Machine Learning

Top LHC France 2024 - LPNHE Paris

Anja Butter, LPNHE

LPNHE PARIS

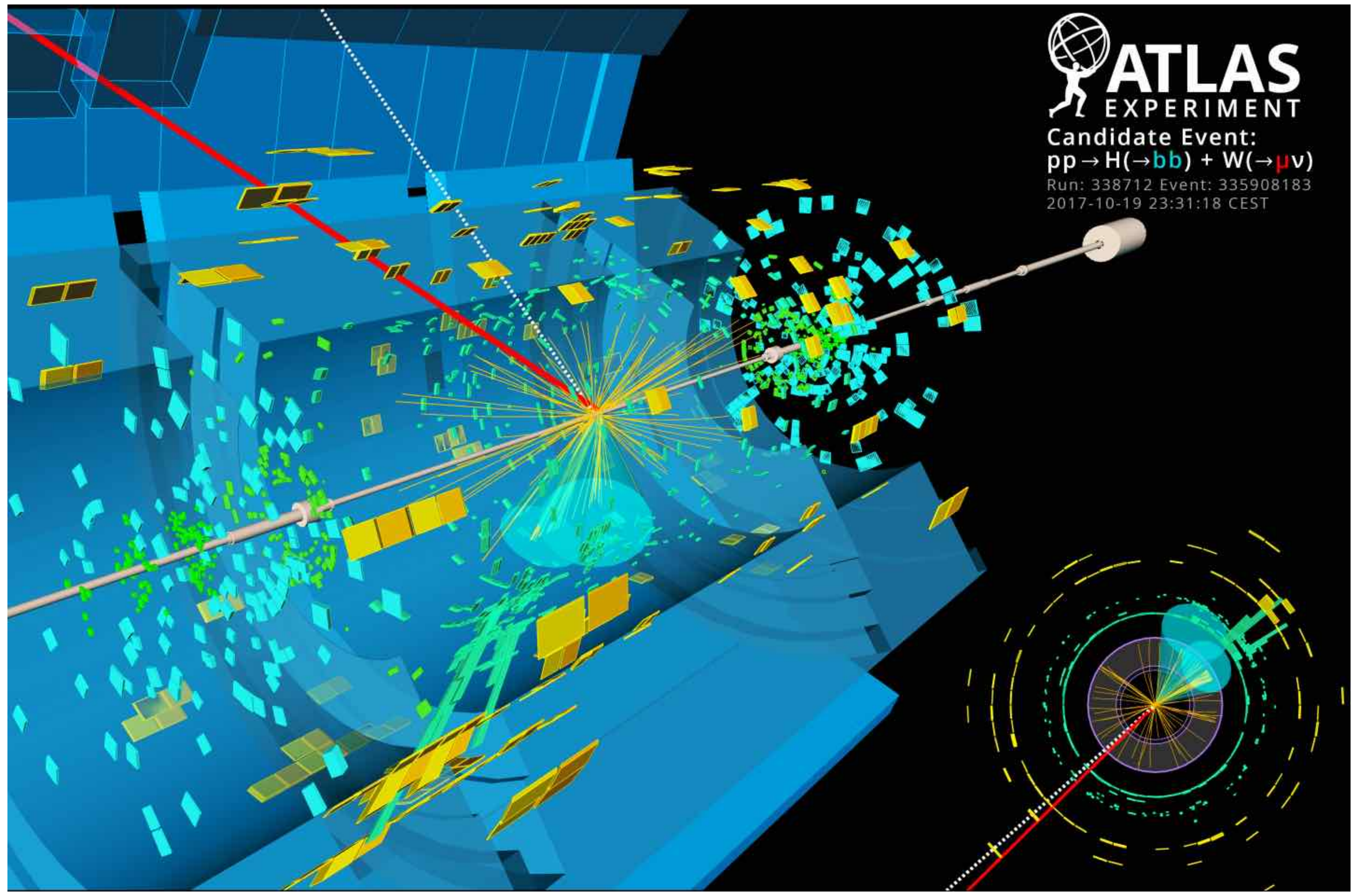
~~Machine Learning~~
Graph networks, Uncertainties and Unfolding

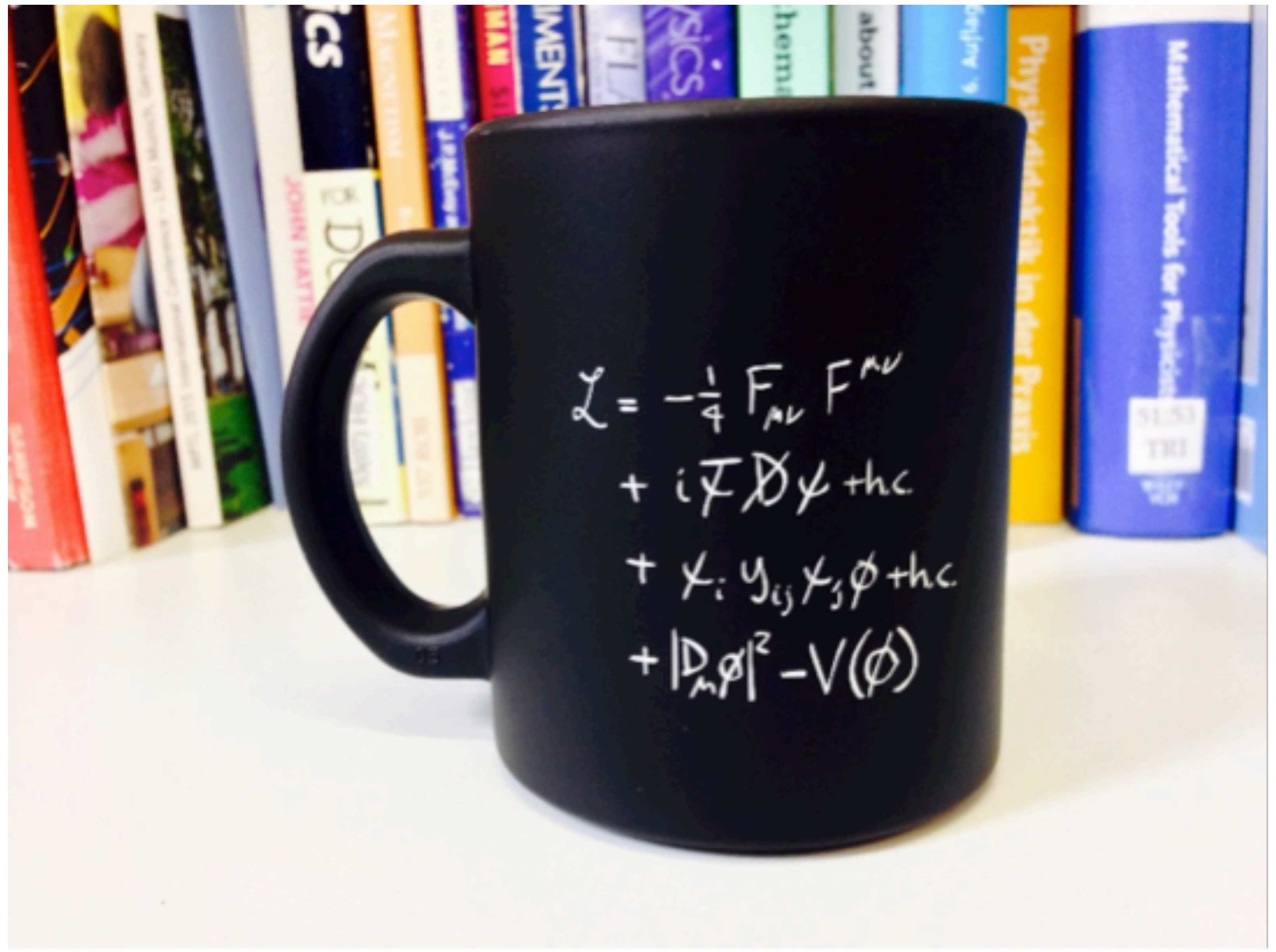
Top LHC France 2024 - LPNHE Paris

Anja Butter, LPNHE

LPNHE
PARIS

From Theory to Data and Back





Setting

- Large Hadron Collider at CERN
- Proton collisions at 13 TeV
- **Huge** dataset ~1Pb/s before trigger selection

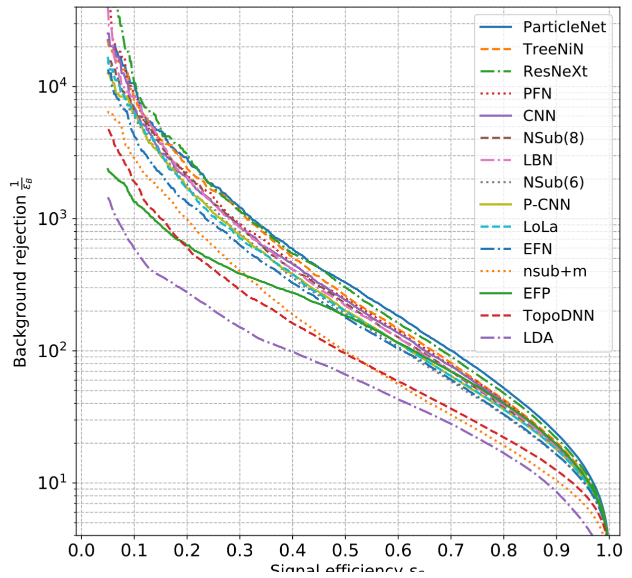
Goal

- Understand full dataset from **1st principles**
- Precision measurements of the SM
- Find signs of new physics (eg dark matter)

Need efficient extraction of all information from data → use data science methods

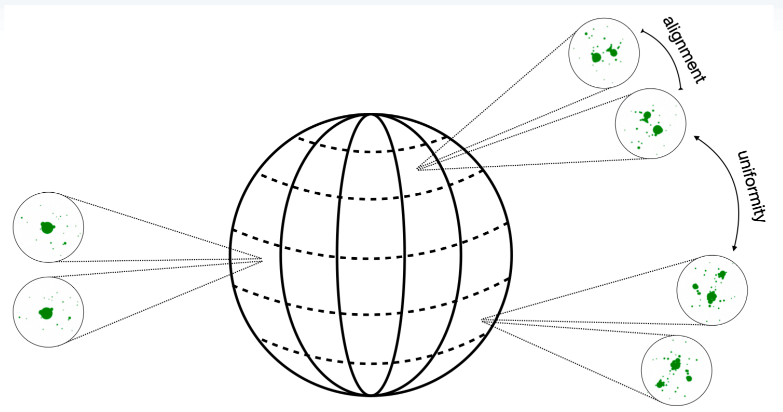
ML for big data in particle physics

Top tagging



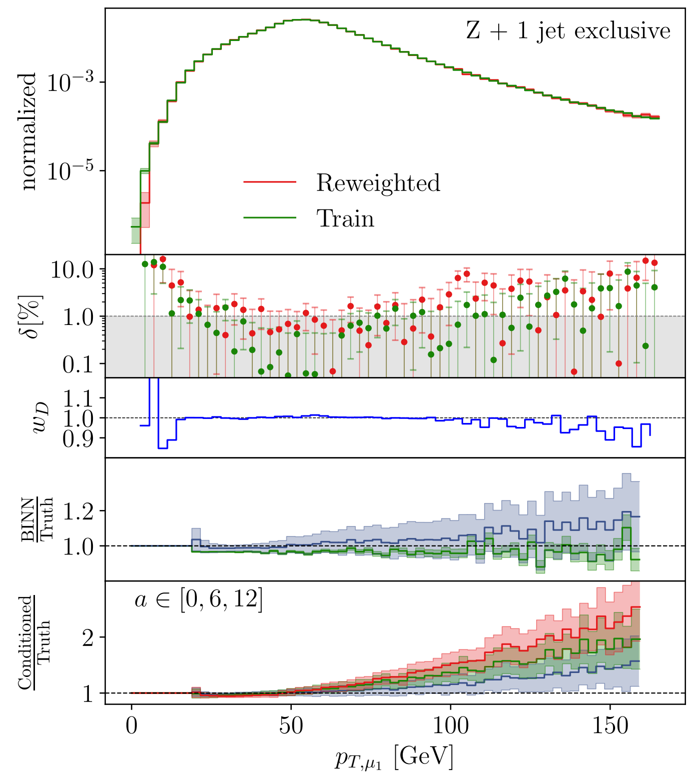
G. Kasieczka et al. [1902.09914]

Anomaly detection



B. Dillon et al. [2108.04253]

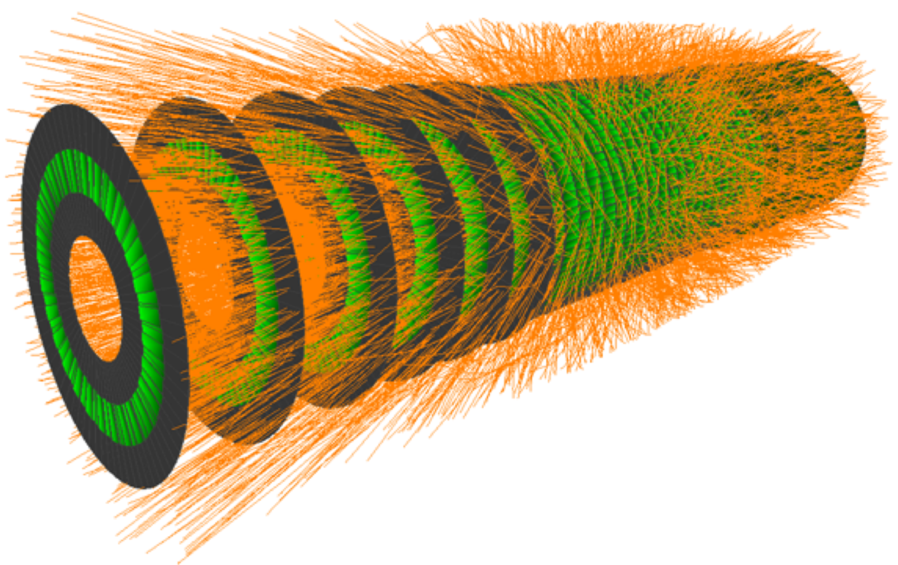
Event generation



A. Butter et al. [2110.13632]

Track reconstruction

Kaggle challenge



Classification

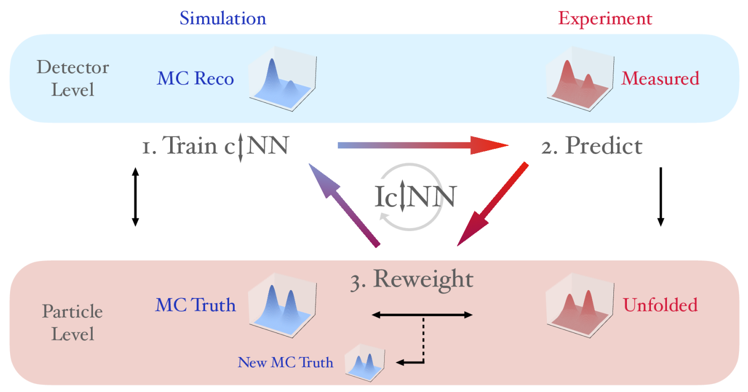
Generative models

Graph networks

Bayesian networks

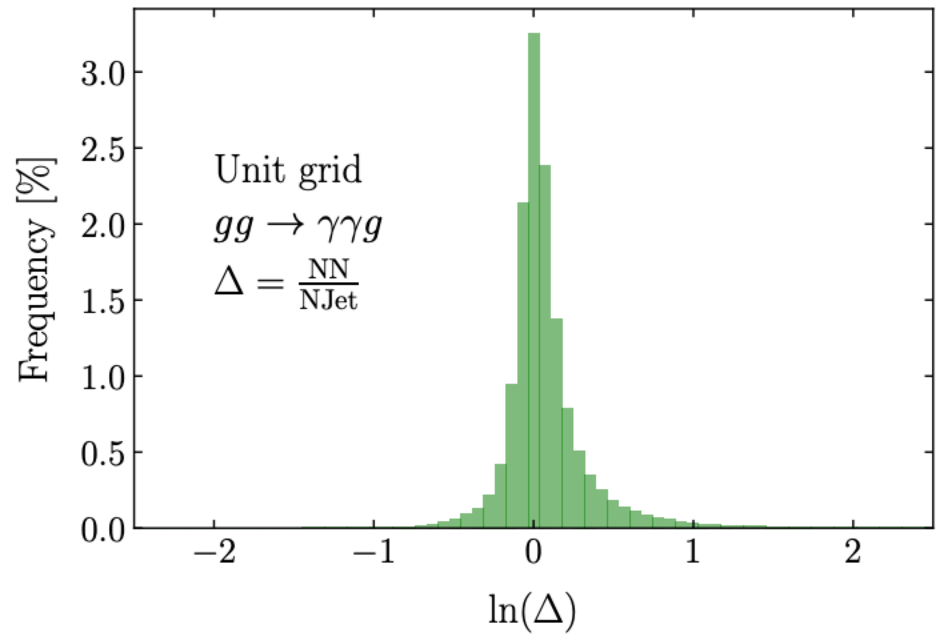
Regression

Unfolding



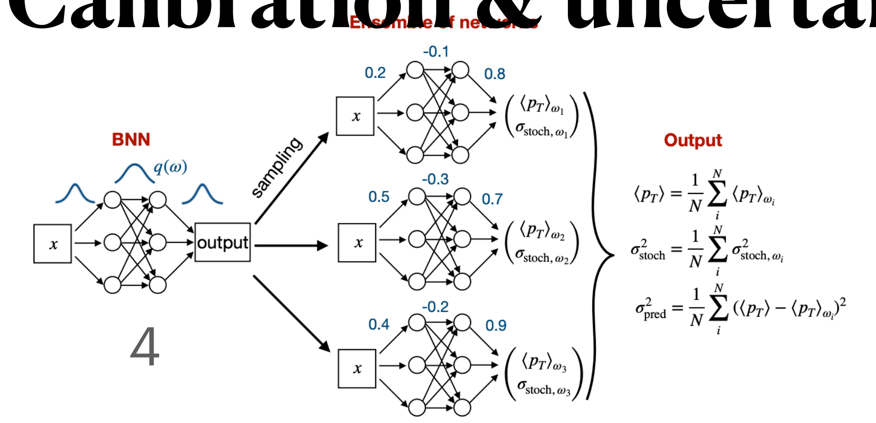
M. Backes et al. [2212.08674]

