From Model-Agnostic Searches to Unfolding: Deep Learning in Particle Physics

Monalisa Patra

IHub-DATA, IIIT Hyderabad

10-08-2024

Frontiers in Particle Physics 2024
IISC, Bangalore

Outline

- Model Agnostic New Physics search
- Detector Simulation Framework
- Unfolding

New Physics?

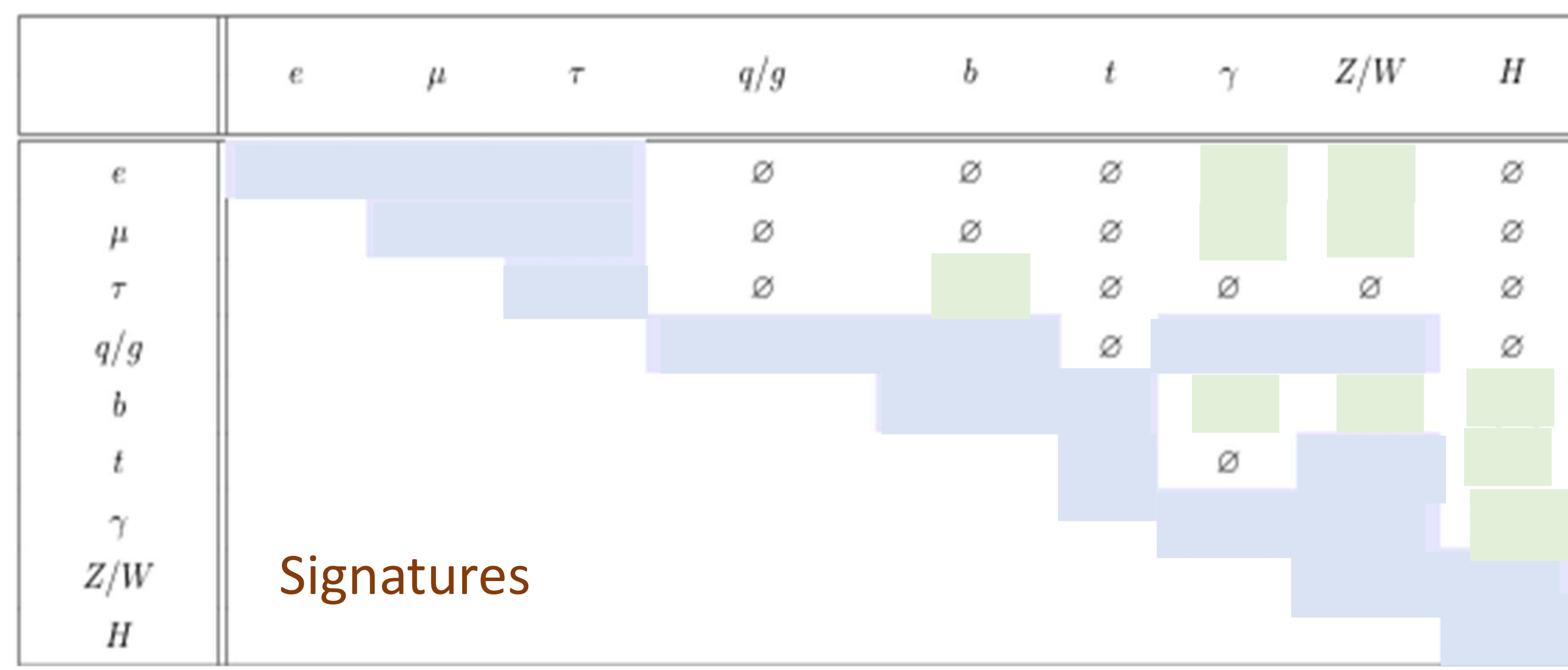
- Copious amounts of data are generated at the LHC, making the discovery potential for new physics enormous .

- Despite thousands of searches for new physics at the LHC, all we have are limits and null results.

- What if new physics is hiding in the data but we haven’t looked in the right places yet?

$A \to BC$	$B = \mathrm{SM}$: e	μ	τ	q/g	b	t	γ	Z/W	H
$C = \mathrm{SM}$: e	Z'	$\not{R}$	$\not{R}$	LQ	LQ	LQ	L^*	L^*	L^*
μ		Z'	$\not{R}$	LQ	LQ	LQ	L^*	L^*	L^*
τ			Z'	LQ	LQ	LQ	L^*	L^*	L^*
q/g				Z'	W'	T'	Q^*	Q^*	Q'
b					Z'	W'	Q^*	Q^*	B'
t						Z'	Q^*	T'	T'
γ							H	H	Z_{KK}
Z/W								H	$H^\pm/A$
H									H

Models

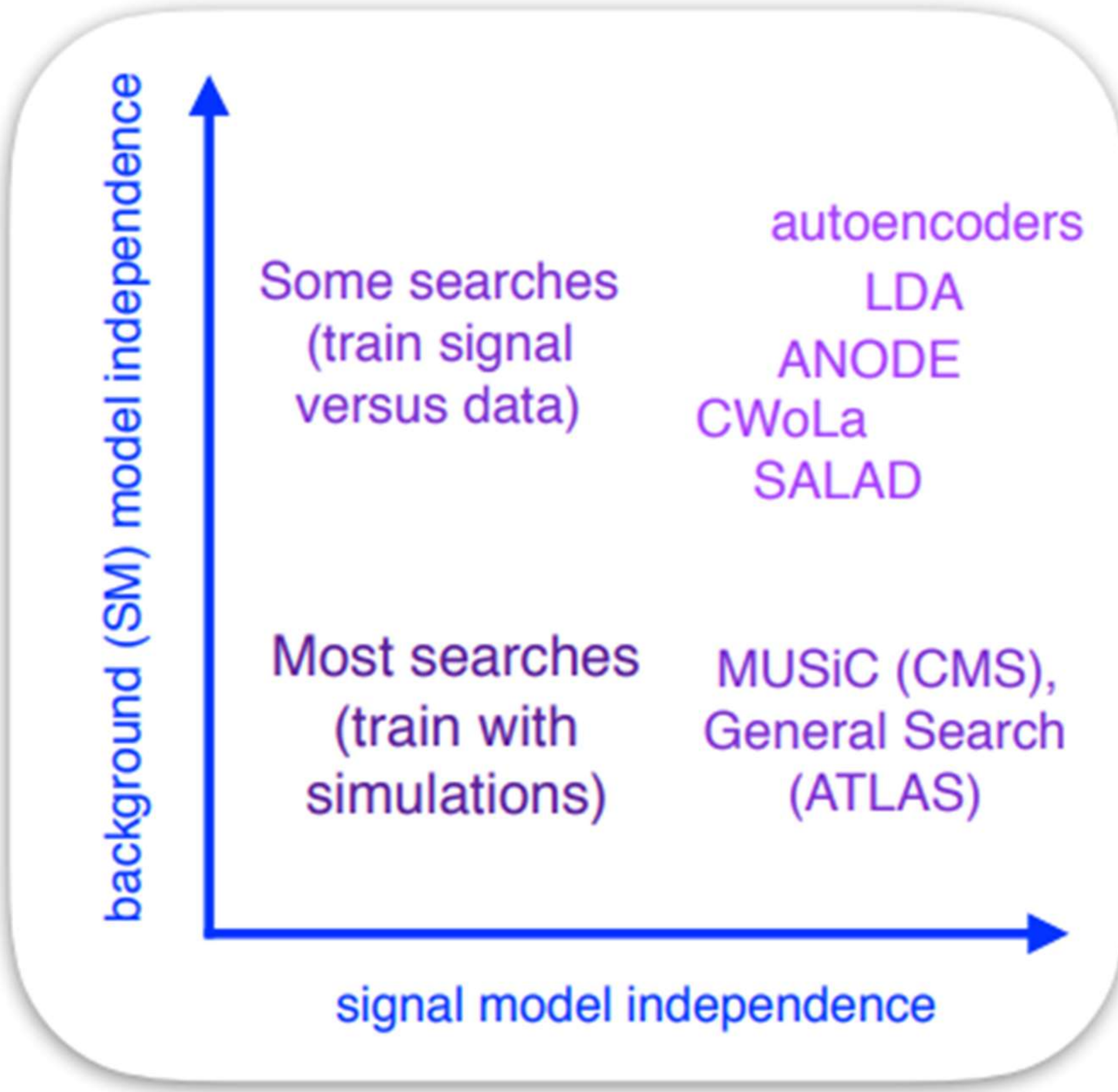


J. Kim, K. Kong, B. Nachman, D. Whiteson, 1907.06659

Model-Agnostic Searches

- Can we search for new physics more model-independently and fill all the gaps in our coverage?

Nachman and Shih, 2001.04990

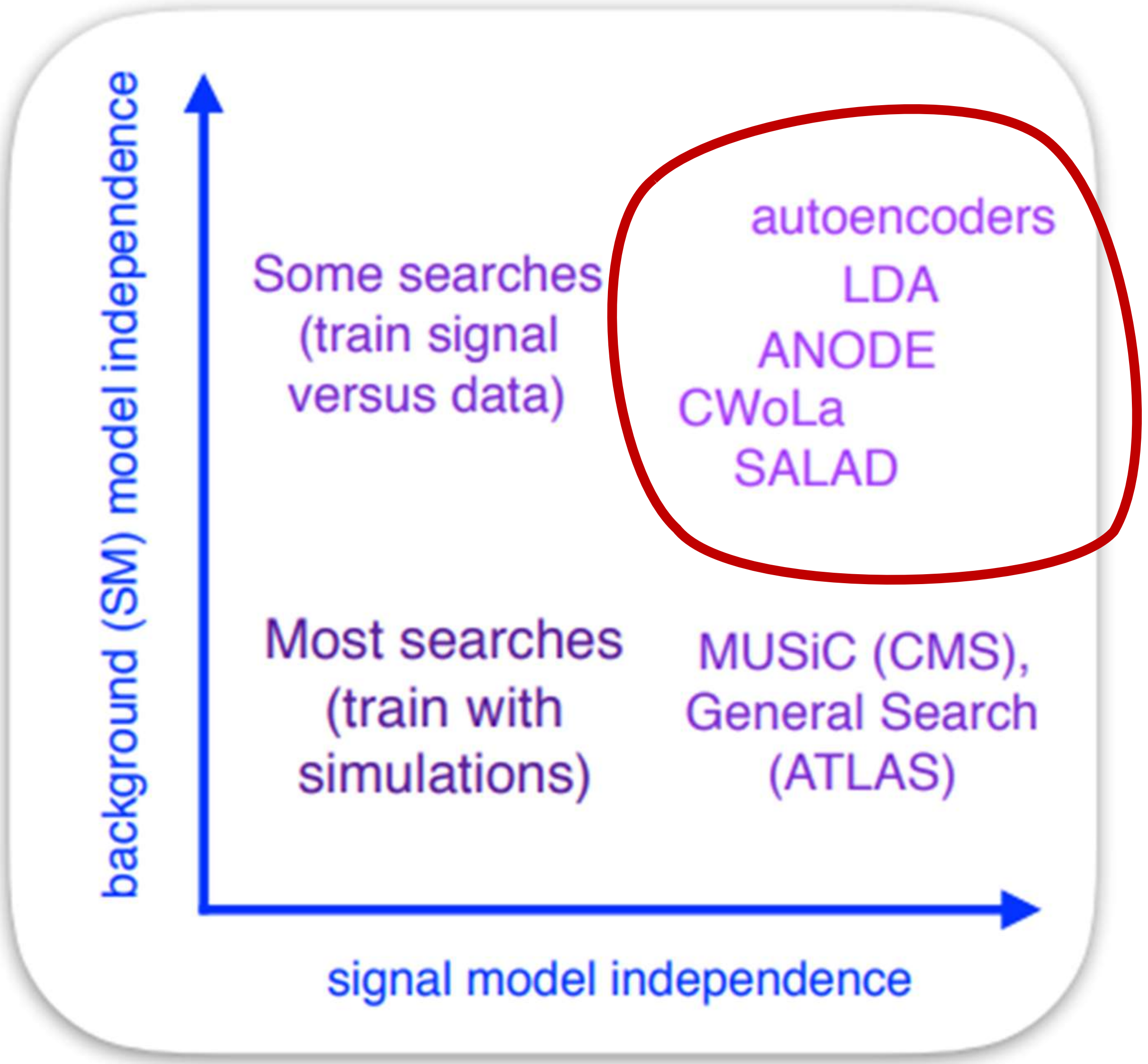


(a) Signal sensitivity

Model-Agnostic Searches

- Can we search for new physics more model-independently and fill all the gaps in our coverage?

