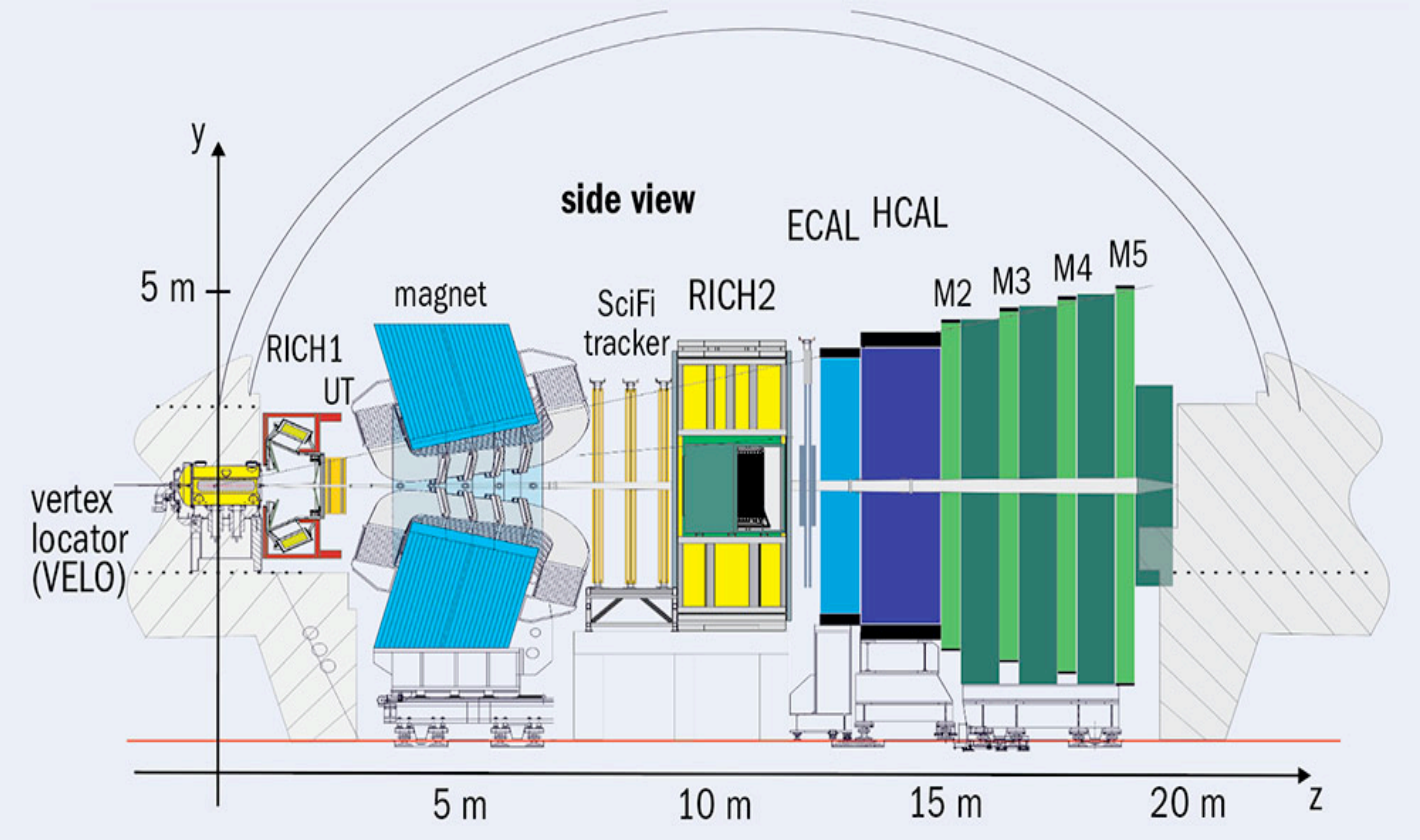
ML in LHCb

Prospecting for New Physics through Flavor, Dark Matter, and Machine Learning

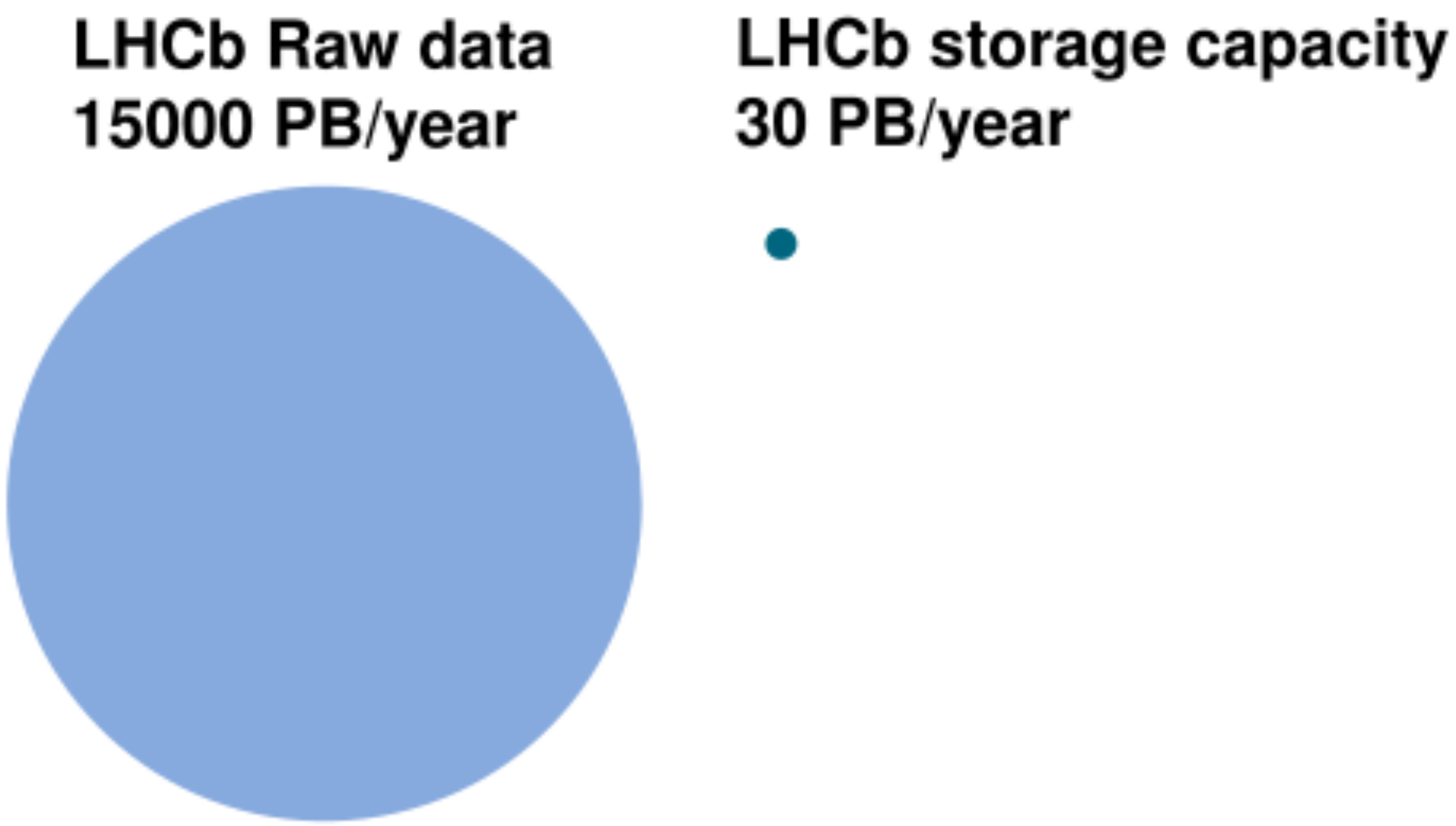
Niklas Nolte - MIT - IAIFI - LHCb - April 2023

Intro



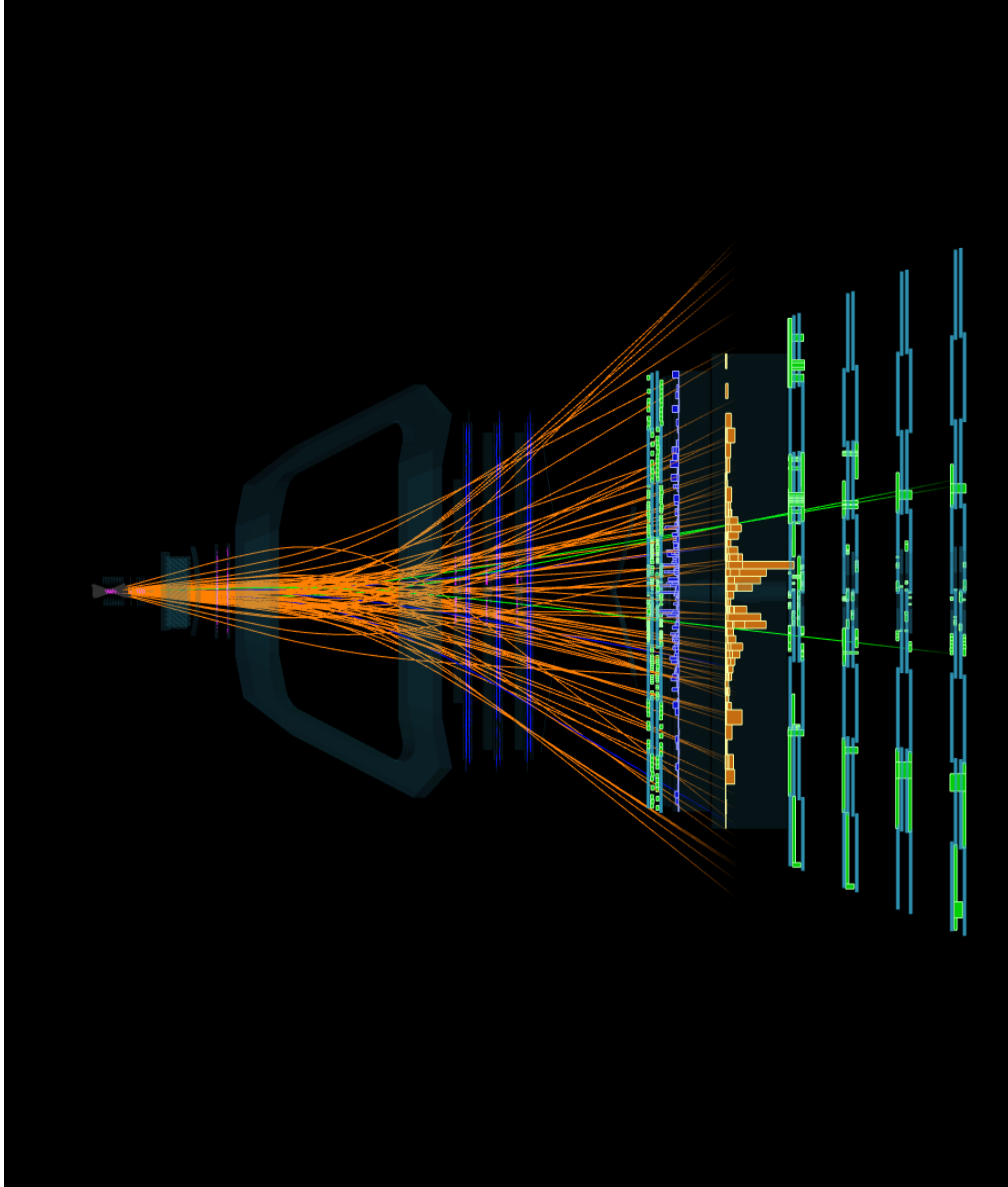
- ML online
  - Trigger
  - ML for HLT
- ML offline
  - Robust PID
  - Flavor Tagging
  - Fast Simulation

