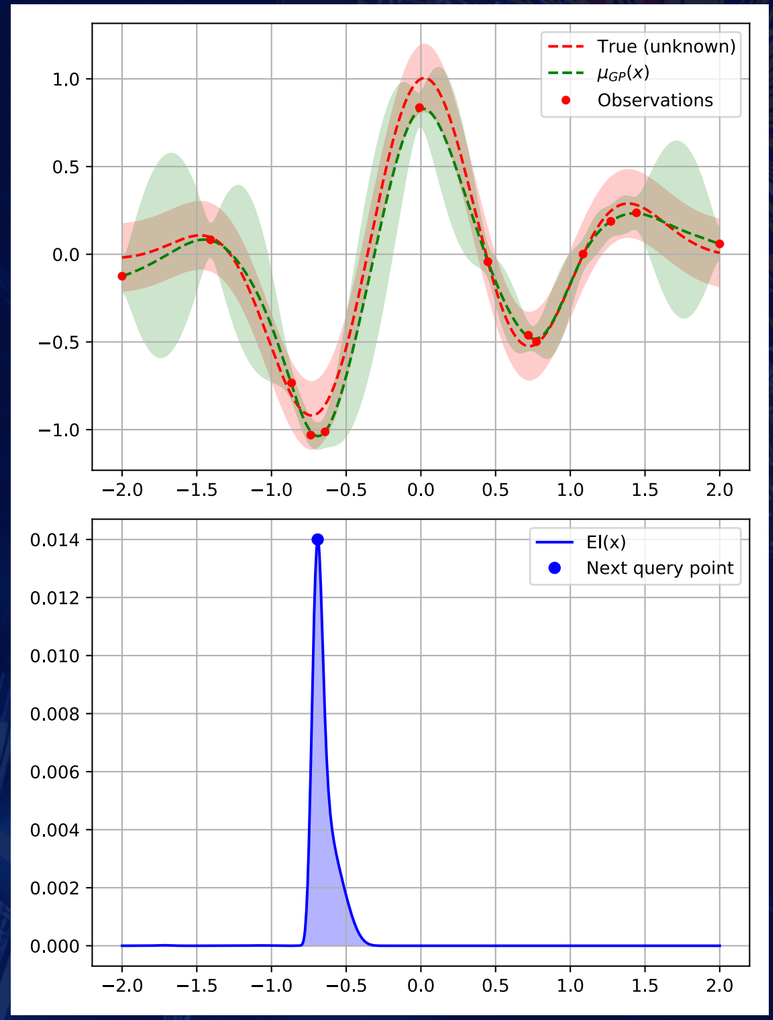
Bayesian optimization (3)

- Let’s visualize the BO process
- In this example we have
  - a Gaussian Process as the surrogate model and
  - use EI as the acquisition function



Bayesian optimization (3)

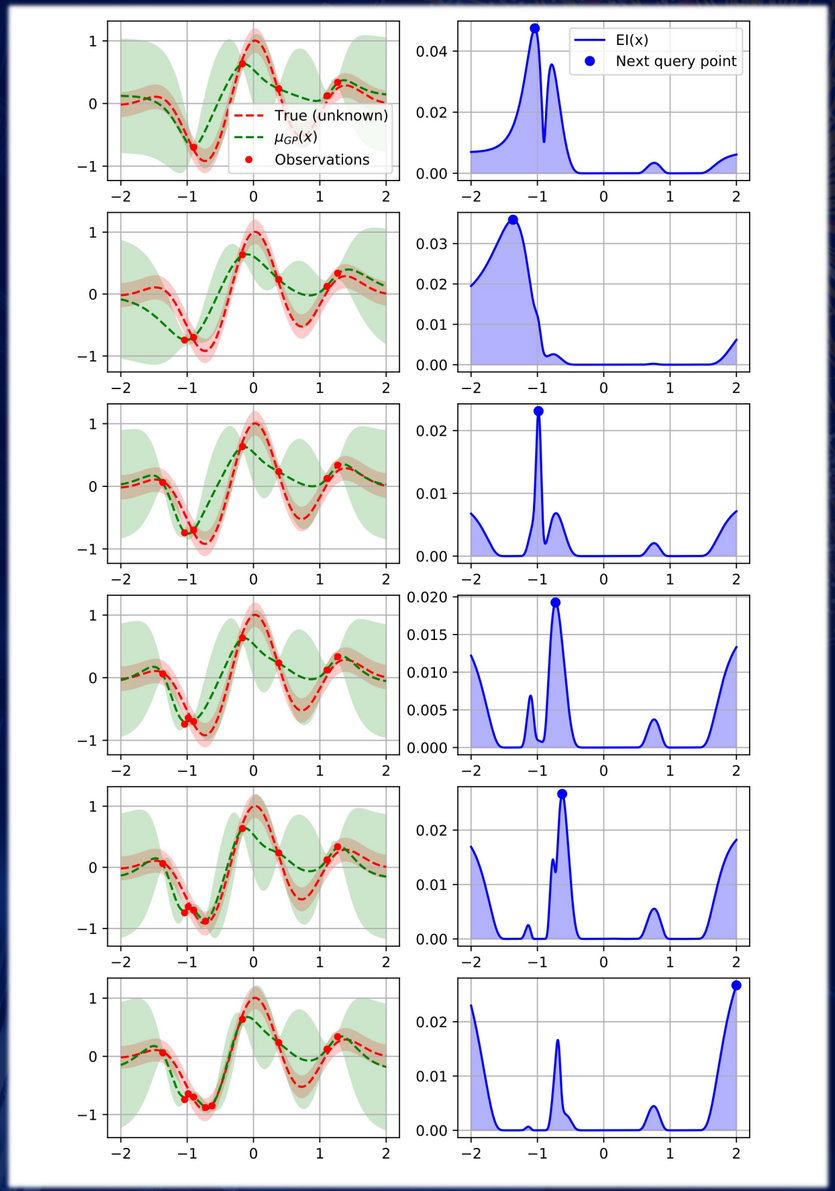
- Let’s visualize the BO process
- In this example we have
  - a Gaussian Process as the surrogate model and
  - use EI as the acquisition function



Bayesian optimization (4)

- Choices of surrogate model
  - Gaussian Process (GP)
    - Closed form
    - Runtime complexity: $O(n^3)$
  - Random Forest
    - Ensemble of decision trees
    - Faster than GP
    - Runtime complexity: $O(nlog(n))$
  - Bayesian Neural Network
    - NN with uncertainty estimates built-in
    - Very flexible
    - Requires more training data
  - Tree-structured Parzen Estimator (TPE)
    - Fast
    - Simple and non-parametric
    - Runtime complexity: $O(nlog(n))$

Example using Gaussian Process and Expected Improvement



Parallel Bayesian optimization

- BO as discussed up until now is sequential, it waits for an evaluation to complete before selecting a new set of HPs to try
- With modern computing and HPC, we can run many trials in parallel
  - Must ensure to never evaluate same θ more than once since that would be very inefficient
- One strategy is to
  - Evaluate some given number of trials to get a set of observations to fit first surrogate
  - Pick next θ as described previously
  - If more resources are available, modify acquisition function to penalize θs that are currently being evaluated but haven't completed yet
    - One way of doing this is by reducing the variance of the surrogate model at those points, θ

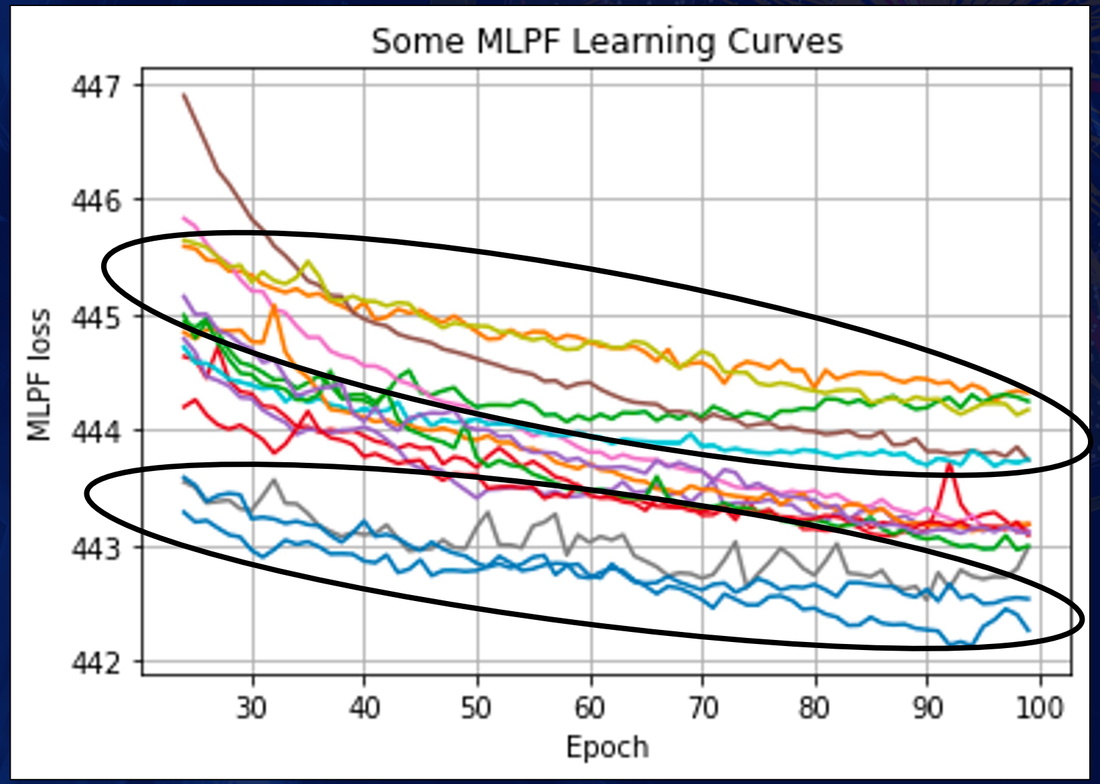
Scheduling algorithms for HPO

Adaptive configuration evaluation

- It is possible to identify badly performing trials early so why train them to convergence?
- Adaptive configuration evaluation strategies terminate badly performing trials early

Some examples include:

- Successive Halving Algorithm (SHA)
  - Terminate some fraction of trials according given stopping rate s
- Hyperband
  - Loop over SHA using different stopping rates s
- Asynchronous Successive Halving Algorithm (ASHA)
  - Async version of SHA



Successive Halving Algorithm (SHA)

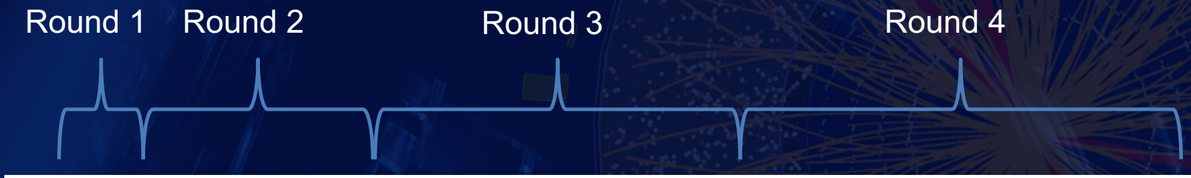
Bracket

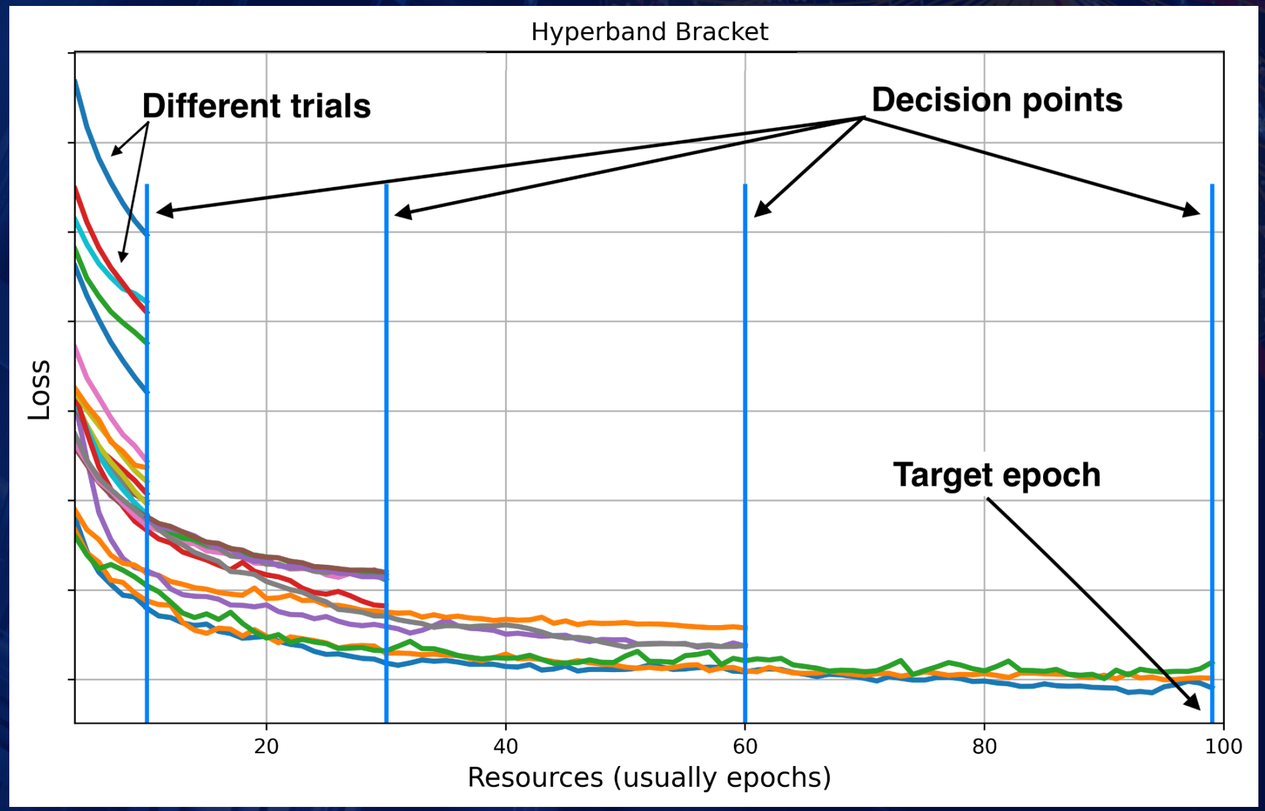
Round

1. Partially train some trials up to a decision point
2. Evaluate performance and throw out worst x%
3. Repeat 1-2 until target epoch is reached or only 1 trial remains

Hyperband adds a 4th step

4. Repeat 1-3 for different sets of decision points

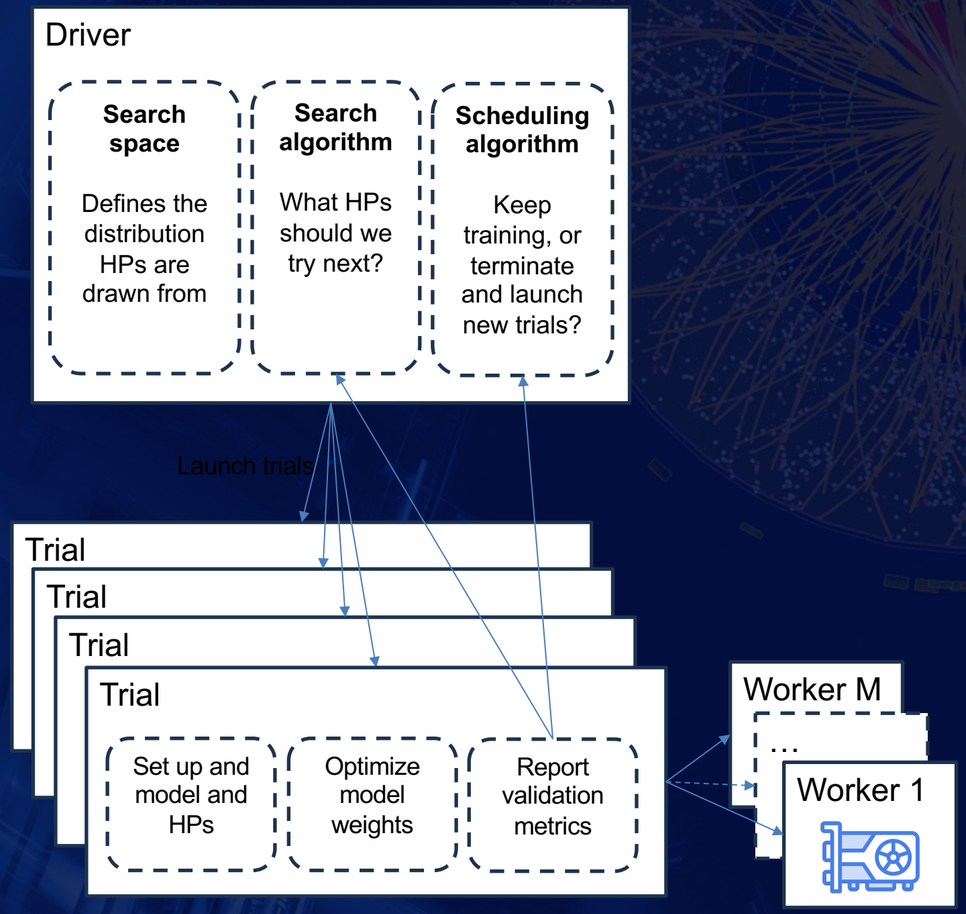




Hyperparameter optimization on HPC in practice

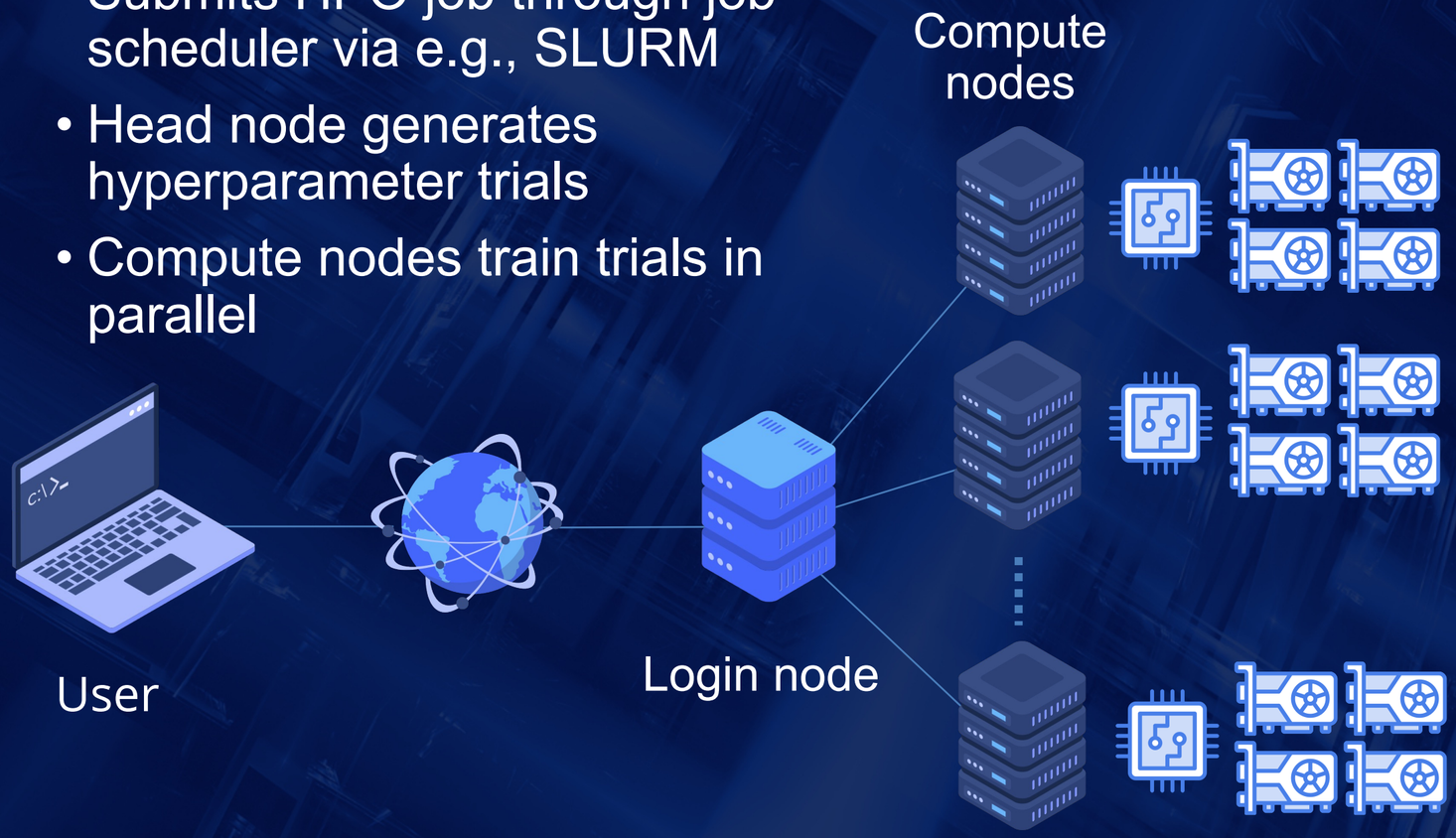
Distributed HPO workflow

- Driver
  - Defines search space
  - Generates hyperparameter trials
  - Launches, monitors, and terminates trials
- Trials
  - Sets up model and HPs
  - Distributes model training across M workers
  - Reports metrics back to driver