Amplitude estimation



J. Aylett-Bullock, et al. [2106.09474]

Calibration & uncertainties

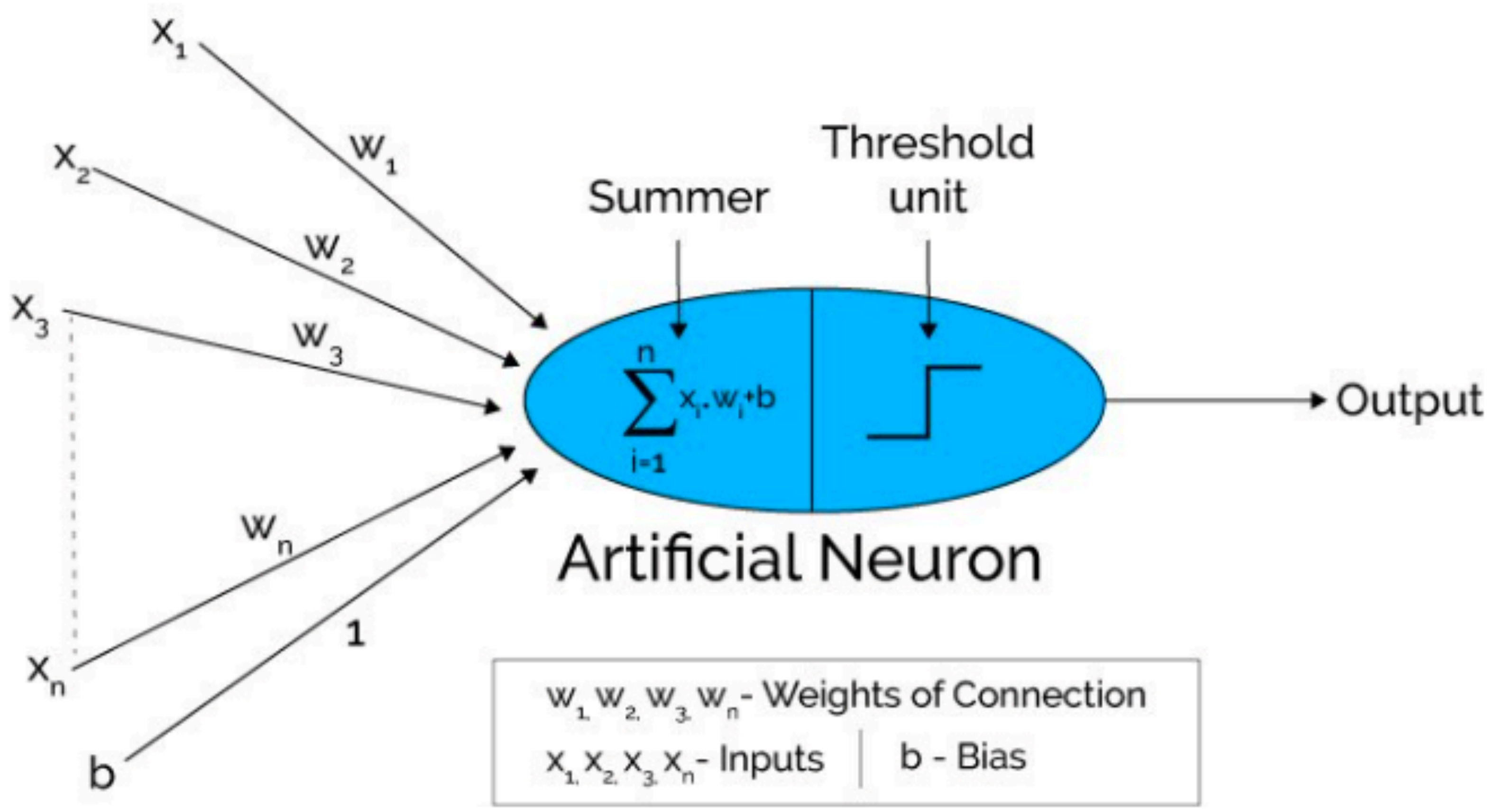


G. Kasieczka et al. [2003.11099]

Complete citations $\mathcal{O}(800)$
https://iml-wg.github.io/HEPML-LivingReview/

Neural networks in a nutshell

Neuron

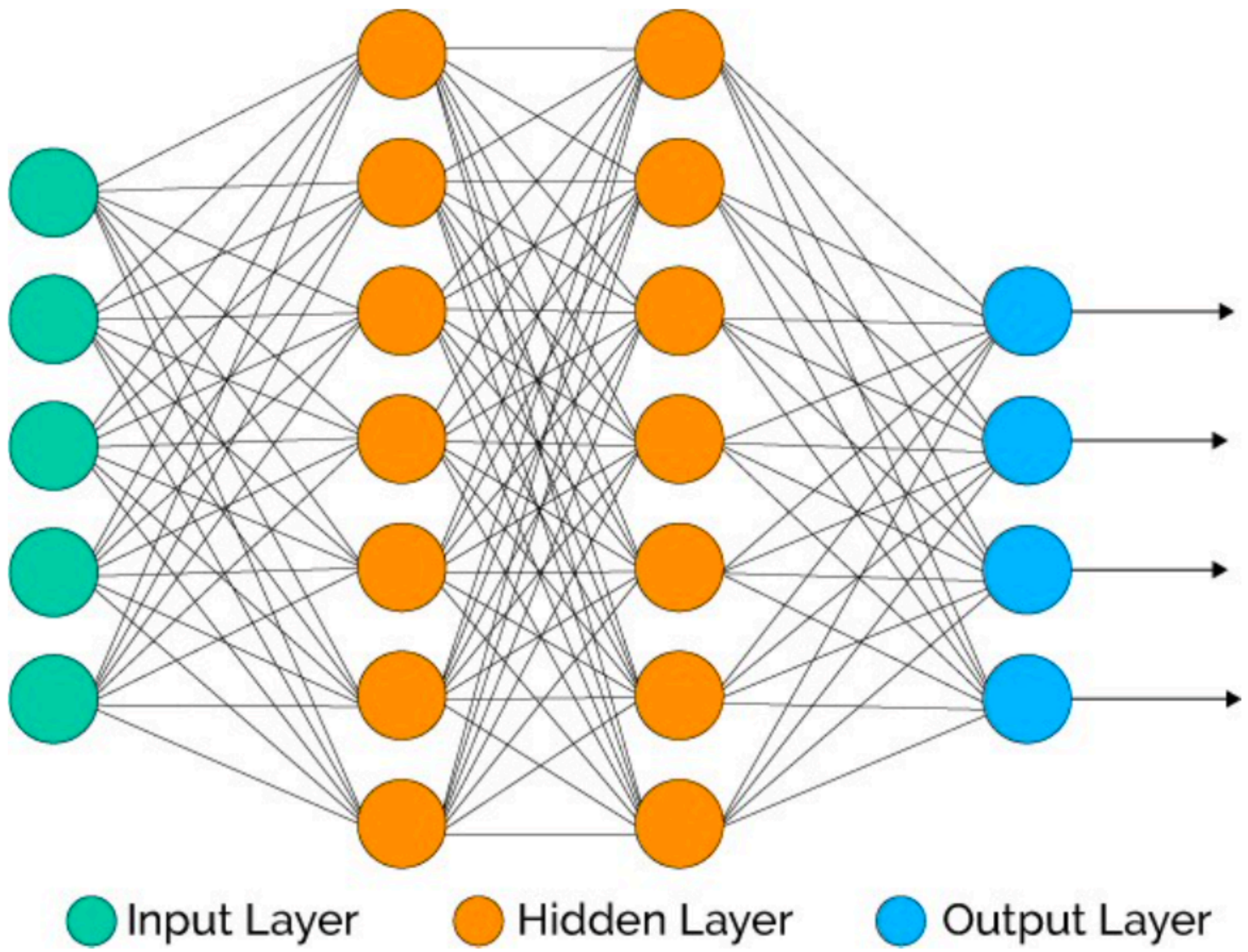


https://hackernoon.com/overview-of-artificial-neural-networks-and-its-applications-2525c1addff7

Popular activation functions:

- Re(ctified) L(inear) U(nit): $\Theta(x)x$
- Sigmoid: $\dfrac{1}{1+e^{-x}}$

Neural network

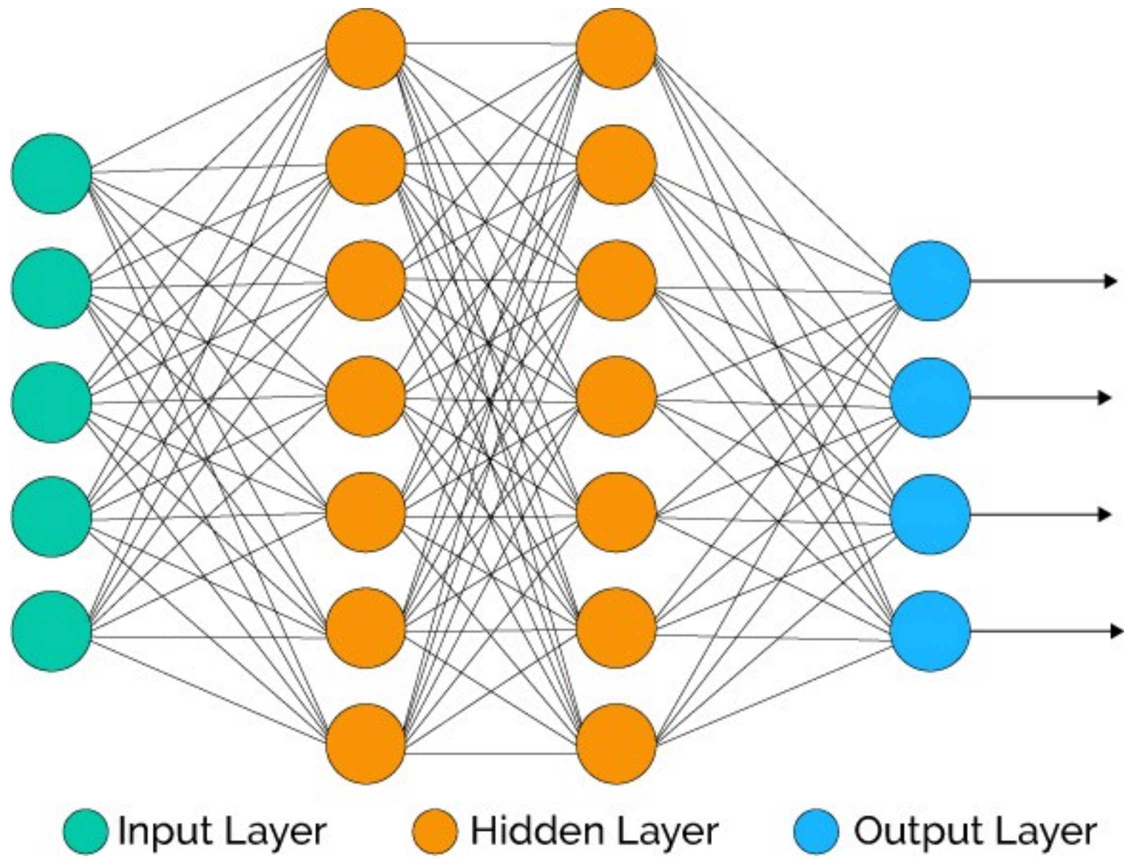


https://hackernoon.com/overview-of-artificial-neural-networks-and-its-applications-2525c1addff7

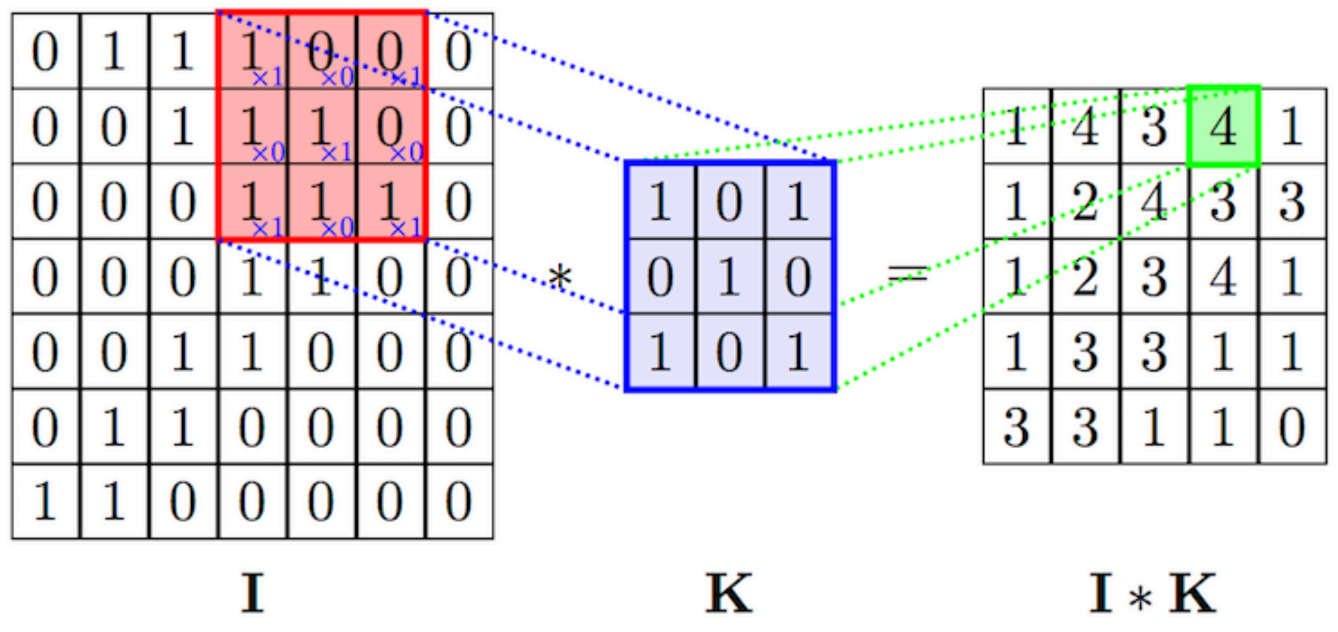
Training concept:

Minimization of loss function with back propagation (gradient descent)

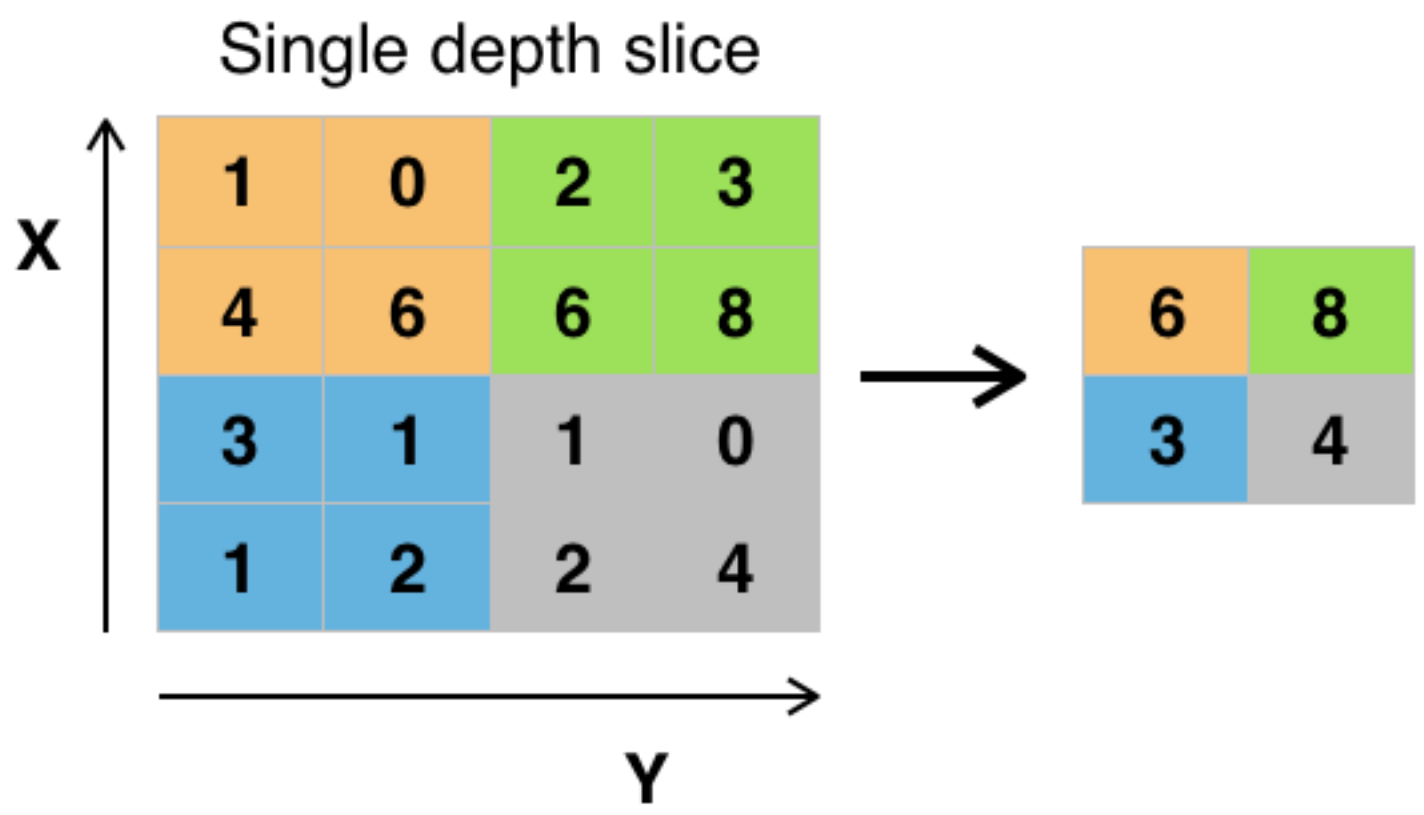
Different types of layers & networks



Dense networks
Standard network



Convolutional neural network (CNN)
Implement **equivariance**

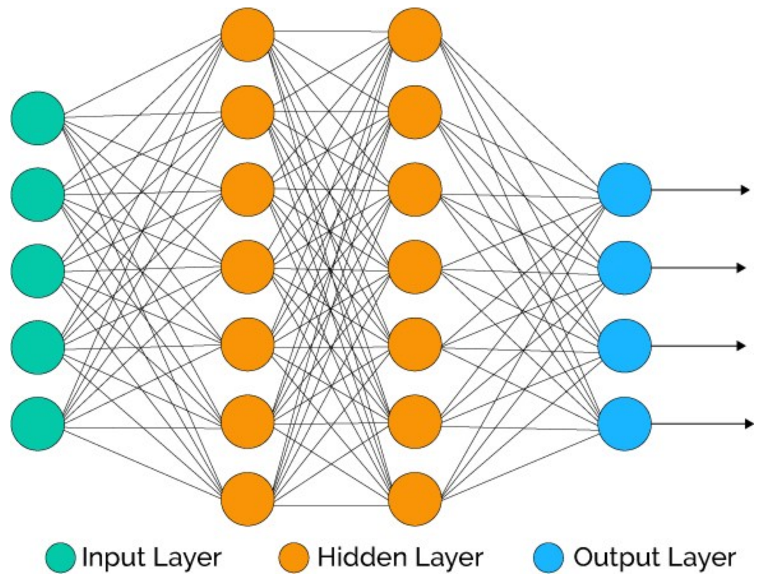


Pooling layer (max/min/mean/std)
Implement **invariance**

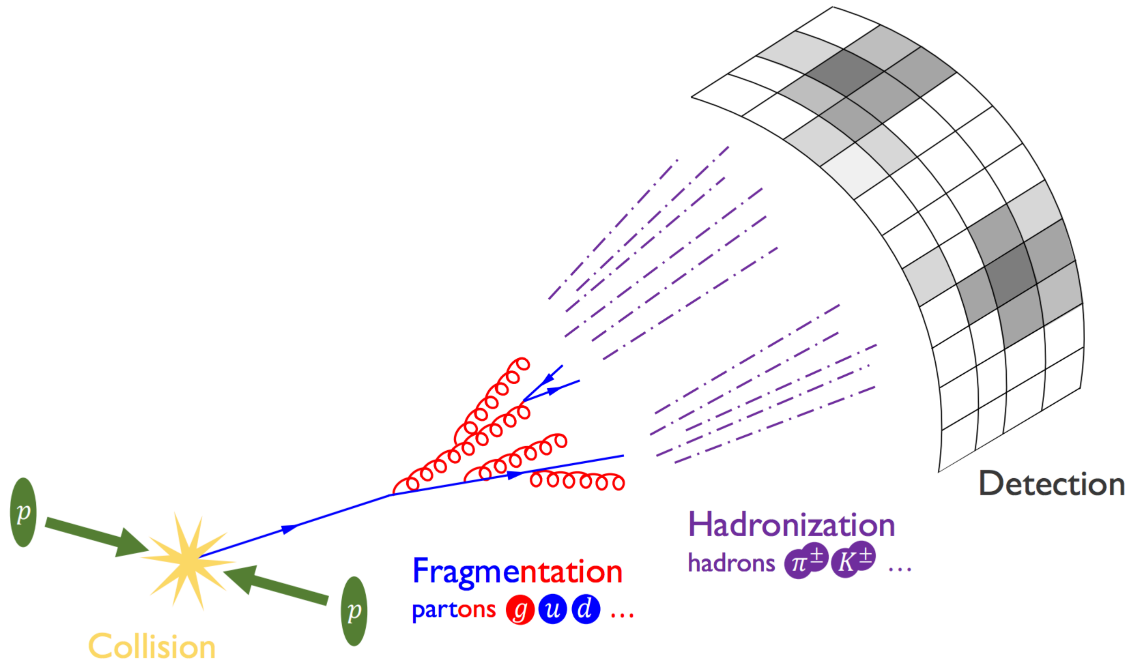
Data determine the network

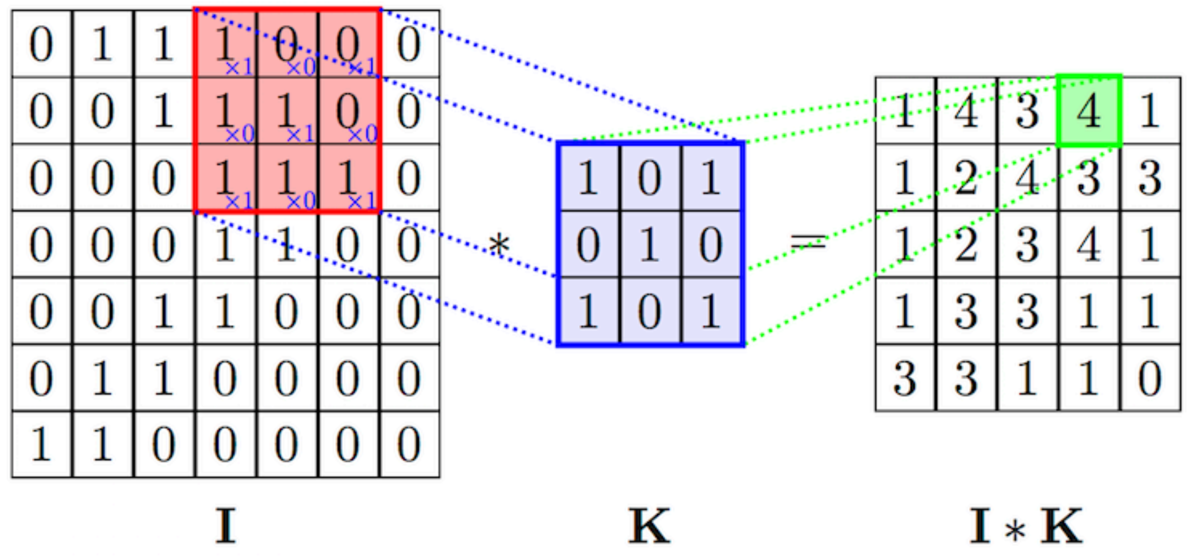
Data with intrinsic order
Example: events with structure