The Trigger at LHCb

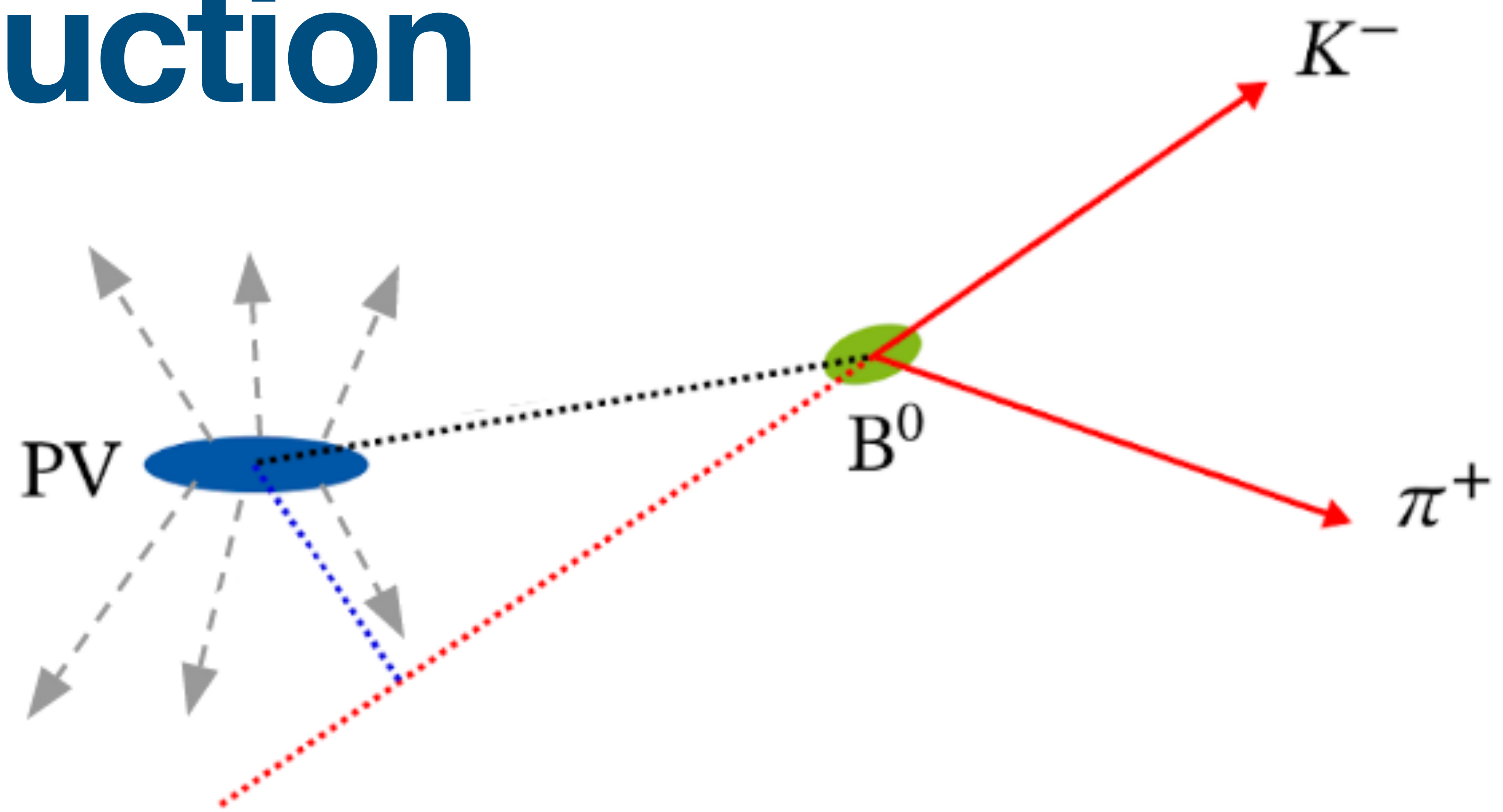


Select **only interesting** events:
Hybrid approach of expert systems and ML

Real time data reduction: 5 TB/s → 10 GB/s



Event Selection -- data reduction



Trigger: mostly an expert system

Many subsystems look for particular decays
→ Strong reduction and purity ✅

Some look for general signatures, weaker selection
→ Achieve good purity with ML classifiers
Need guarantees to employ these! **No room for error**

Guarantees needed:
1. Robustness w.r.t small changes
2. Monotonicity in certain features for "out of distribution" (OOD) guarantees

(Adversarial) Robustness

many SOTA ML models are proven to be highly unstable

Robustness $:= \quad f(x+\epsilon) = f(x) + O(\epsilon)$

I want deterministic robustness, i.e. provably robust networks!

Deterministic Robustness

NeurIPS ML4PS 21
arXiv:2112.00038

WLOG: Binary classifier: $F : \mathbb{R}^n \to \mathbb{R}$

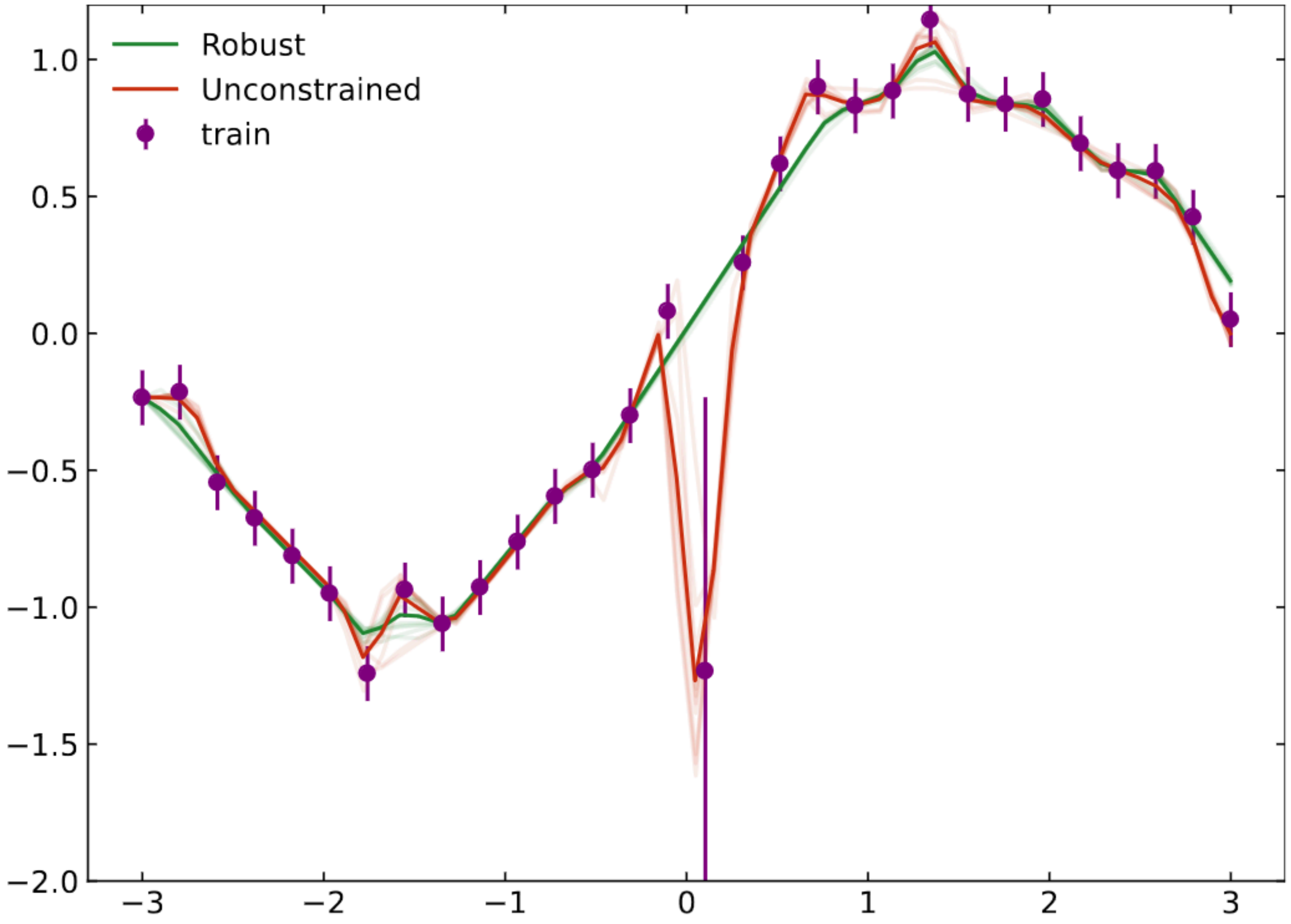
Constrain gradient wrt inputs, i.e. make it

Lipschitz-L $\quad ||\nabla F|| \leq L$

A perturbation ϵ to an input x needs certain magnitude to flip the sign:

$$\operatorname{sign} F(x+\epsilon) = -\operatorname{sign} F(x)$$

$$\Rightarrow ||\epsilon|| > \frac{|F(x)|}{L}$$



Lipschitz Networks

NeurIPS ML4PS 21
arXiv:2112.00038

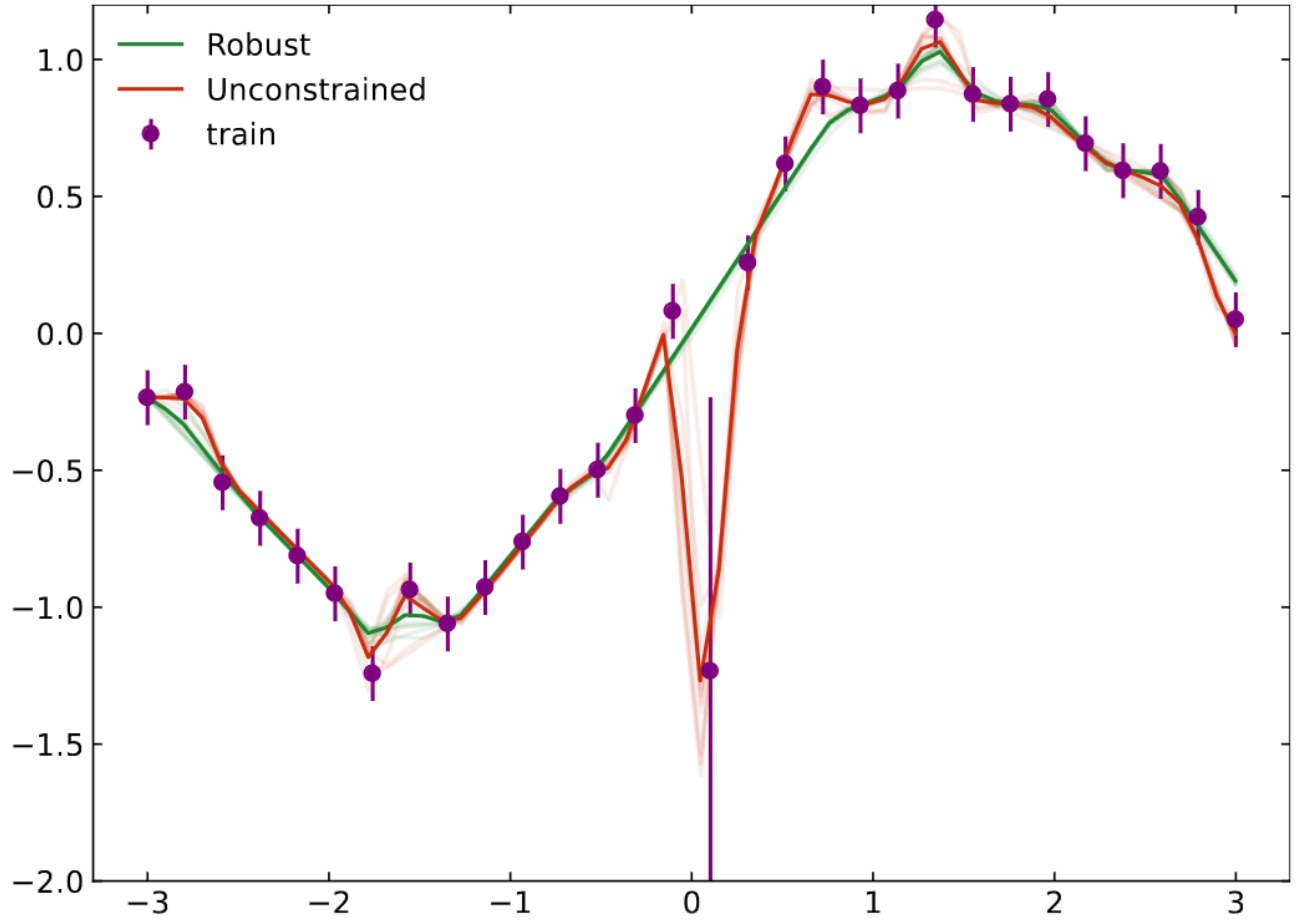
$||\nabla F|| \leq L$ can be enforced by constraining weights