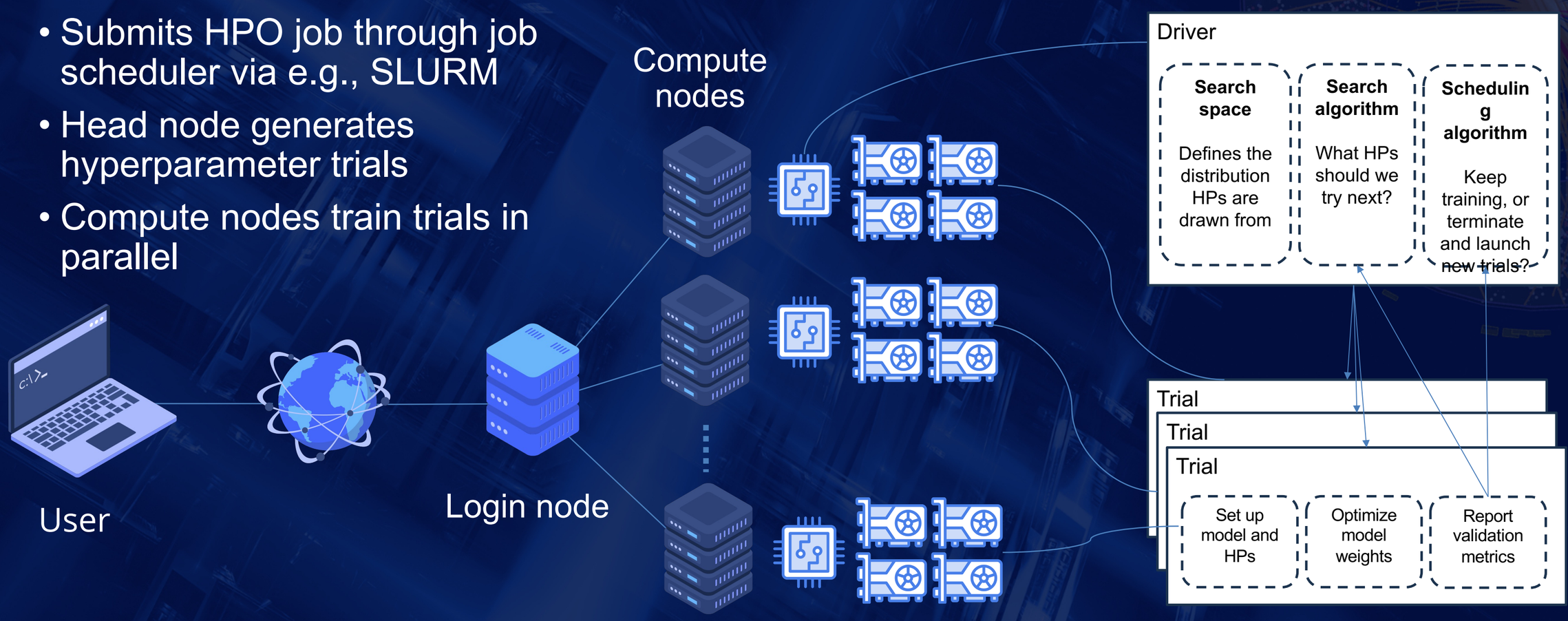
Distributed HPO on HPC systems

- User connects to HPC via ssh
- Submits HPO job through job scheduler via e.g., SLURM
- Head node generates hyperparameter trials
- Compute nodes train trials in parallel

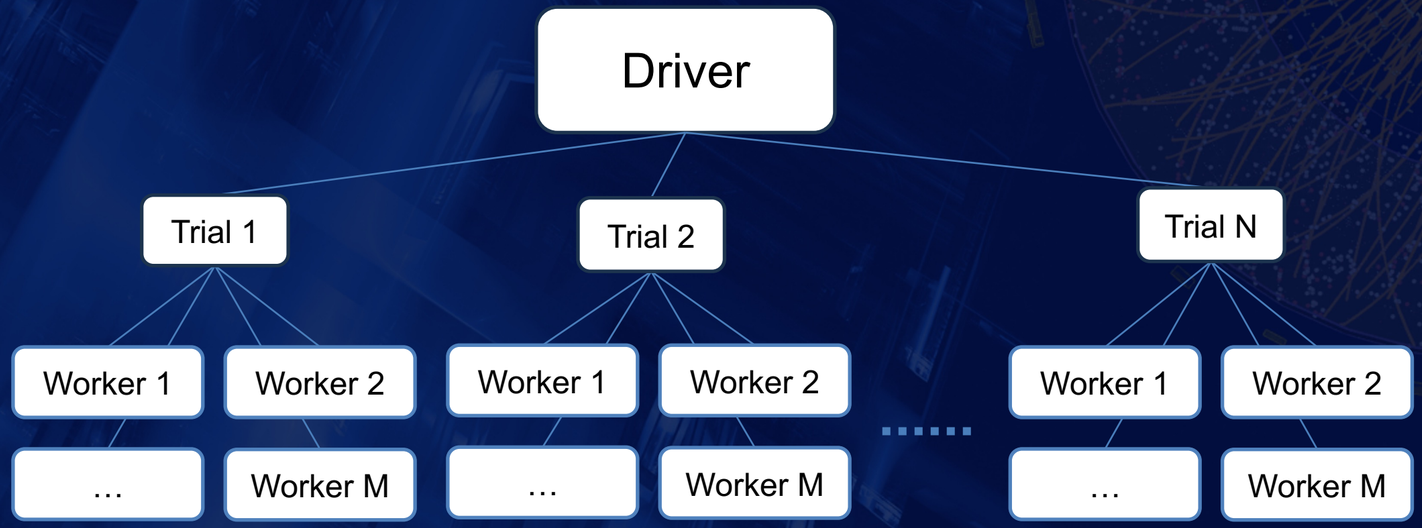


Distributed HPO on HPC systems

- User connects to HPC via ssh
- Submits HPO job through job scheduler via e.g., SLURM
- Head node generates hyperparameter trials
- Compute nodes train trials in parallel