$$event = [p_{T,e^+}, p_{T,e^-}, \eta_{e^+}, \eta_{e^-}, p_{T,j}]$$

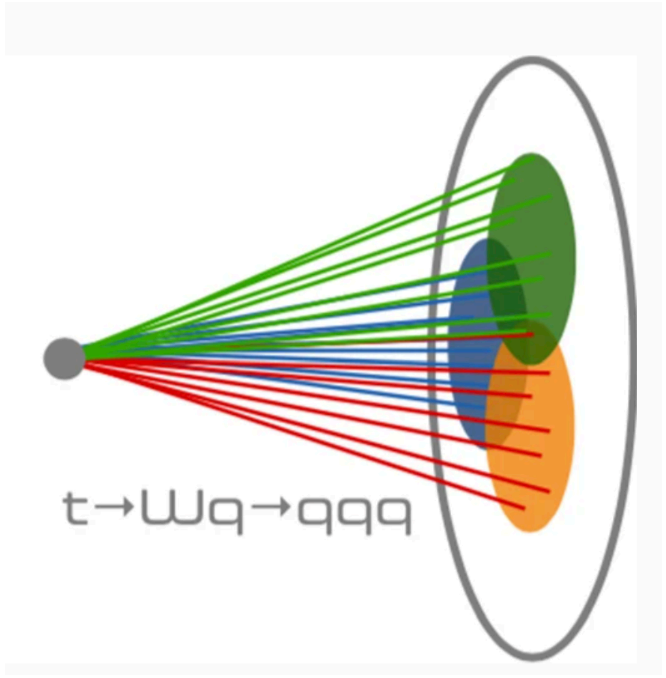


Images
Example: Calorimeter cells





Unordered sets
Example: Jet constituents

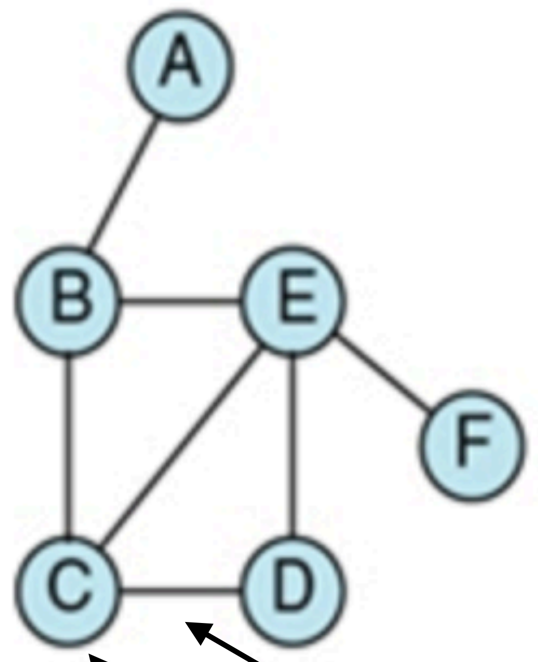


?

Graph networks

Graph networks

How to represent a graph



A
B
C
D
E
F

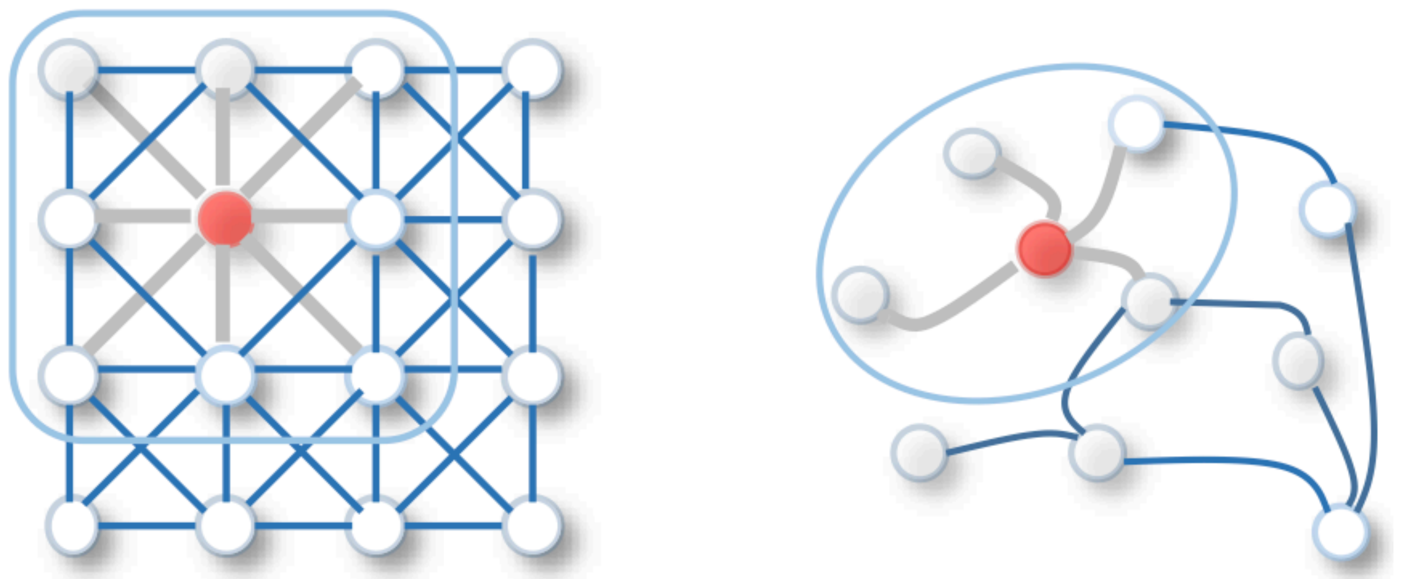
Vertex vector

	A	B	C	D	E	F
A	0	1	0	0	0	0
B	1	0	1	0	1	0
C	0	1	0	1	1	0
D	0	0	1	0	1	0
E	0	1	1	1	0	1
F	0	0	0	0	1	0

Adjacency matrix

Node Edge

Image vs Graph

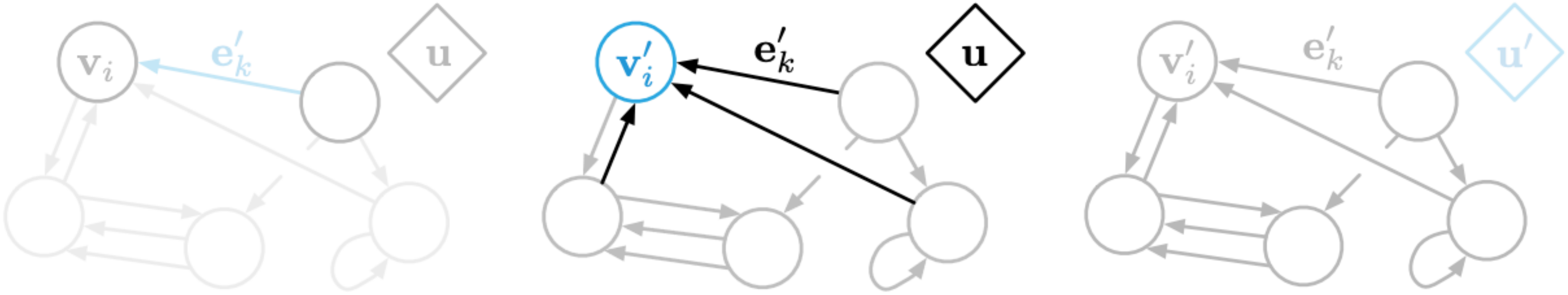


pixels → node

neighbouring pixel → neighbouring node (graph edges)

Graph networks

1806.01261



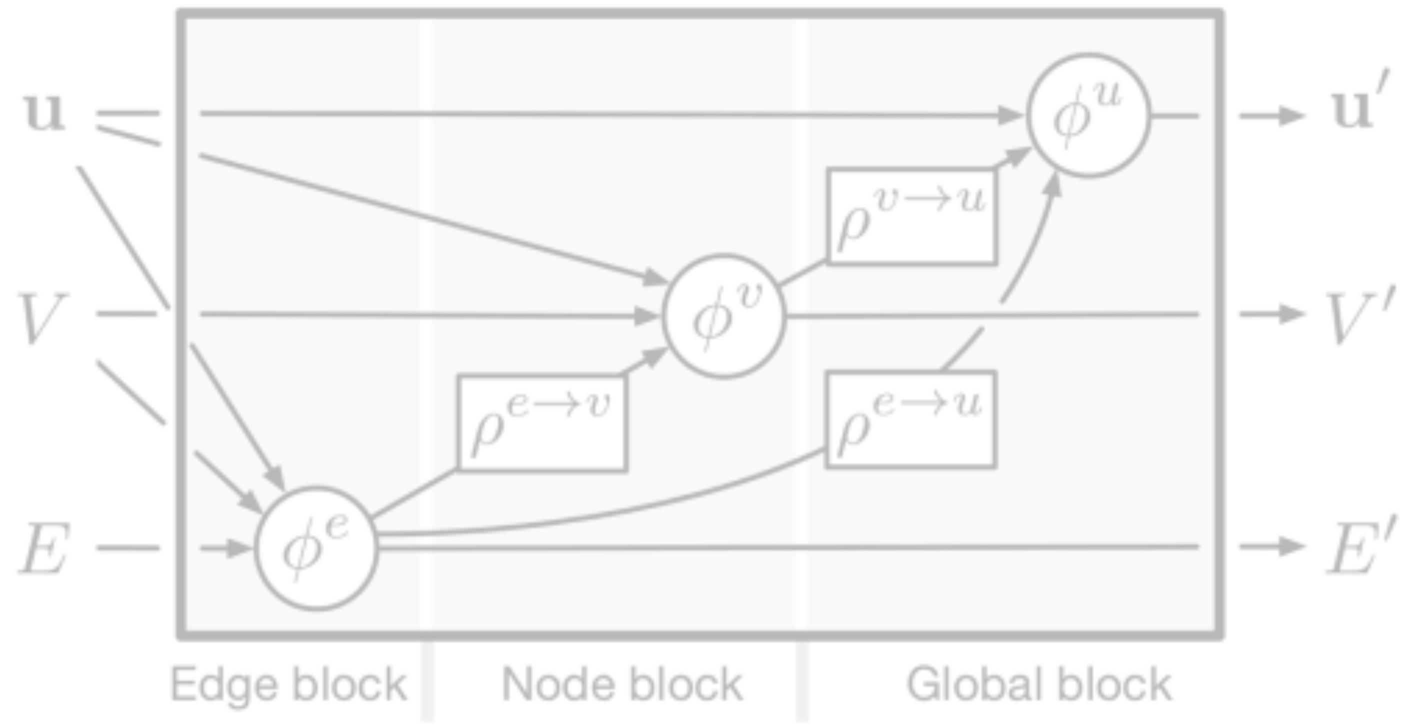
(a) Edge update (b) Node update (c) Global update

→ edge convolution

$$\vec{v}_i' = \frac{1}{k}\sum_{j=1}^{k} h_\Theta(\vec{v}_i, \vec{v}_{i_j} - \vec{v}_i)$$

Aggregation function

h **is independent of** i, j



(a) Full GN block

What can we do with graph networks?

Examples

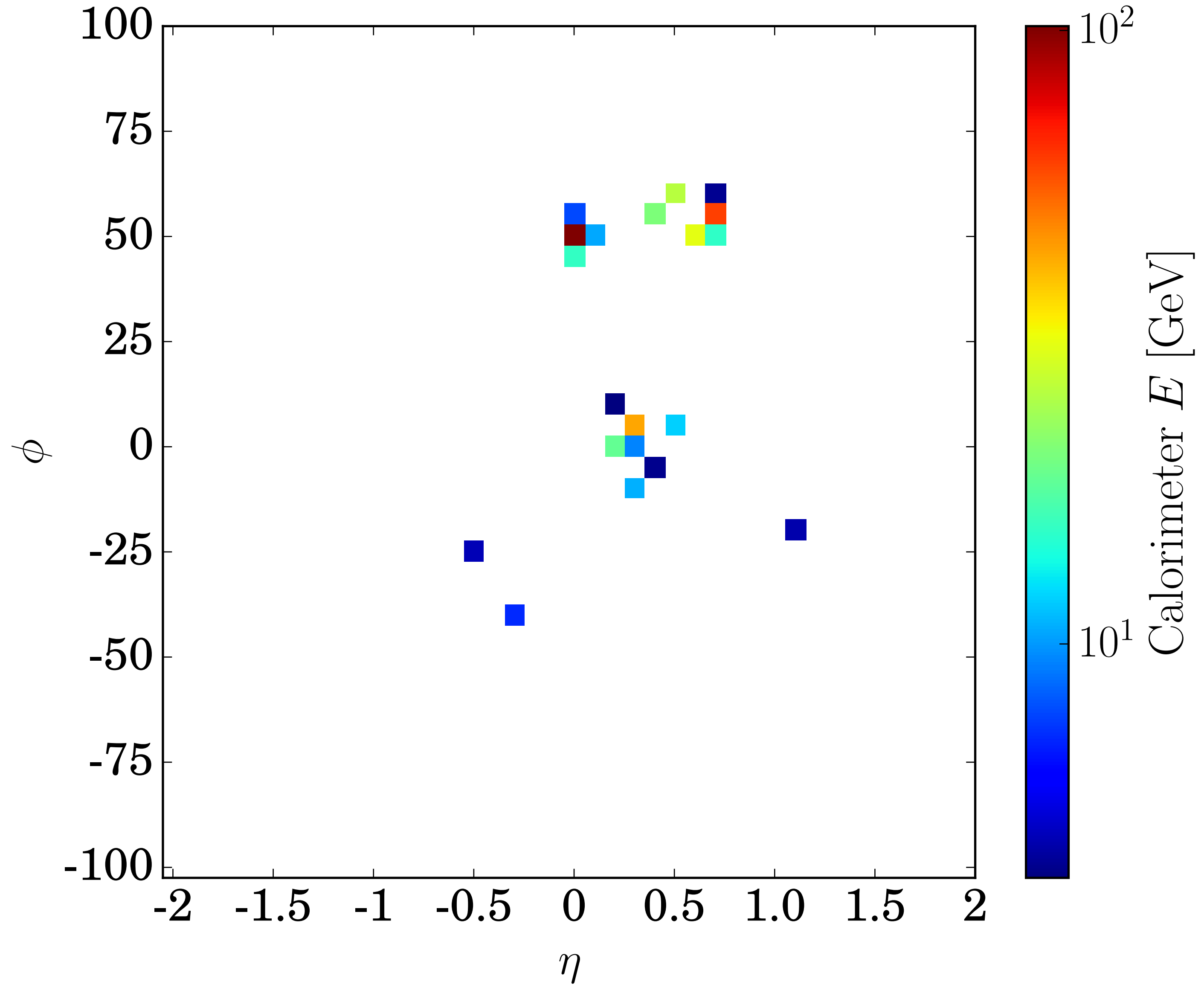
- Node classification (assign label to a node)
  - *Does this hit belong to my track?*
- Graph classification (assign label to graph)
  - *Top vs QCD jet*
  - *B-jet identification*
  - *Event classification (Signal vs Background)*
- Graph generation
  - *Generate new jet*
- Embedding into alternative space for better interpretation

Top jet classification

1707.08966

Data set

- Top vs QCD
- Calorimeter image & Particle Flow objects
- Pythia8 + Delphes 3
- FastJet3 anti-kt with R = 1.5
- $|\eta_{fat}| < 1.0$, $p_{T,jet} = 350 \ \dots \ 450$ GeV



Calorimeter image:
Mostly empty & No tracking information

→ CNN not suited

Instead:
→ Set of particle flow objects
→ They become set of nodes

Optional: Build graph for instance from nearest neighbors

Lorentz Layer

Physics inspired layer that acts on nodes [1707.08966]

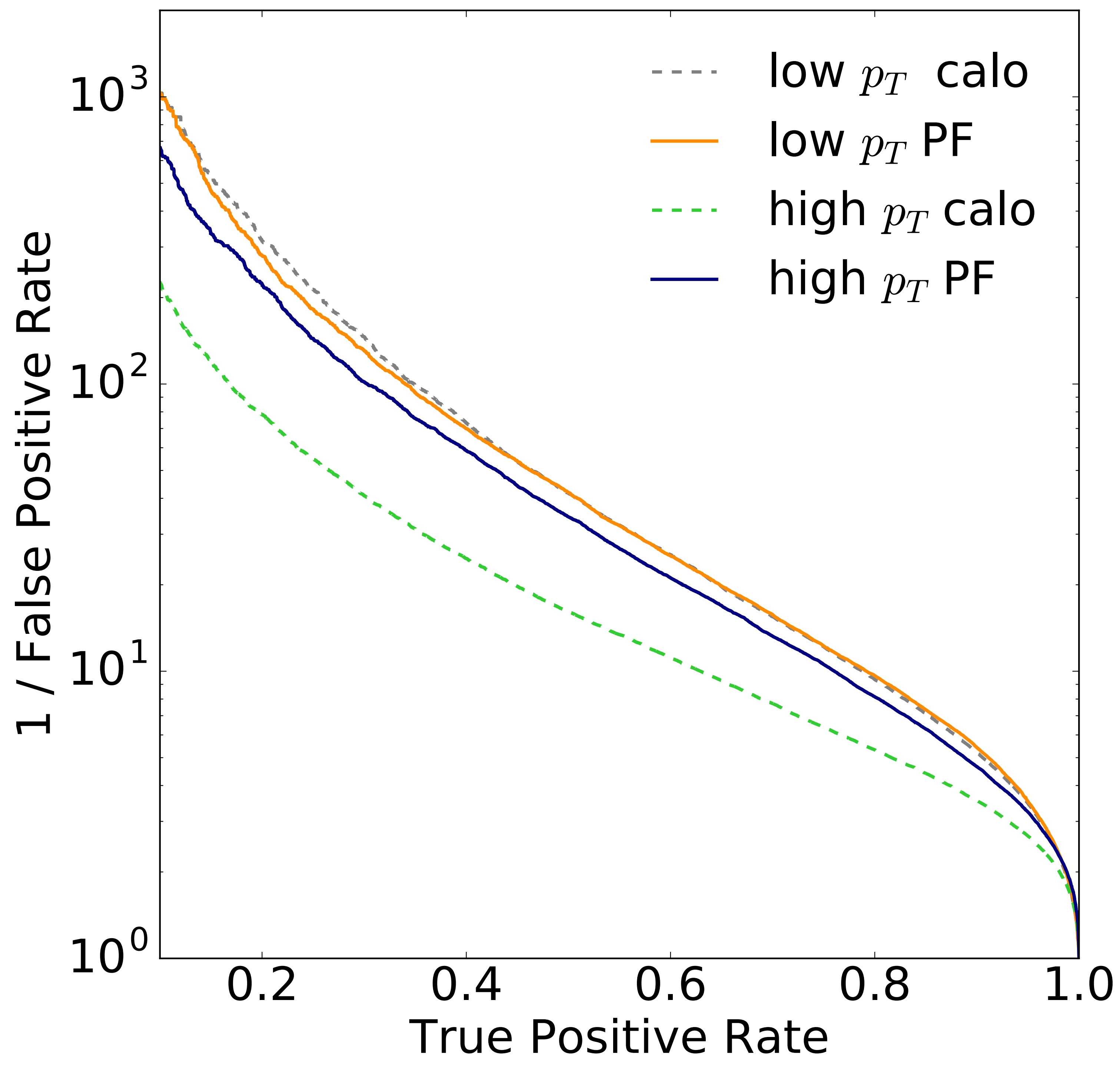
Transform Lorentz vectors into physics motivated objects.

$$\tilde{k}_j \xrightarrow{\text{LoLa}} \hat{k}_j = \begin{pmatrix} m^2(\tilde{k}_j) \\ p_T(\tilde{k}_j) \\ w_{jm}^{(E)} \, E(\tilde{k}_m) \\ w_{jm}^{(d)} \, d_{jm}^2 \end{pmatrix}$$

$$d_{jm}^2 = (\tilde{k}_j - \tilde{k}_m)_\mu \, g^{\mu\nu} \, (\tilde{k}_j - \tilde{k}_m)_\nu$$

Transformation in place
Aggregation over other objects
Distance d_{jm} encodes edge information

Not exactly graph concept, as weights are index dependent



At high p_T:
PF based network outperforms CNN
→ tracking information is crucial !