In an MLP with Lipschitz-1 activations:

$$L \leq \prod ||W^i||$$

(Toeplitz matrix for CNNs)

One possibility:
maintain $||W^i|| \leq \sqrt[d]{L}$ in every layer



Monotonic Networks

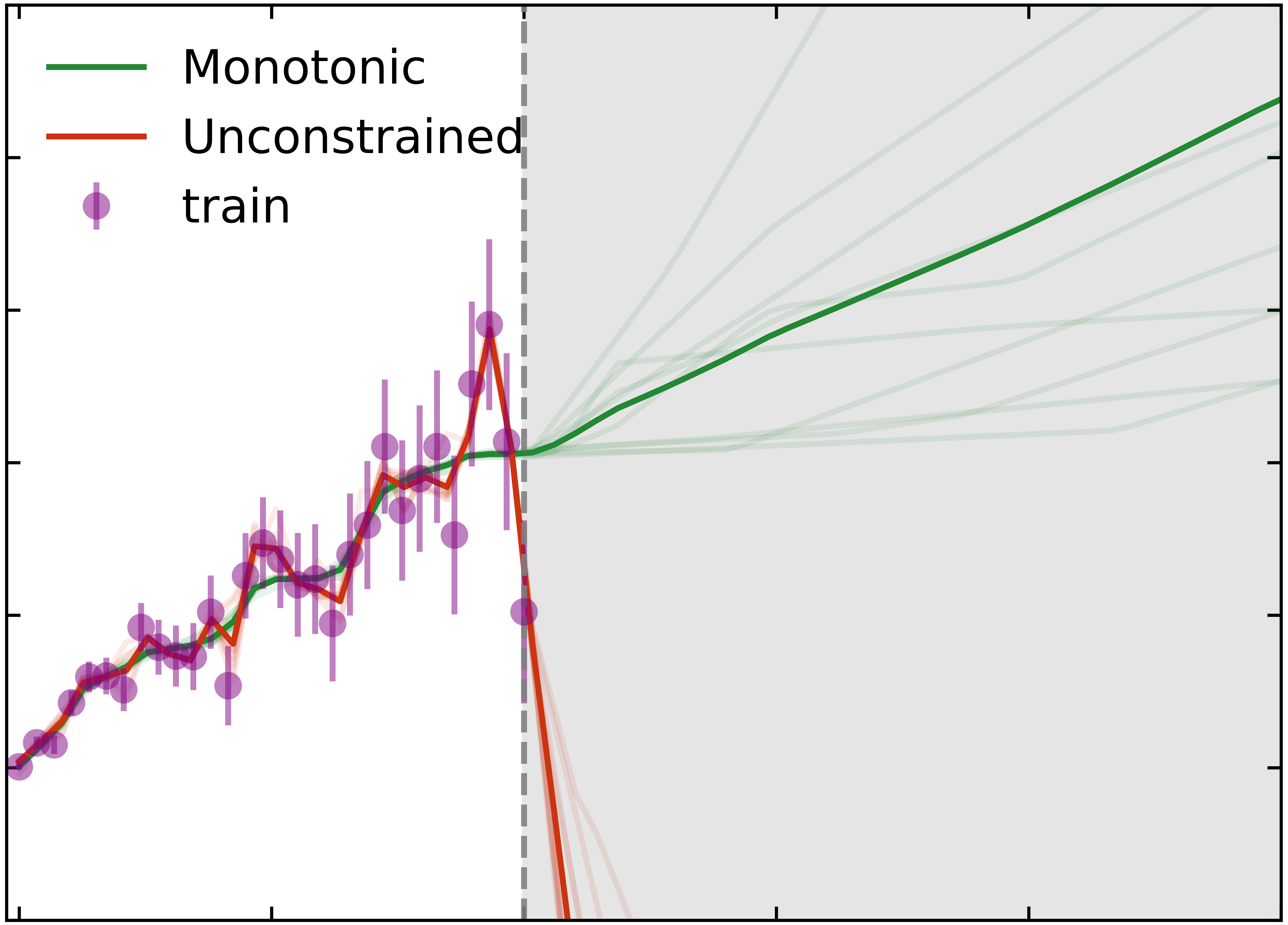
NeurIPS ML4PS 21
arXiv:2112.00038

Care about tails!

Guarantees about OOD
with monotonicity

Expressive monotonic networks
are not obvious.

Easier with robustness!



Monotonic Lipschitz Networks

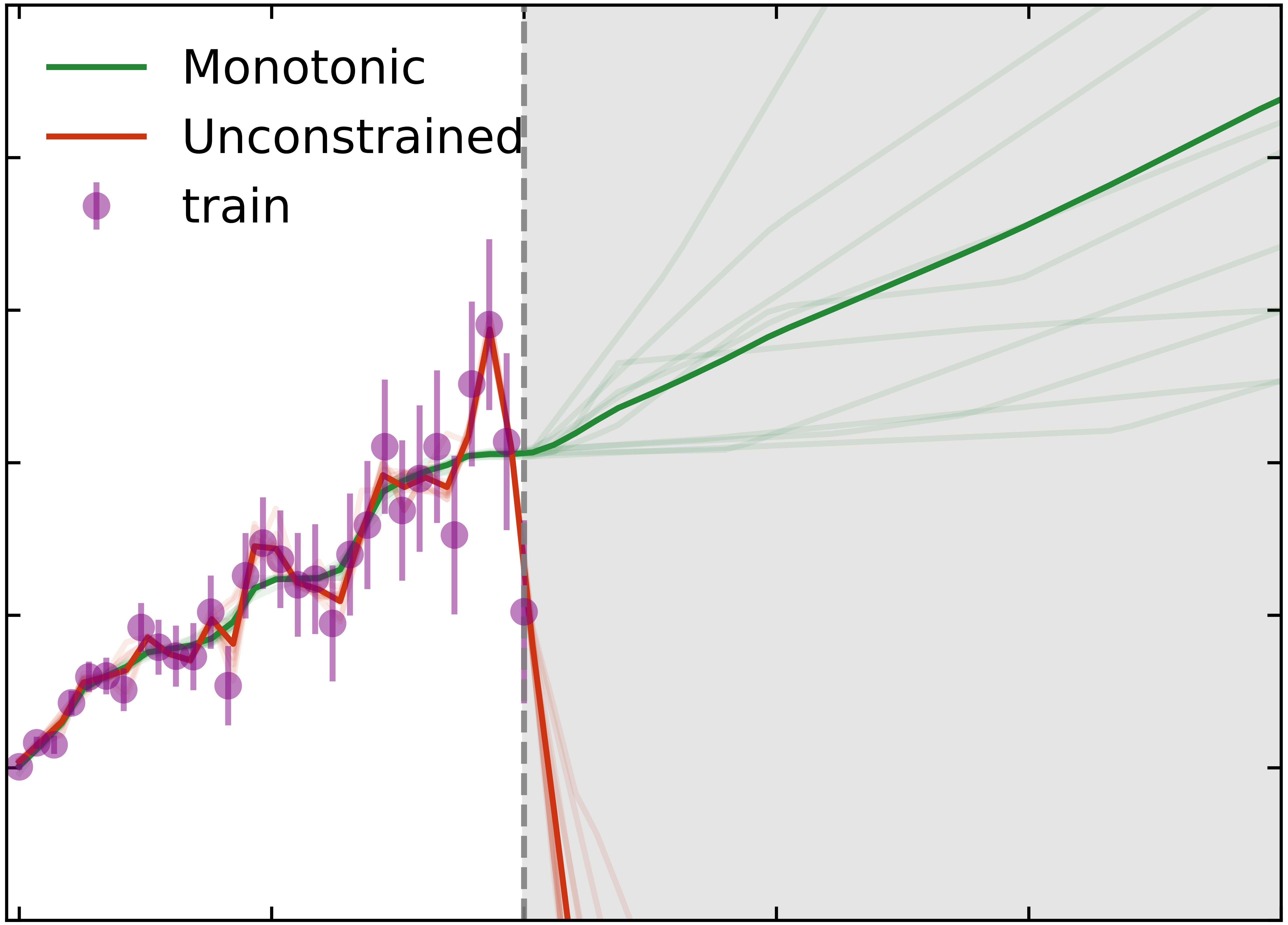
NeurIPS ML4PS 21
arXiv:2112.00038

Combine Lipschitz networks with monotonicity!

$$M(x) = F(x) + L\sum_i x_i$$

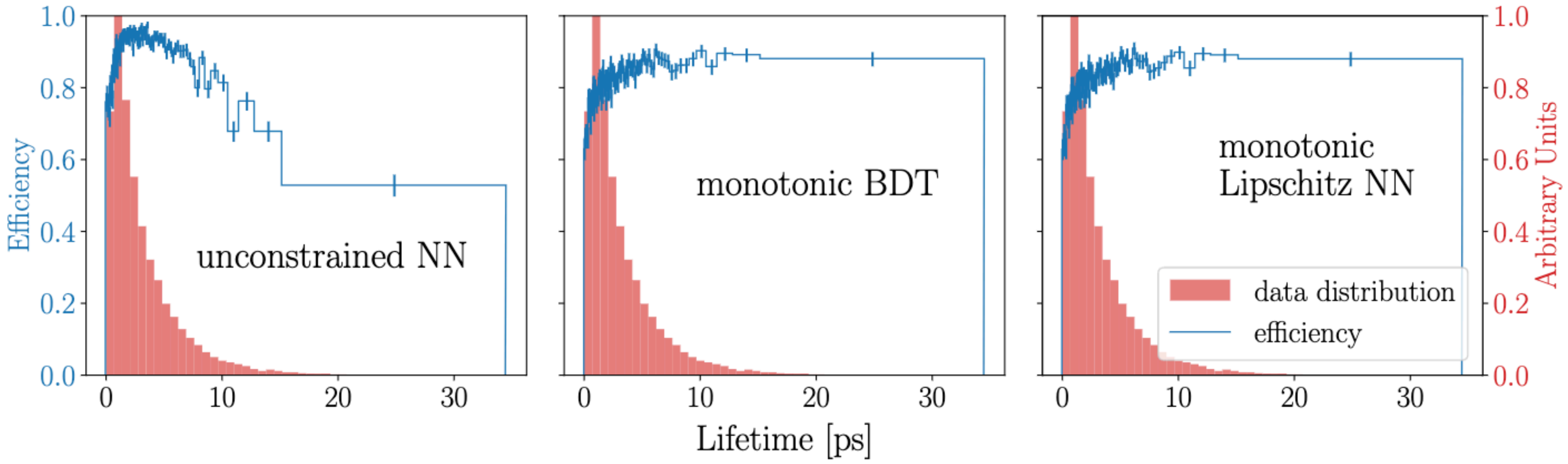
$$\frac{\partial M}{\partial x_i} = \frac{\partial F}{\partial x_i} + L \geq 0$$