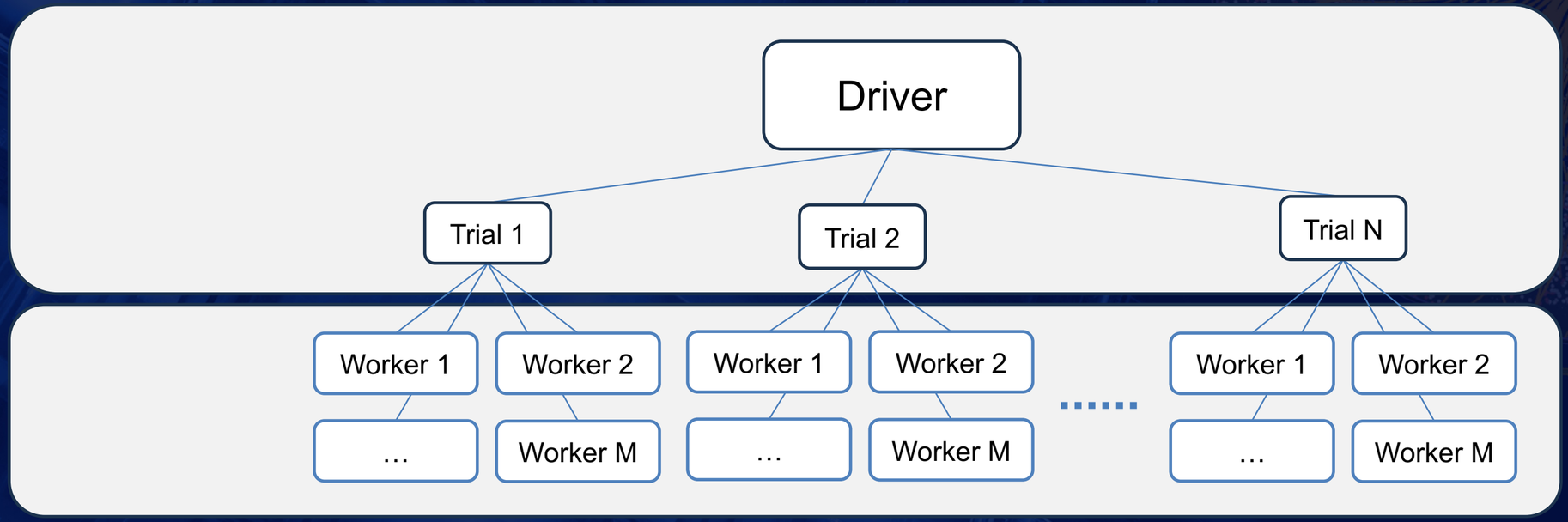
Two levels of parallelization



Two levels of parallelization

Hyperparameter Optimization

Distributed training



AI & HPC in practice: Distributed training and HPO for MLPF

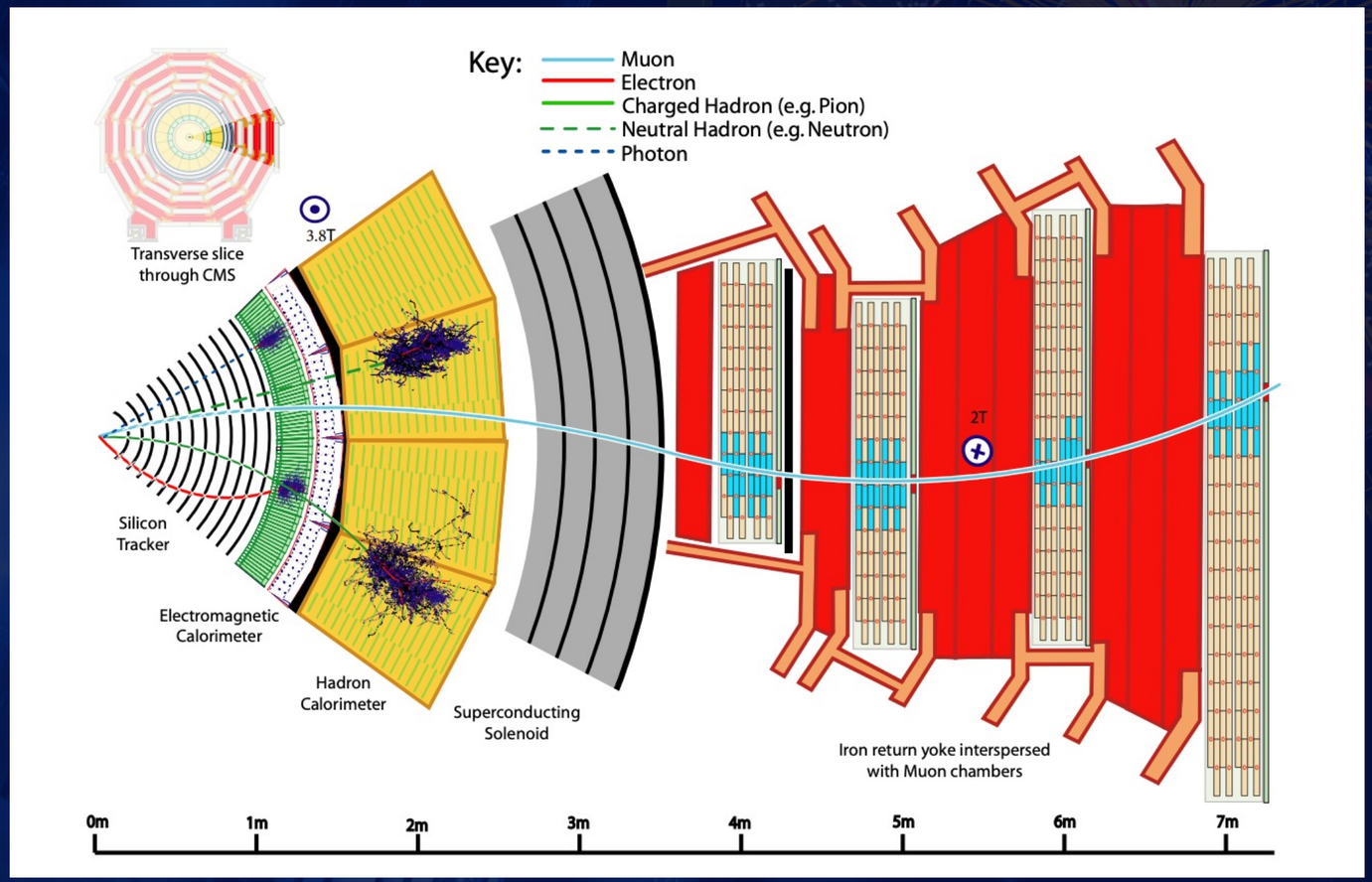
In collaboration with the CMS Experiment.
Eric and David were supported by the EC-funded CoE RAISE project.

Javier Duarte, Dolores Garcia, Maria Girone, Michael Kagan, Farouk Mokhtar, Joosep Pata, David Southwick, Eric Wulff Mengke Zhang

Event reconstruction at the LHC (1/3)

- Particle detectors at the LHC are extremely complex, with many subdetectors
- Particles interact with the detectors and leave tracks and energy deposits
- Information from subdetectors are combined to produce a particle-level interpretation of the event
- Event reconstruction is the process of inferring higher-level physics objects from detector signals

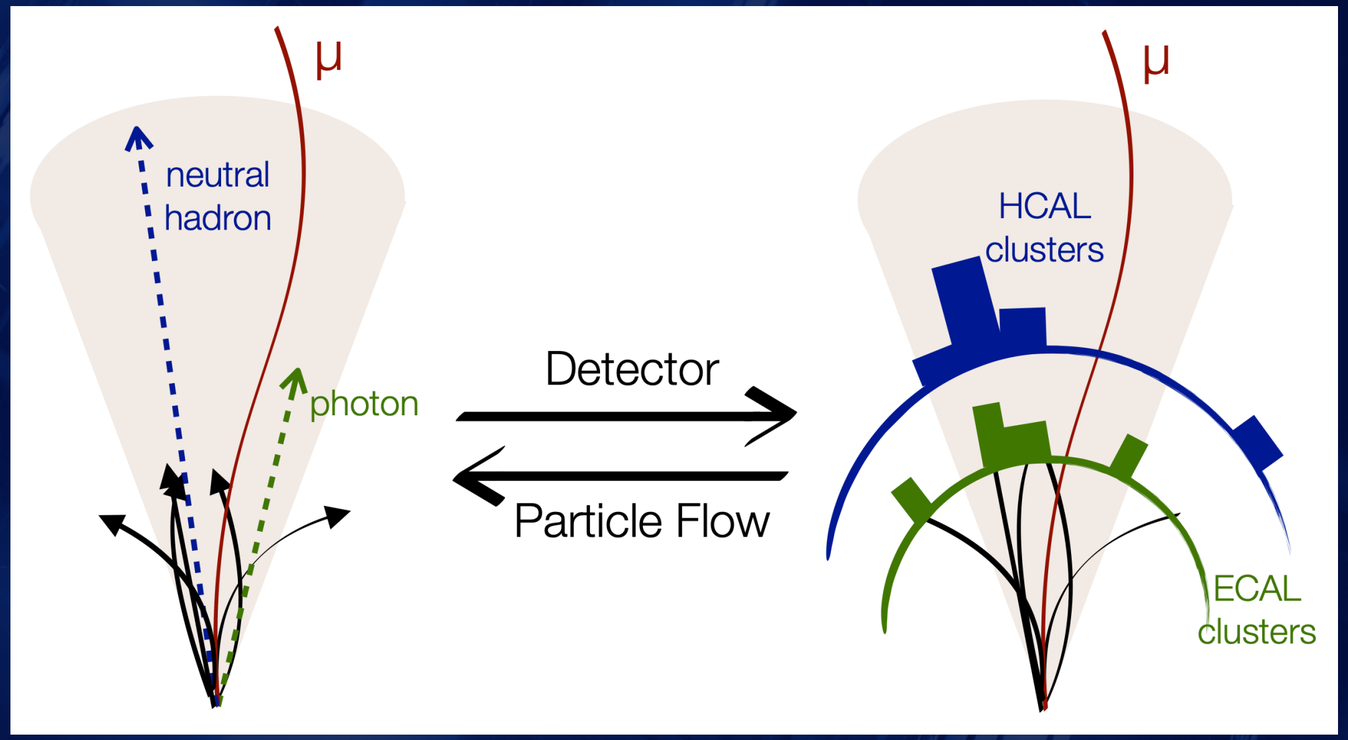
Transverse slice through the CMS detector



JINST 12 (2017) P10003

Event reconstruction at the LHC

- Event reconstruction attempts to solve the inverse problem of particle-detector interactions, i.e., going from detector signals back to the particles that gave rise to them
- Particle-flow (PF) reconstruction takes tracks and clusters of energy deposits as input and gives particle types and momenta as output

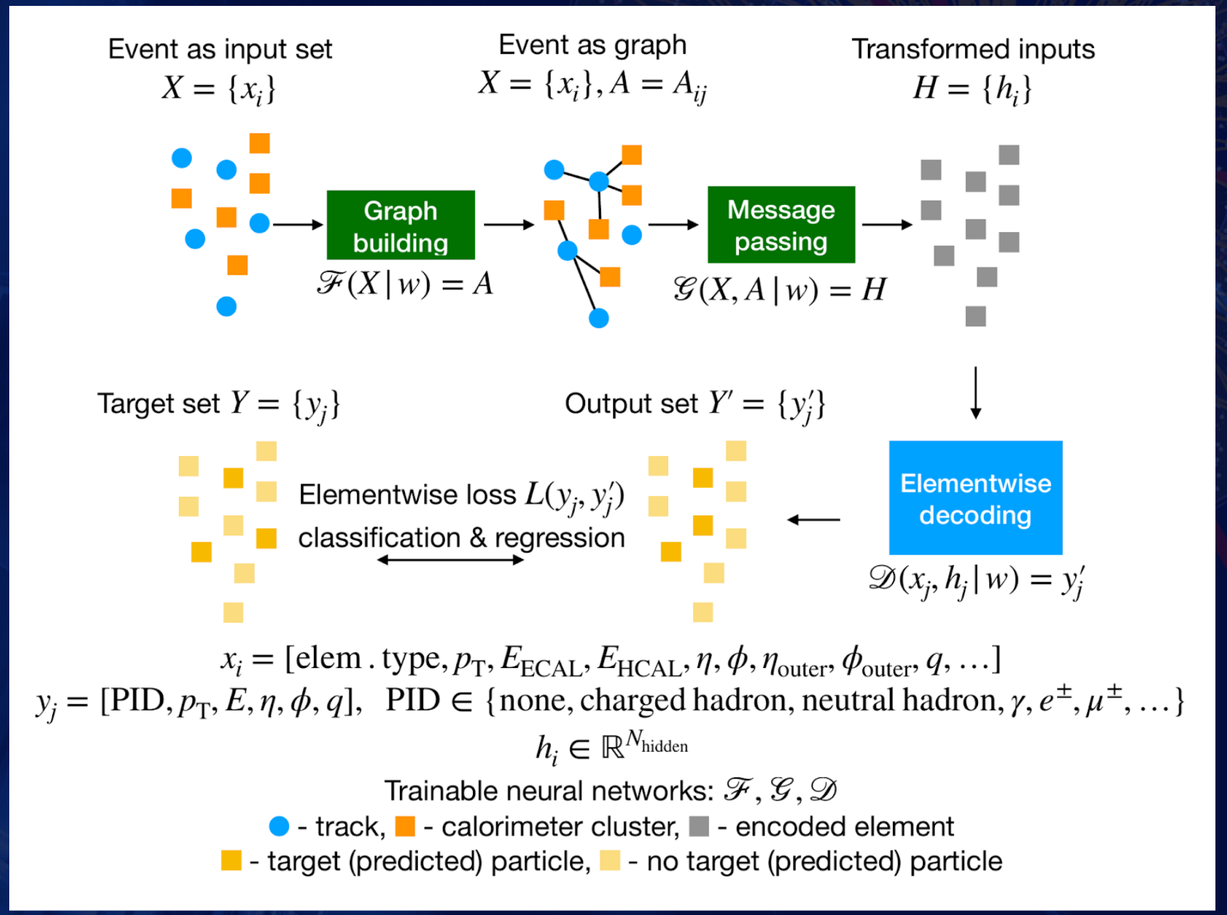


Machine-Learned Particle Flow (MLPF)