Nachman and Shih, 2001.04990

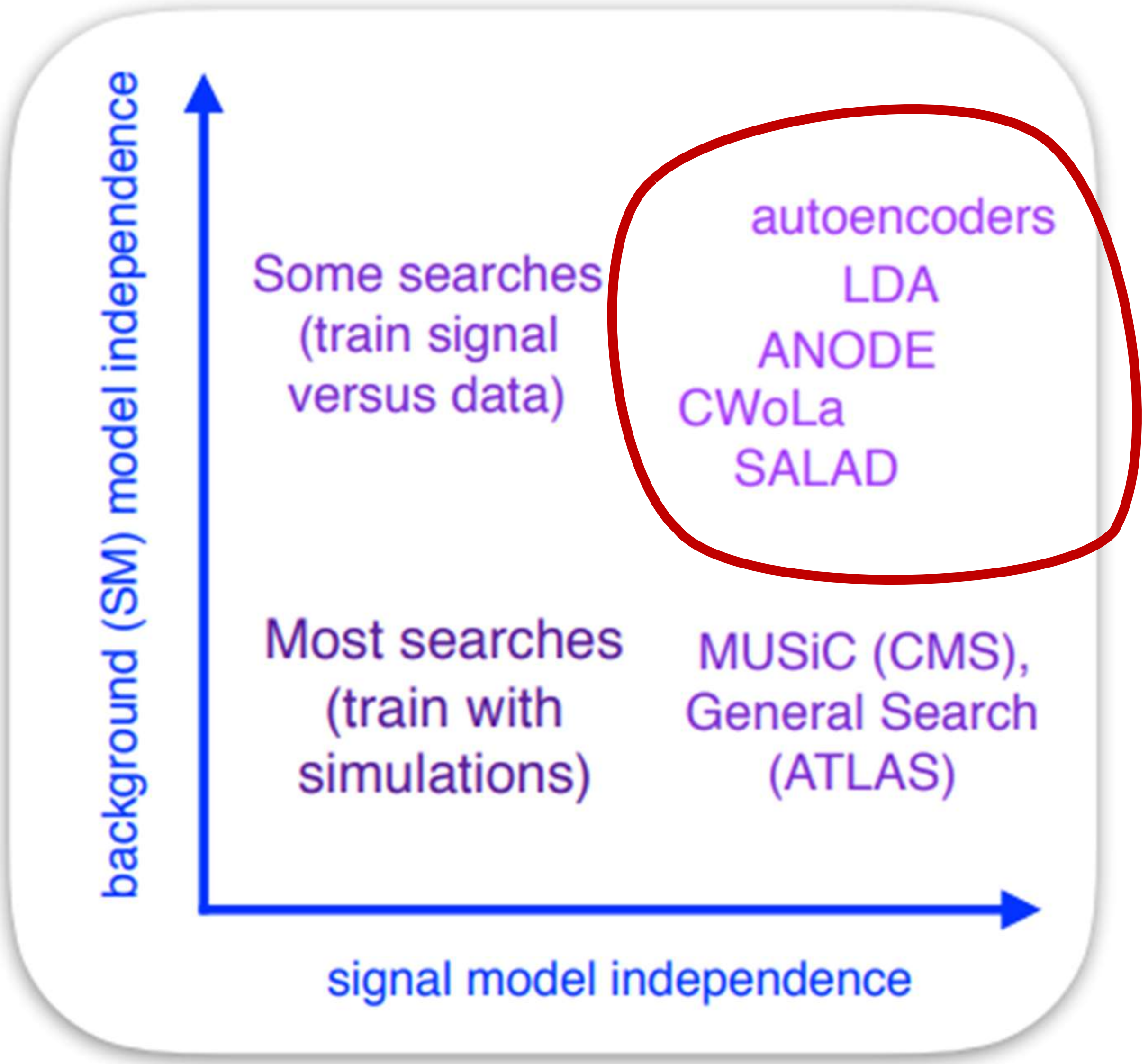


(a) Signal sensitivity

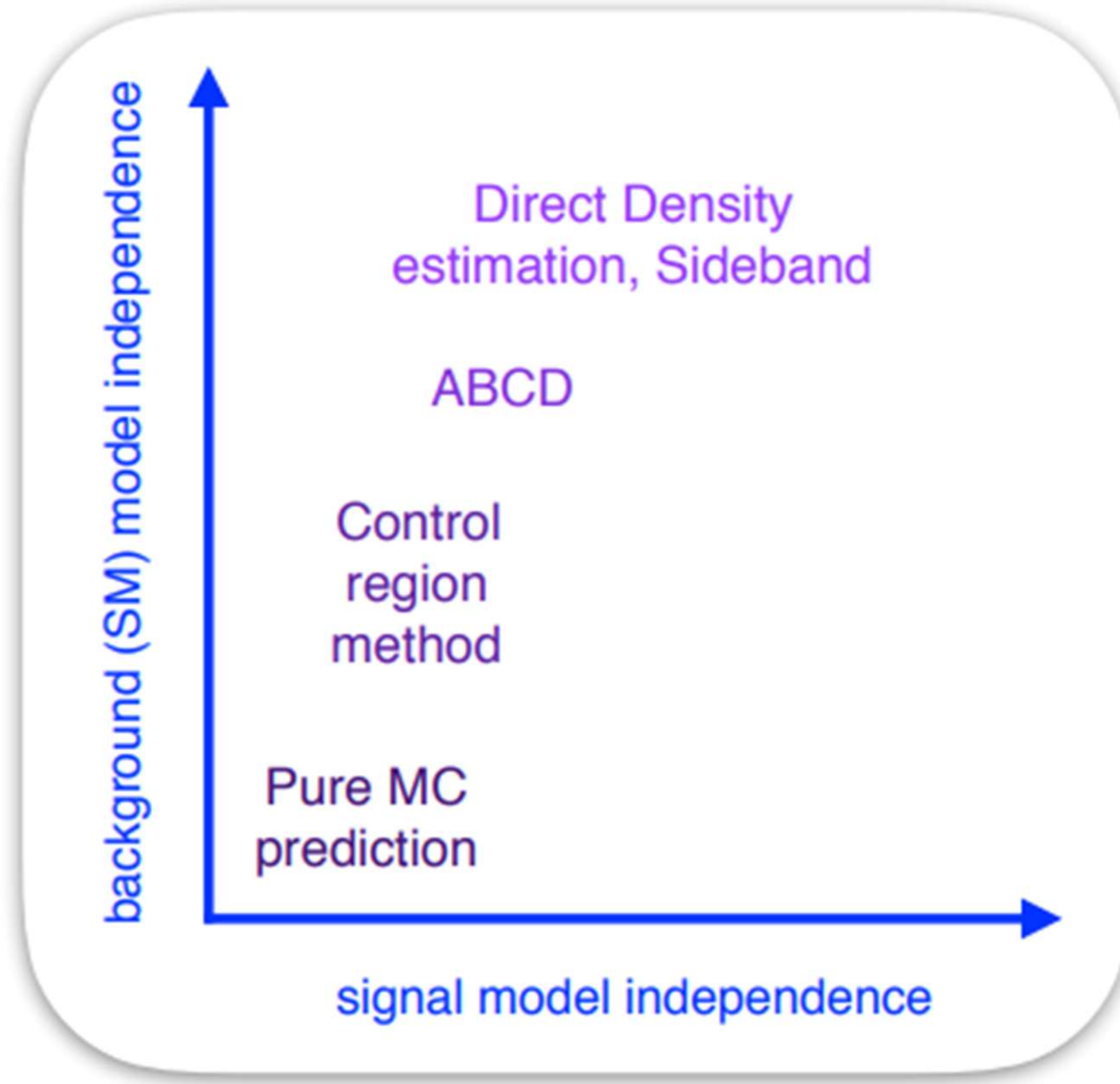
Model-Agnostic Searches

- Can we search for new physics more model-independently and fill all the gaps in our coverage?

Nachman and Shih, 2001.04990



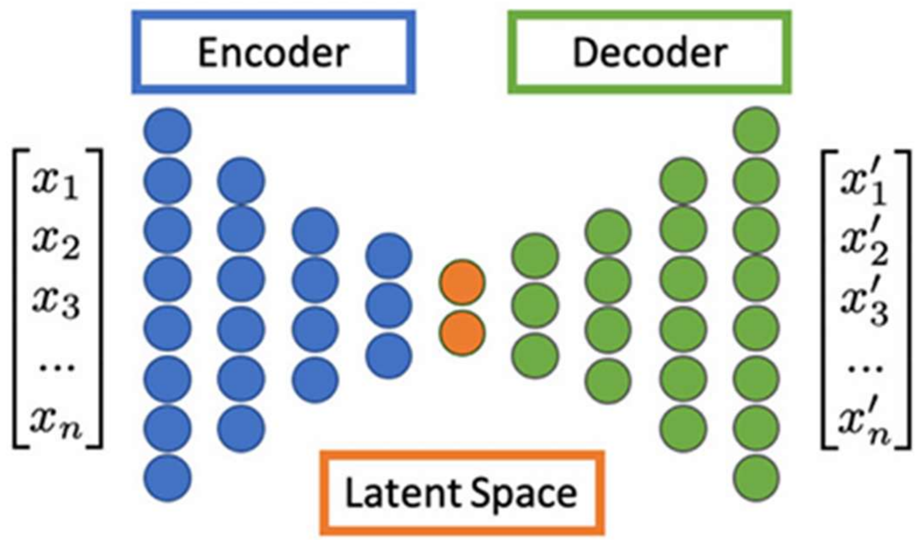
(a) Signal sensitivity

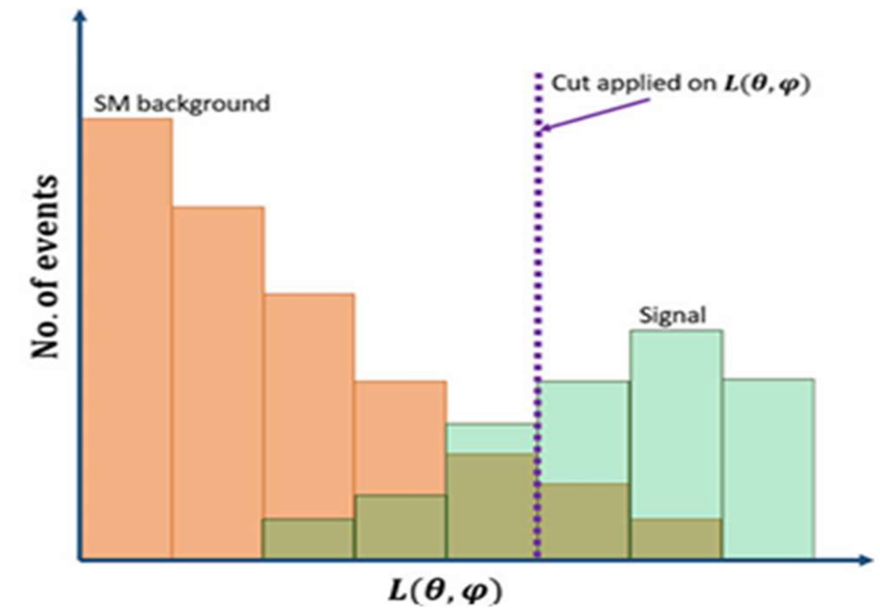


(b) Background specificity

Autoencoders (AEs) as Anomaly Detectors

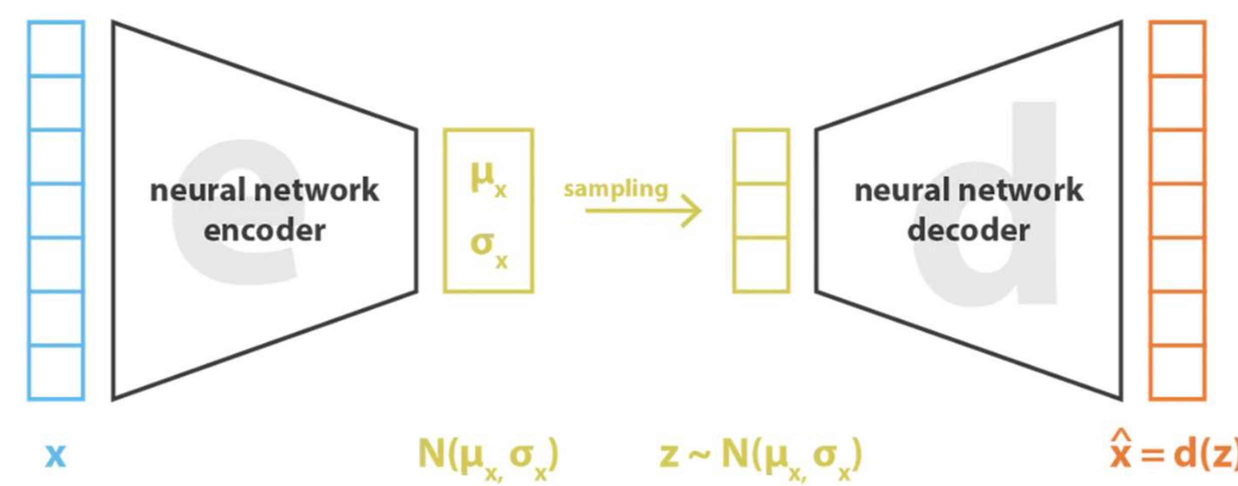
- Autoencoders work by learning compression to a latent space which preserves the original information.