Lipschitz-L : $\|\nabla F\|_\infty \leq L$



Example

NeurIPS ML4PS 21
arXiv:2112.00038



Applications

HLT1 (on GPU)

1. fast inference with inclusive 1 & 2 trajectory beauty and charm selections
2. rejecting mis-reconstructed trajectories early in online reconstruction

HLT2

1. inclusive 2, 3 track beauty selection, Topological Trigger
2. in muon & and other more specialized signature selections

→ many instances of very small networks
(orthogonal to most current efforts in ML)

Bonus: Monotonicity as inductive bias appears often!
Applications in Medicine, Criminal Justice

ICLR 23
`https://openreview.net/forum?id=w2P7fMy_RH`

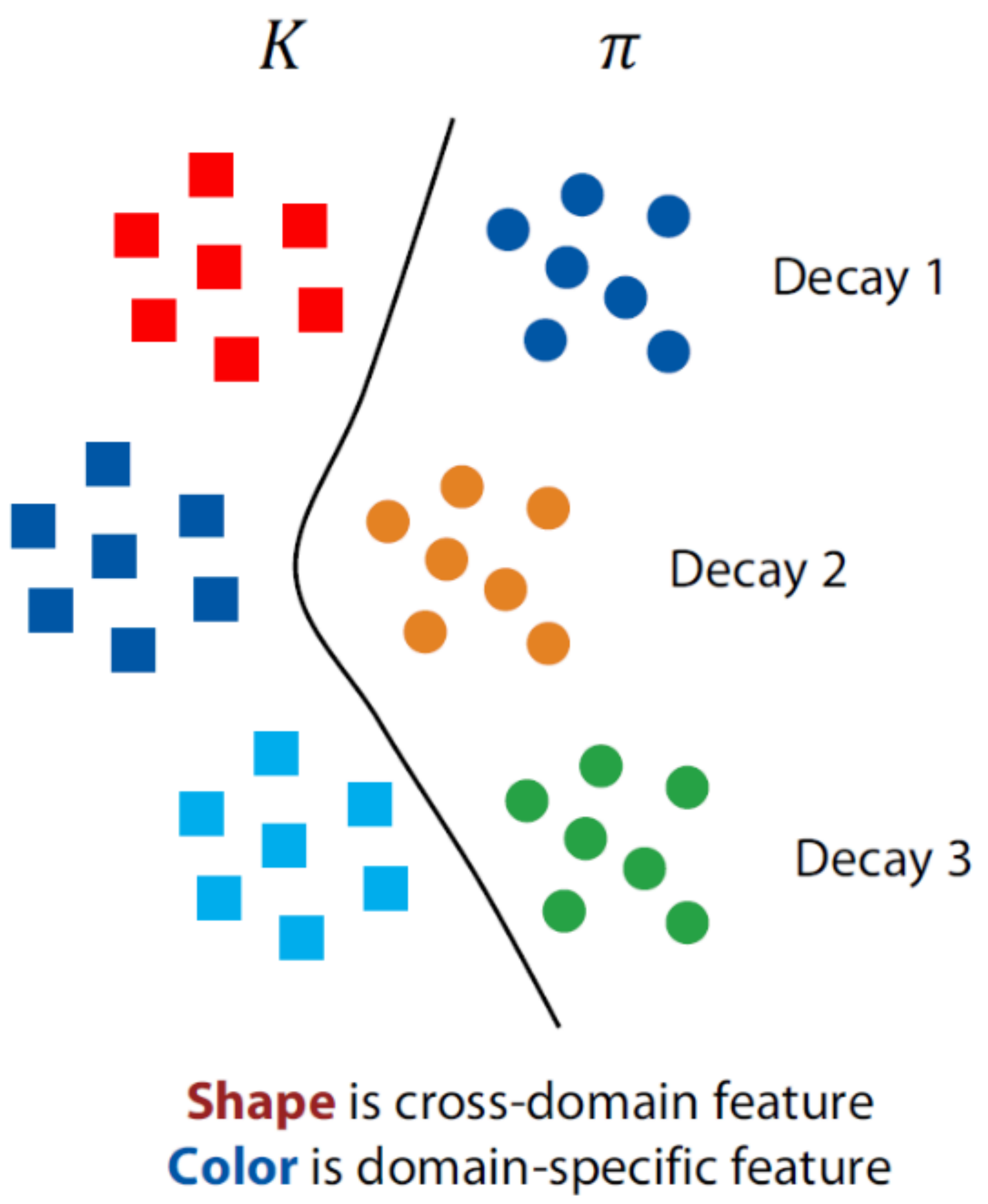
ML offline

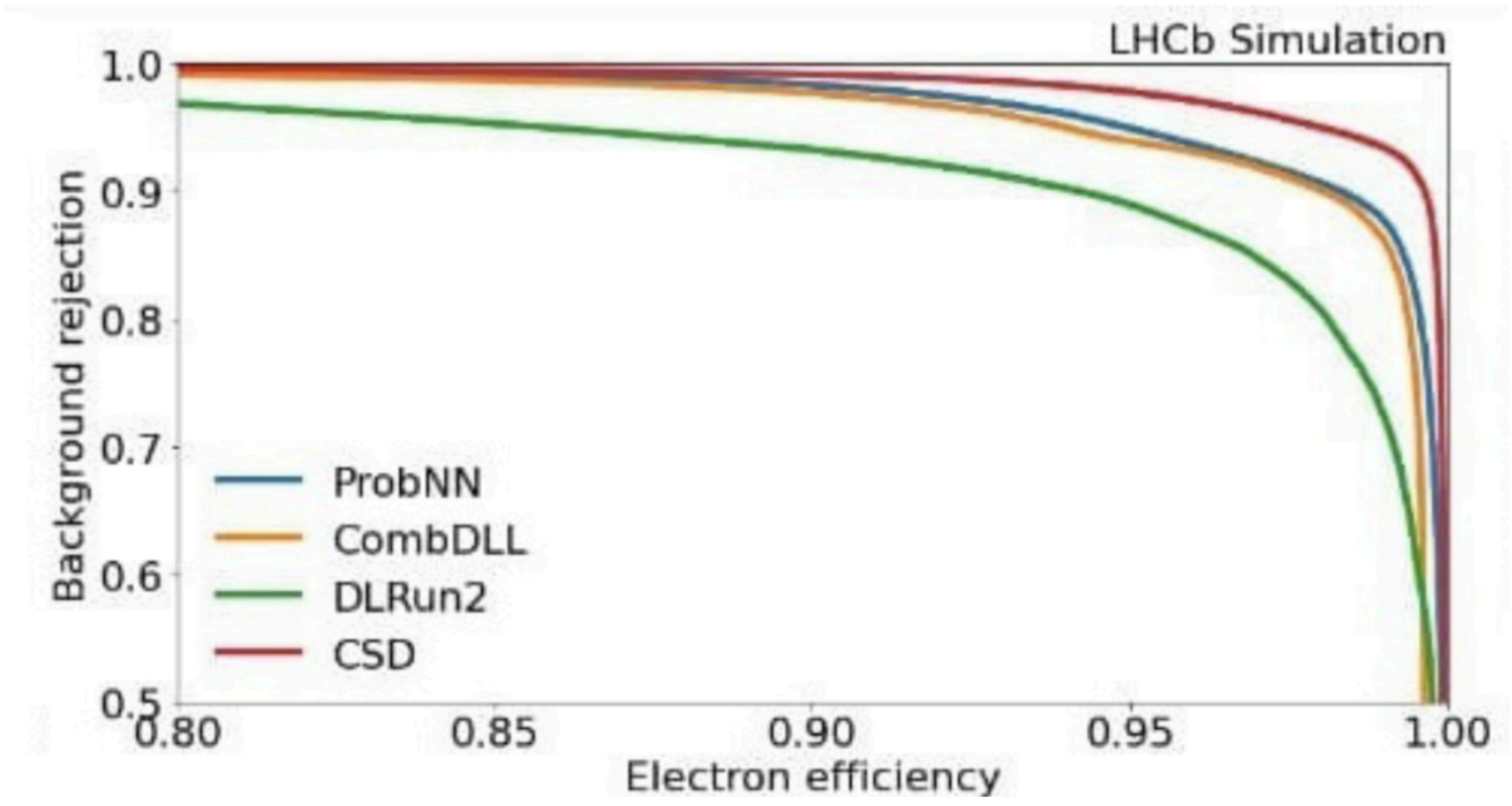
1. Robust PID
2. Flavor Tagging
3. Fast Simulation
4. More

Robust Particle Identification

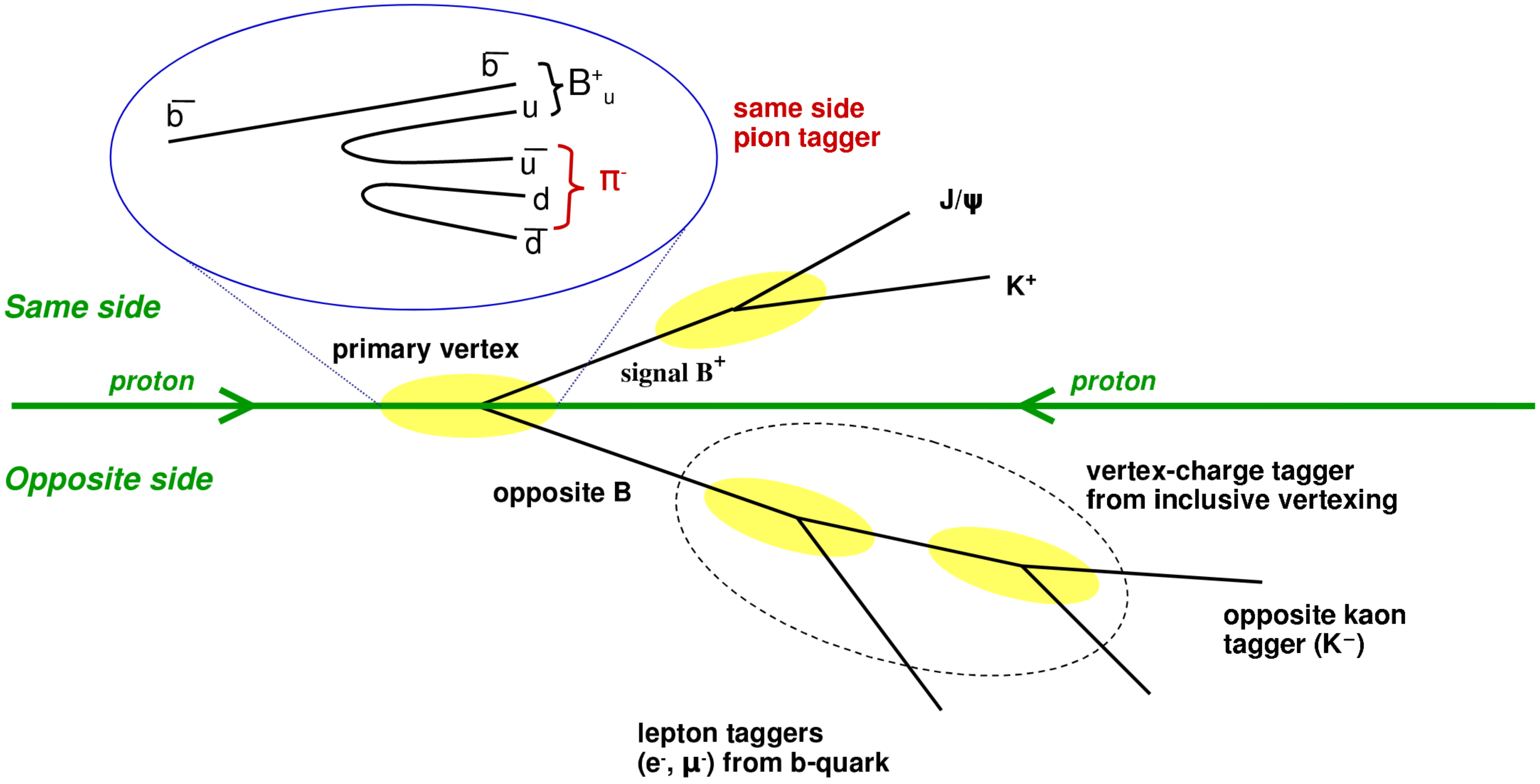
arXiv:2212.07274

Slightly different definition of robust: w.r.t. domain shifts via Common-Specific Decomposition (CSD)