- The Particle Flow (PF) Algorithm [1]
  - Tries to identify and reconstruct all stable individual particles from collision events by combining information from different subdetectors (tracks, calorimeter clusters)
- Machine-Learned Particle-Flow (MLPF) [2]
  - GPU accelerated, NN-based algorithm for PF
  - Code available on GitHub
    - Details on TensorFlow GNN model: ACAT2021 talk by J. Pata (and proceedings)
    - Details on HPO: ACAT 2021 talk by E. Wulff (and proceedings)
    - Details on PyTorch Transformer model: EPS-HEP talk by F. Mokhtar
    - MLPF in CMS, CLIC and CLD: FCC-ee workshop presentation by F. Mokhtar
  - See Nature Commun. Phys. 2024 and Phys. Rev. D 111, 092015, 2025 for latest published results.
  - Latest public CMS results: CMS-DP-2025-033
    - We aim for a CMS paper in the near future.

[1] CMS Collaboration https://cds.cern.ch/record/1194487?ln=en

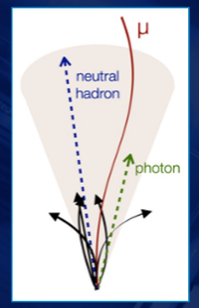
The MLPF model



[2] Pata, J., Duarte, J., Vlimant, JR. *et al.* MLPF: efficient machine-learned particle-flow reconstruction using graph neural networks. *Eur. Phys. J. C* **81,** 381 (2021). https://doi.org/10.1140/epjc/s10052-021-09158-w

ML-based particle flow reconstruction workflow

Physics simulation

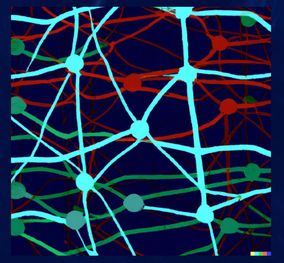


Data selection

Dataset creation

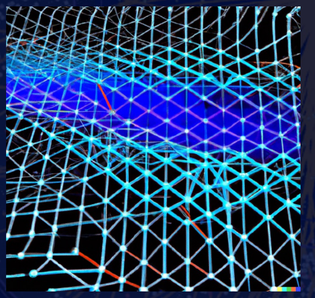
Data pre-processing

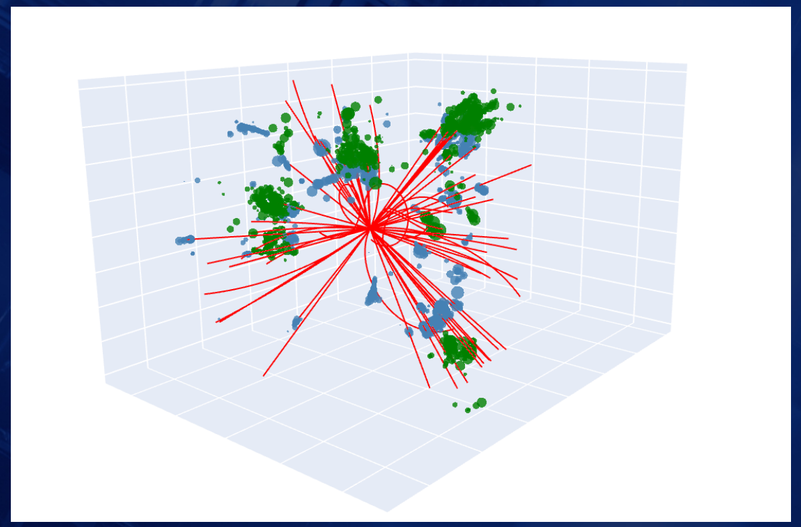
ML training



Model export

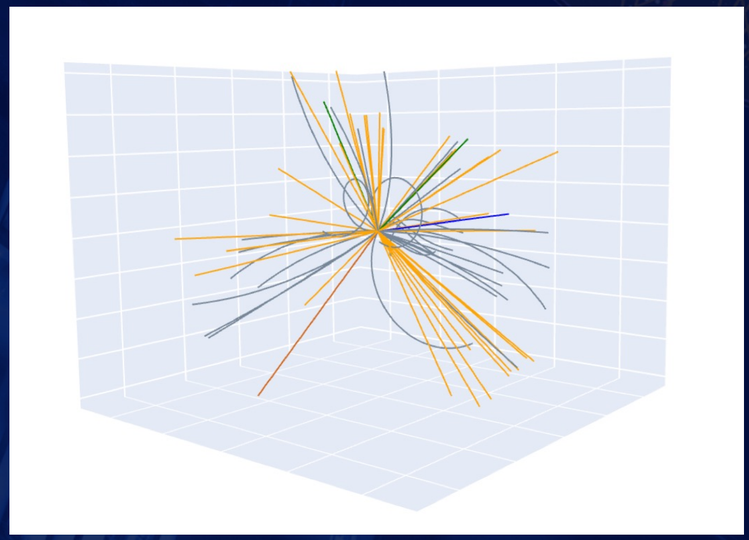
Trained model





Tracks and calorimetry

Event reconstruction



MLPF event reconstruction

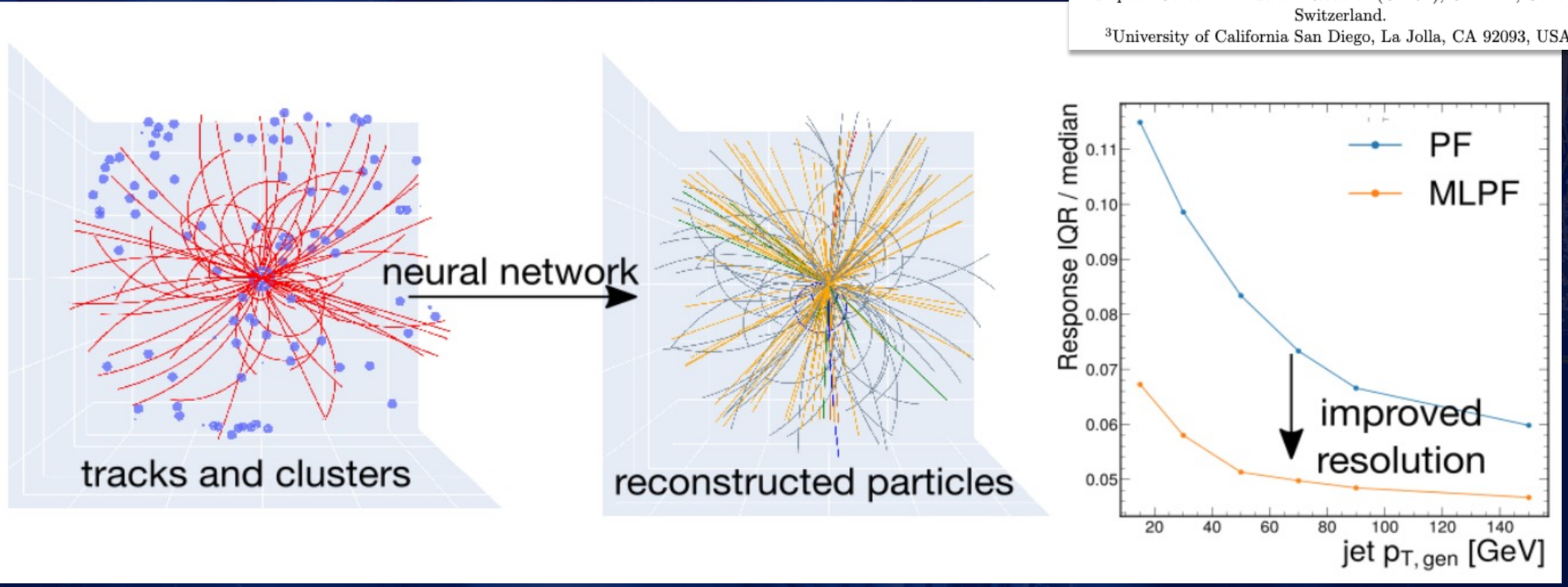
MLPF

Improves event reconstruction using ML while keeping models portable and aimed at possible future deployment scenarios

Improved particle-flow event reconstruction with scalable neural networks for current and future particle detectors

Joosep Pata[1*], Eric Wulff[2], Farouk Mokhtar[3], David Southwick[2], Mengke Zhang[3], Maria Girone[2], Javier Duarte[3]

[1*]National Institute of Chemical Physics and Biophysics (NICPB), Rävala pst 10, 10143 Tallinn, Estonia.
[2]European Center for Nuclear Research (CERN), CH 1211, Geneva 23, Switzerland.
[3]University of California San Diego, La Jolla, CA 92093, USA.