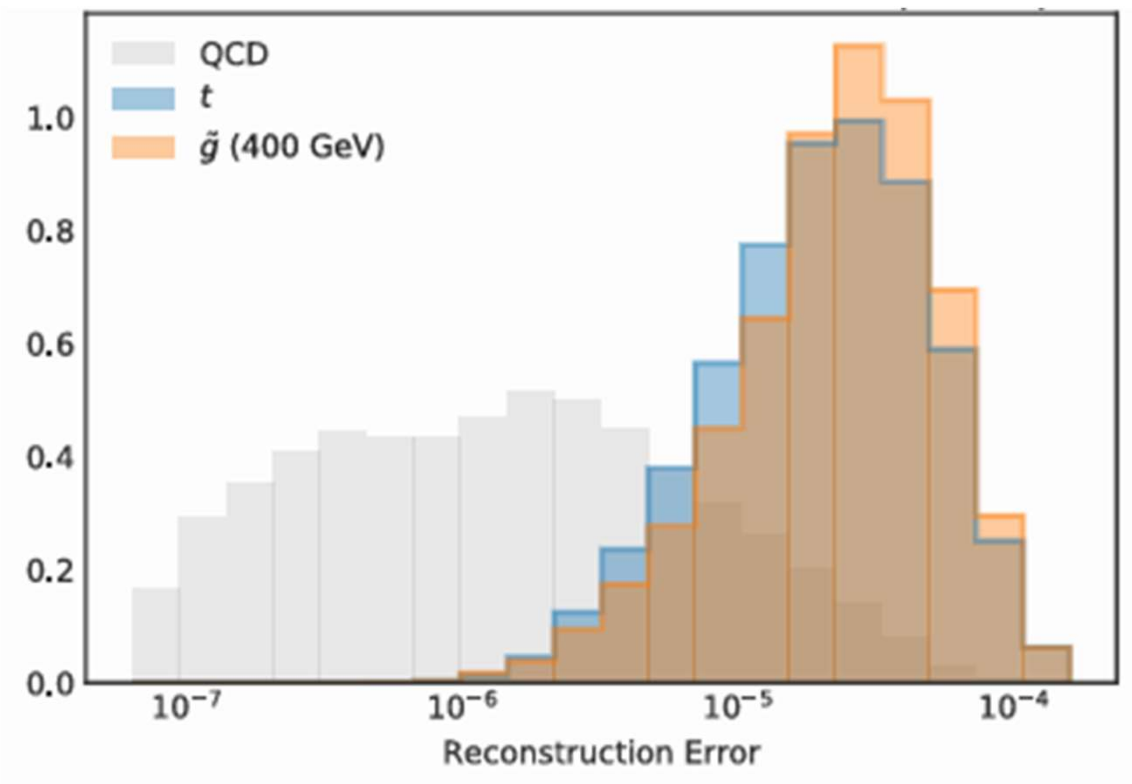
- The reconstruction fidelity gives an anomaly score.

- Variational AE has encoder, decoder architecture like AE. Encoder of VAE attempts to generate parameters of the parametric probability distribution in latent space. Typically, distribution is chosen to be gaussian.

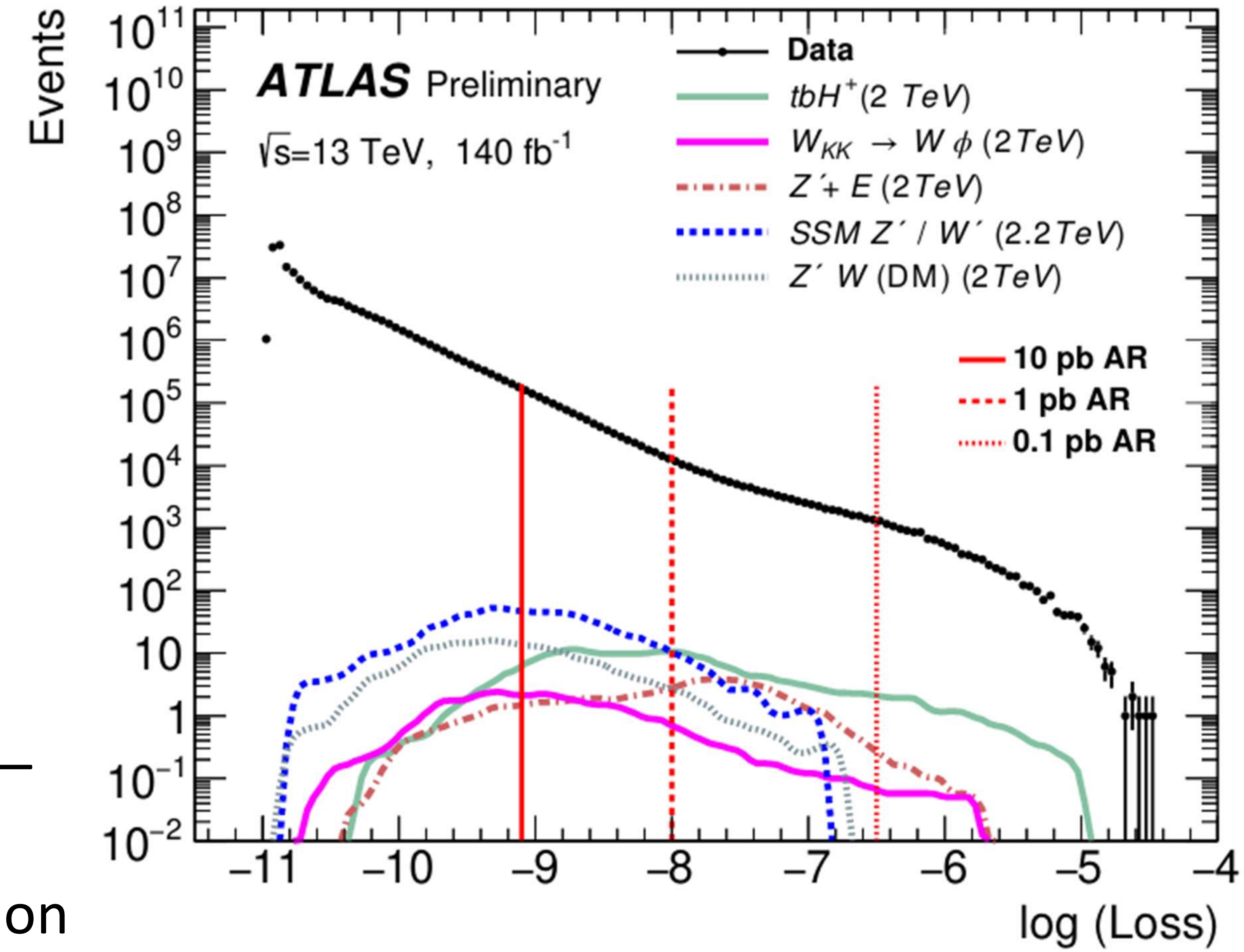


https://towardsdatascience.com/

Unsupervised Anomaly Detection



M. Farinia, Y. Nakai, D. Shih,
PRD 101, 075021 (2020)



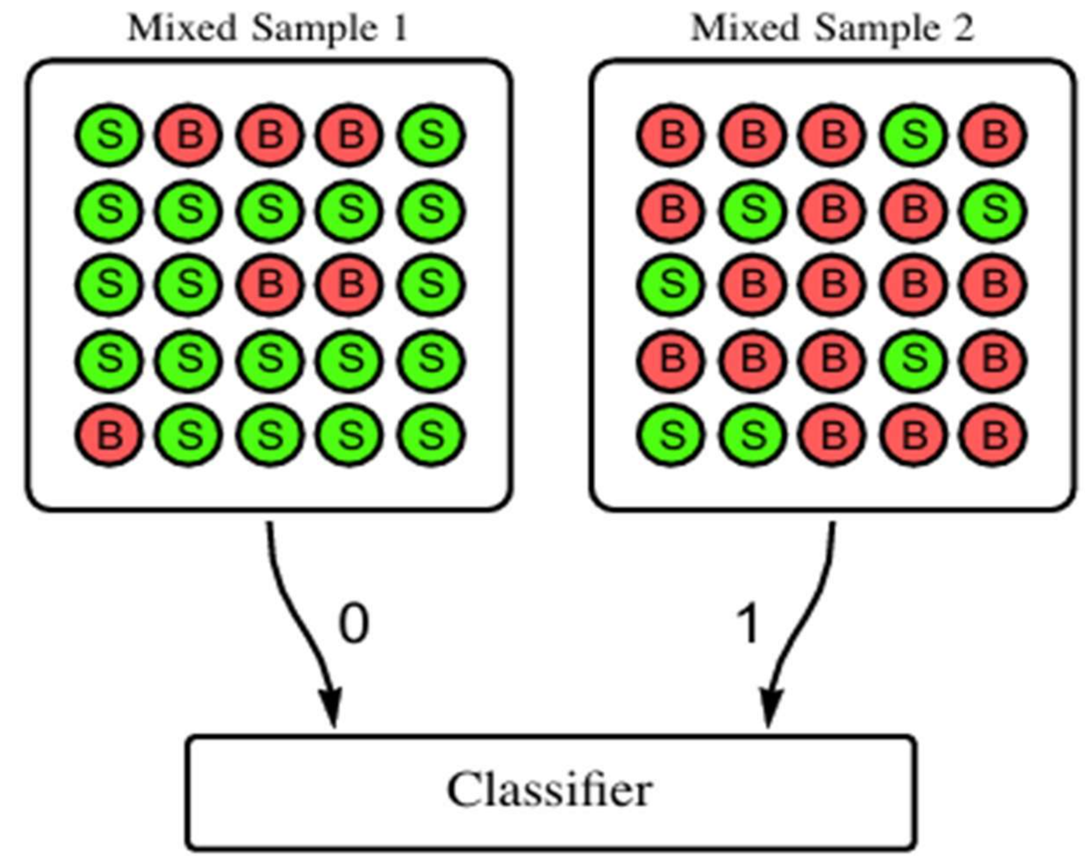
ATLAS-CONF-2023-022

- Not relying on specific signal hypothesis — model independent search.
- Unsupervised anomaly detection trained on data — no MC modelling dependence

CWoLA Bump Hunt – Weak Supervision

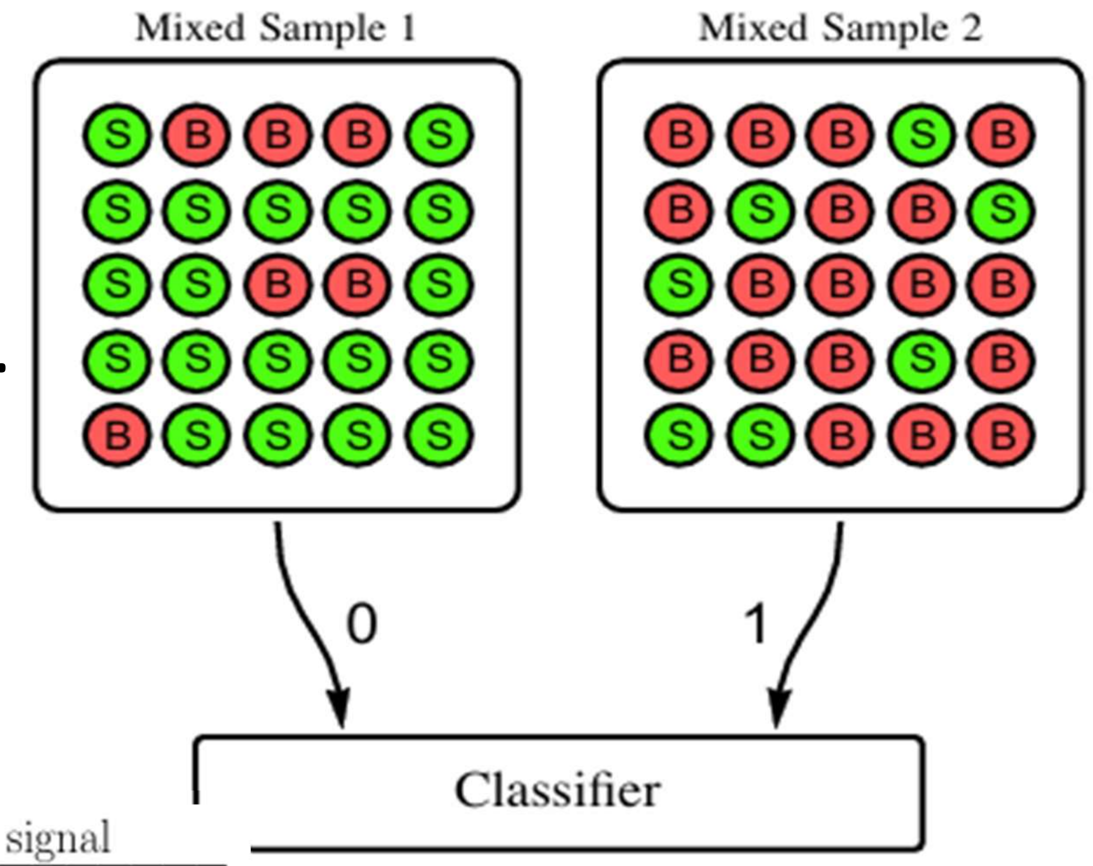
[Metodiev et al: 1708.02949]

- Trained on two data samples with different signal fractions.
- Classifier is also optimal for distinguishing signal vs background because optimal classifier is the likelihood ratio.