Flavor Tagging



doi:10.22323/1.321.0230

Input
The full event
[#tracks, #features]

Output
B_0 or $\overline{B_0}$

Classical taggers
Find particles that indicate the B flavor

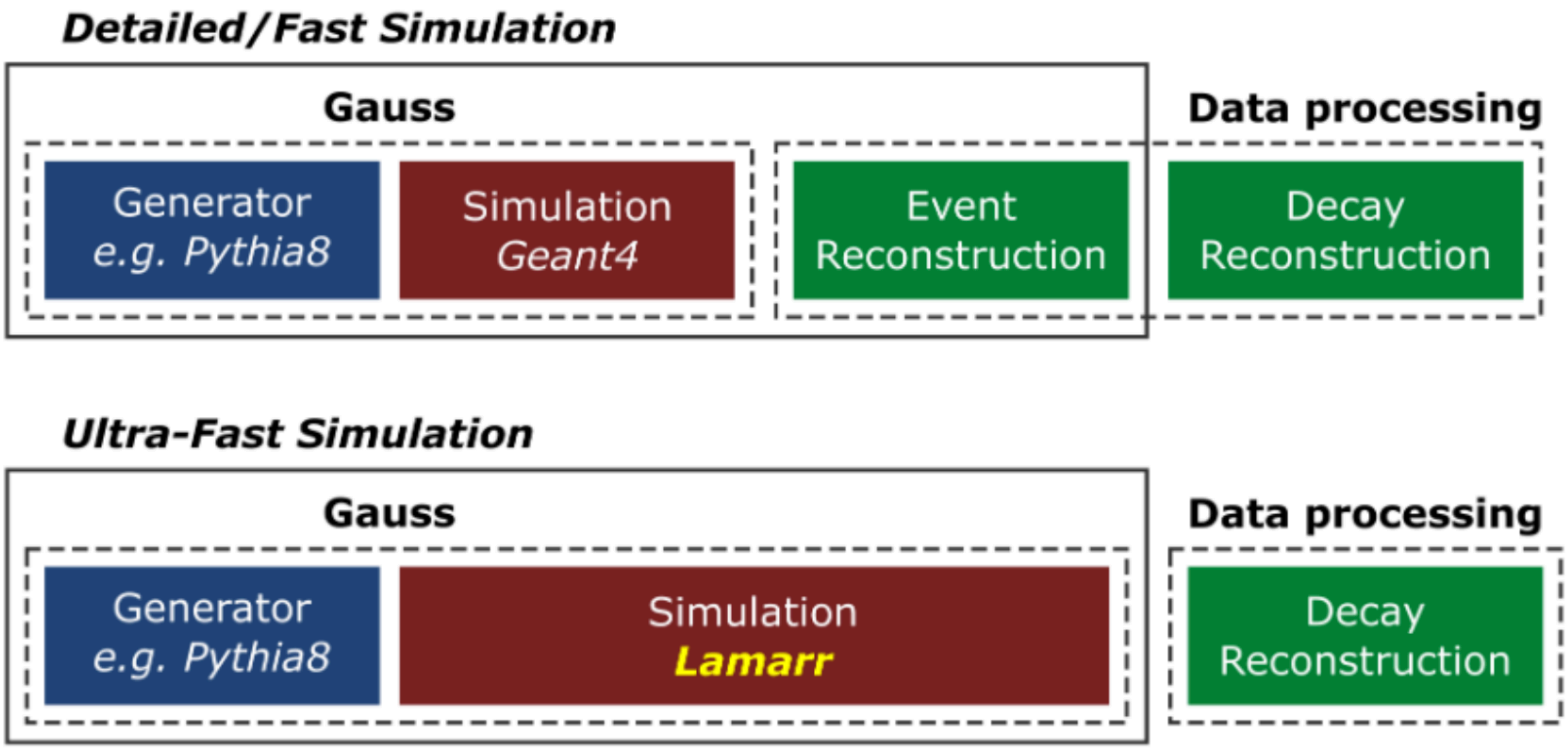
ML Taggers
Predict flavor directly

RNNs
DeepSets
Transformers

up to **50% more tagging power** with ML taggers!

doi:10.22323/1.414.0233

Fast Simulation - Parametrize the Sim Pipeline



doi:10.22323/1.414.0233

Two Approaches

Efficiencies

Gradient Boosted Decision Trees (GBDTs) trained on simulated data with *Binary* or *Categorical Cross Entropy* to predict the fraction of "*good**" candidates, *i.e.* the "efficiency"of a specific step as a function of generator-level quantities.

- GBDTs are robust and easy to train
- Almost no preprocessing is needed

* either "accepted", "reconstructed", "selected"... depending on the context

Reconstructed quantities

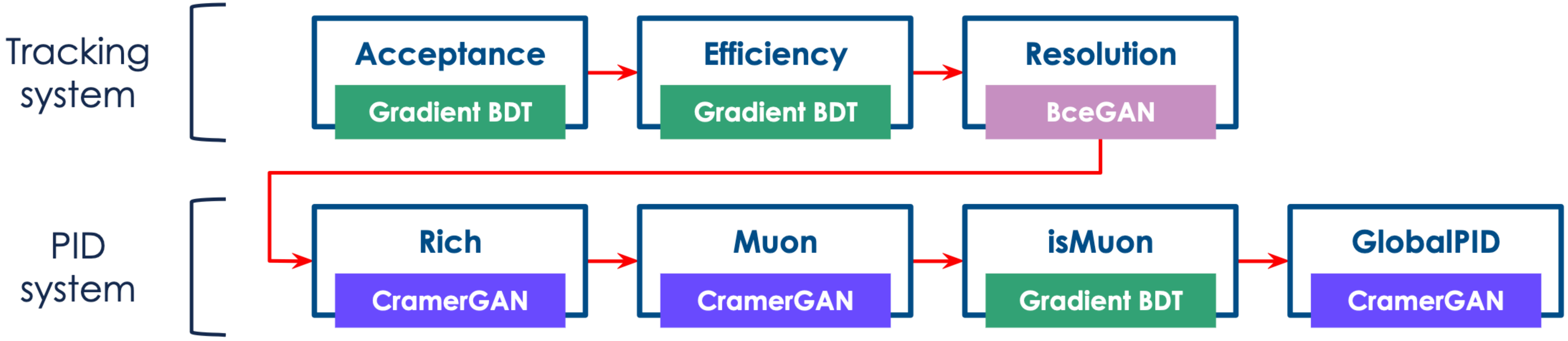
Conditional **Generative Adversarial Networks** trained on either simulated or calibration data.

Various GAN flavours adopted for different parameterizations balancing between accuracy and robustness.

Training is performed on **opportunistic GPU resources** provided to the Collaboration.