ParticleNet

[1902.08570, H. Qu, L. Gouskos]

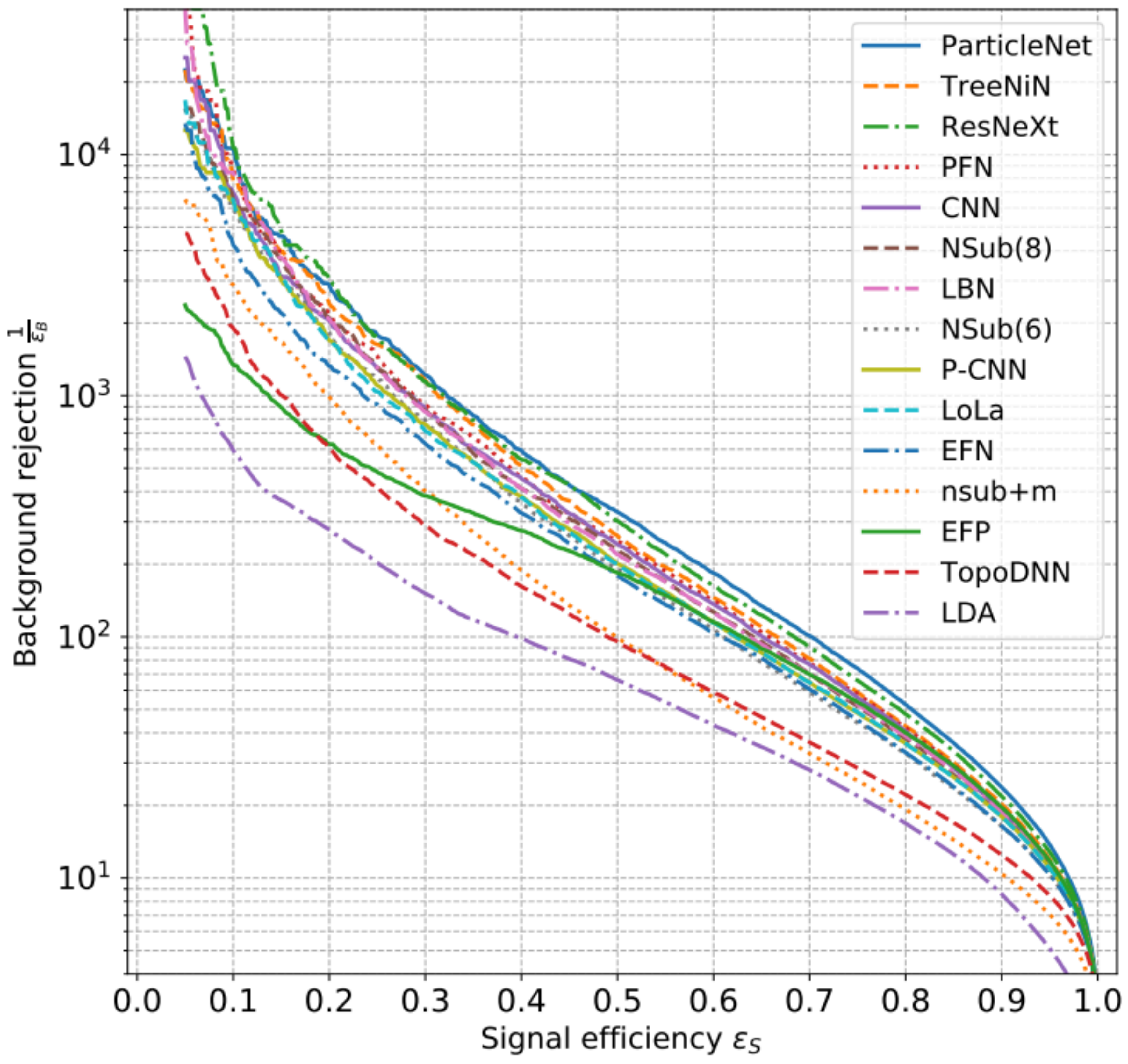
- Jet = unordered set of particles
- Particle cloud (permutation invariant)
- Translational symmetry

- K-nearest neighbours define local patch
  $x_i' = \boxdot_{j=1}^{k} \phi_\theta(x_i, x_{i_j} - x_i)$
  - $\boxdot$ indicates an aggregation function (max, **mean**, sum, ...)
  - ϕ_θ is a 3 layer MLP

- Dynamically update edges for each layer

- Hyperparameter:
  - # neighbors, latent dim, dropout, batchnorm, learning rate, ...

Variable
$\Delta\eta$
$\Delta\phi$
$\log p_T$
$\log E$
$\log \frac{p_T}{p_T(\text{jet})}$
$\log \frac{E}{E(\text{jet})}$
ΔR
q
`isElectron`
`isMuon`
`isChargedHadron`
`isNeutralHadron`
`isPhoton`



Lorentz Net

2201.08187, S. Gong et al.

Combination of graph network and physics knowledge

Lorentz Net encodes Lorentz **equivariance**

$$x_i^{l+1} = x_i^l + c \sum_{j\in[N]} \phi_x(m_{ij}^l) \cdot x_j^l$$

$$m_{ij}^l = \phi_e \left(h_i^l, h_j^l, \psi(\|x_i^l - x_j^l\|^2), \psi(\langle x_i^l, x_j^l \rangle) \right)$$

x^0 are the 4-momenta
h^0 embedds charge, PID, etc.
$\langle \cdot , \cdot \rangle$ Minkowski product
$\psi(\,\cdot\,) = \mathrm{sgn}(\,\cdot\,)\ \log(|\cdot| + 1)$
ϕ_x are neural networks

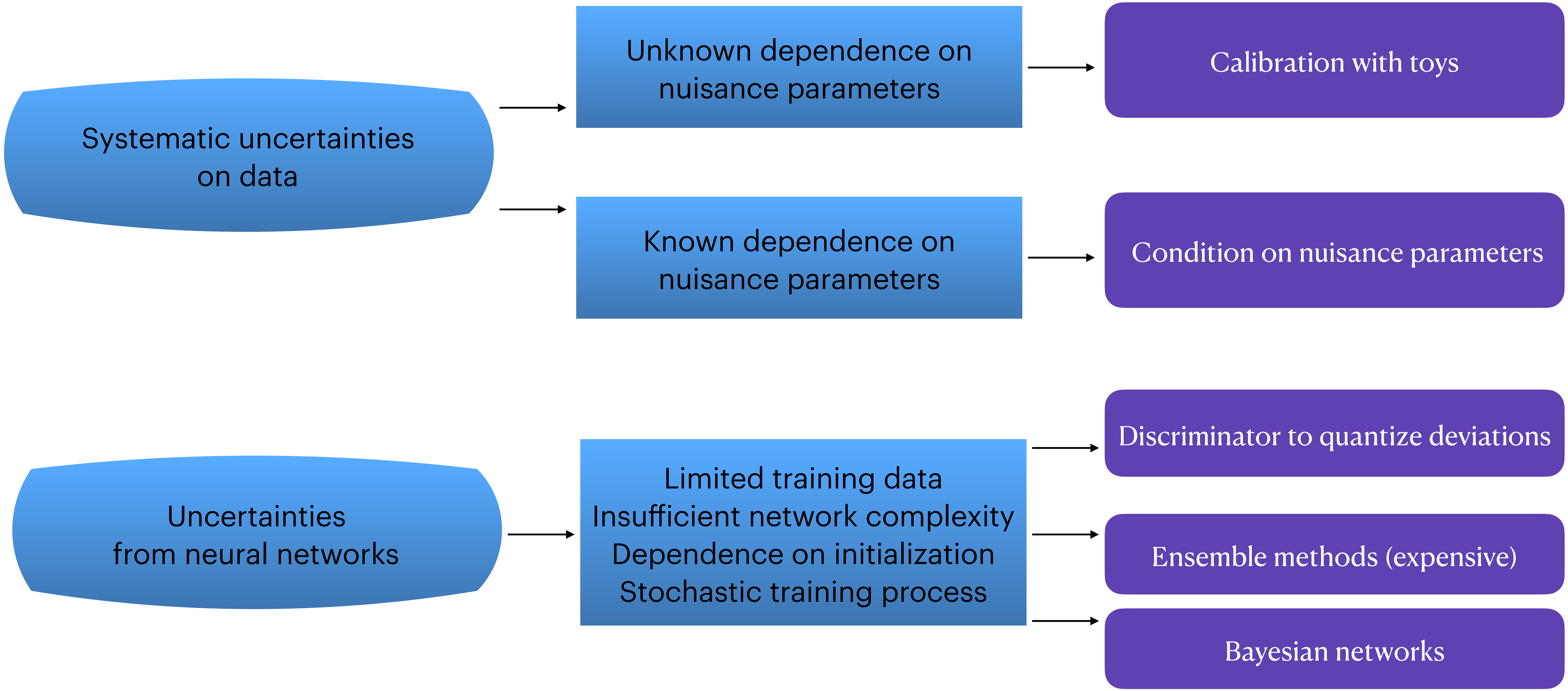
Top tagging dataset

Training Fraction	Model	Accuracy	AUC	$1/\varepsilon_B$ $(\varepsilon_S = 0.5)$	$1/\varepsilon_B$ $(\varepsilon_S = 0.3)$
0.5%	ParticleNet	0.913	0.9687	77 ± 4	199 ± 14
	LorentzNet	**0.929**	**0.9793**	$\mathbf{176 \pm 14}$	$\mathbf{562 \pm 72}$
1%	ParticleNet	0.919	0.9734	103 ± 5	287 ± 19
	LorentzNet	**0.932**	**0.9812**	$\mathbf{209 \pm 5}$	$\mathbf{697 \pm 58}$
5%	ParticleNet	0.931	0.9807	195 ± 4	609 ± 35
	LorentzNet	**0.937**	**0.9839**	$\mathbf{293 \pm 12}$	$\mathbf{1108 \pm 84}$

→ Physics layers enable better performance for smaller datasets

What about uncertainties?

Uncertainties vs methods



Limitations of a standard network

Example
$gg \to \gamma\gamma g(g)$ @LO

90k training amplitudes
870k test amplitudes

Standard approach

Training data
$T =$ (phase space points x, Amplitudes $A'(x)$)

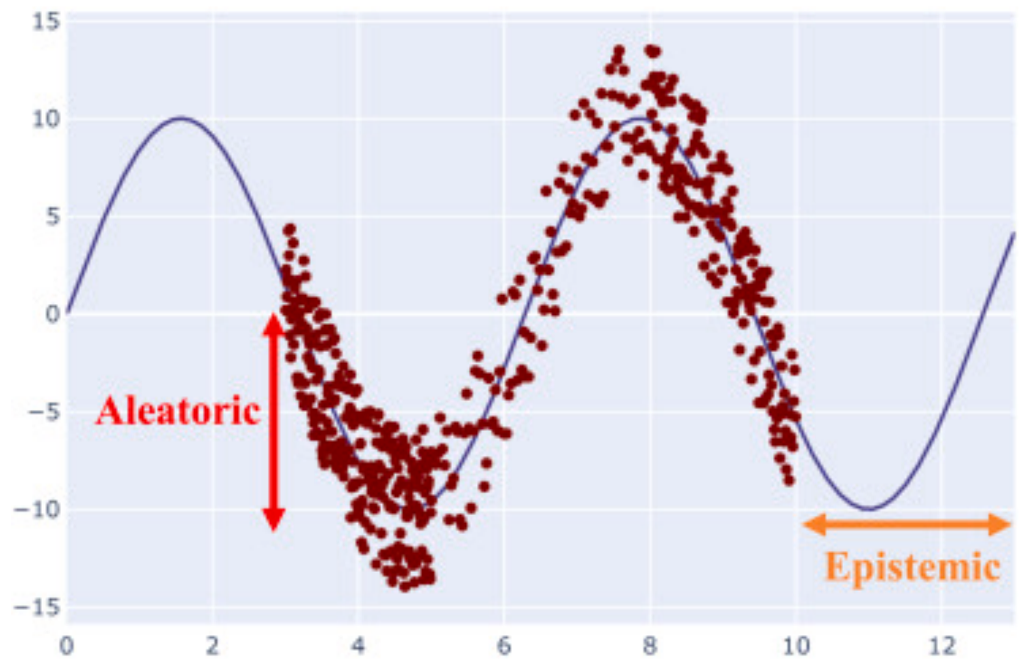
Loss
$\mathscr{L} = (A'(x) - NN(x))^2$

PROBLEM: For limited data there is **no unique solution**

→ Need better formulation of the problem

→ Find $p(A \mid x, T)$ (from now on x is implicit)

→ $p(A) = \int \mathrm{d}w \; p(A \mid w) p(w \mid T)$

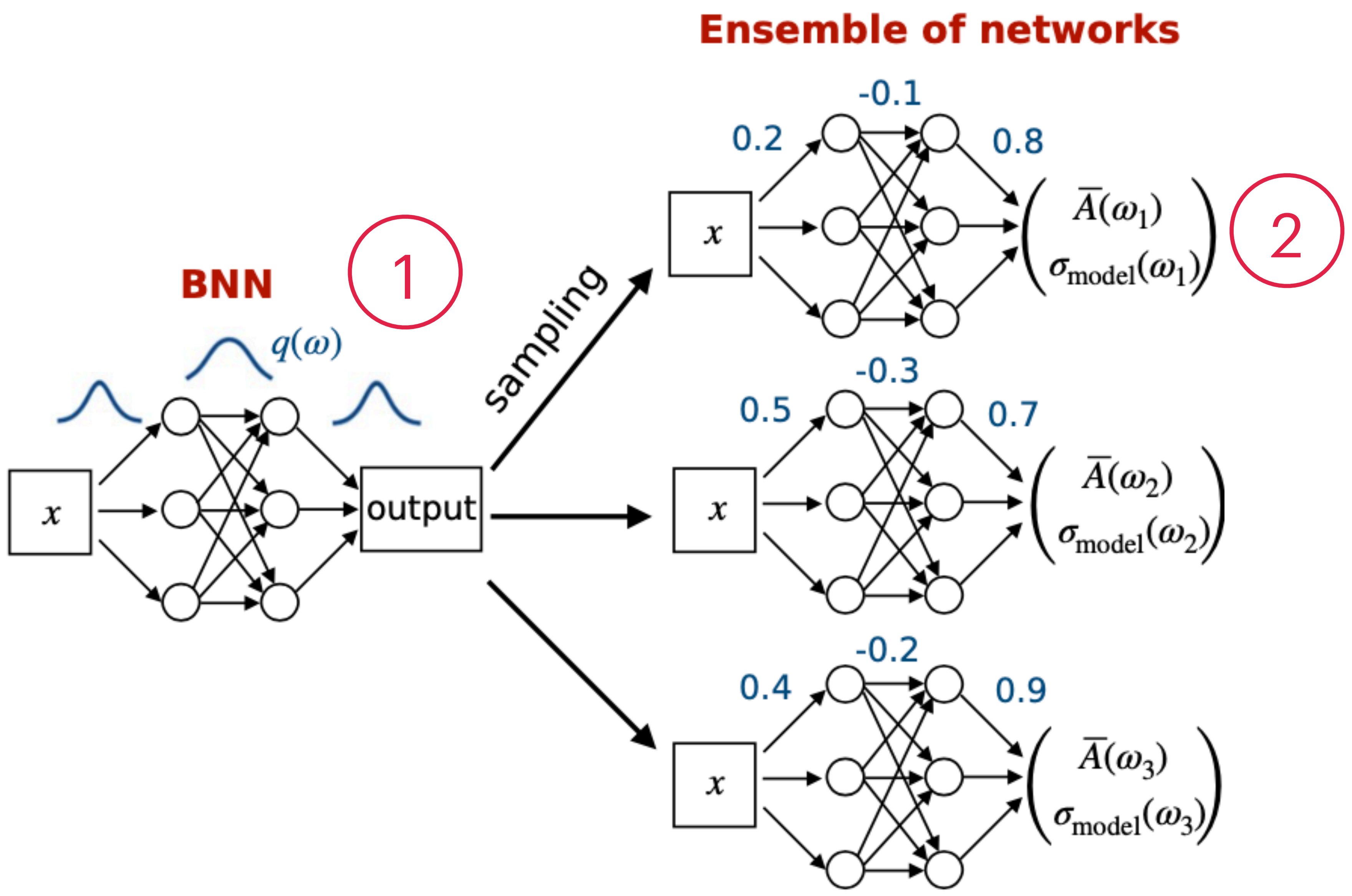


Capturing probabilities with Bayesian networks

$$\boxed{p(A) = \int \mathrm{d}w\ p(A\,|\,w)p(w\,|\,T) \approx \int \mathrm{d}w\ p(A\,|\,w)q(w)}$$

2 1

Bayesian network



Building the loss function

Approximate $q(w)$ by minimizing KL divergence

$$\begin{aligned}\mathscr{L}_{BNN} &= \mathrm{KL}[q(w), p(w\,|\,T)] \\ &= \int dw\ q(w)\ \log \frac{q(w)}{p(w\,|\,T)} \\ &= \int dw\ q(w)\ \log \frac{q(w)p(T)}{p(w)p(T\,|\,w)} \\ &= \mathrm{KL}[q(w), p(w)] - \int dw\ q(w)\ \log p(T\,|\,w)\end{aligned}$$