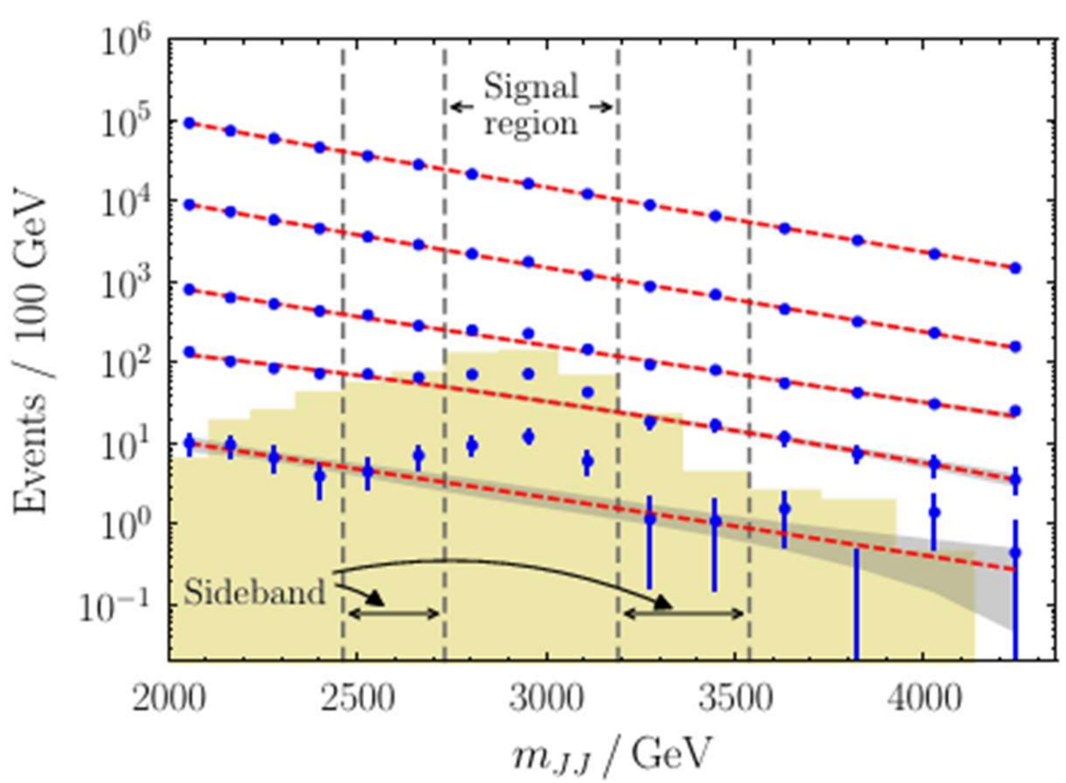


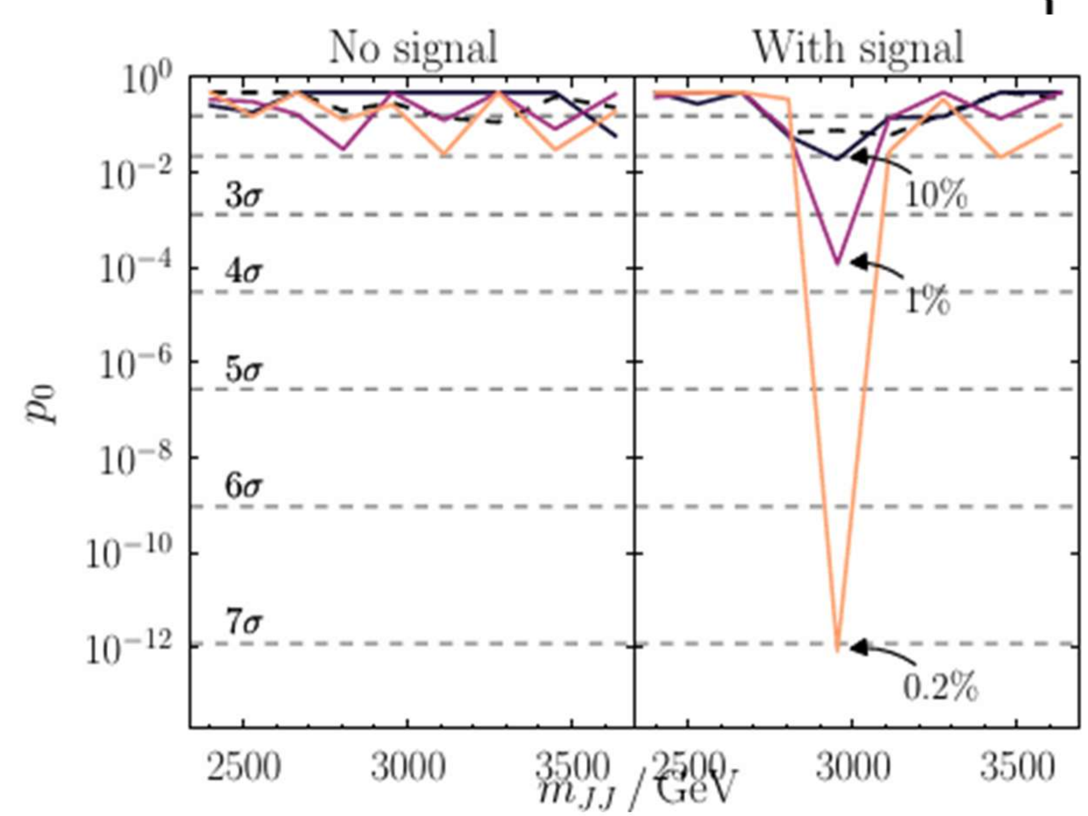
CWoLA Bump Hunt – Weak Supervision

[Metodiev et al: 1708.02949]



- Trained on two data samples with different signal fractions.
- Classifier is also optimal for distinguishing signal vs background because optimal classifier is the likelihood ratio.
- Can also be used for a weakly supervised bump hunt:
  - Train a classifier between signal region and side bands
  - Apply a threshold cut on the classifier output and perform a bump hunt

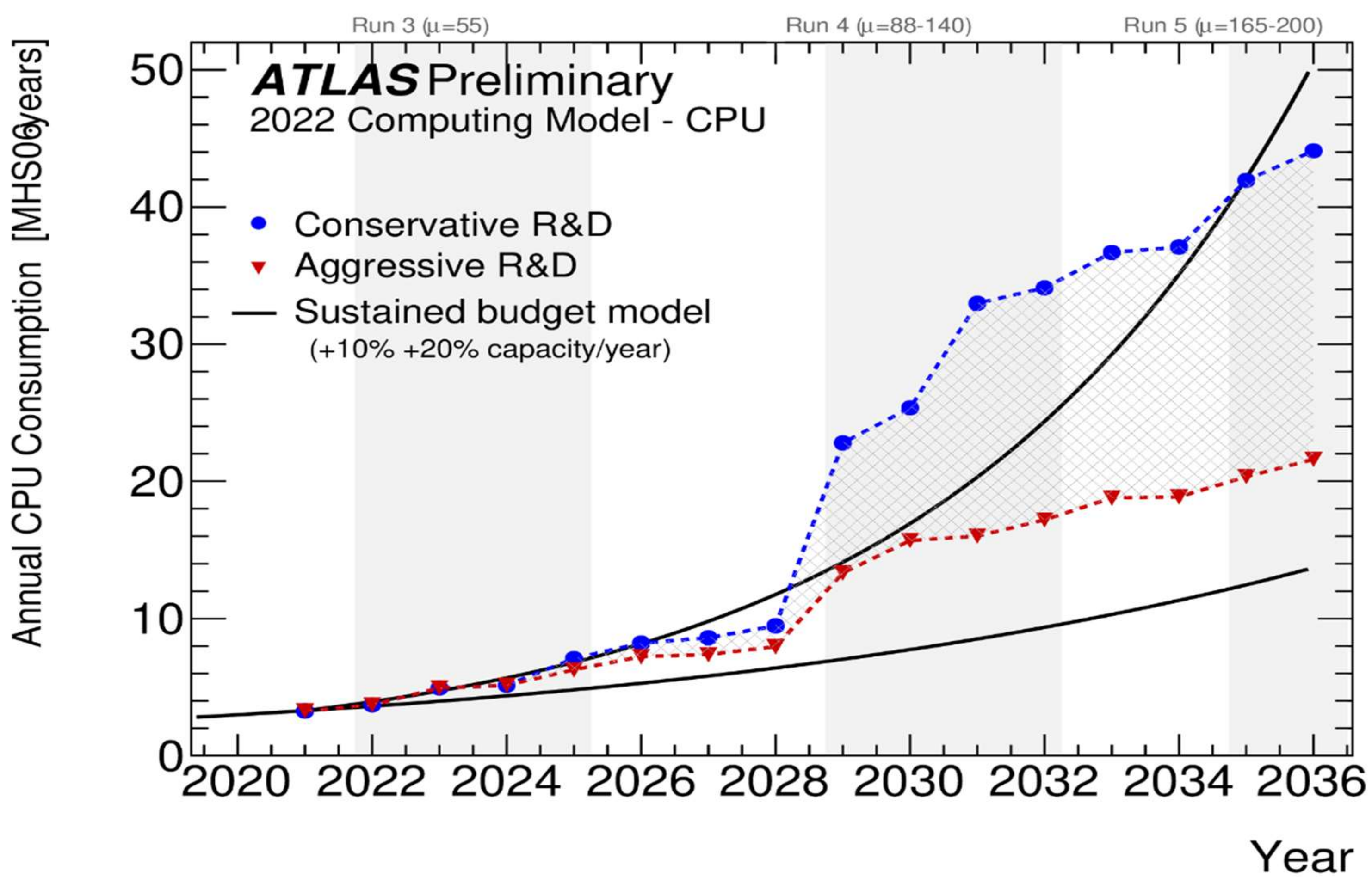




CWoLa hunting in the dijet final state
[Collins et al: 1902.02634]

LHC Run3

- At the end of LHC Run3, the computational needs will exceed the available budget.
- A large fraction goes into simulation.



[ATLAS Software and Computing HL-LHC Roadmap (cern.ch)](#)

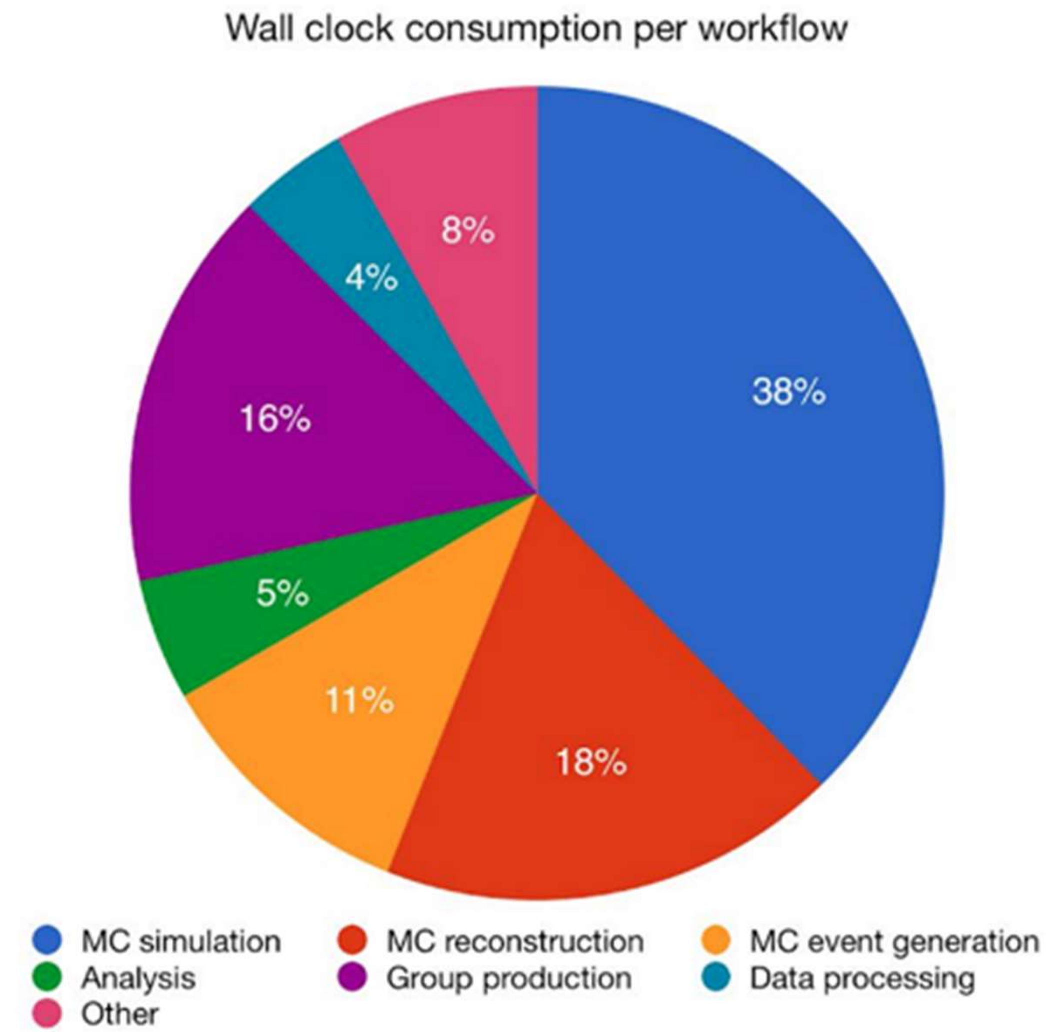
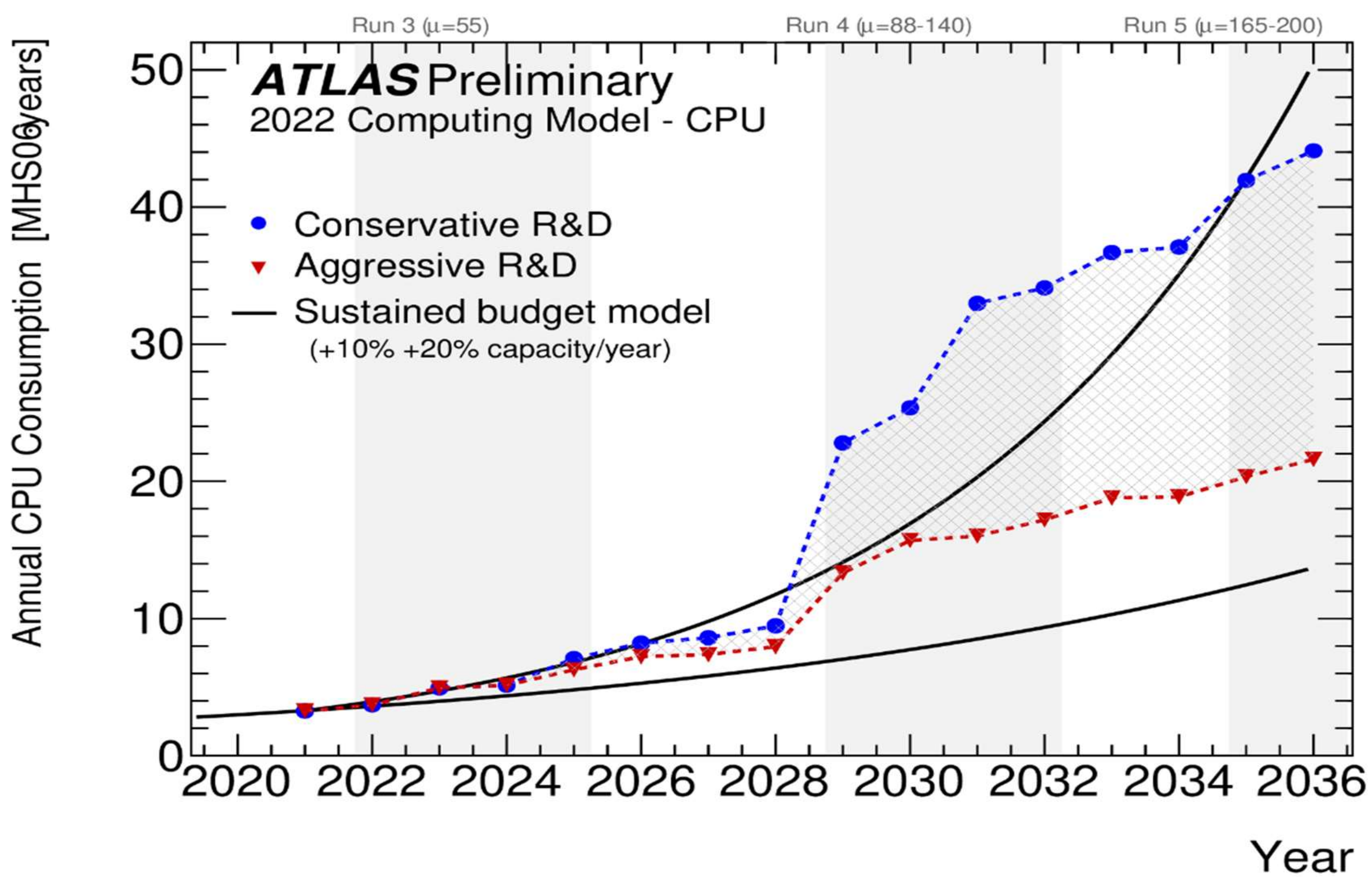