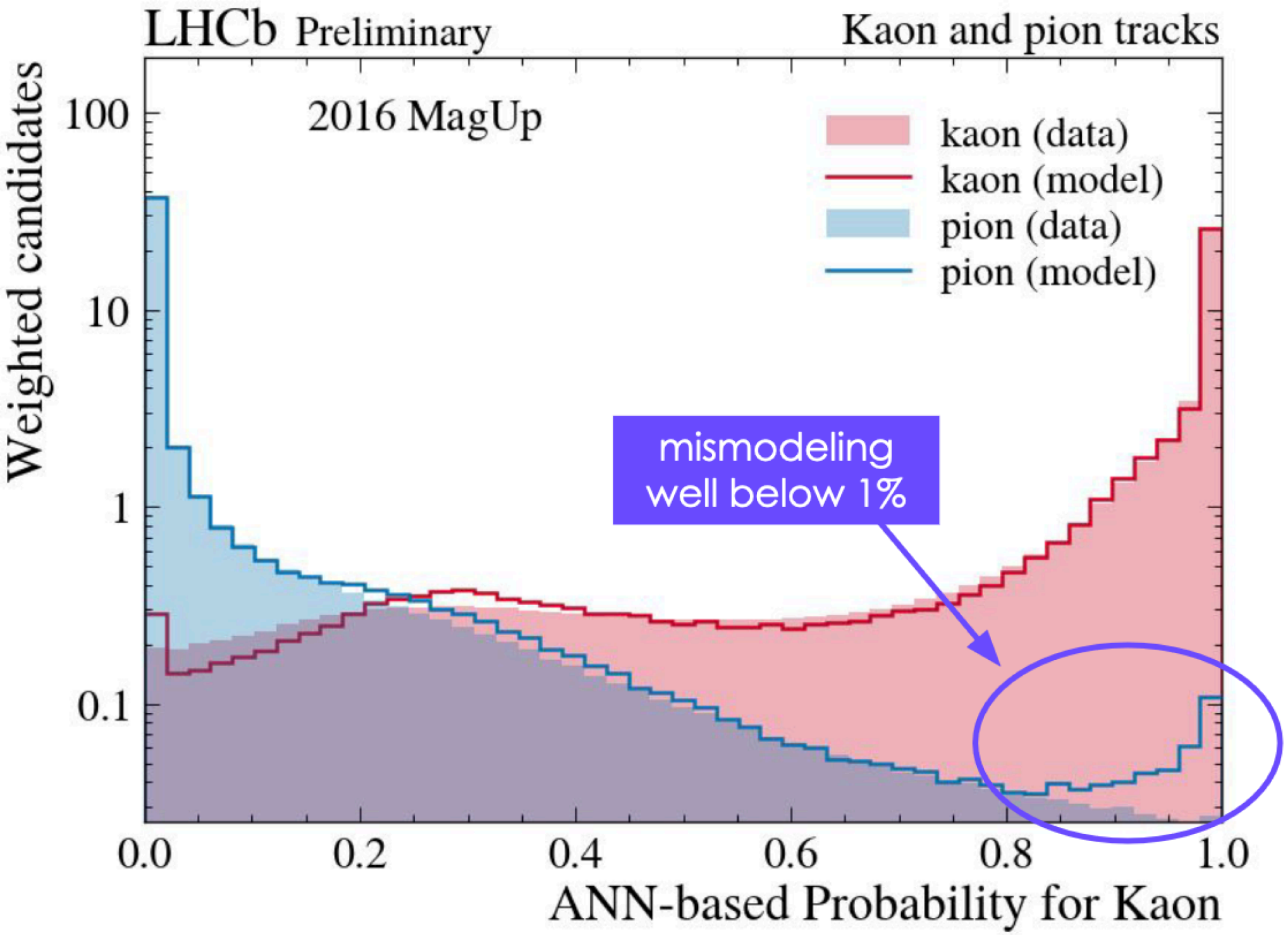
doi:10.22323/1.414.0233

Pipelines of Parametrizations

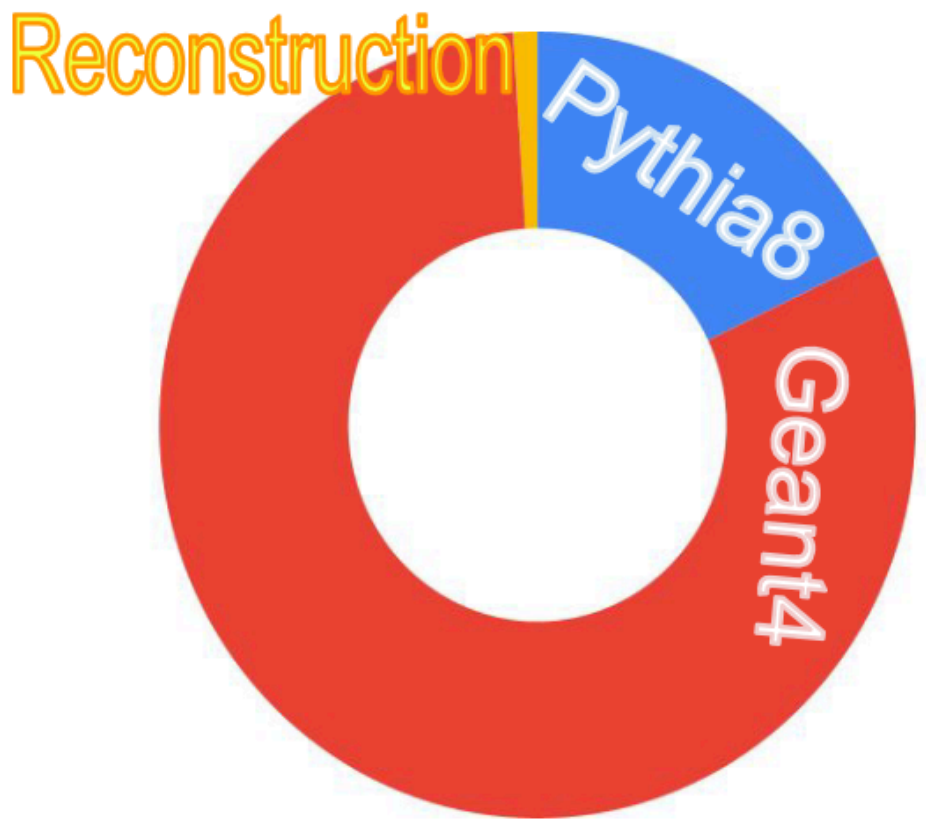


doi:10.22323/1.414.0233

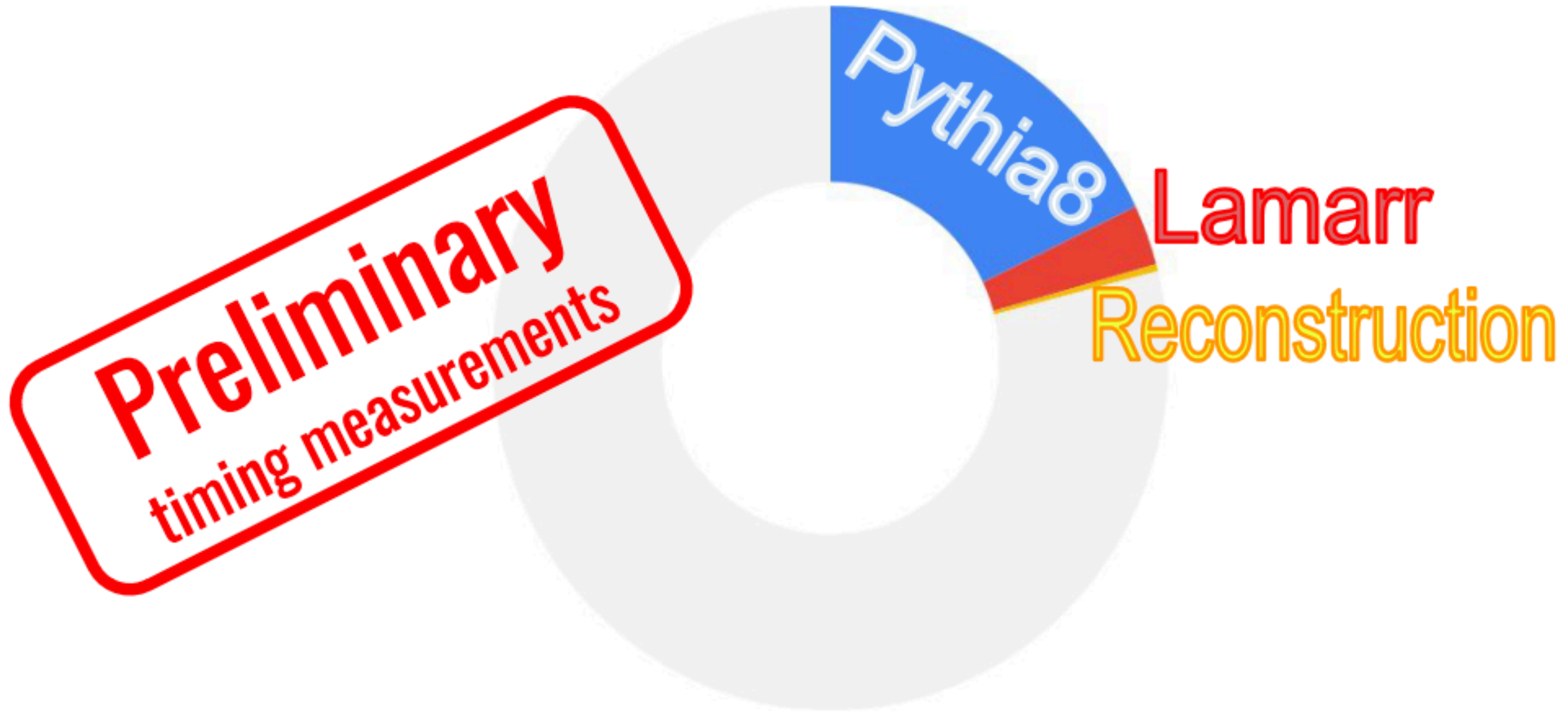
Pretty impressive!



Detailed simulation: Pythia8 + Geant4
1M events @ 2.5 kHS06.s/event ≈ 80 HS06.y



Ultra-fast simulation: Pythia8 + Lamarr
1 M events @ 0.5 kHS06.s/event ≈15 HS06.y



Etc Etc...

1. RoboShifter - Classify the quality of a run during data taking
   doi:10.1088/1742-6596/898/9/092027

2. Particle ID with neural networks in the trigger
   doi:10.1051/epjconf/201921406011

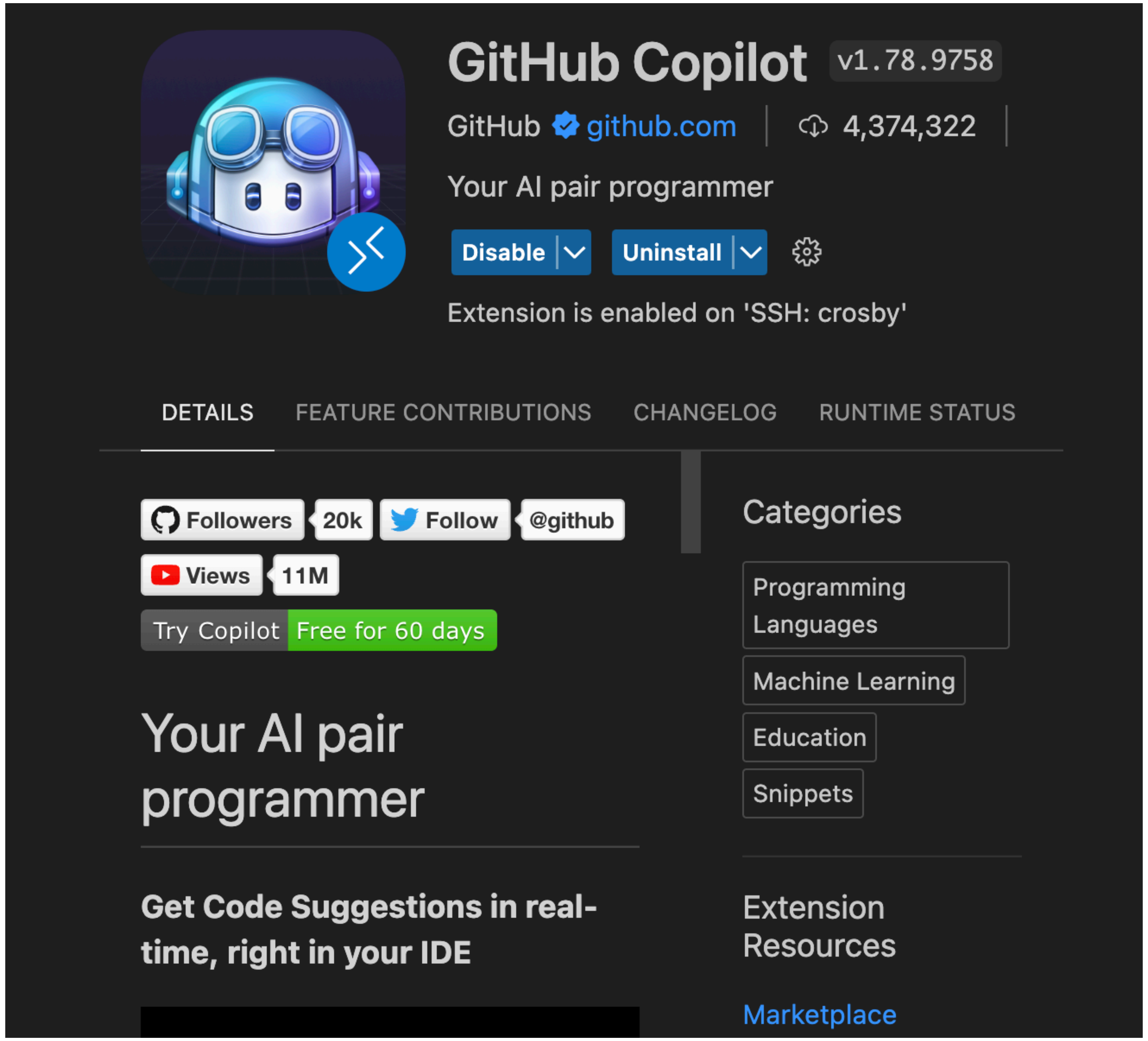
3. Particle ID with gaussian mixture models in the fixed target program (SMOG)
   arxiv:2110.10259v2