1 Gaussian prior

$$\boxed{\frac{\sigma_q^2 - \sigma_p^2 + (\mu_q - \mu_p)^2}{2\sigma_p^2} + \log \frac{\sigma_p}{\sigma_q}}$$

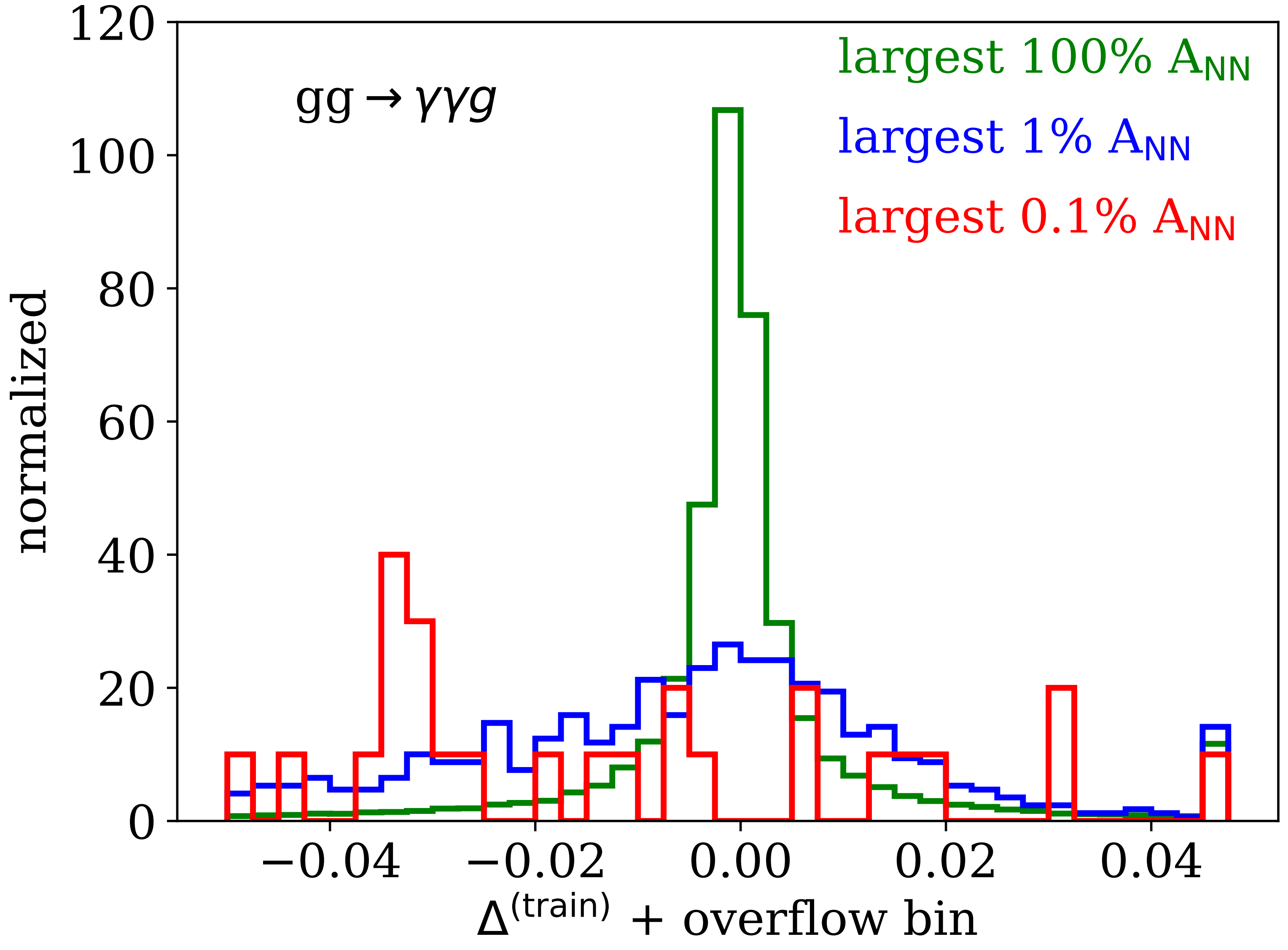
2 Gaussian uncertainty

$$\boxed{\frac{\left|\bar{A}_j(\omega) - A_j^{(\text{truth})}\right|^2}{2\sigma_{\text{model},j}(\omega)^2} + \log \sigma_{\text{model},j}(\omega)}$$

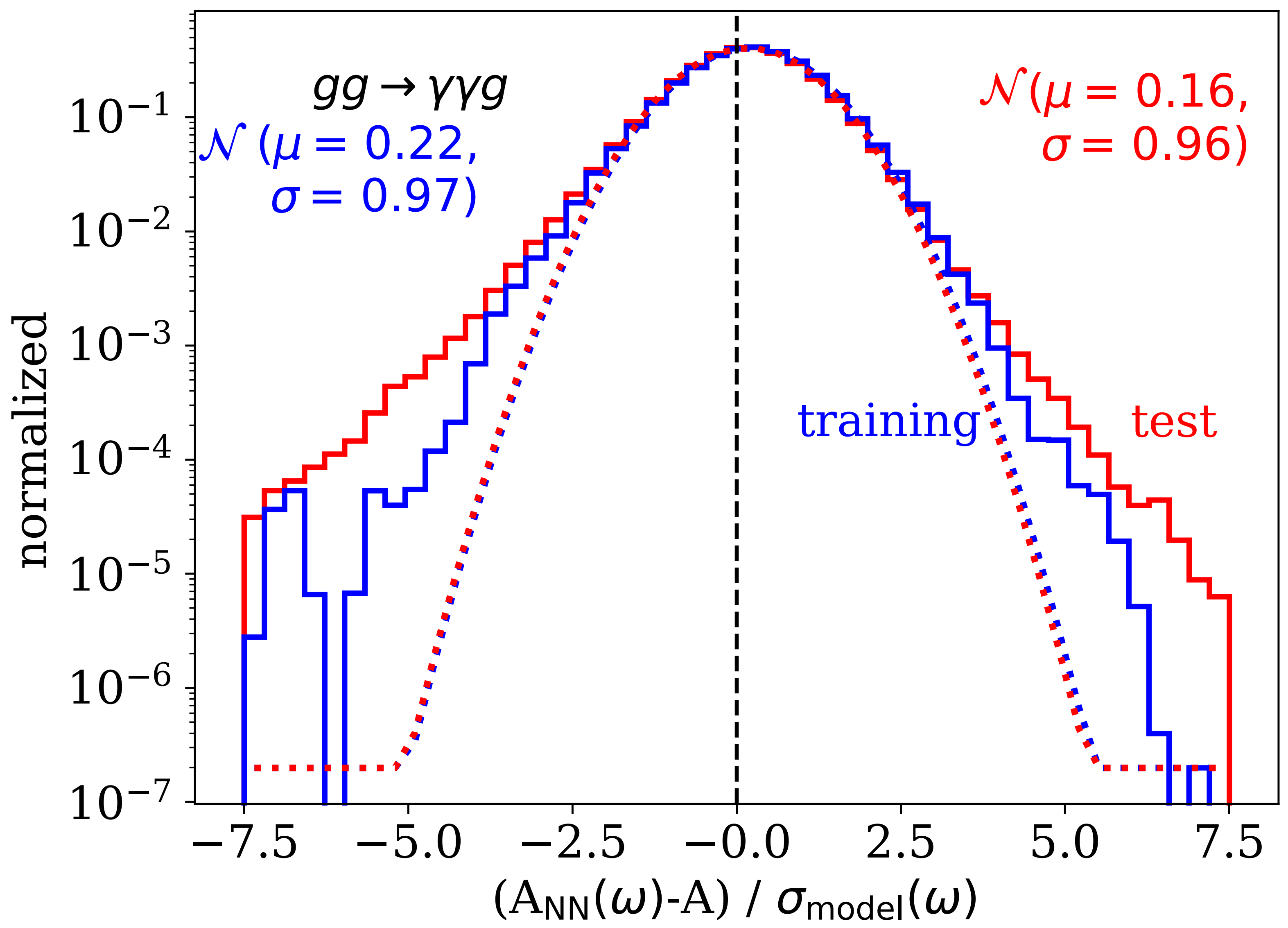
Results - out of the box

+ Deviations at 1 percent level

Precision $\Delta^{(train)} = \dfrac{A_{NN} - A_{train}}{A_{NN}}$



Calibration $\Delta^{(train)} = \dfrac{A_{NN} - A_{train}}{A_{NN}}$

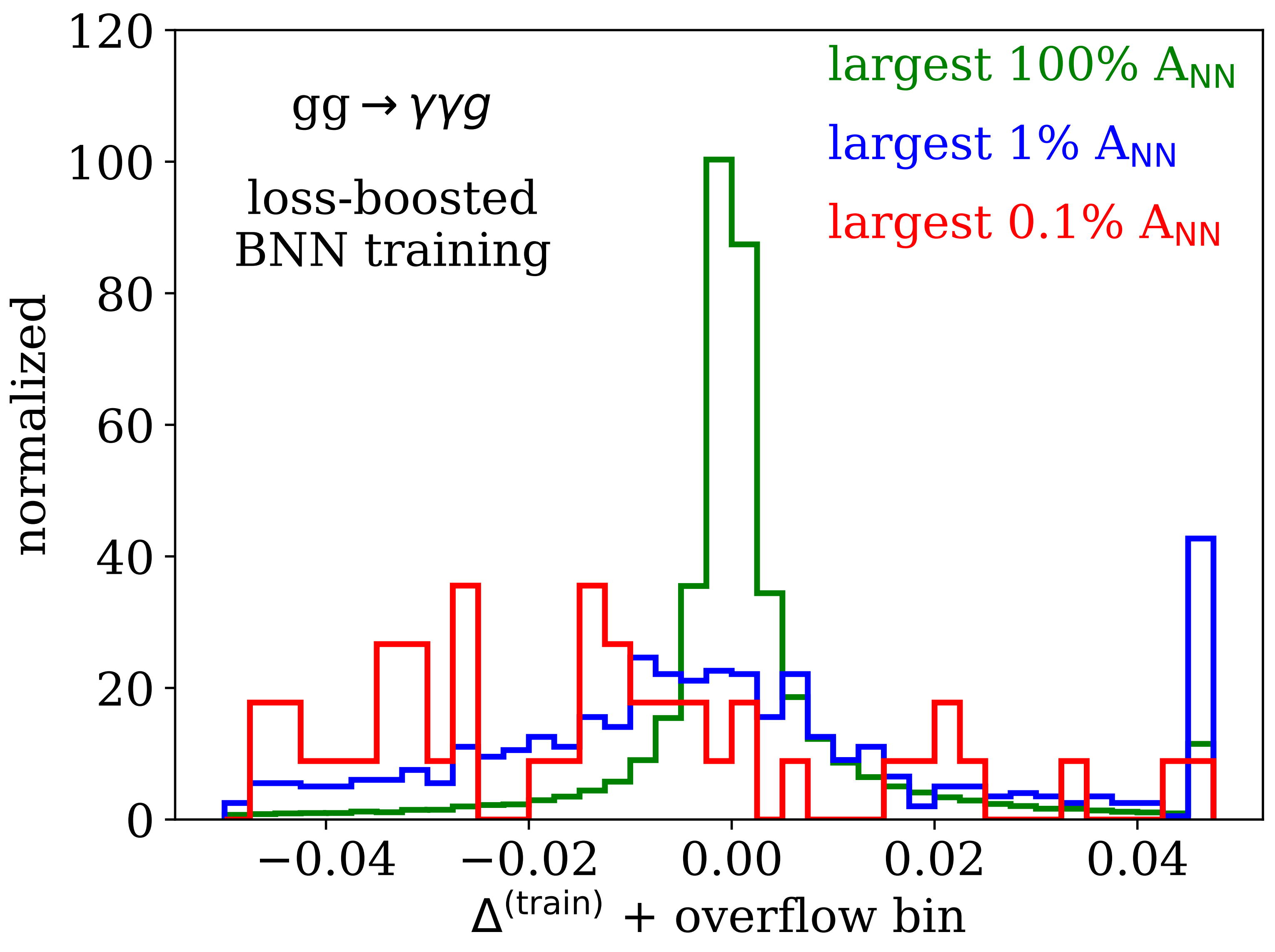


Performance worse for rare points with large amplitudes (collinear)

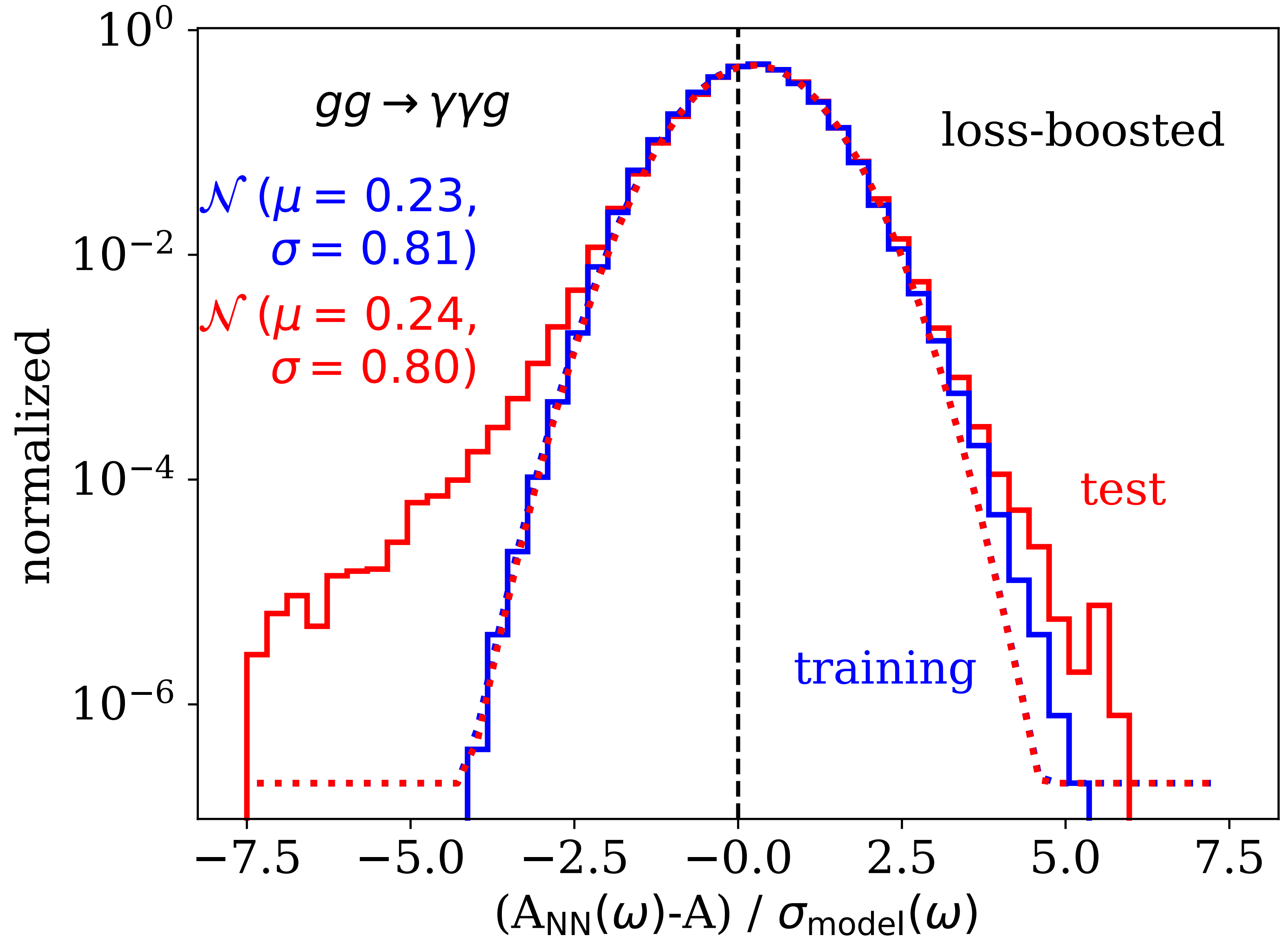
Roughly Gaussian but enhanced tails

Loss boosting

Enforce training on samples with $\Delta A > 2\sigma$
→ include them 5 times in each epoch
→ Repeat 4 times



No change in performance



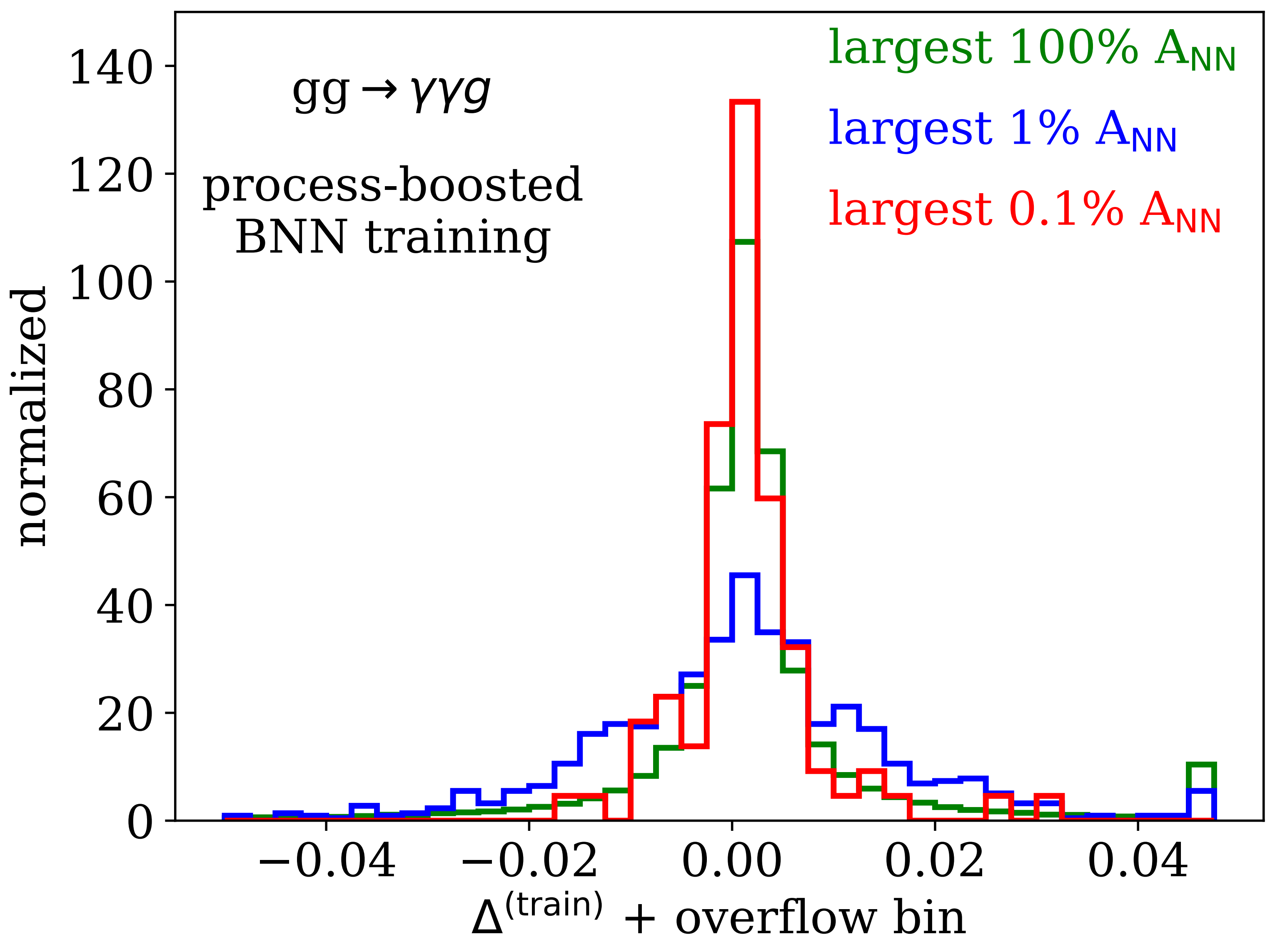
Tails reproduced for training data
Improvement for test data