Figure 1: ATLAS CPU hours used by various activities in 2018

CERN-LHCC-2020-015; LHCC-G-178

LHC Run3

- At the end of LHC Run3, the computational needs will exceed the available budget.
- A large fraction goes into simulation.



[ATLAS Software and Computing HL-LHC Roadmap (cern.ch)]

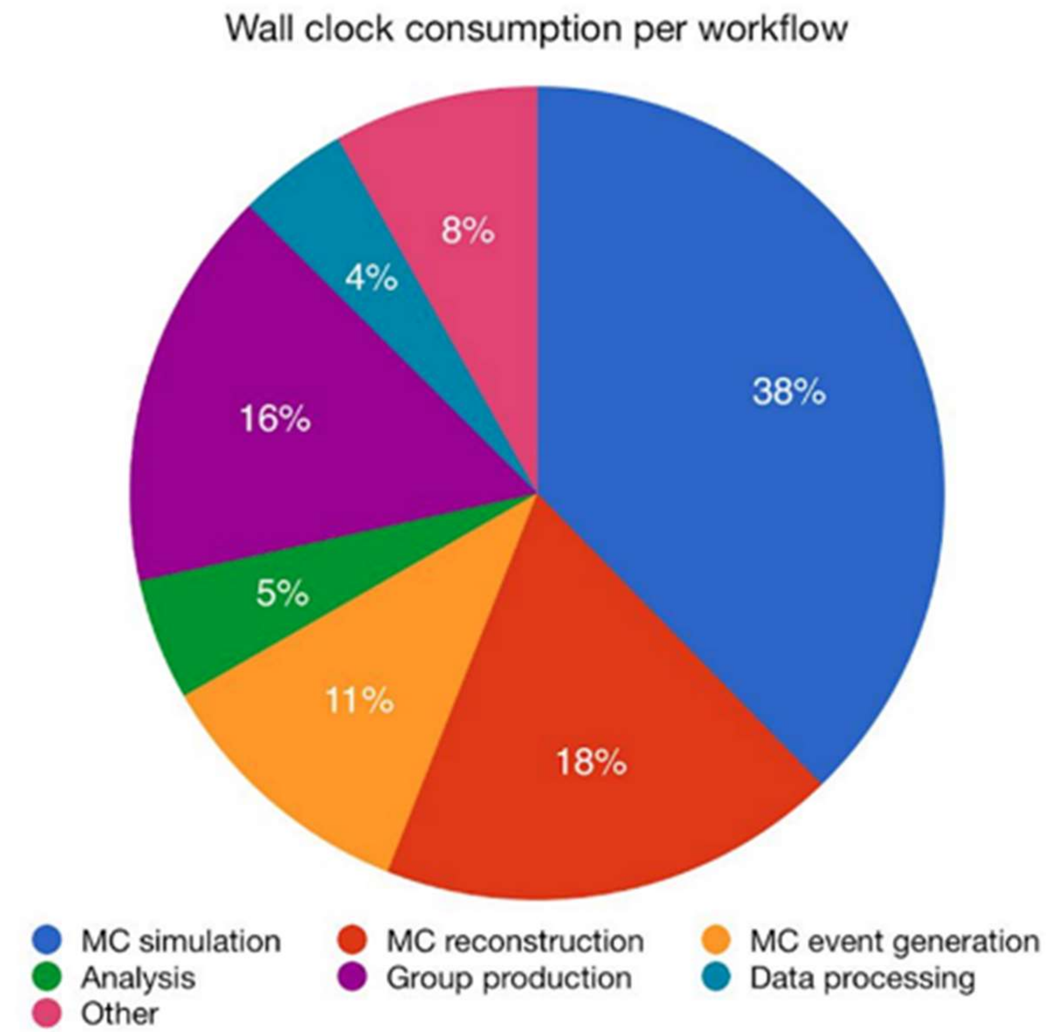


Figure 1: ATLAS CPU hours used by various activities in 2018

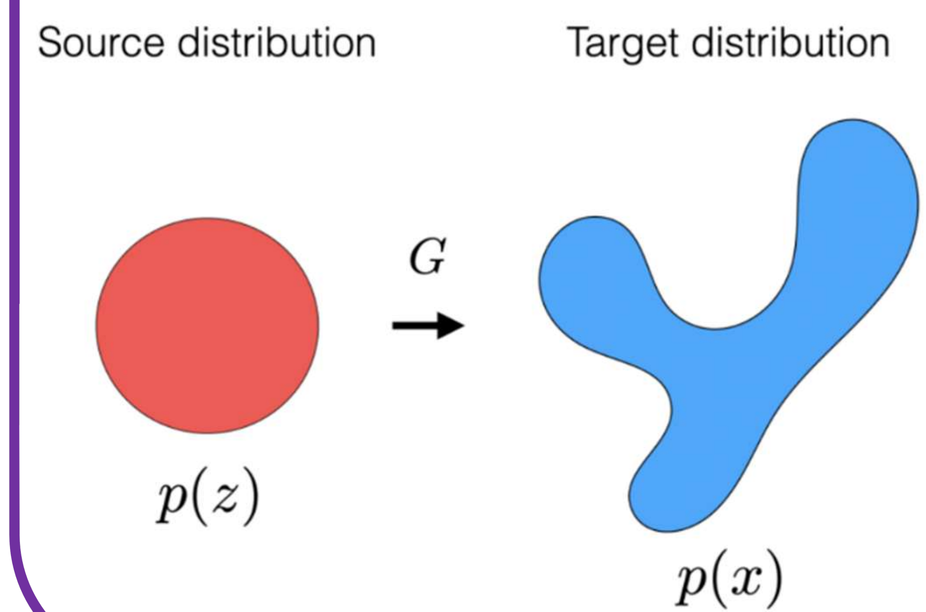
CERN-LHCC-2020-015; LHCC-G-178

Calls for Detector Simulation which will be fast and faithful

Deep Generative Models

Learns the distribution of data and generates new data from the distribution.

Sampling from Noise



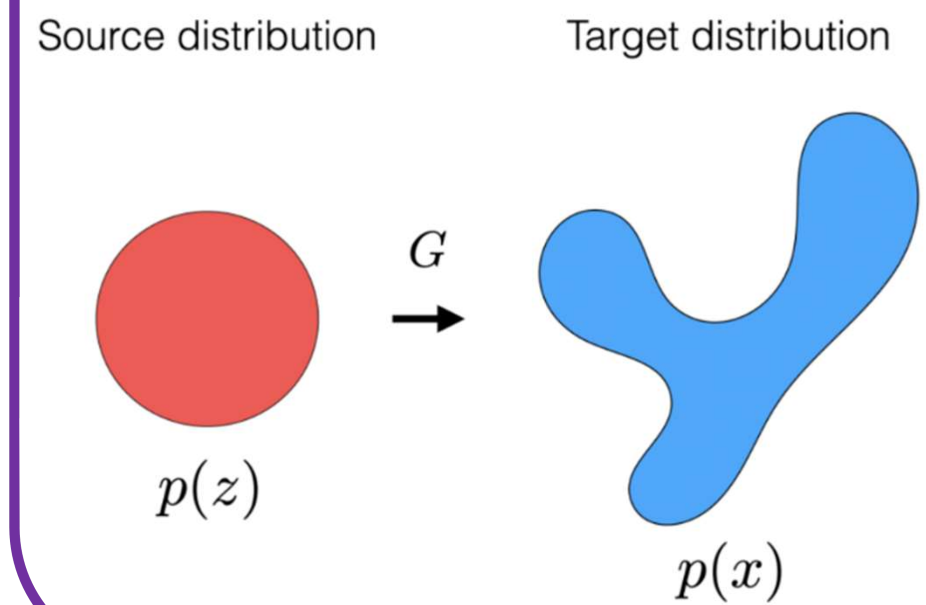
Deep Generative Models

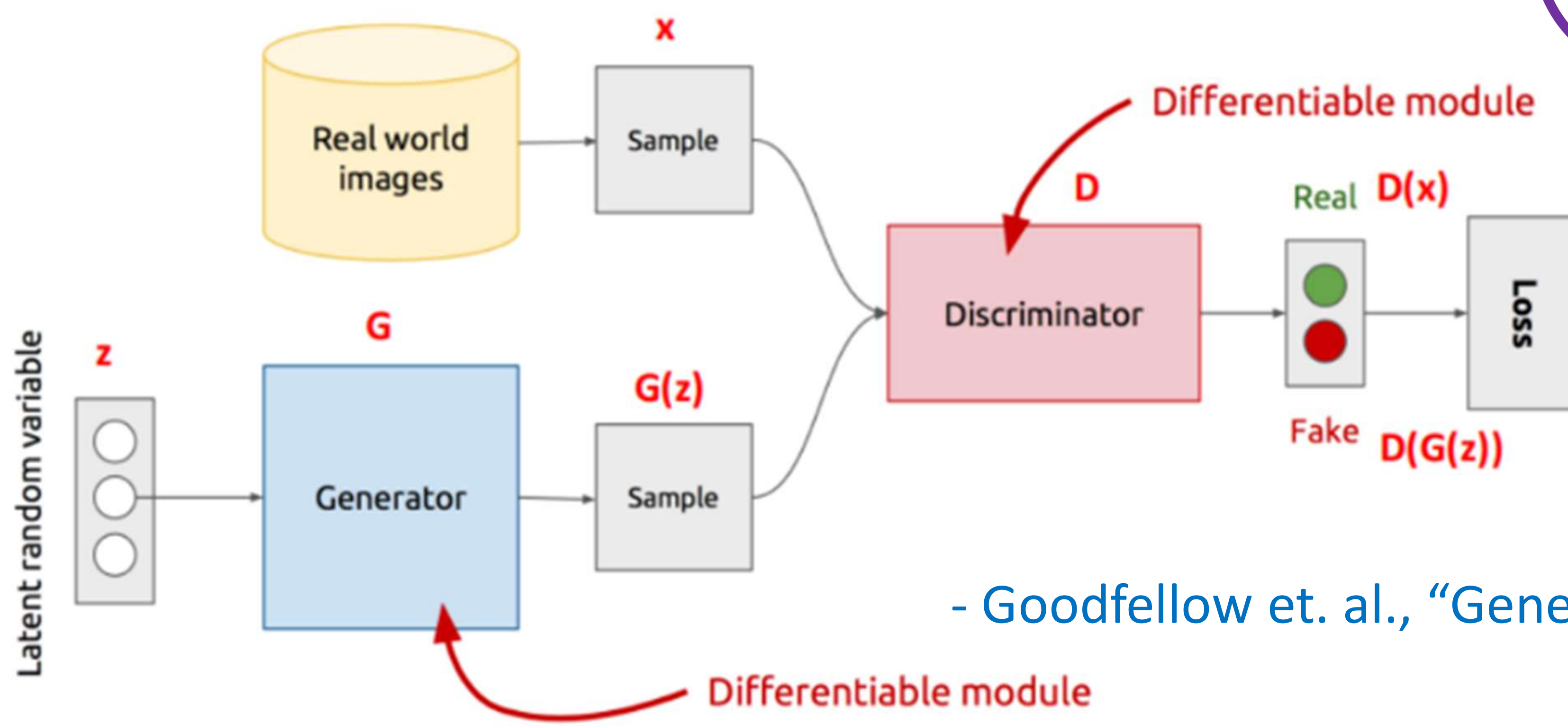
Learns the distribution of data and generates new data from the distribution.