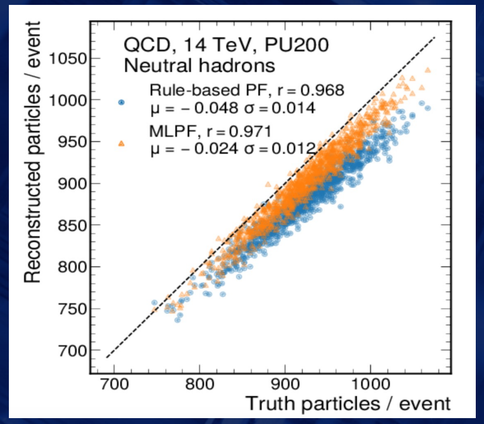
Based on electron-positron collision events in a CLIC-based detector geometry

Dataset: https://zenodo.org/records/8260741

Pata, J., Wulff, E., Mokhtar, F. et al. Improved particle-flow event reconstruction with scalable neural networks for current and future particle detectors. Commun Phys 7, 124 (2024). https://doi.org/10.1038/s42005-024-01599-5

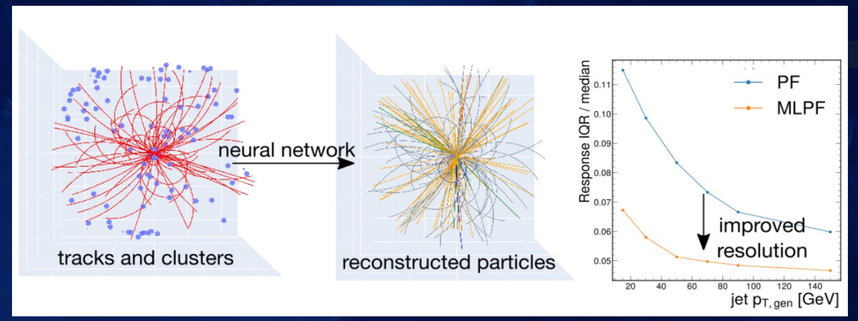
MLPF timeline

Proof of concept



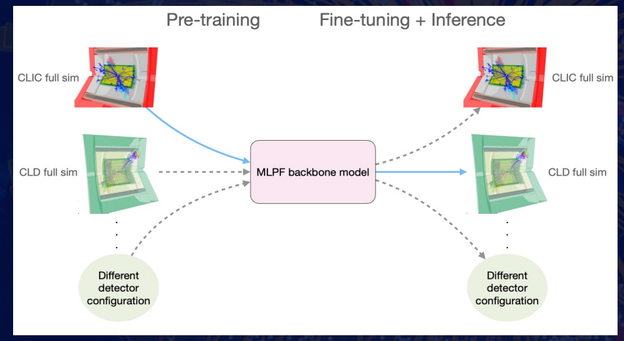
Eur. J. Phys. doi:10.1140/epjc/s10052-021-09158-w

Event-level SOTA on full simulation



Commun. Phys doi:10.1038/s42005-024-01599-5

Fine-tuning from one detector to another



Phys. Rev. D. doi:10.48550/arXiv.2503.00131

Open data

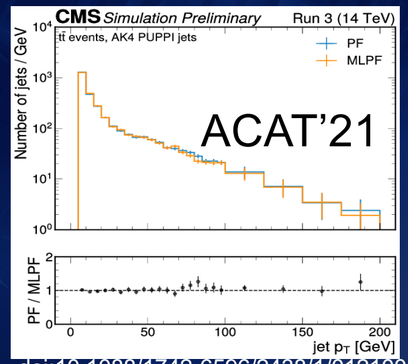
FastML'23, ML4Jets'23, ACAT'24

2020 2021 2022 2023 2024 2025

⏳ **CMS paper**

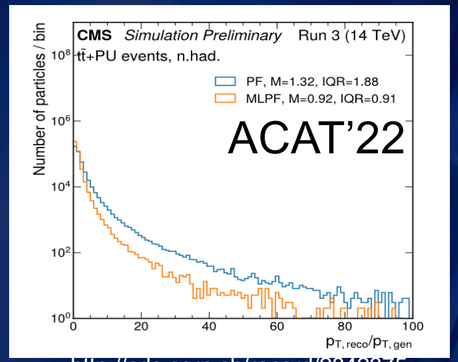
Real detector

Proof of concept

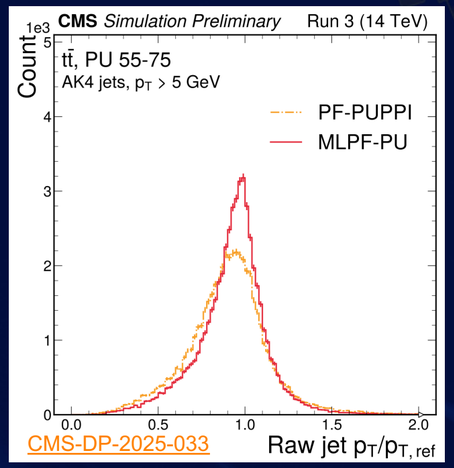


doi:10.1088/1742-6596/2438/1/012100
doi:10.1088/1742-6596/2438/1/012092

Particle-level SOTA



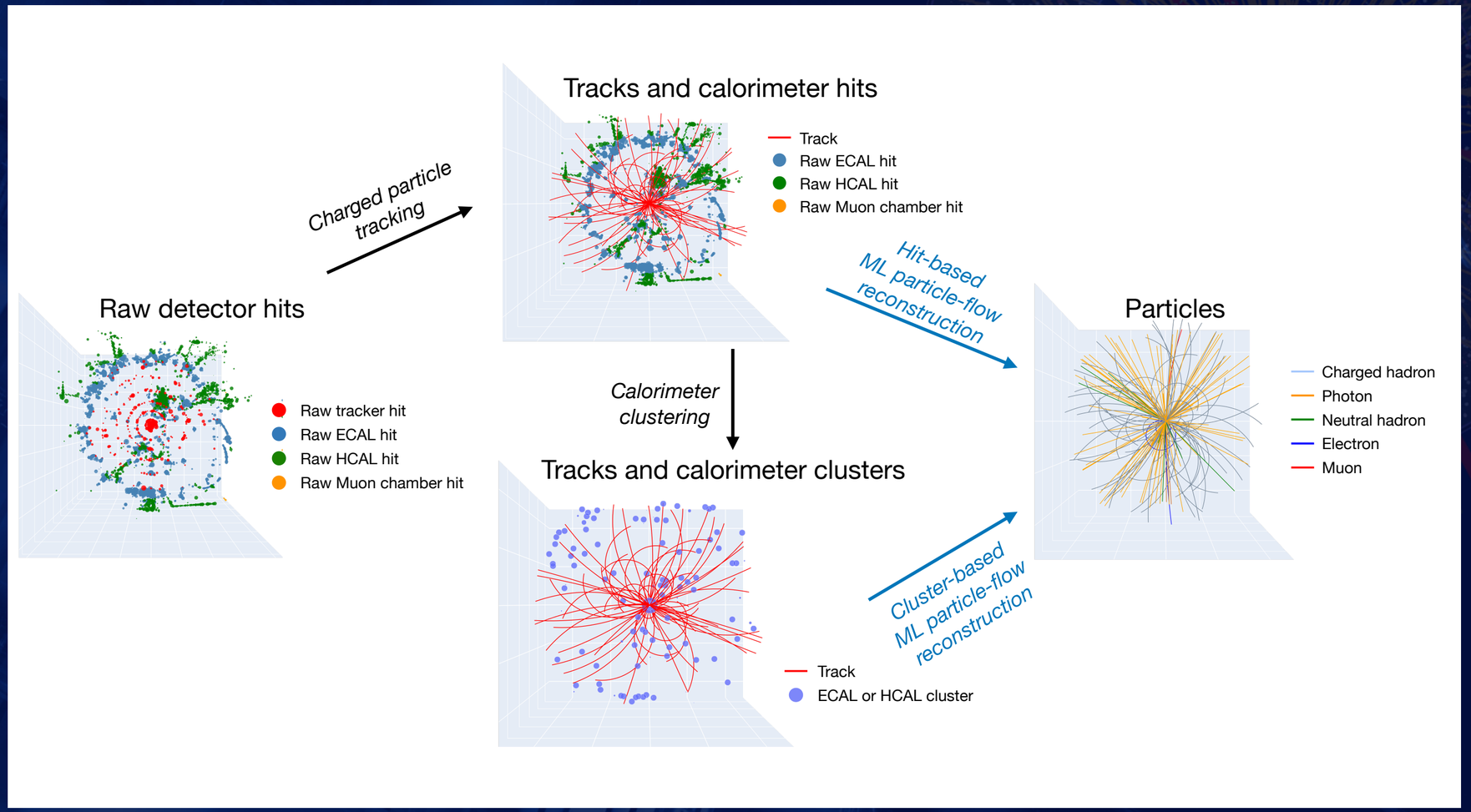
http://cds.cern.ch/record/2842375
https://arxiv.org/abs/2303.15053



CMS-DP-2025-033

Highly diverse and granular datasets

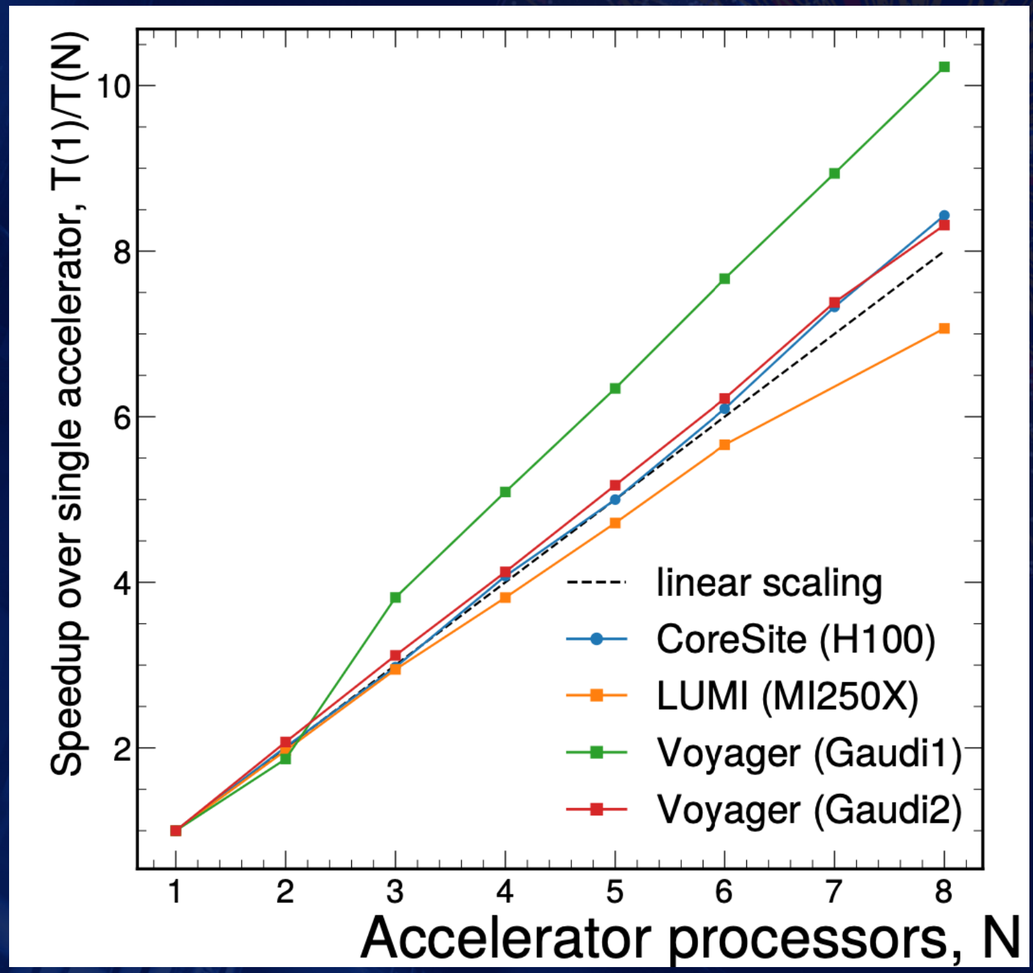
- Highly diverse and granular datasets suitable for tracking, clustering and particle flow reconstruction
- Can be used for both supervised and unsupervised learning