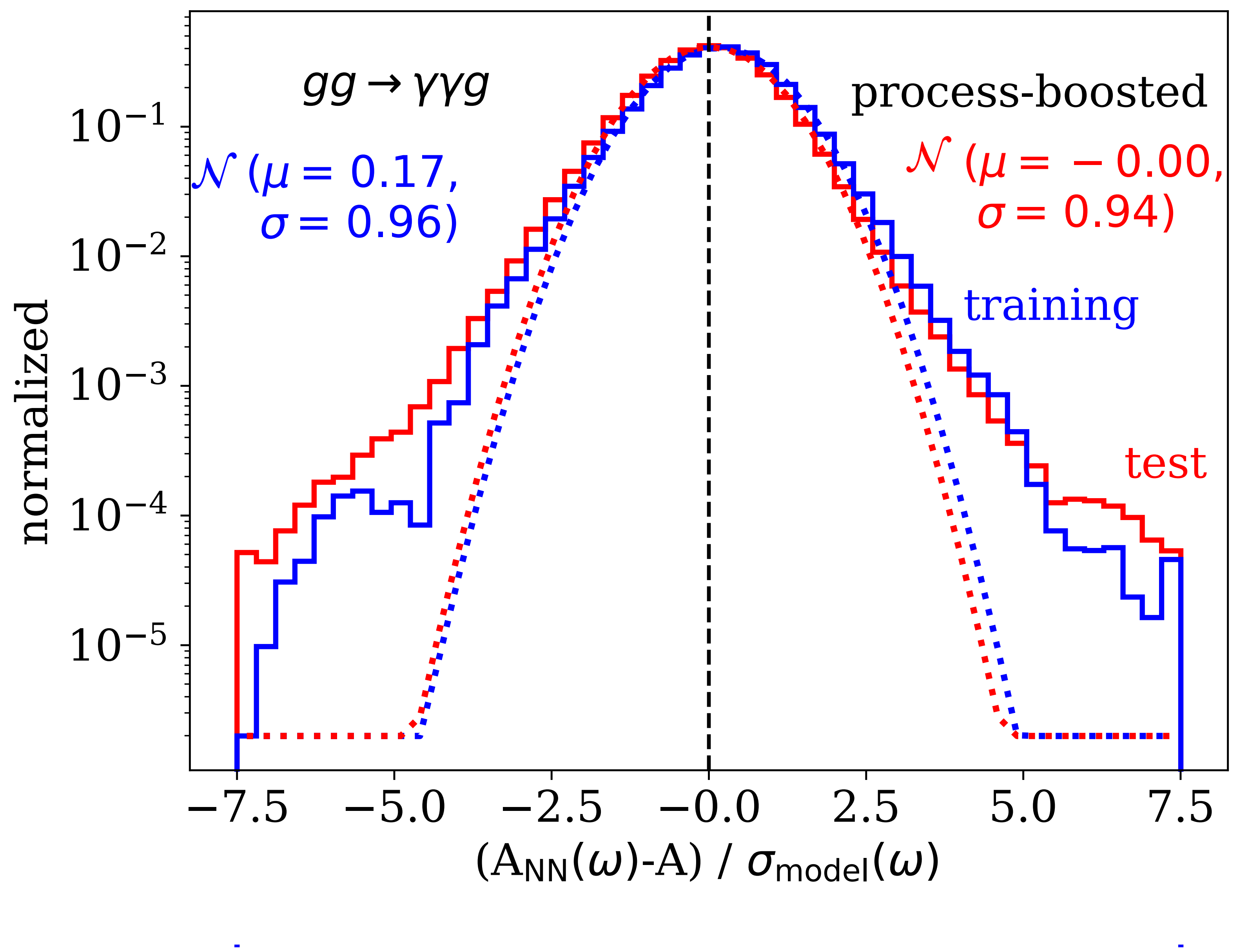
Performance boosting

Enforce training on 200 samples with largest uncertainty σ_{tot}

→ include them +3 times in each epoch

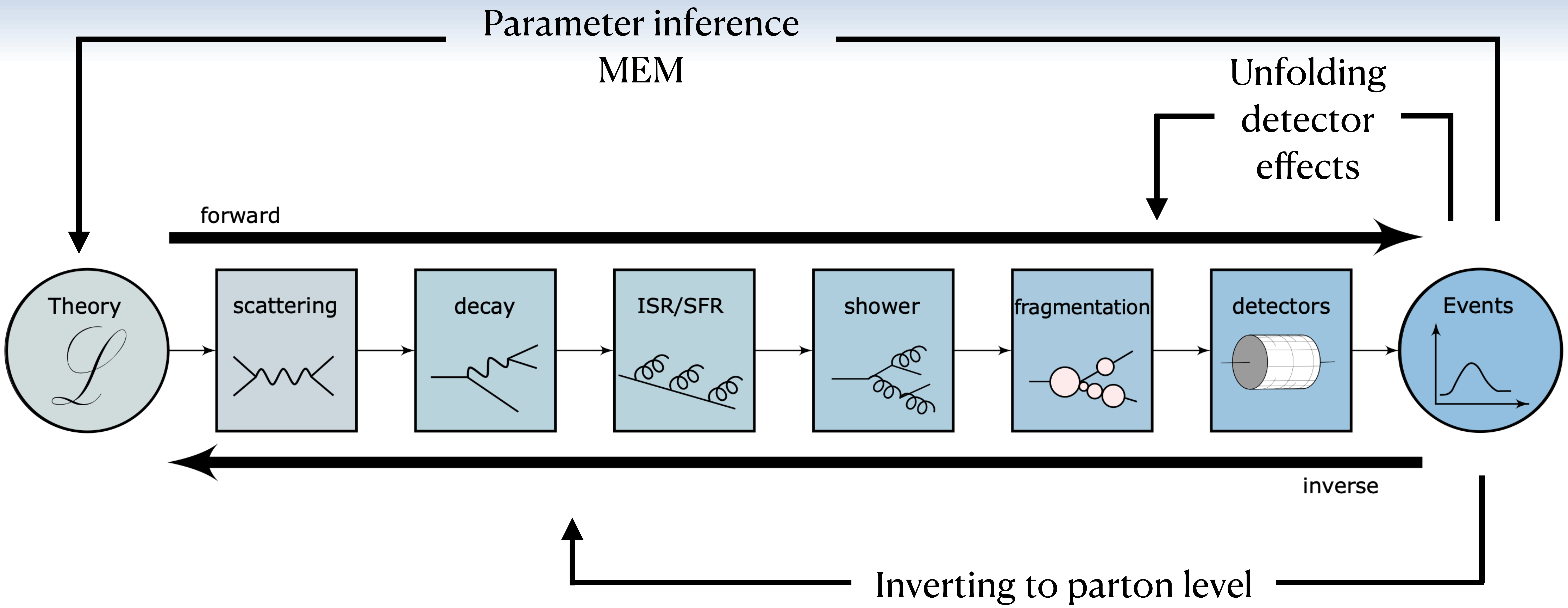
→ Repeat 20 times





Significant improvement in performance

Inverting the simulation chain

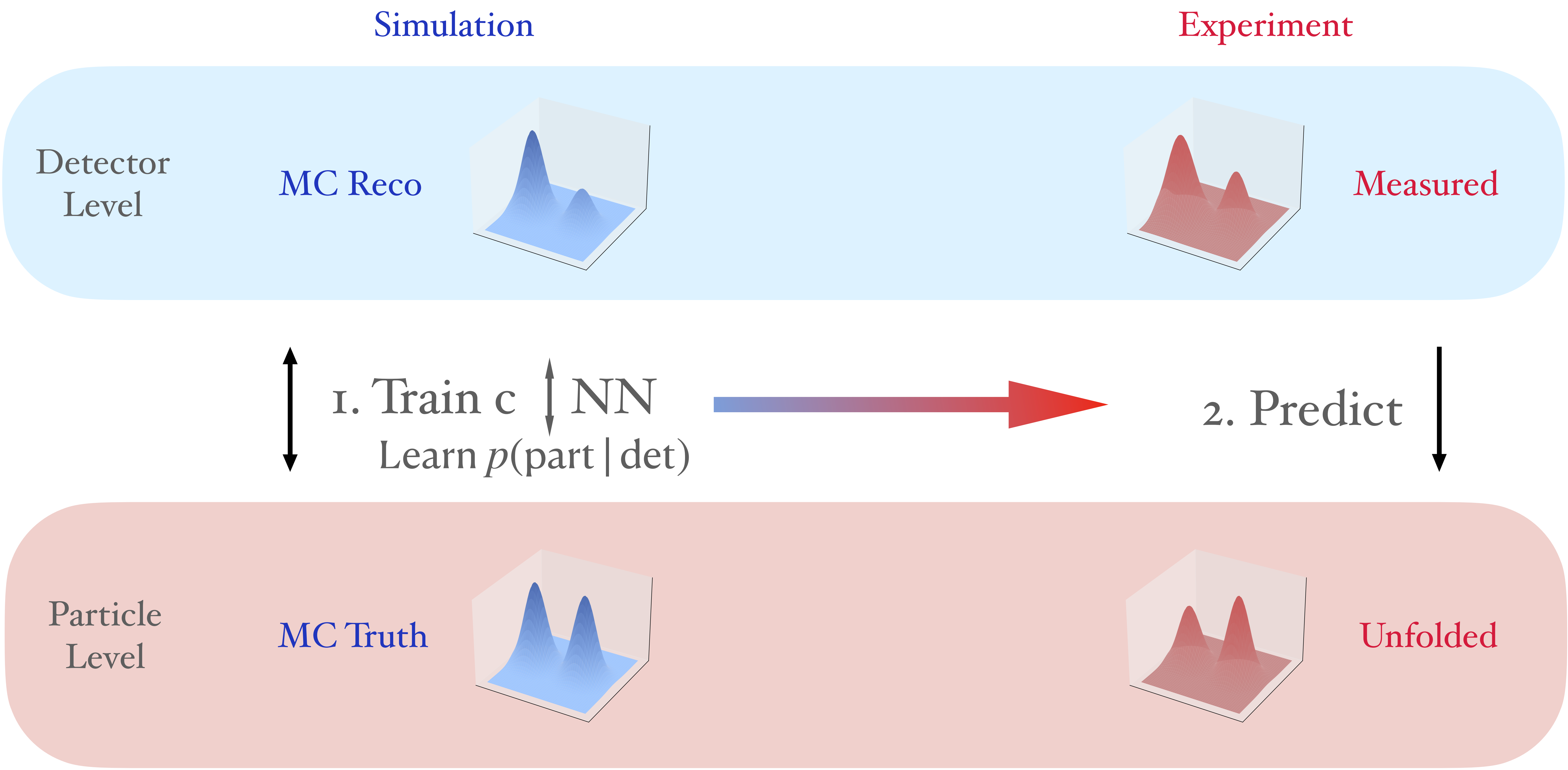


Requirements

- ☐ High - dimensional
- ☐ Bin - independent
- ☐ Statistically well defined

Unfolding

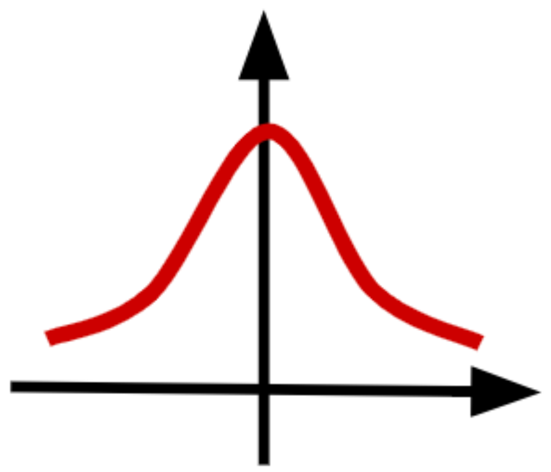
Flow based unfolding methods



cINN unfolding

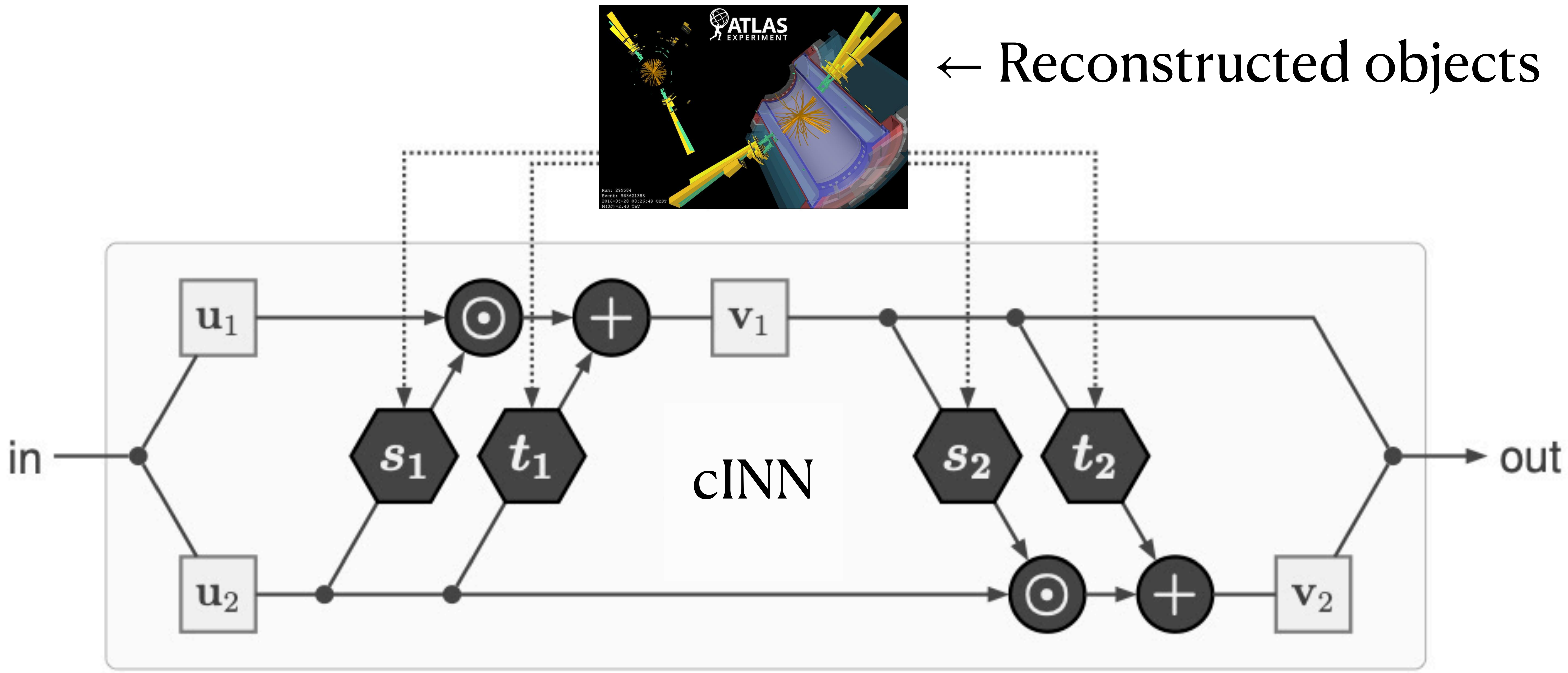
High-dimensional. Bin independent. Robust.

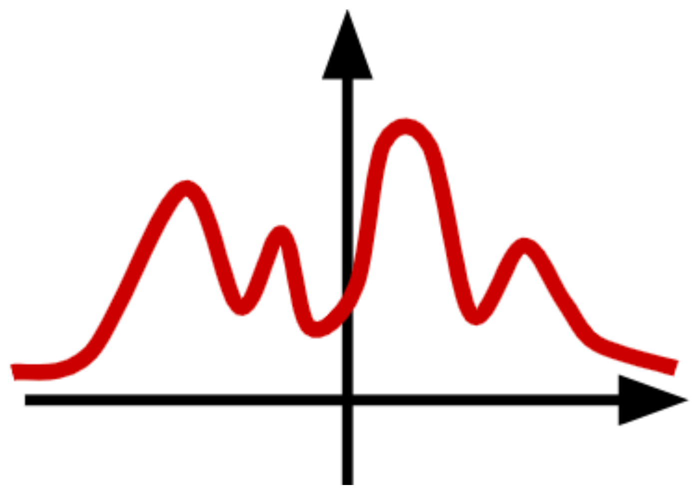
Given a reconstructed event:
What is the probability distribution at particle level?



Normal distribution

$z \sim \mathcal{N}$





Probability distribution

$x \sim P_{\text{parton}}$

Training

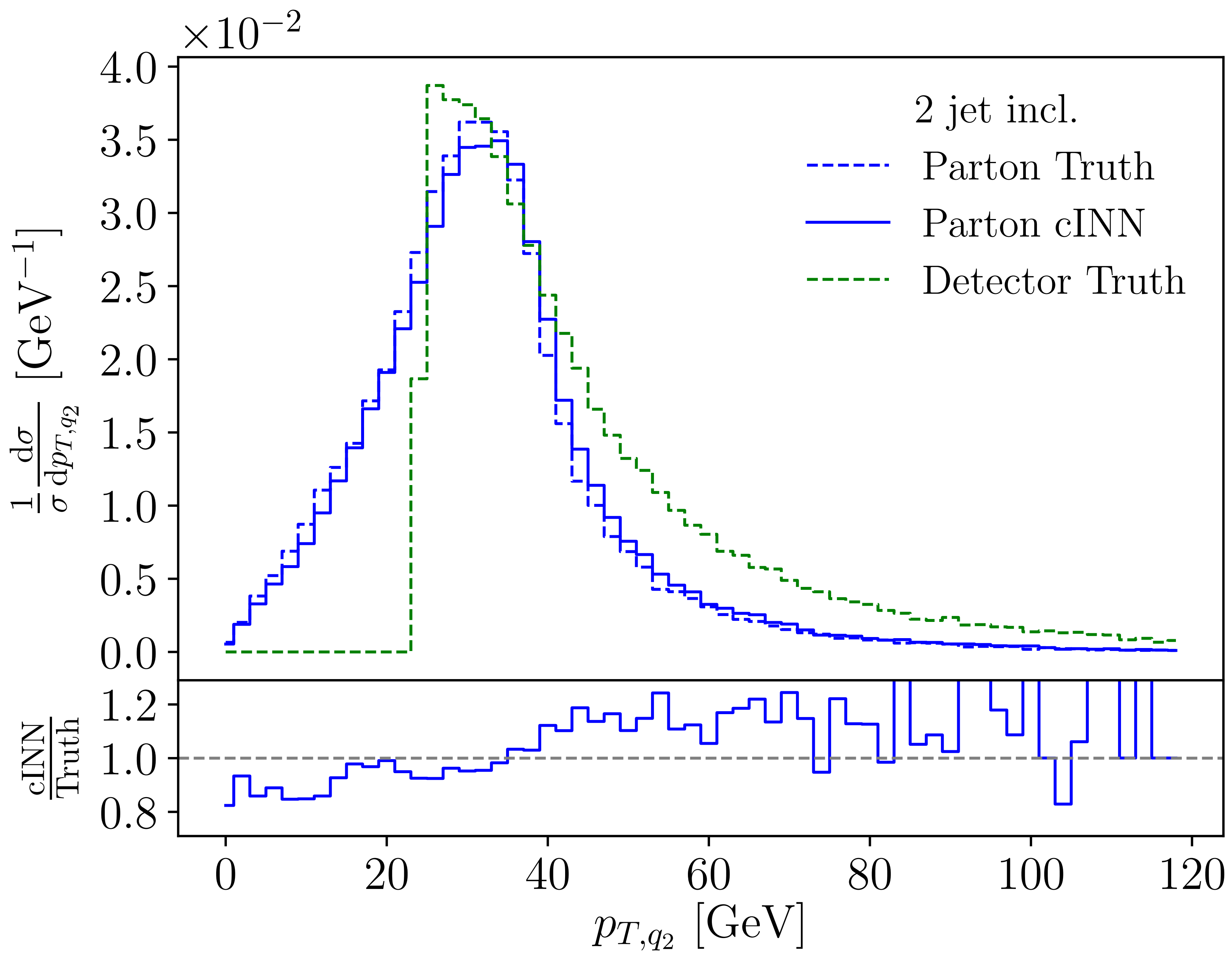
Unfolding

$$\mathscr{L} = \log p(\theta \,|\, x, reco)$$
$$= \log p(z \,|\, \theta, reco) + \log J_{NN} + p(\theta)$$