- Generative Adversarial Networks (GANs) are a way to make a generative model by having two neural networks compete with each other

- **Generator**: generates fake samples, tries to fool the Discriminator
- **Discriminator**: tries to distinguish between real and fake samples
- Train them against each other
- Repeat this and we get better Generator and Discriminator

Sampling from Noise





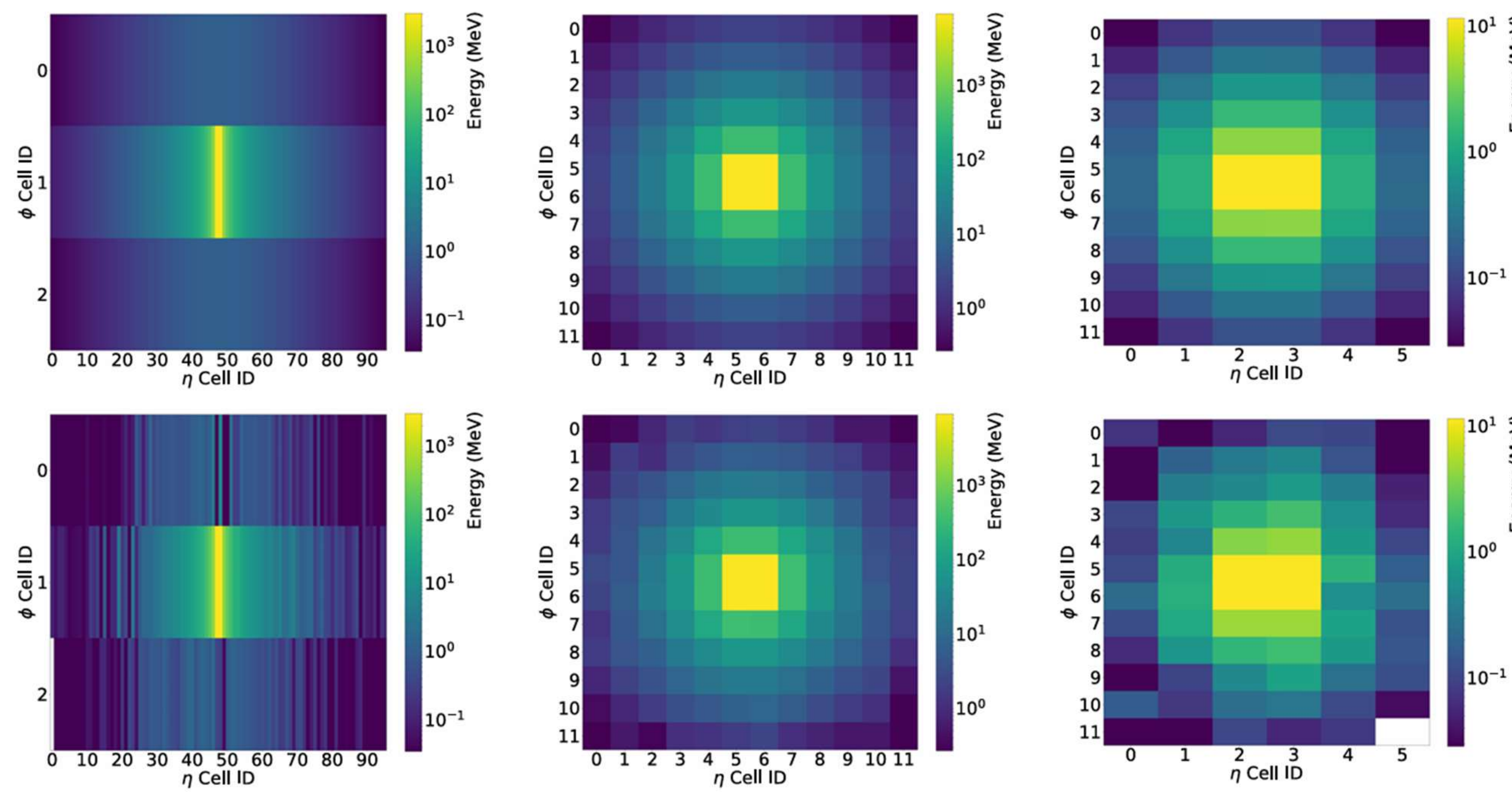
- Goodfellow et. al., “Generative Adversarial Networks” (2014)

GANs

https://research.nvidia.com/publication/2018-04_progressive-growing-gans-improved-quality-stability-and-variation

GANs

https://research.nvidia.com/publication/2018-04_progressive-growing-gans-improved-quality-stability-and-variation

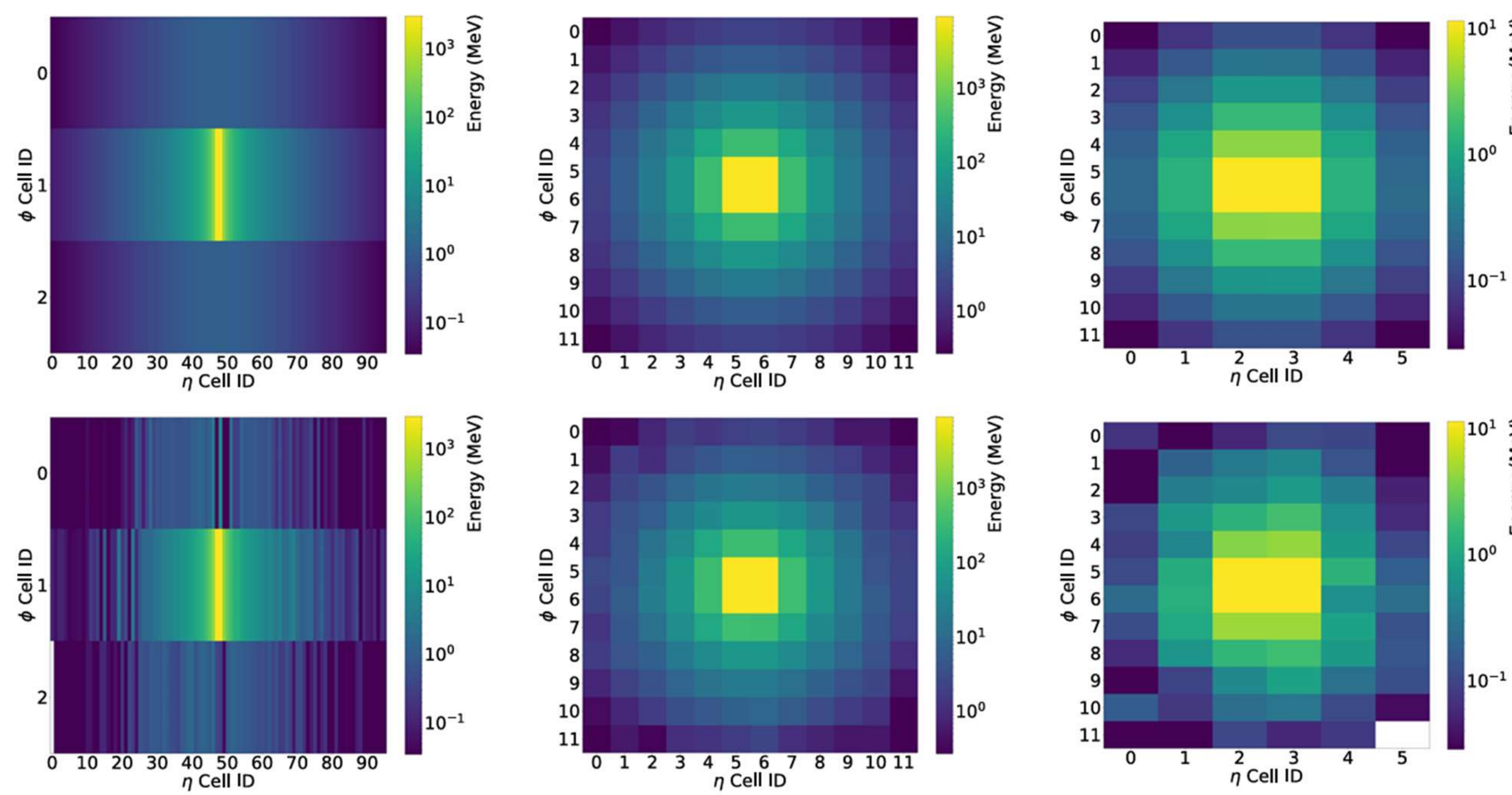


CaloGAN: Paganini, de Oliveira, Nachman [1705.02355; 1712.10321]

GANs

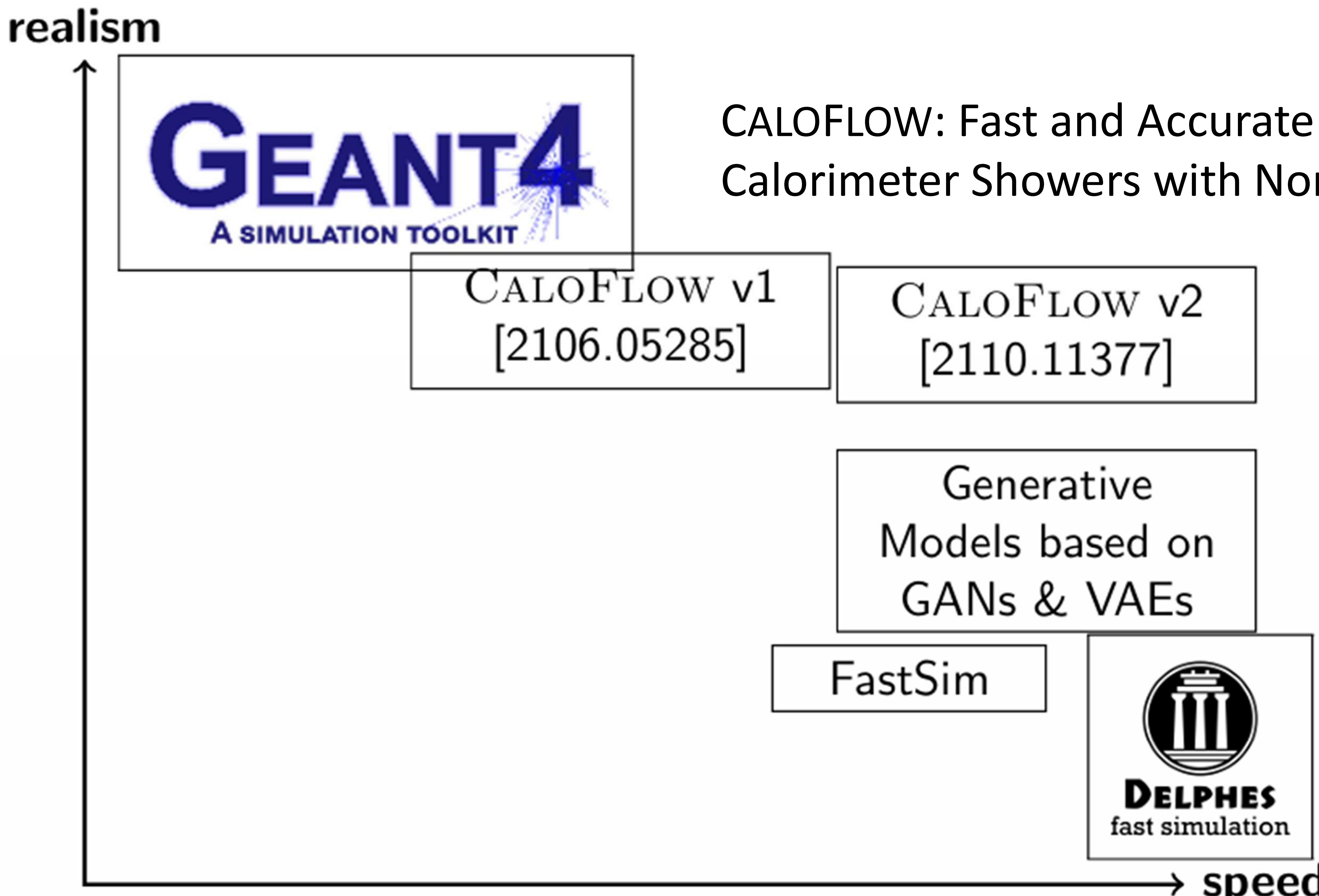
https://research.nvidia.com/publication/2018-04_progressive-growing-gans-improved-quality-stability-and-variation

- GANs have been demonstrated to be capable of reproducing Geant4 calorimeter images with reasonable accuracy (both at the individual image level and **at the distributional level**), while gaining up to 5 orders of magnitude in computational speed.