Pata, J., Wulff, E., Mokhtar, F. et al. Improved particle-flow event reconstruction with scalable neural networks for current and future particle detectors. Commun Phys 7, 124 (2024). https://doi.org/10.1038/s42005-024-01599-5

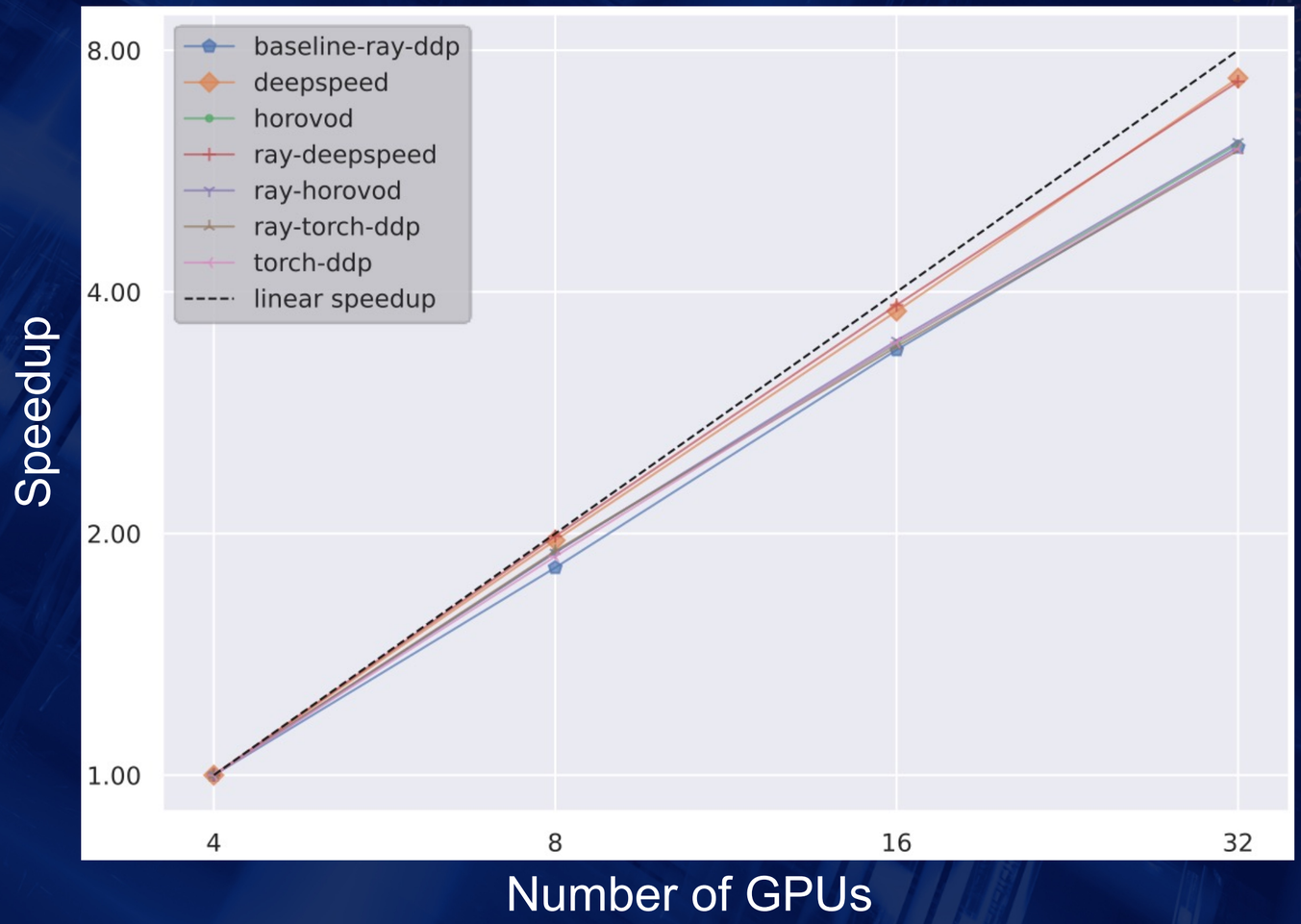
Portable distributed training

- The HPC AI chip landscape is diversifying, we need flexible and portable codes to make use of them
- MLPF training is portable and runs on CPU, NVIDIA and AMD GPUs as well as Intel Habana Gaudi cards
- We use PyTorch DDP to implement multi-GPU training



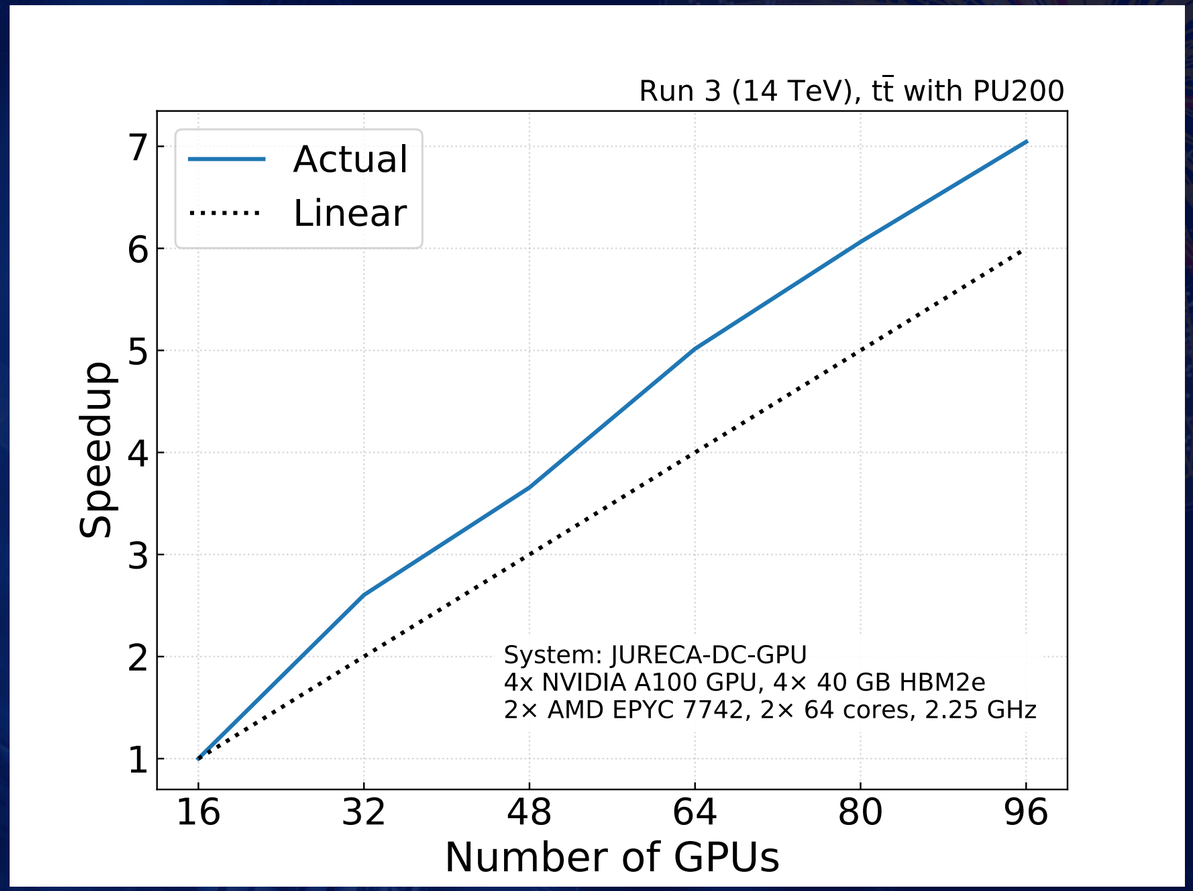
Pata, J., Wulff, E., Mokhtar, F. et al. Improved particle-flow event reconstruction with scalable neural networks for current and future particle detectors. Commun Phys 7, 124 (2024). https://doi.org/10.1038/s42005-024-01599-5