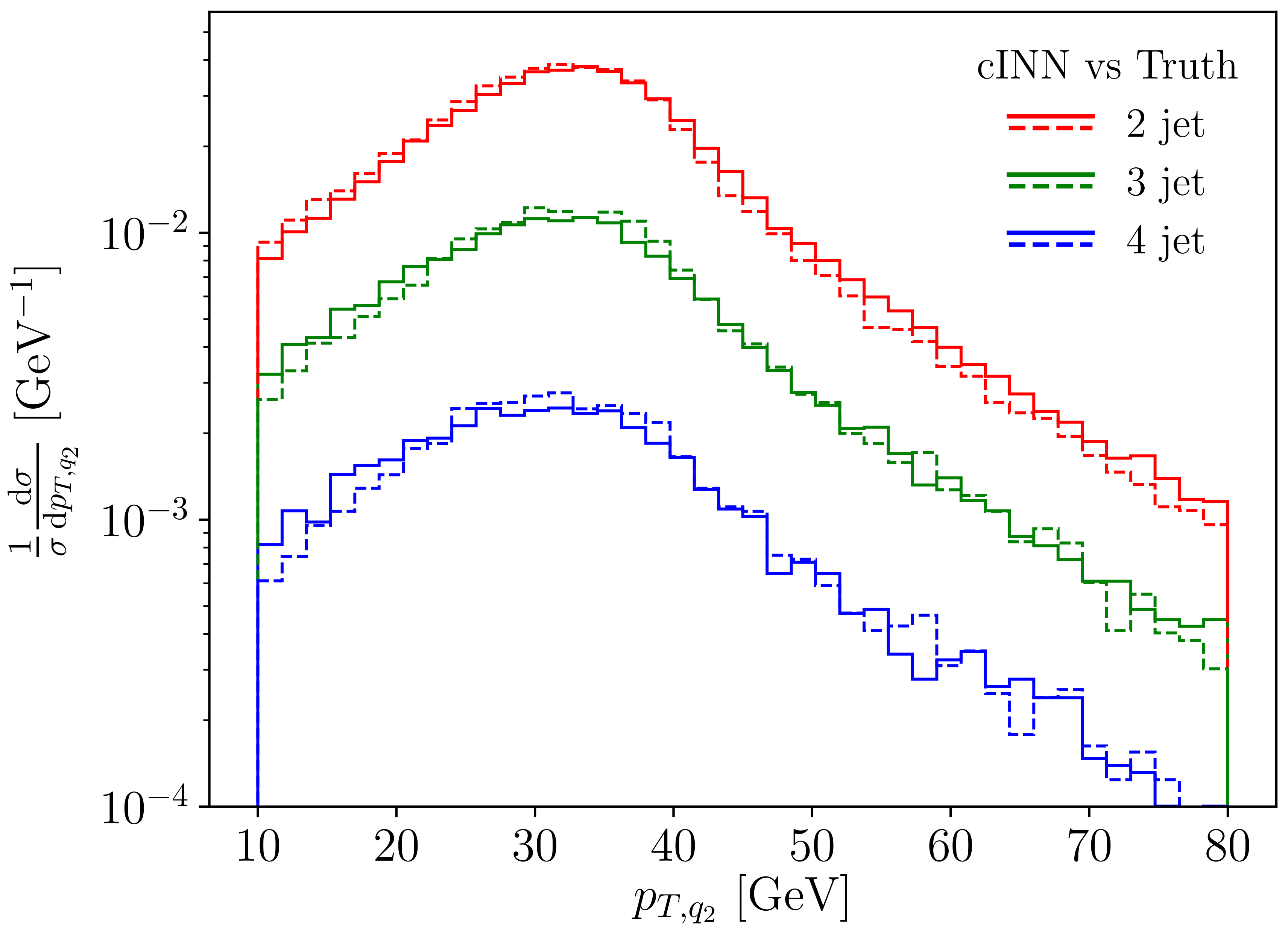
Inverting inclusive distributions

$pp > WZ > q\bar{q}l^+l^- +$ ISR → 2/3/4 jet events

Training on inclusive dataset



Evaluate exclusive 2/3/4 jet events

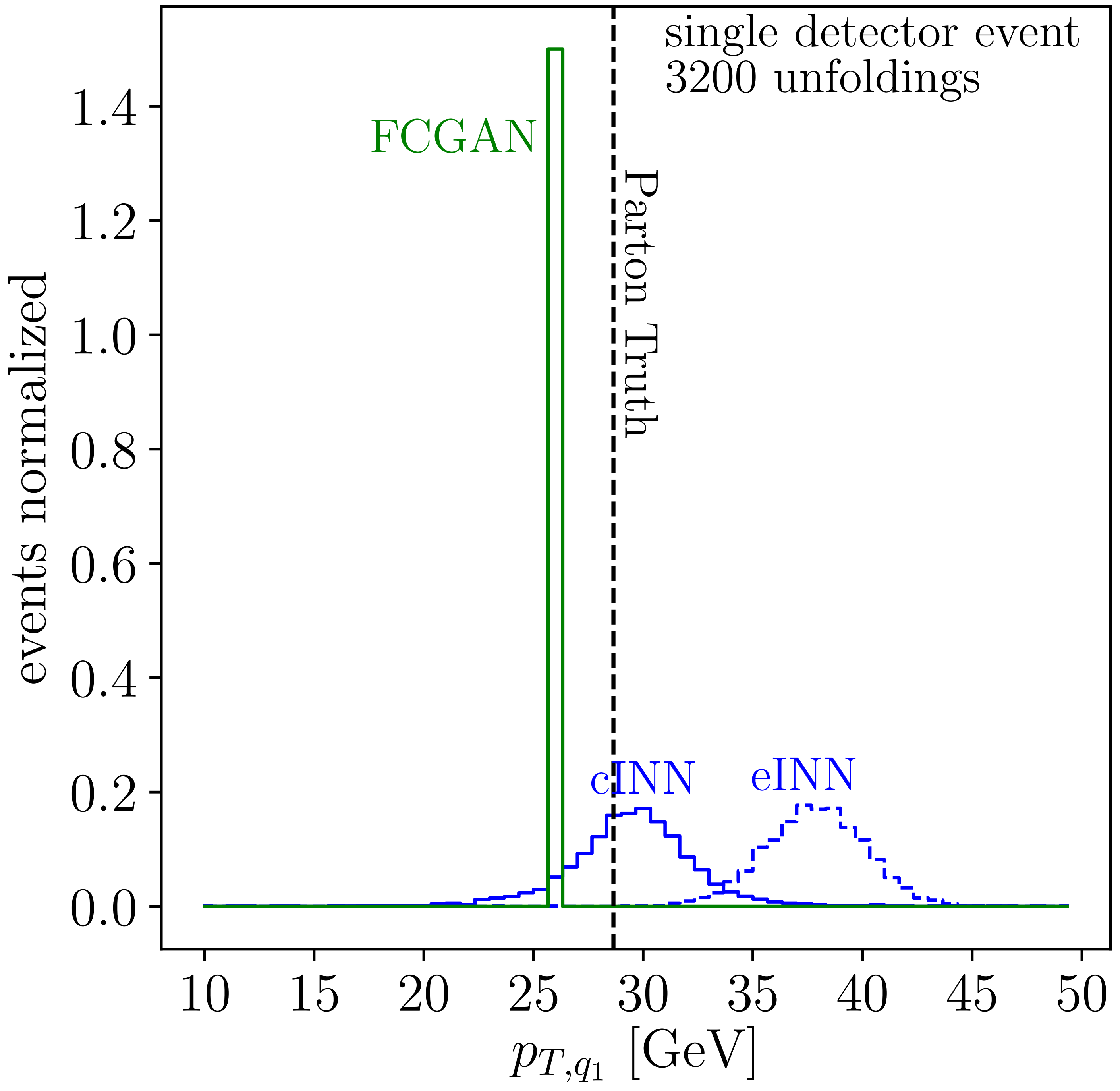


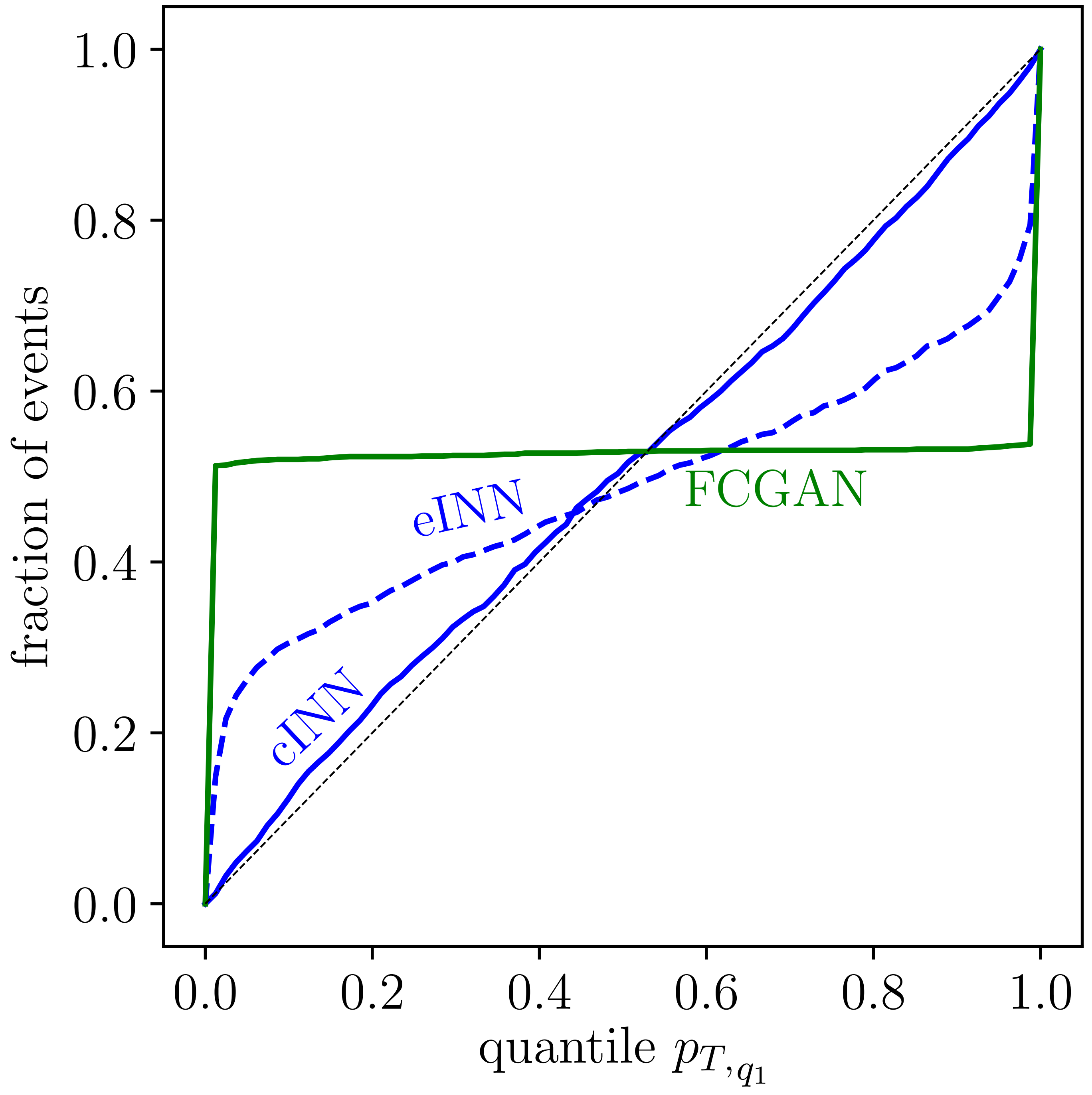
- ☑ High-dimensional
- ☑ Bin-independent
- ☐ Statistically well defined ?

M. Bellagente et al. [2006.06685]

Event-wise unfolding

No deterministic mapping!
Check calibration of probability density for individual event unfolding



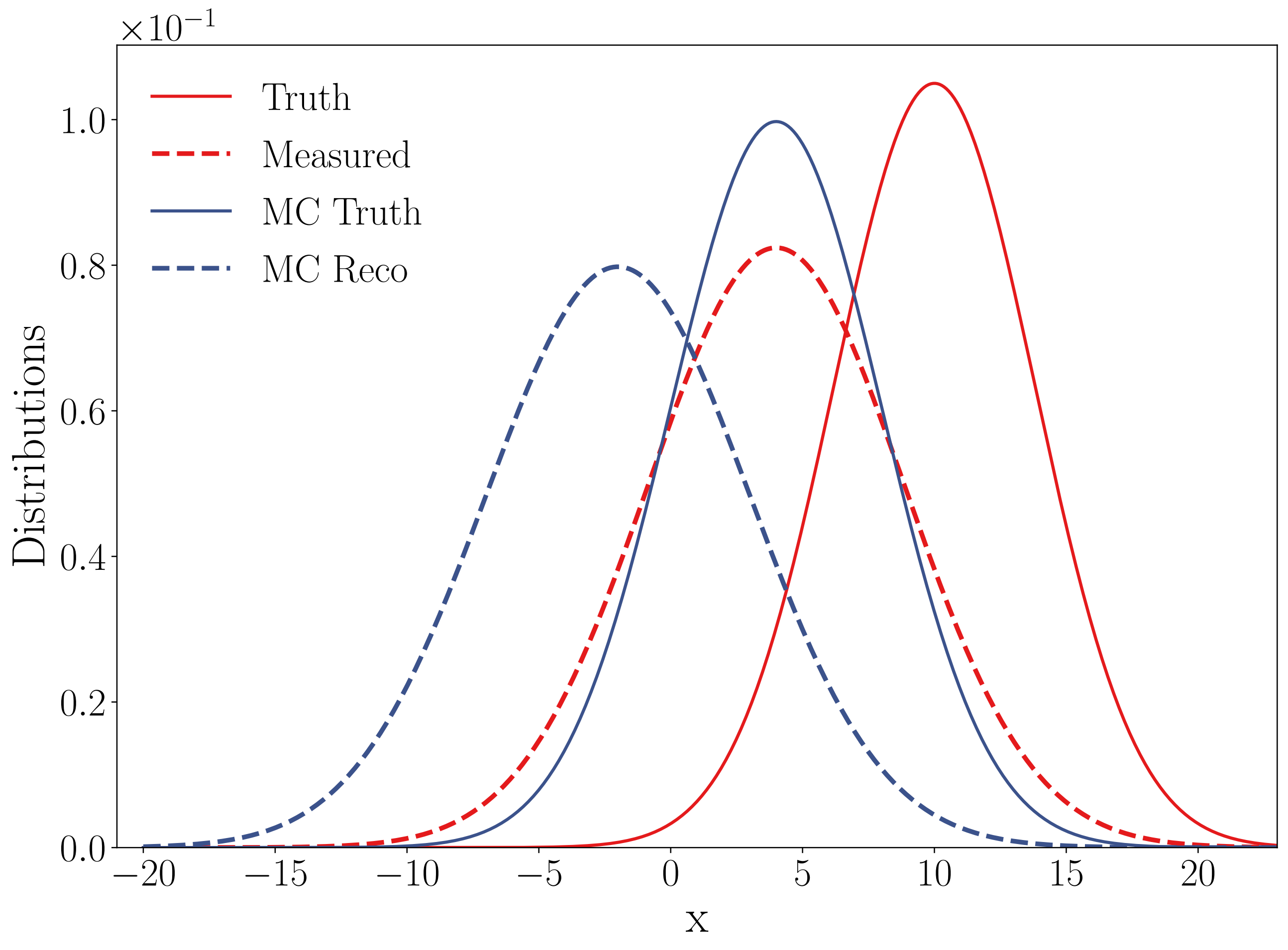


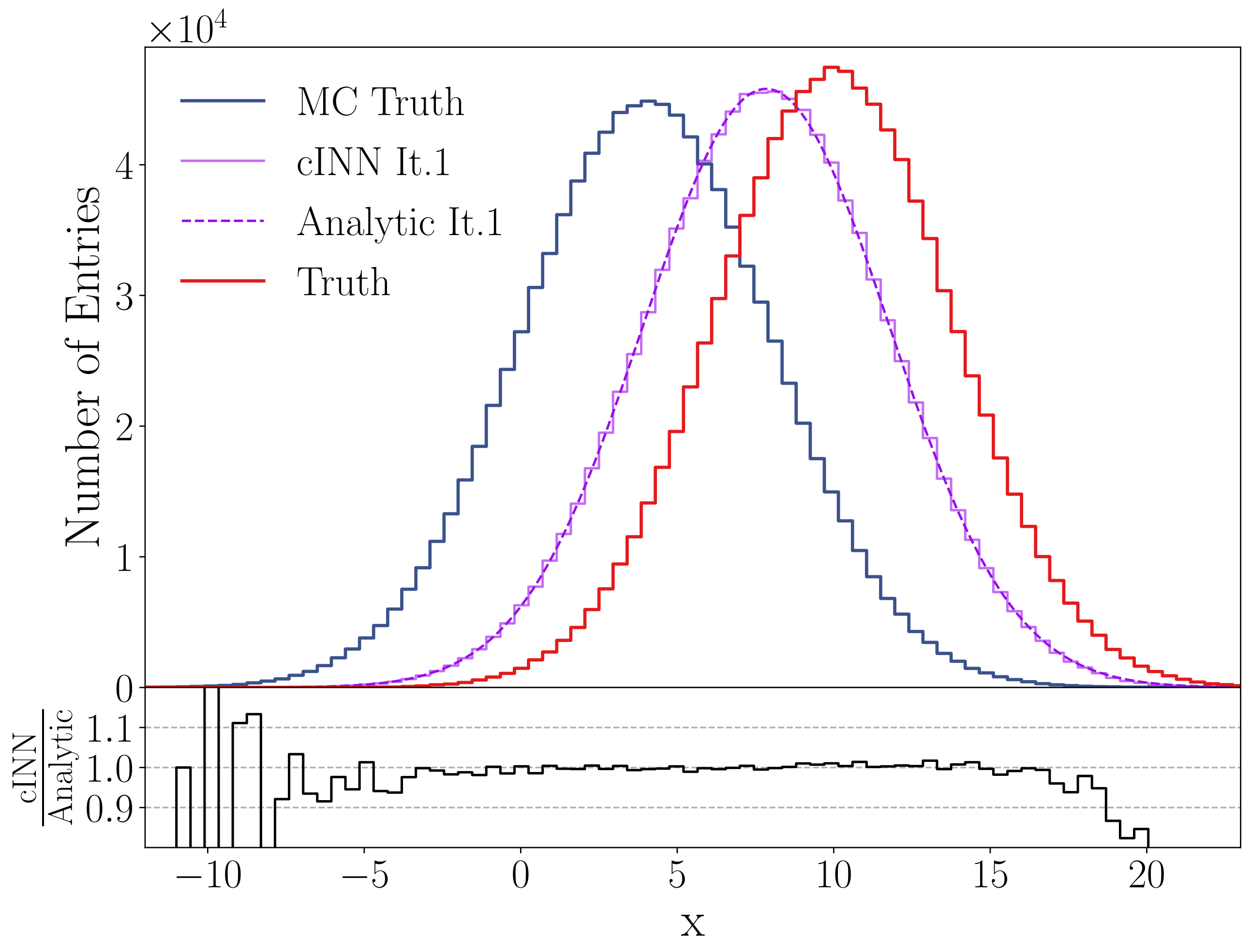
- ☑ High-dimensional
- ☑ Bin-independent
- ☑ Statistically well defined