CaloGAN: Paganini, de Oliveira, Nachman [1705.02355; 1712.10321]

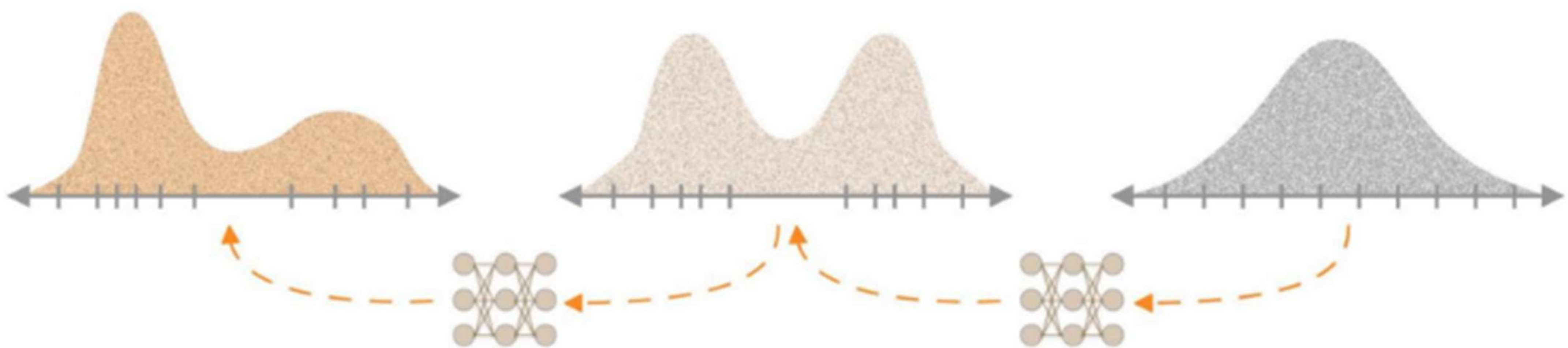
CALOFLOW: Fast and Accurate Generation of Calorimeter Showers with Normalizing Flows



Plot from Claudius Krause's slide

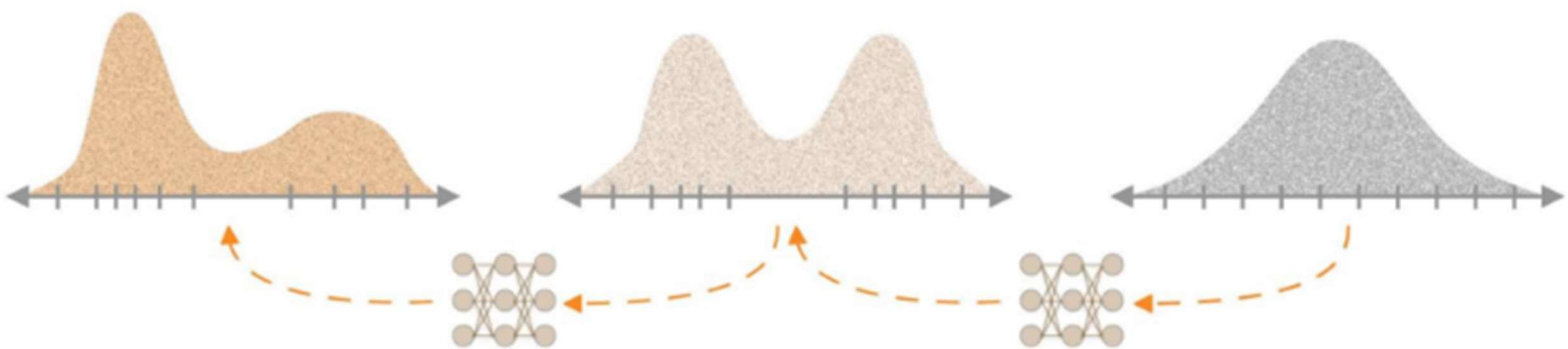
Normalizing Flows

- A transformation of a simple probability distribution into a more complex distribution by a sequence of invertible and differentiable mappings.



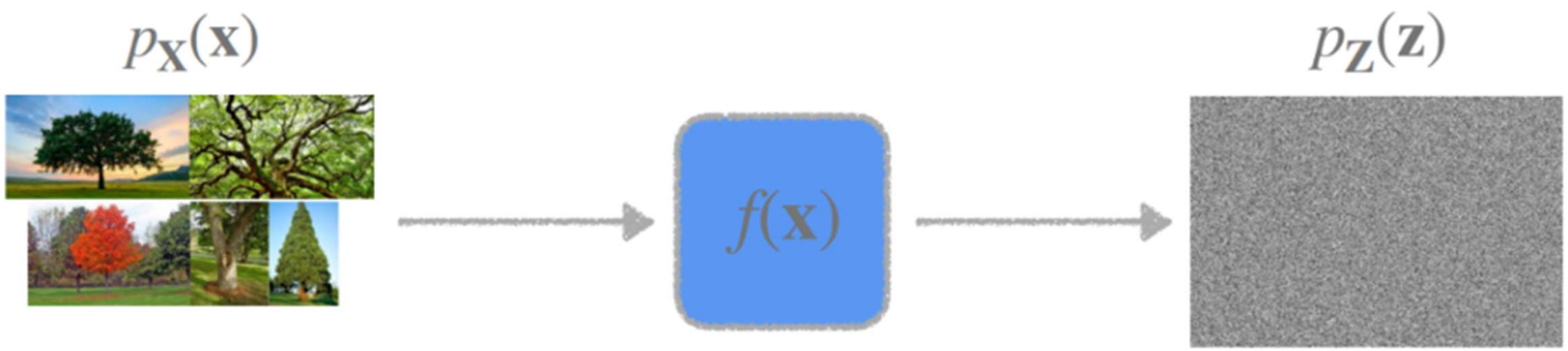
Normalizing Flows

- A transformation of a simple probability distribution into a more complex distribution by a sequence of invertible and differentiable mappings.



Probability Transformation in NFs

Learn $f(x)$ to transform $p_X(x)$ to $p_Z(z)$



Volume correction

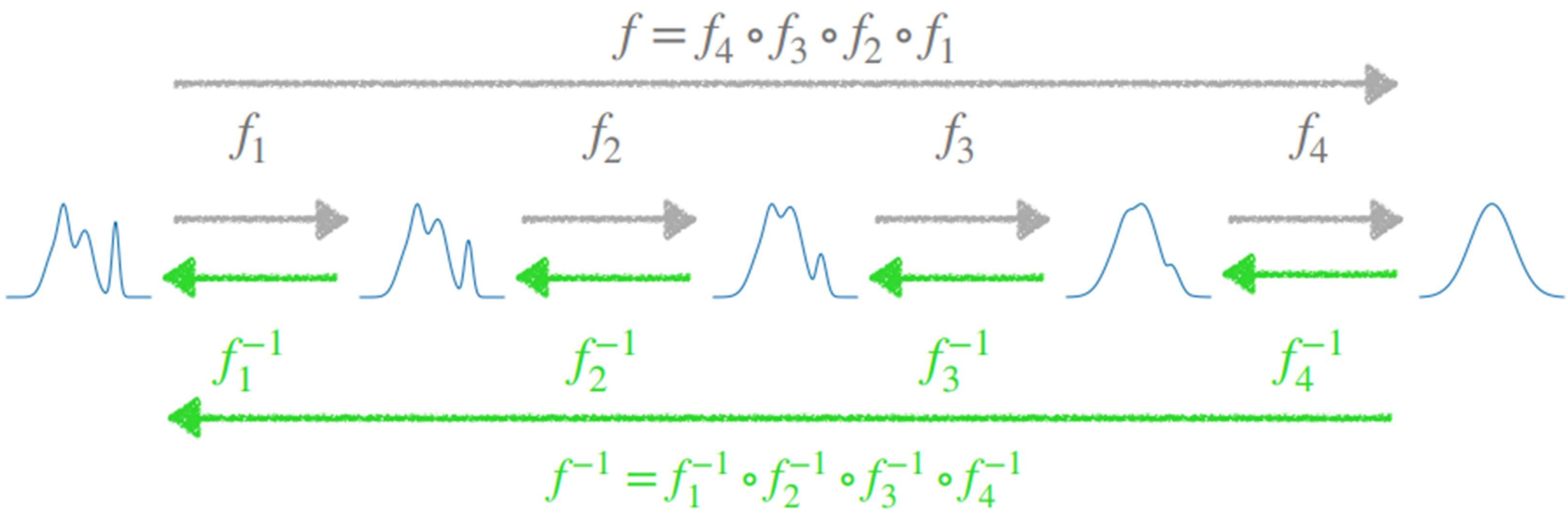
$$p_X(x) = p_Z(f(x))|\det \mathrm{J}(f(x))|$$

Invertible, differentiable function

https://mbrubake.github.io/cvpr2021-nf_in_cv-tutorial/Introduction%20-%20CVPR2021.pdf

Building Flows by Composition

- Invertible, differentiable functions are closed under composition.
- A complex flow is built from composition of simpler flows. $\mathbf{f} = \mathbf{f_K} \circ \mathbf{f_{K-1}} \circ \cdots \circ \mathbf{f_2} \circ \mathbf{f_1}$



Determinant of Jacobian

$$\det \mathrm{J}(f) = \det \prod_{k=1}^{K} \mathrm{J} f_k = \prod_{k=1}^{K} \det \mathrm{J} f_k$$

Likelihood

$$\max_{\theta} \sum_{i=1}^{N} \log p_Z(f(x_i|\theta)) + \sum_{k=1}^{K} \log|\det \mathrm{J}\, f_k(x_i|\theta)|$$

CaloFlow

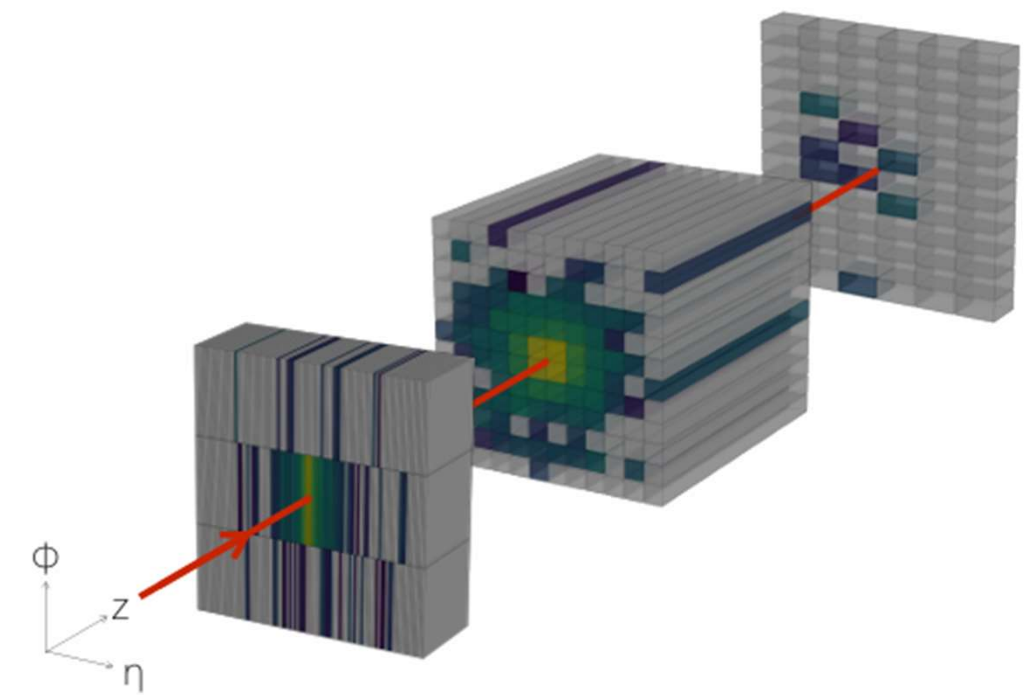
Flow I

- learns $p_1(E_0, E_1, E_2 | E_{tot})$

Flow II

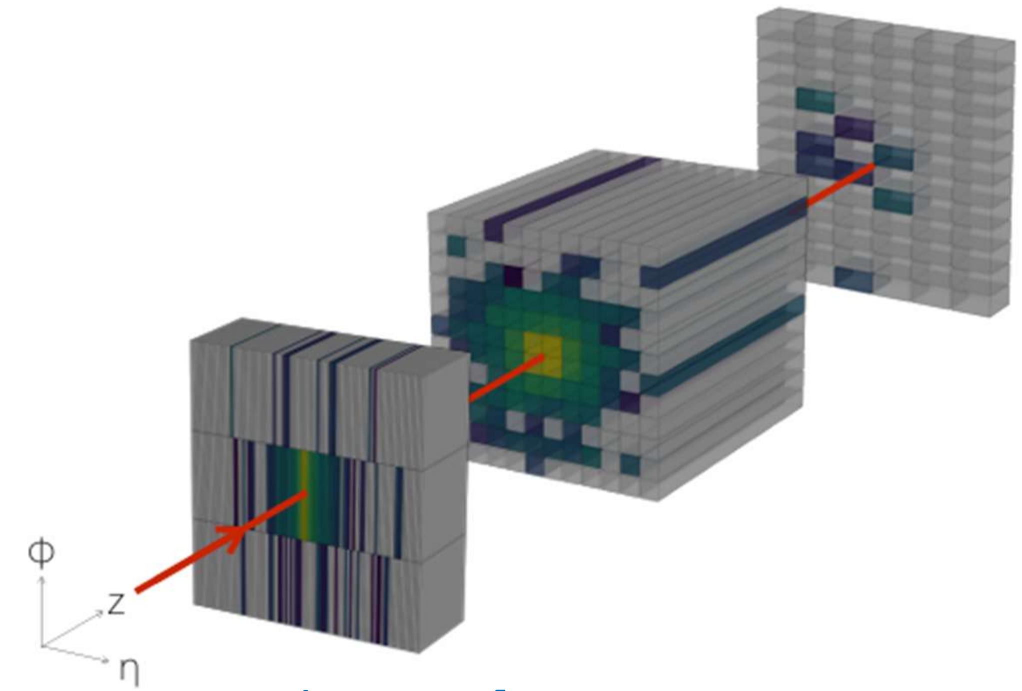
- learns $p_2(\vec{\mathfrak{T}} | E_0, E_1, E_2, E_{tot})$ of normalized showers