MLPF scaling on VEGA



Multi-node scaling of MLPF HPO

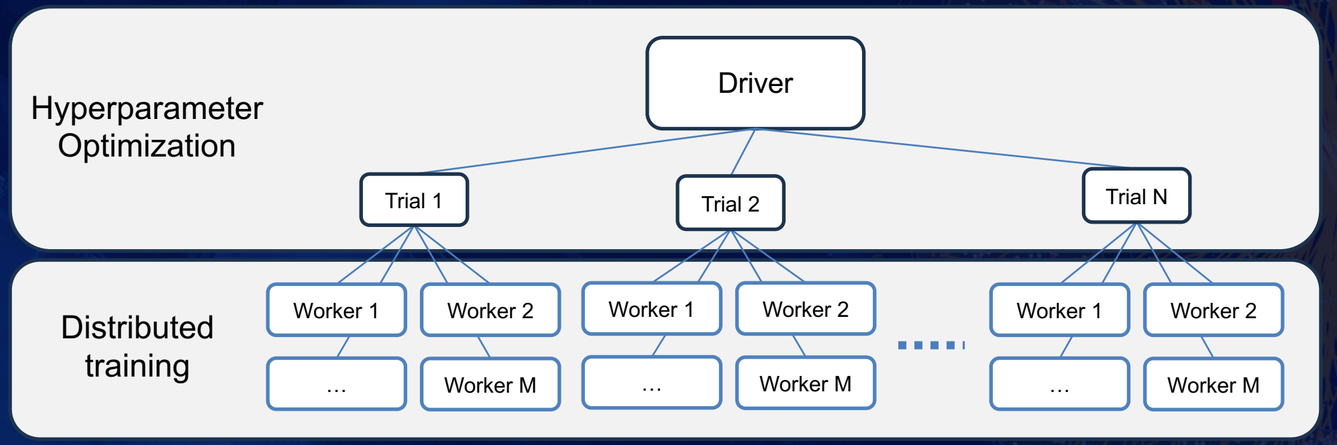
- Scaling of an HPO run of MLPF on the JURECA-DC-GPU system at the Jülich Supercomputer Centre (JSC), 4 NVIDIA A100 and 2× 64 cores AMD EPYC 7742 per node
- **Superlinear** scaling due to ASHA re-loading models when using fewer nodes
- ASHA + Bayesian optimization
- HPO lends itself very well to parallelization



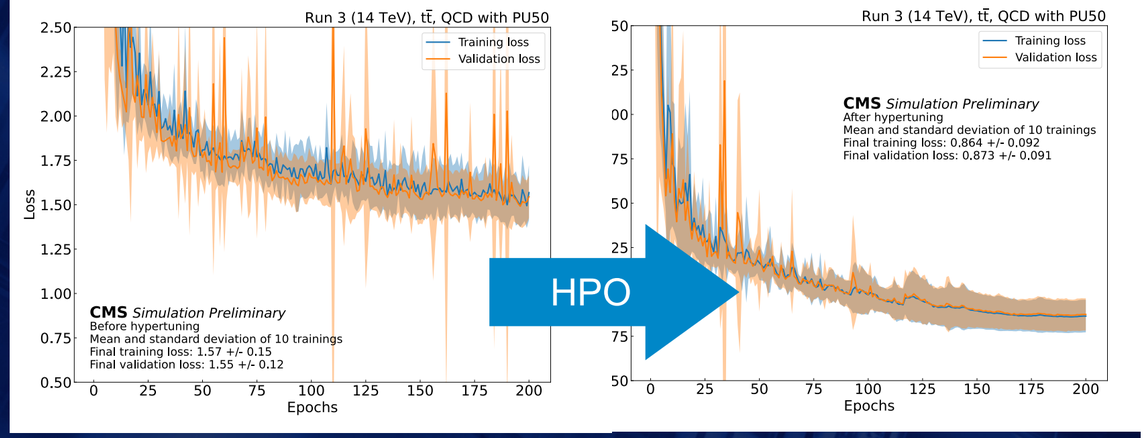
Dataset used: *Simulated particle-level events of ttbar and QCD with PU200 using Pythia8+Delphes3 for machine learned particle flow (MLPF),* https://doi.org/10.5281/zenodo.4559324

Distributed hyperparameter optimization

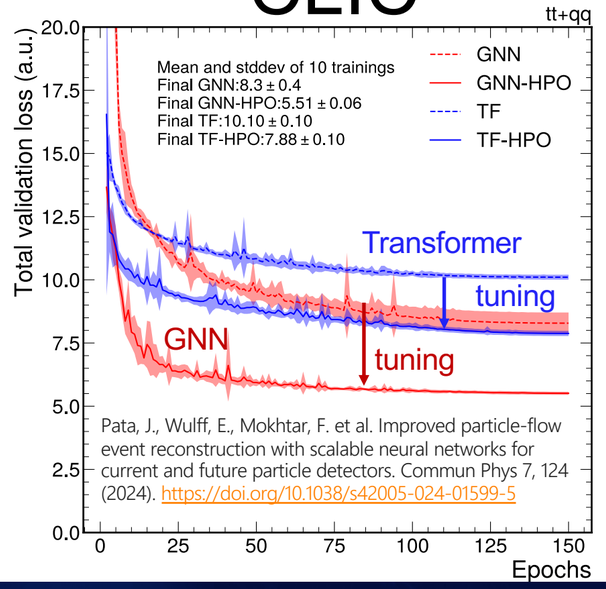
- Final validation loss decreased by **~44%** in CMS and **~34%** in CLIC resulting in significant performance increases
- Runs on 24-96 A100 GPUs at the Flatiron Institute and JSC
- Used ~2400 GPU-hours for the CMS tuning and ~5000 GPU-hours for CLIC



CMS



CLIC



Pata, J., Wulff, E., Mokhtar, F. et al. Improved particle-flow event reconstruction with scalable neural networks for current and future particle detectors. Commun Phys 7, 124 (2024). https://doi.org/10.1038/s42005-024-01599-5

Joosep Pata, Javier Duarte, Farouk Mokhtar, Eric Wulff, Jieun Yoo, Jean-Roch Vlimant, Maurizio Pierini and Maria Girone on behalf of the CMS Collaboration, doi:10.1088/1742-6596/2438/1/012100

Eric Wulff, Maria Girone and Joosep Pata, doi:10.1088/1742-6596/2438/1/012092

Moving Forward

- HPC systems provide large numbers of powerful GPUs suitable for AI training
  - Distributed training speeds up model development cycles and model training
  - HPO significantly increased model performance in the example of MLPF
- We have already begun to see the benefits from engaging with the HPC community
  - CoE RAISE, interTwin, SPECTRUM, ODISSEE, etc.
- The SPECTRUM project is working on facilitating HPC access to the HEP and RA communities in Europe
- We are submitting more applications for EuroHPC JU research calls in the near future
- Computing is experiencing a rapid evolution of technology and techniques with large improvements in HW accelerators and software applications

Vlaams Supercomputer Centrum User Day - Brussels

Thank you

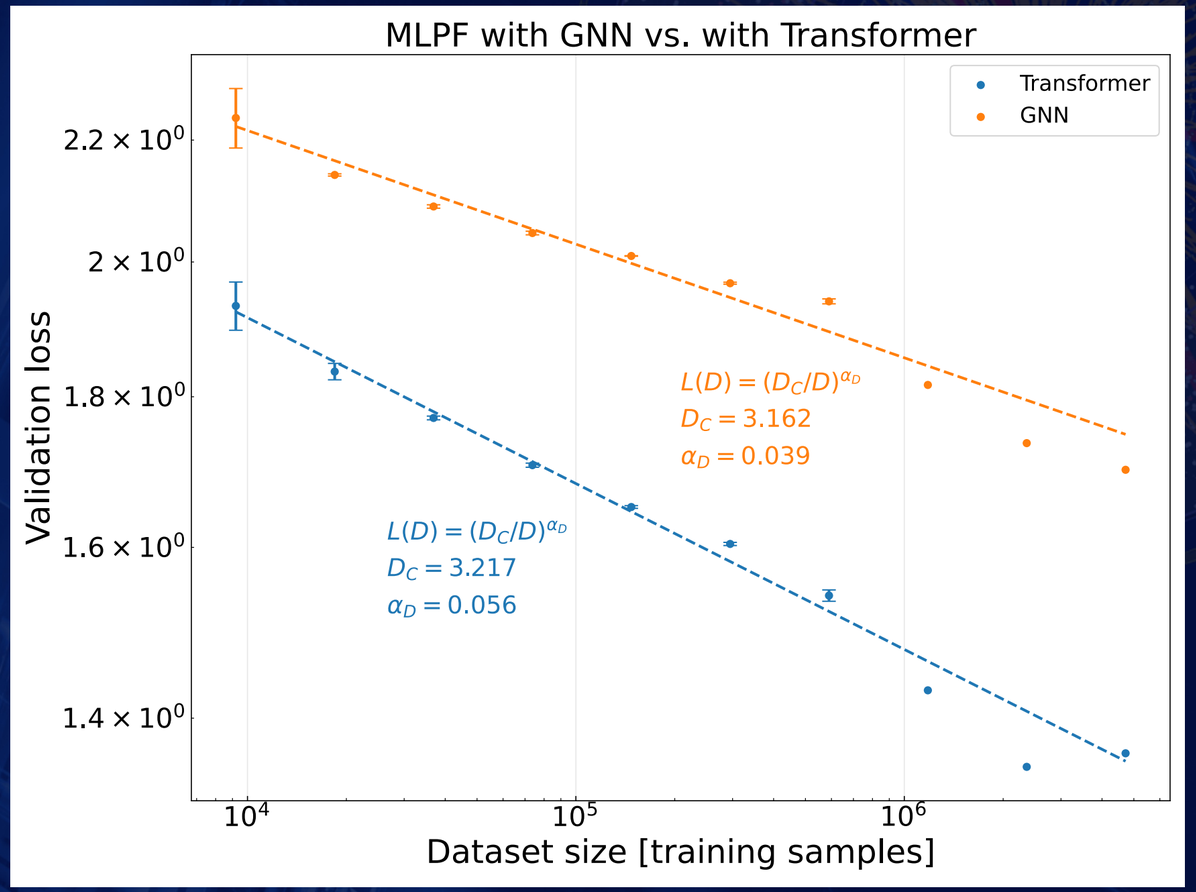
17th December 2025

Eric Wulff
CTO for AI on HPC, CERN openlab
CERN IT Department

Backup

Scaling laws for the MLPF models

- We trained models on datasets of varying size and fit power laws to the relationship between validation loss and dataset size
- As expected, larger datasets translates to better performance
- This motivates us to
  - generate larger datasets in the future
  - further improve training speed to keep runs within a reasonable time window
  - consider multi-node training



Eric Wulff, Joosep Pata, Maria Girone, PASC24, https://pasc24.pasc-conference.org/presentation/?id=pos141&sess=sess158