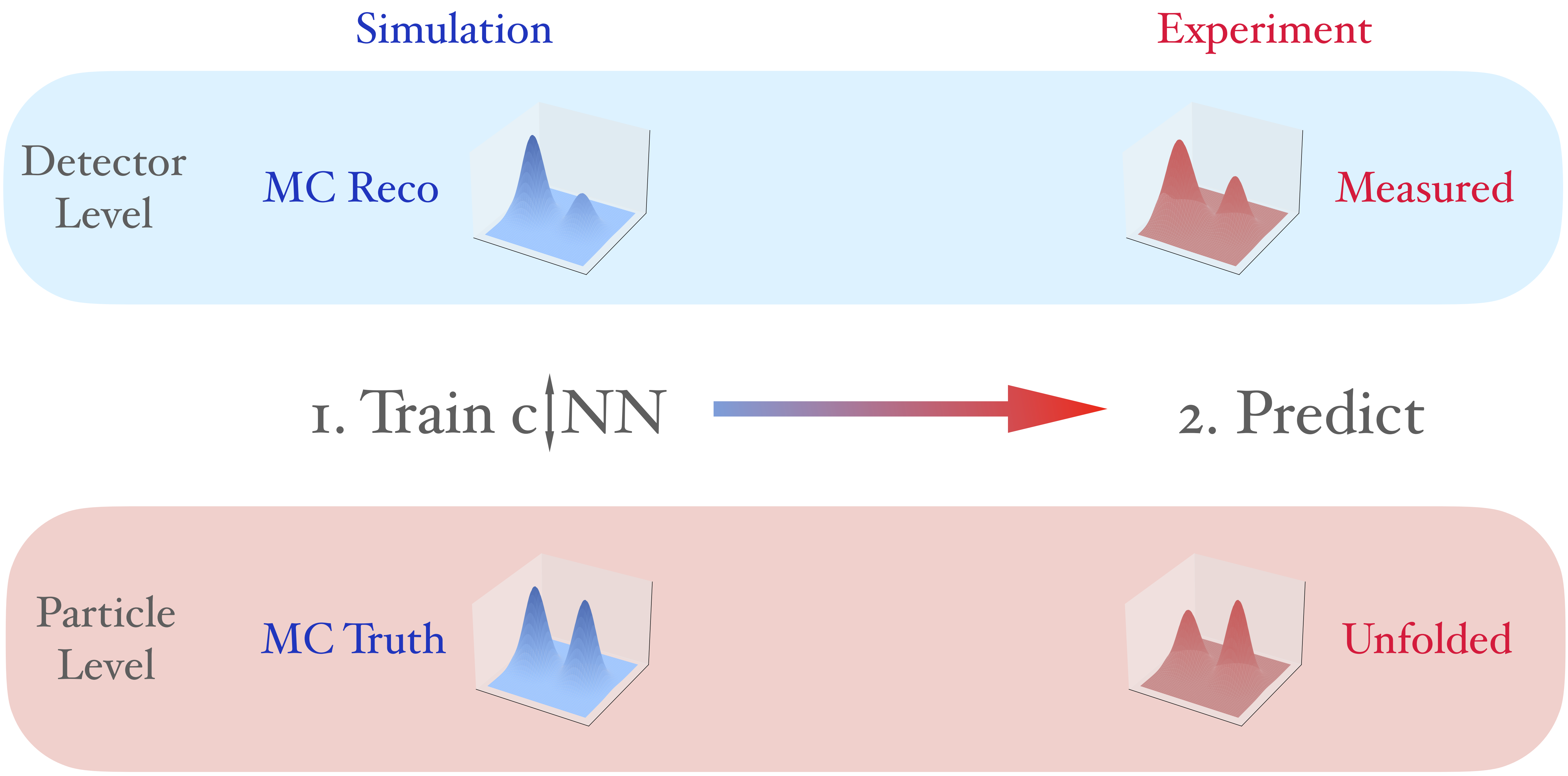
M. Bellagente et al. [2006.06685]

Conditional iterative unfolding

Current work in progress



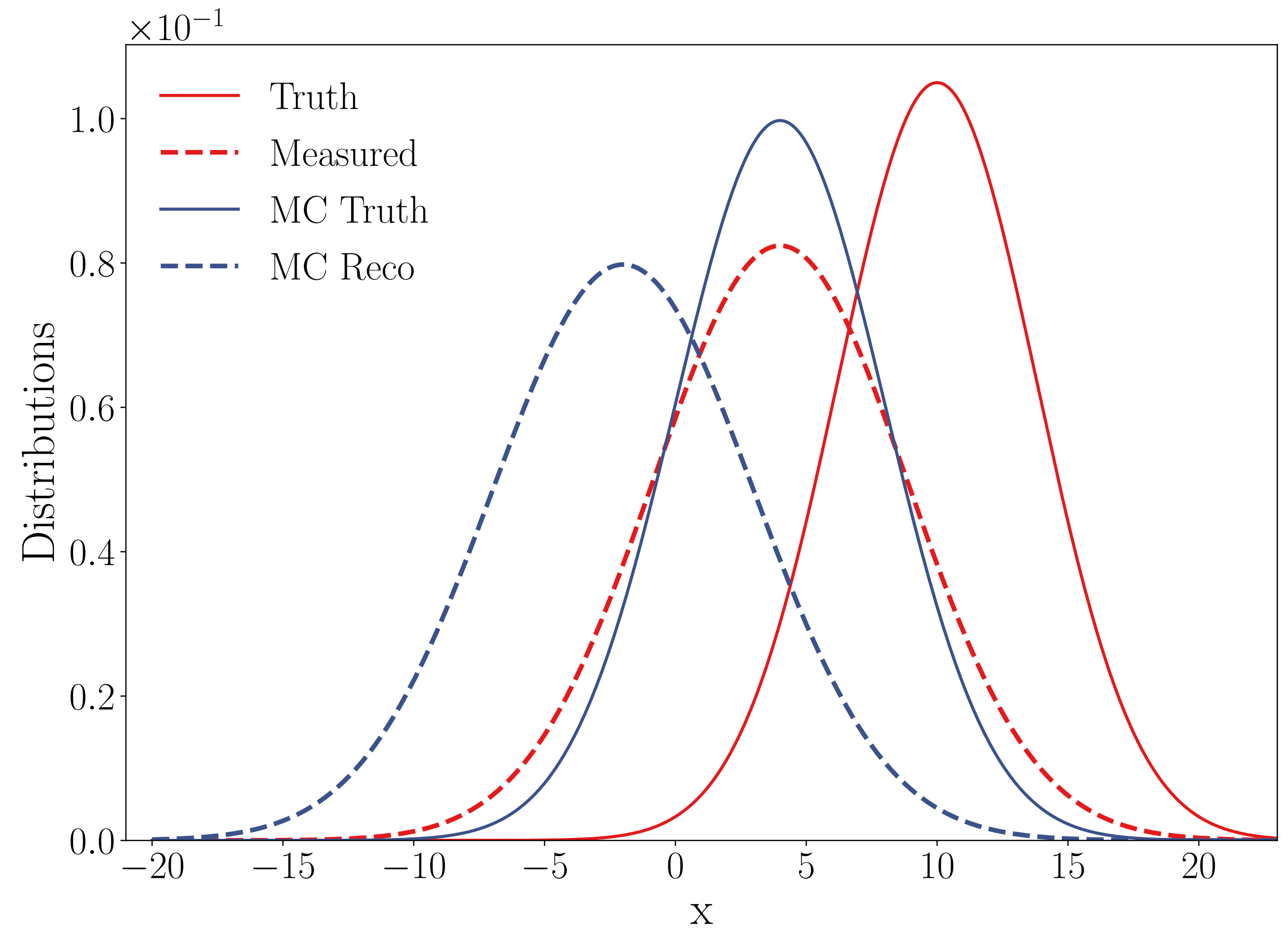


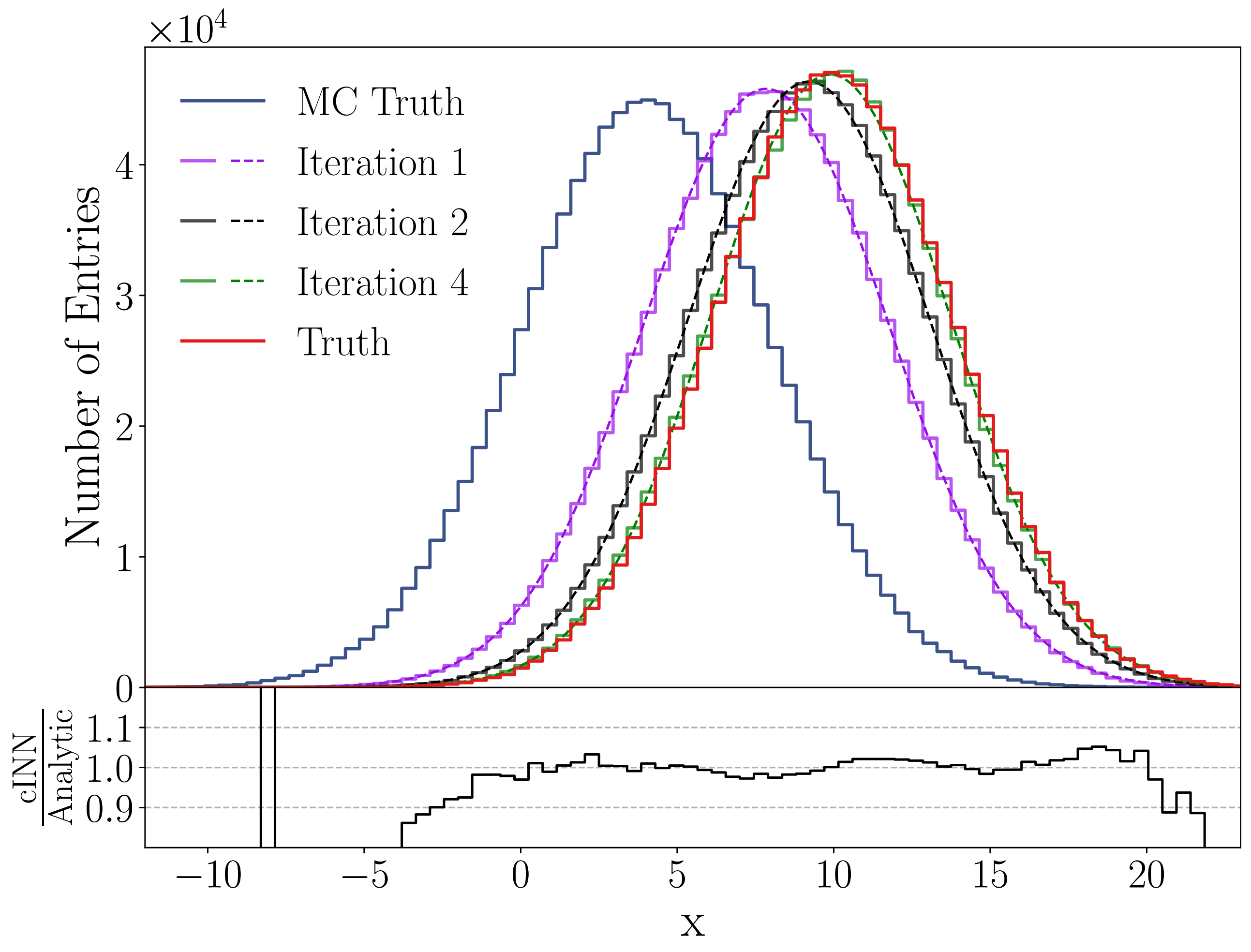


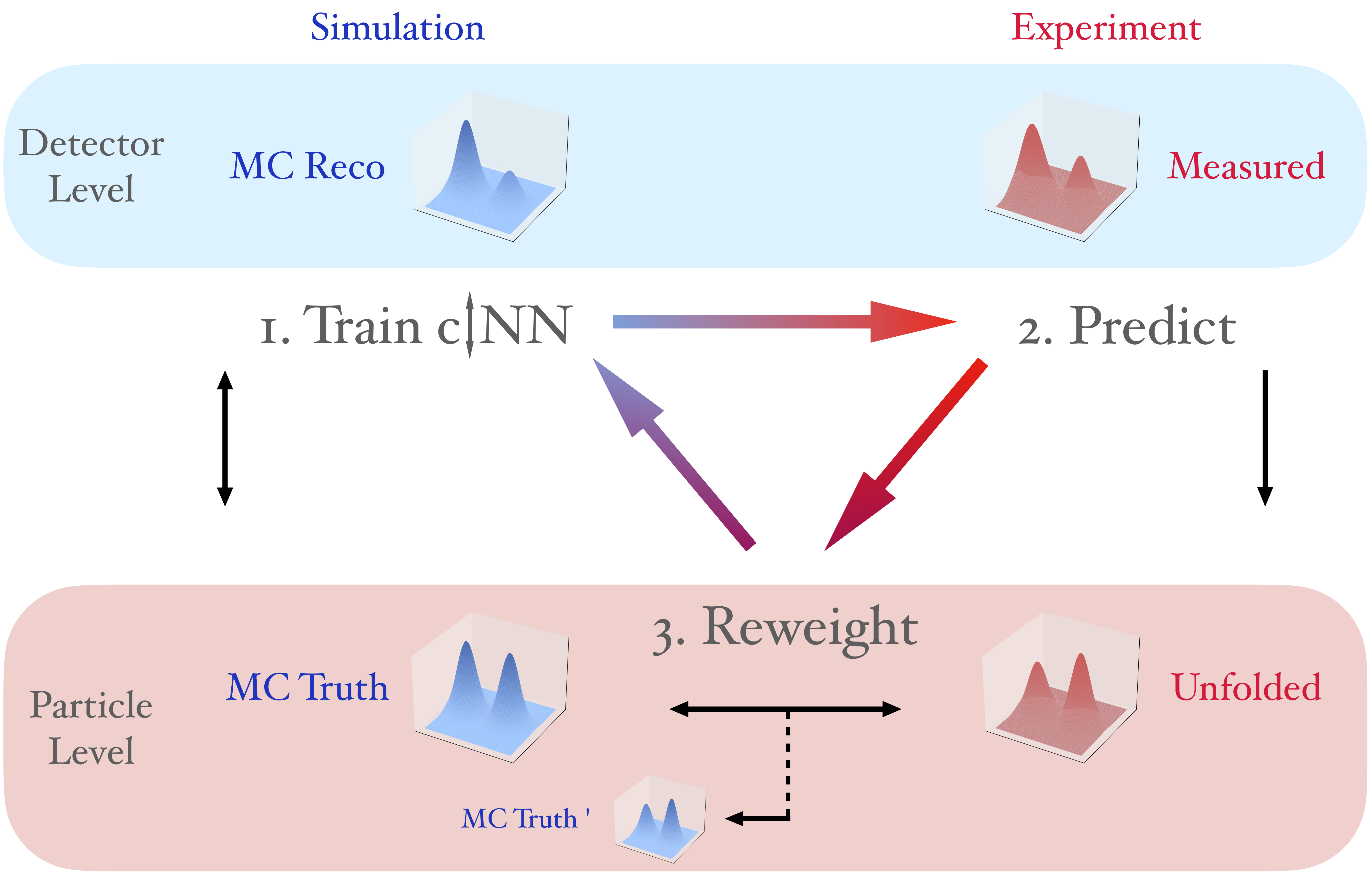
$$p(\text{part}\,|\,\text{det}) = \frac{p(\text{det}\,|\,\text{part}) \cdot p(\text{part})}{p(\text{det})}$$

Conditional iterative unfolding

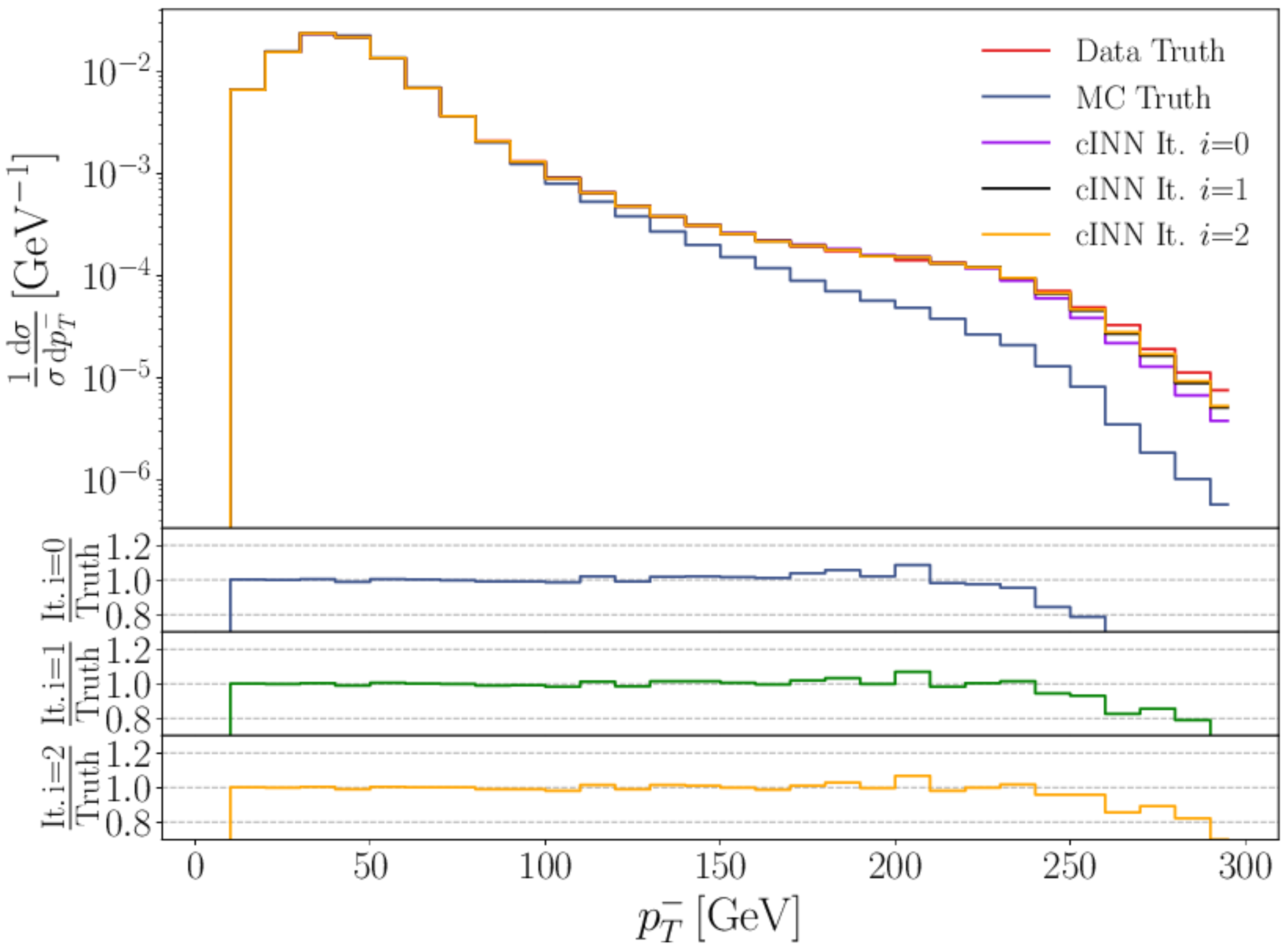
Current work in progress







Example: $Z\gamma\gamma$ + EFT



Summary

Graph networks

- Particularly suitable for unordered sets of objects
- Various applications from top tagging to track reconstruction
- Including physics based layers makes networks more efficient!

Uncertainties

- Same validation as for other techniques (closure tests, toys, etc.)
- Additional tools to evaluate stability and uncertainties (Bayesian networks, ensembles,..)

Unfolding

- ML enables new analysis methods for high-dim. data
  - Unfolding with generative and reweighting methods
  - MEM and many more