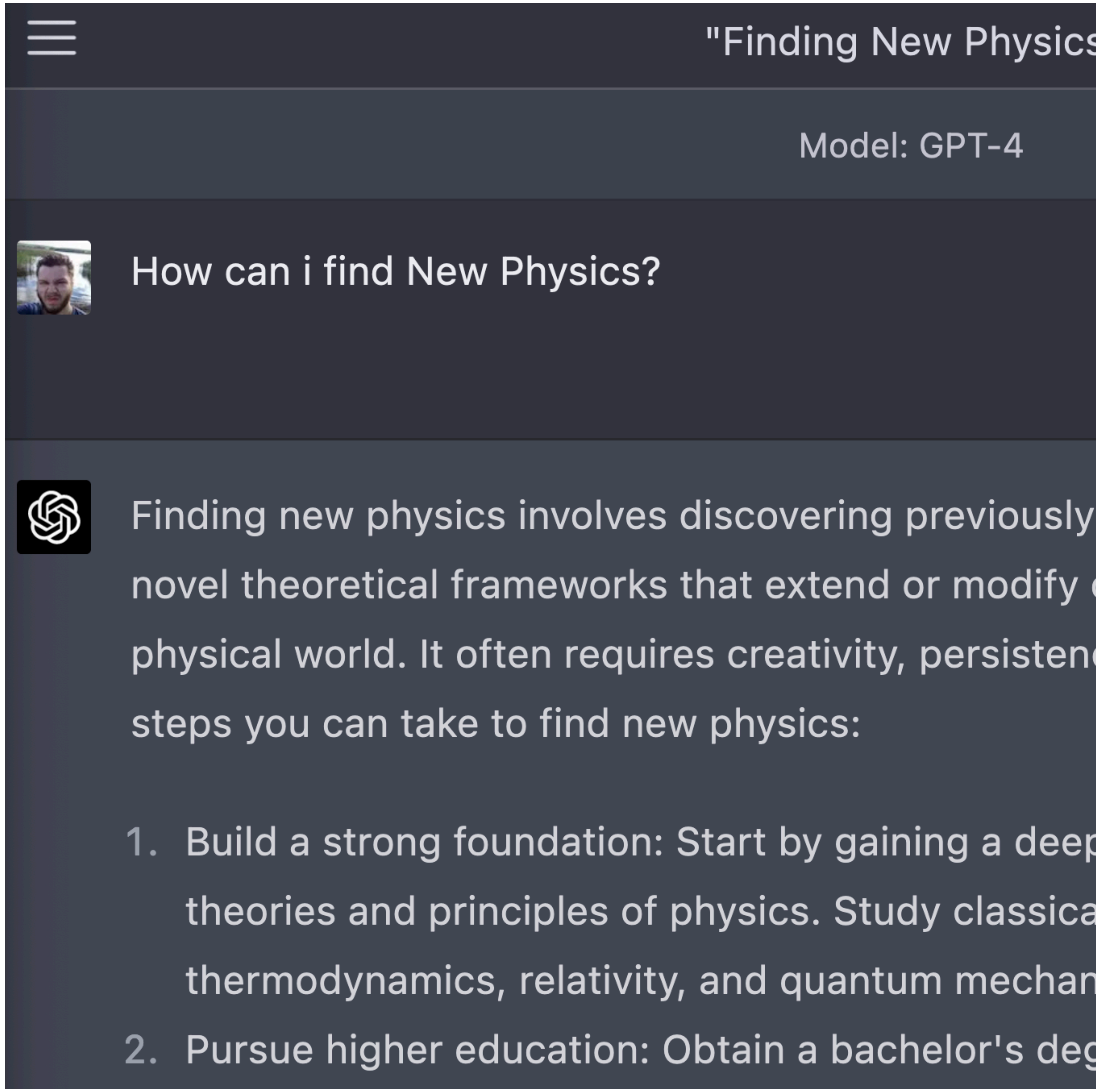
4. Jet Tagging doi:10.1088/1748-0221/10/06/P06013

5. Decorrelation methods for ML in analyses arXiv:2010.09745, arXiv:1305.7248

6. And everything else that people come up with in analyses!

How we actually use ML





Backup

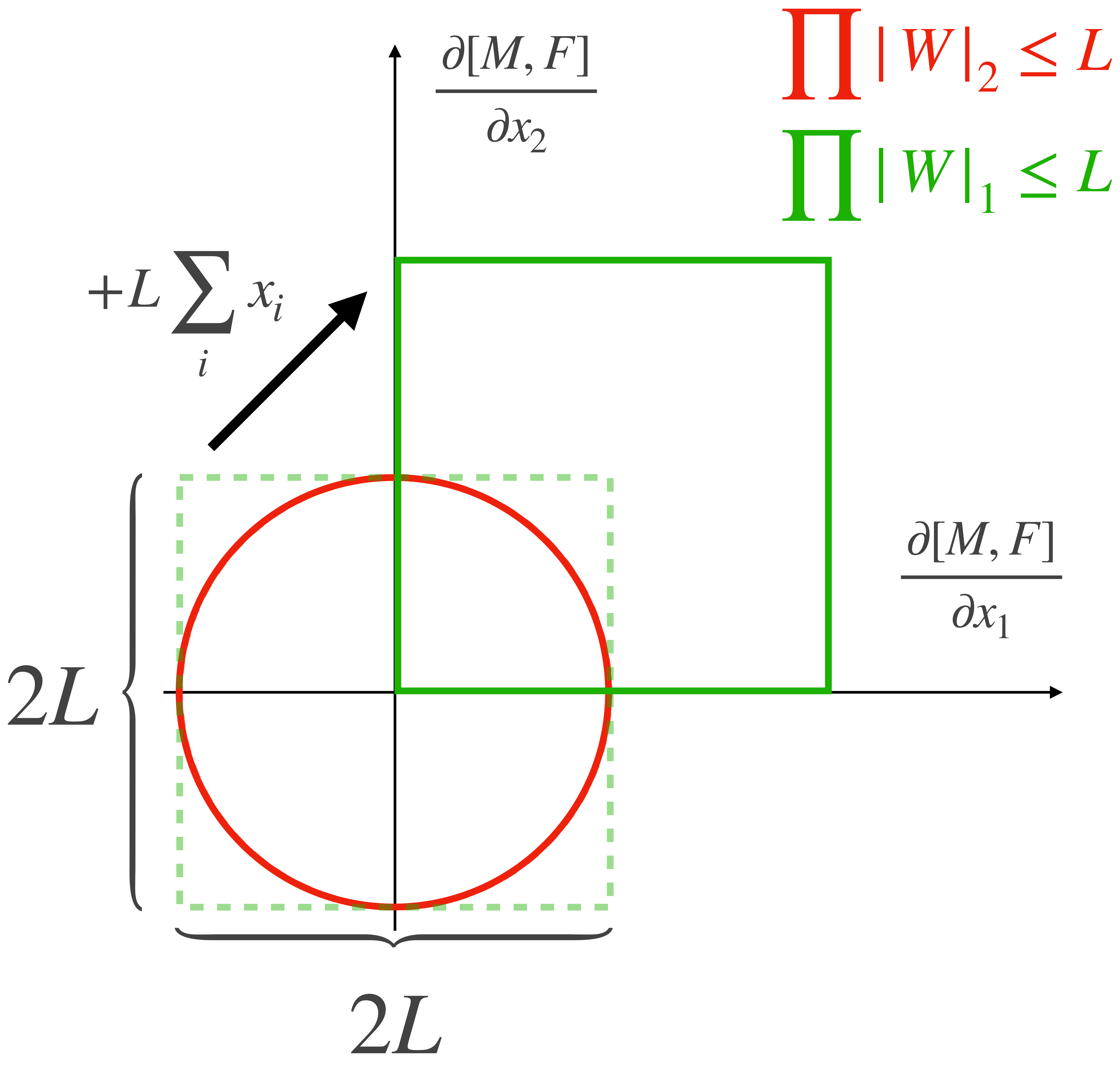
Monotonic Lipschitz Networks

$$M(x) = F(x) + L\sum_i x_i$$

$$\frac{\partial M}{\partial x_i} = \frac{\partial F}{\partial x_i} + L$$

$+L$ contribution in every direction x_i
$\|\nabla F\| \leq L$ is not good enough

We want $\|\nabla F\|_\infty \leq L$!



Monotonic Lipschitz Networks

This architecture is

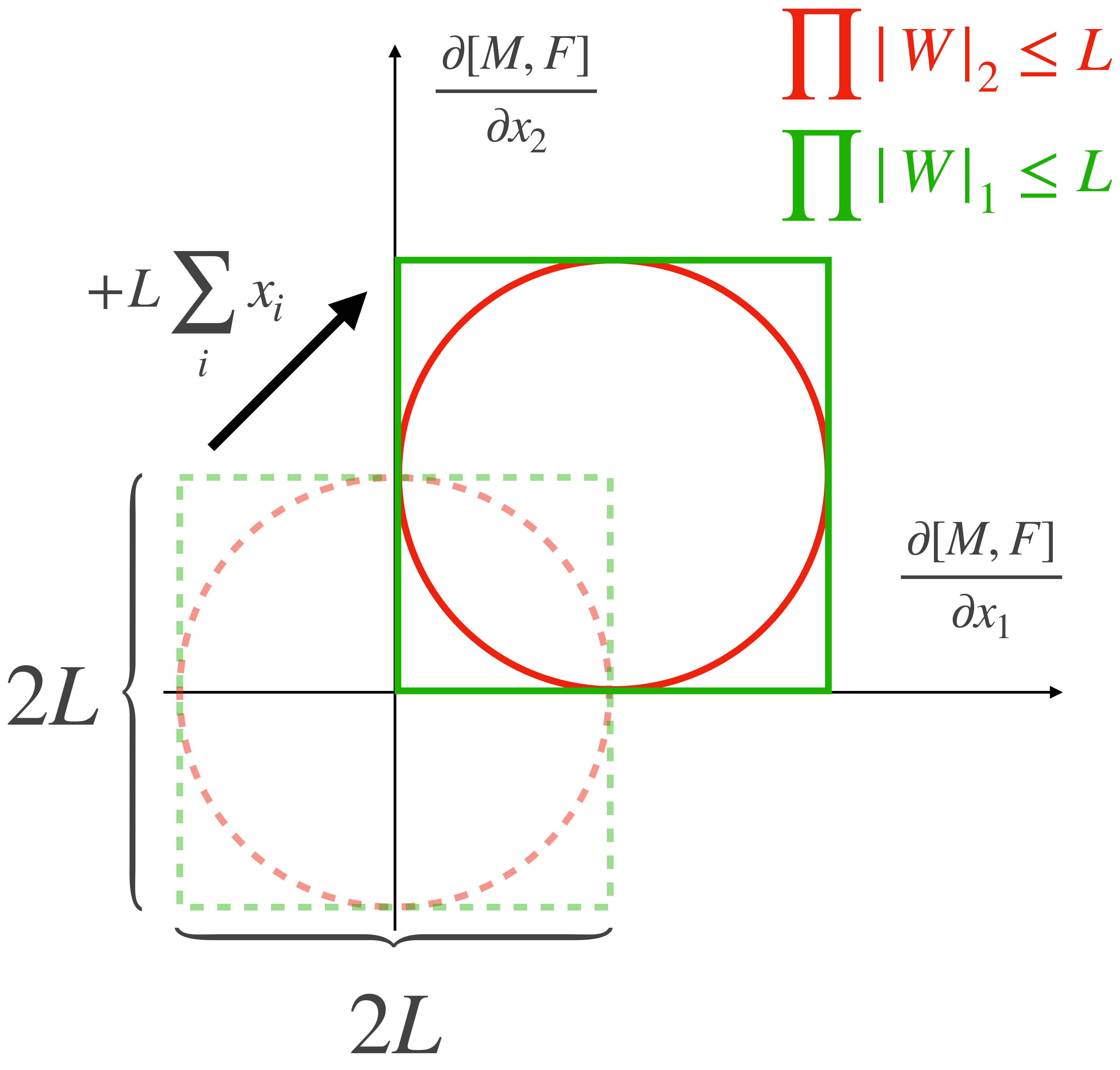
1. provably robust

2. provably monotonic

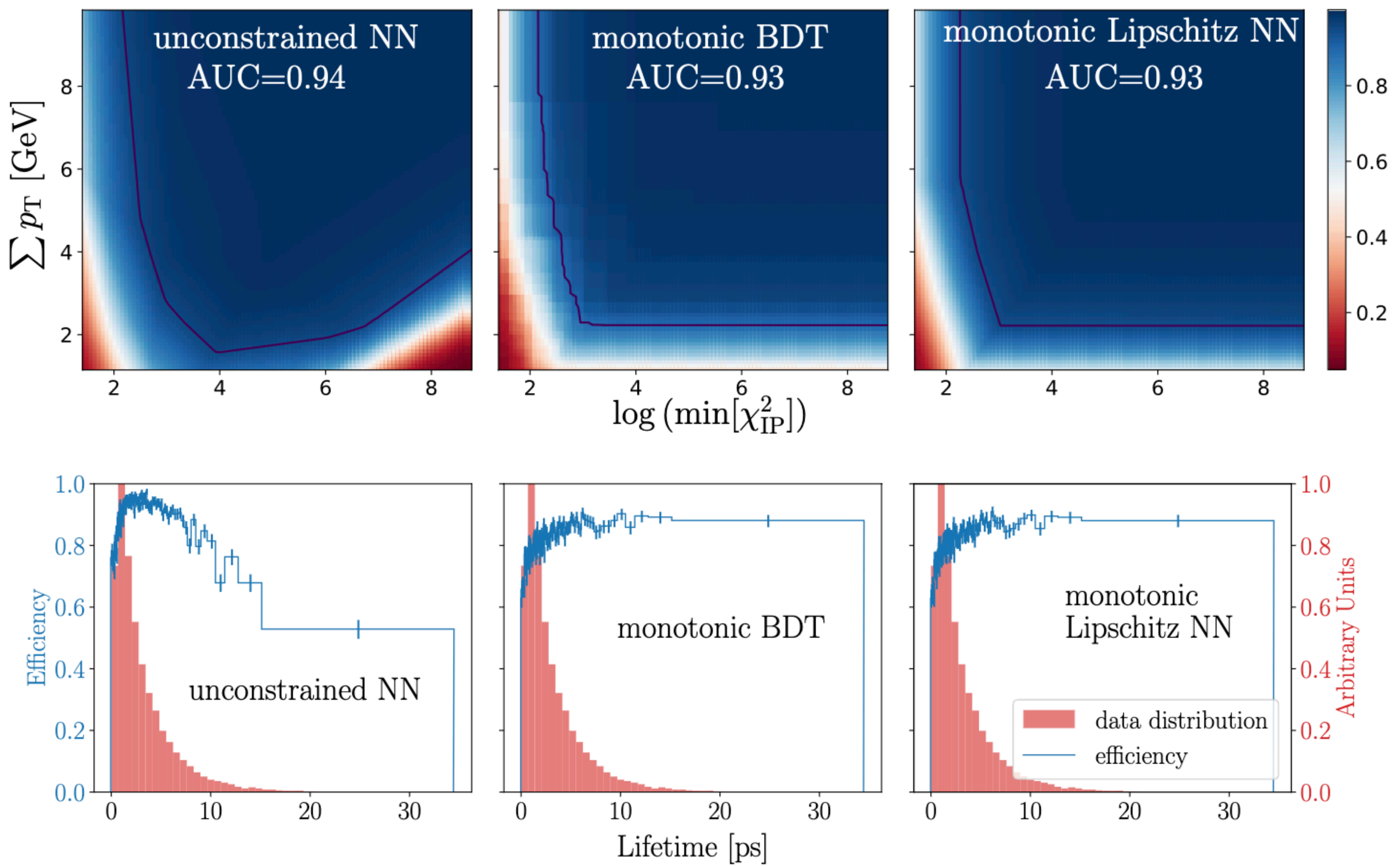
3. universally approximating
   the target function class

4. working well in practice

→ Implemented in the LHCb trigger

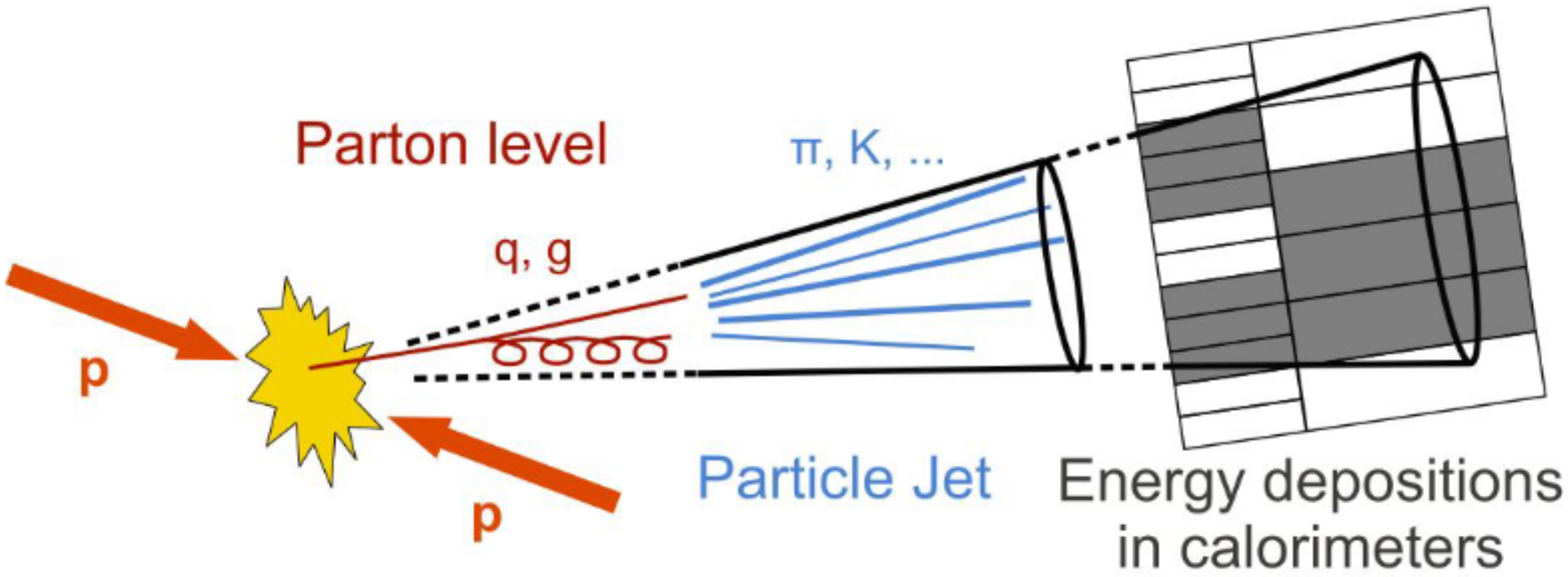


Example


unconstrained NN
AUC=0.94
monotonic BDT
AUC=0.93
monotonic Lipschitz NN
AUC=0.93
$\sum p_T$ [GeV]
$\log(\min[\chi^2_{IP}])$
Efficiency
Arbitrary Units
unconstrained NN
monotonic BDT
monotonic Lipschitz NN
data distribution
efficiency
Lifetime [ps]

Jet Tagging

Similar Task to Flavor Tagging, but for single jet



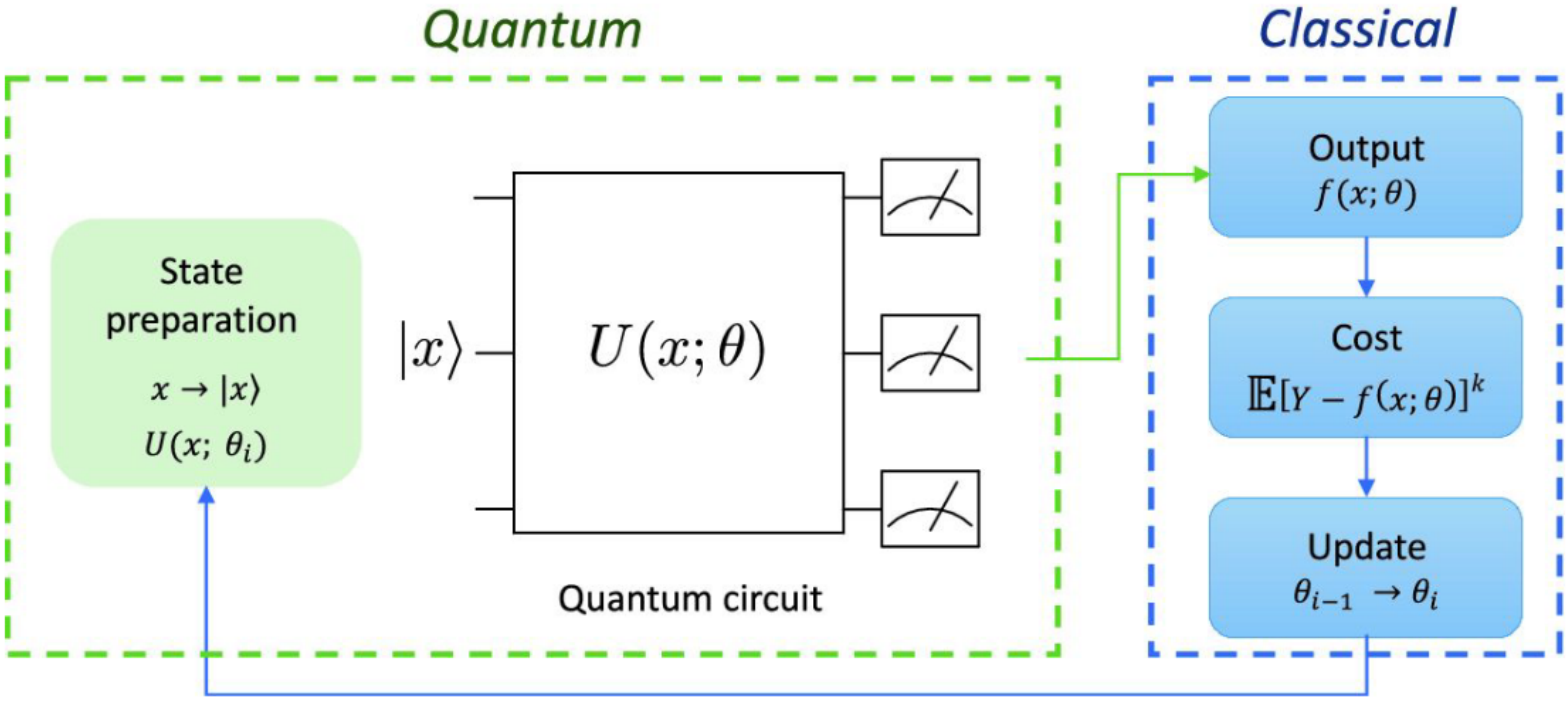
Input
16 Features of the Jet

Output
Jet Tag (Binary)

Standard classification problem

Explorative study: QML for Physics

classification problem = **Variational Quantum Classifier**



Data are fed into variational quantum circuit.

Measurements of qubits are mapped to probabilities for labels.

Probabilities are used to estimate a cost function which is optimized through a classical optimizer

Competitive!