Claudius Krause and David Shih, 2106.05285



CaloGAN: [1705.02355; 1712.10321]

CaloFlow



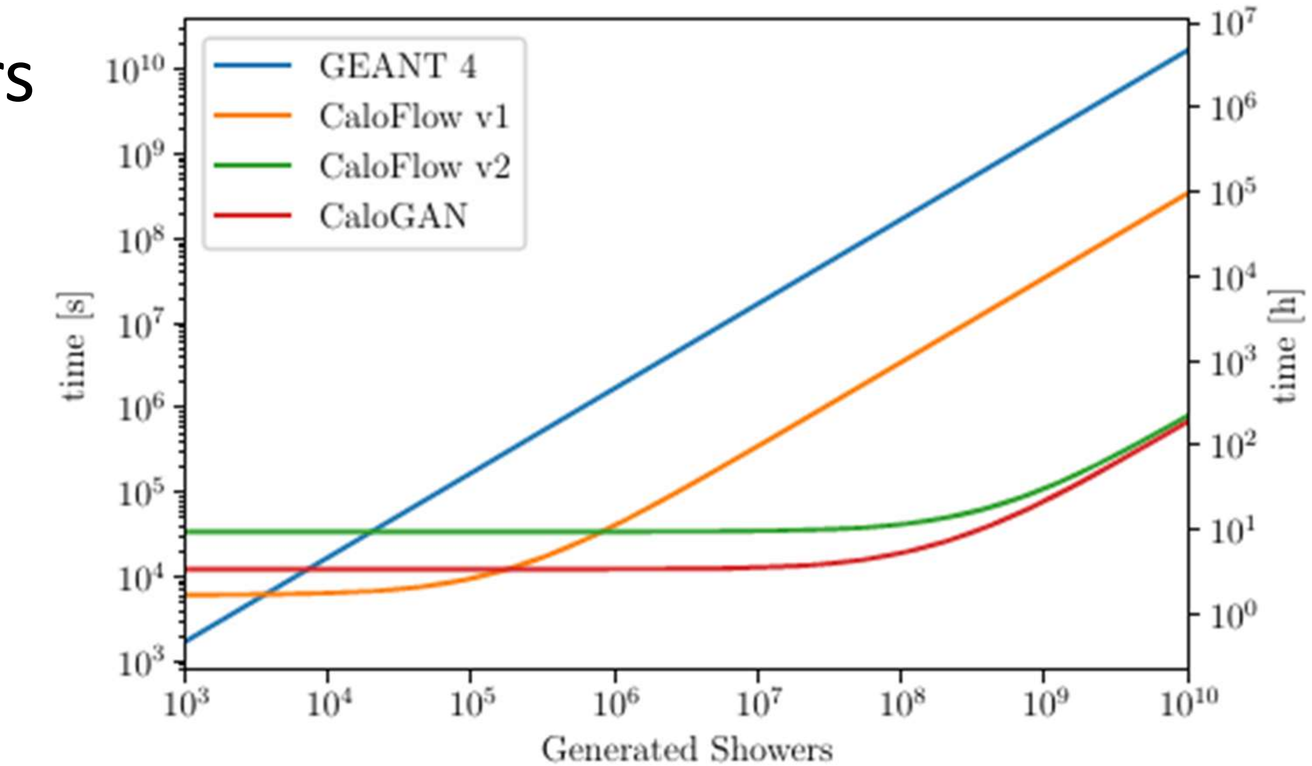
CaloGAN: [1705.02355; 1712.10321]

Flow I

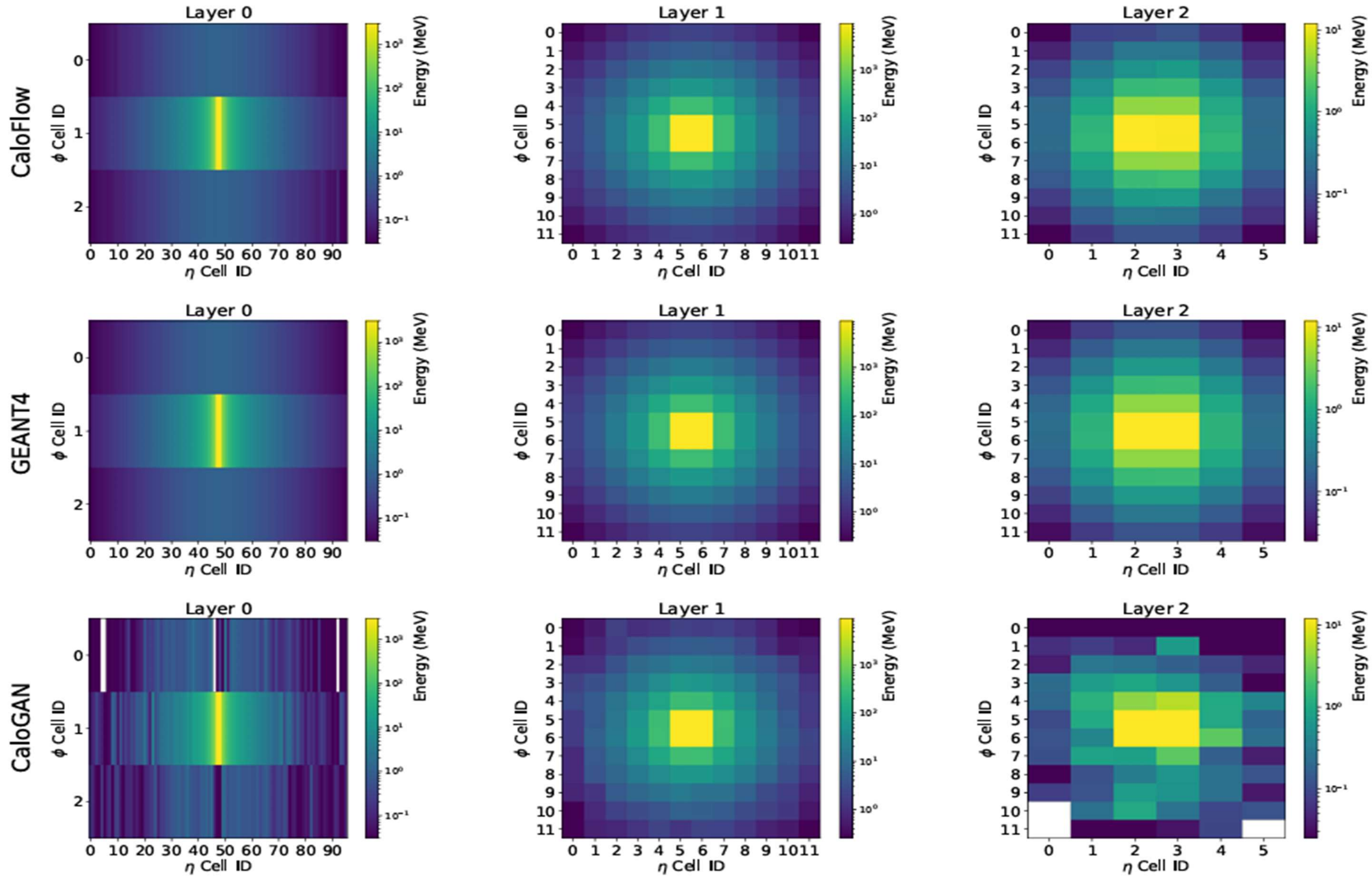
- learns $p_1(E_0, E_1, E_2 | E_{tot})$

Flow II

- learns $p_2(\vec{\mathfrak{T}} | E_0, E_1, E_2, E_{tot})$ of normalized showers
- in CaloFlow v1:
  - Slow in sampling (≈ 500× slower than CaloGAN)
  - Impressive quality!

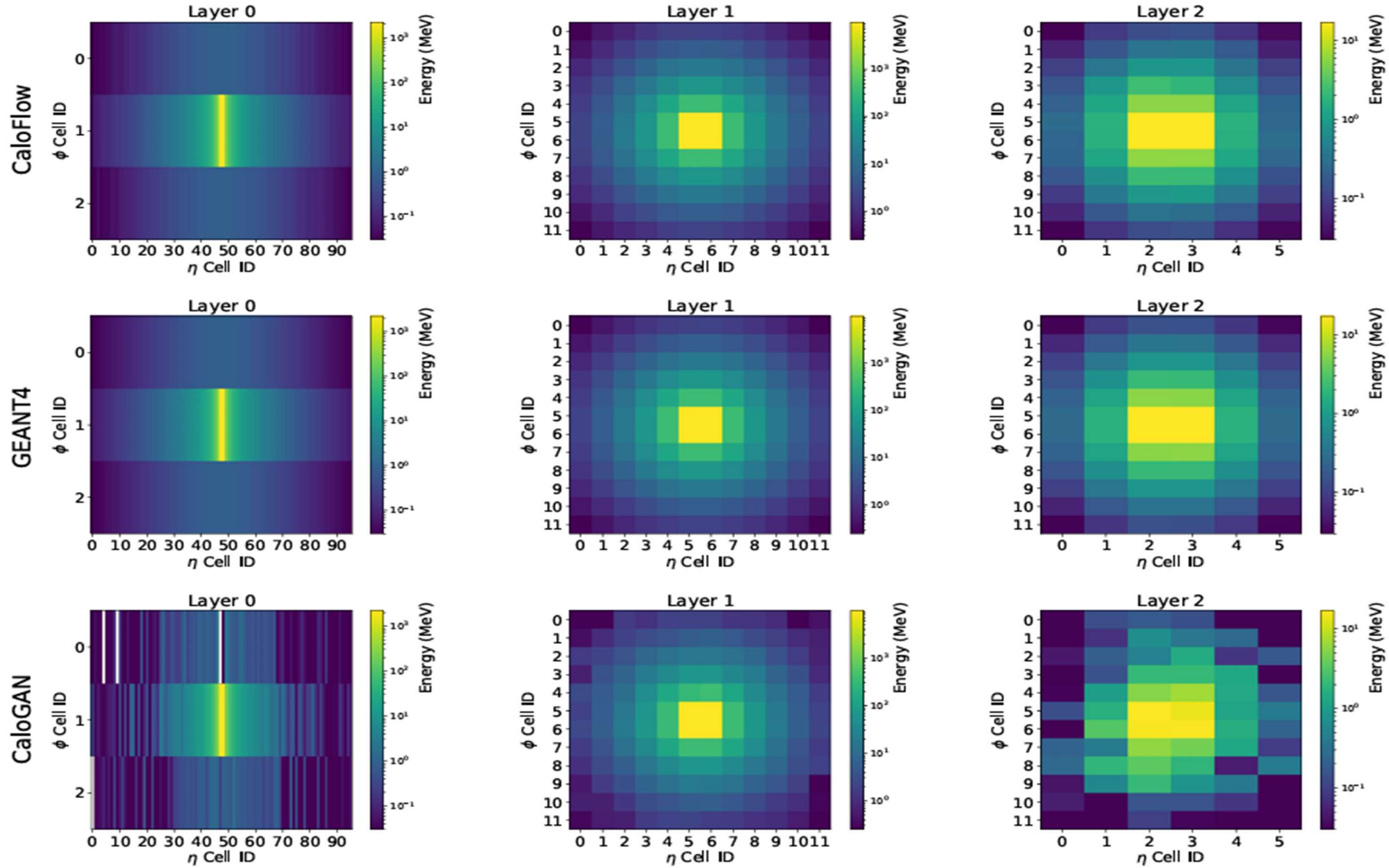


Claudius Krause and David Shih, 2106.05285



Average shower shapes for e^+. Columns are calorimeter layers 0 to 2

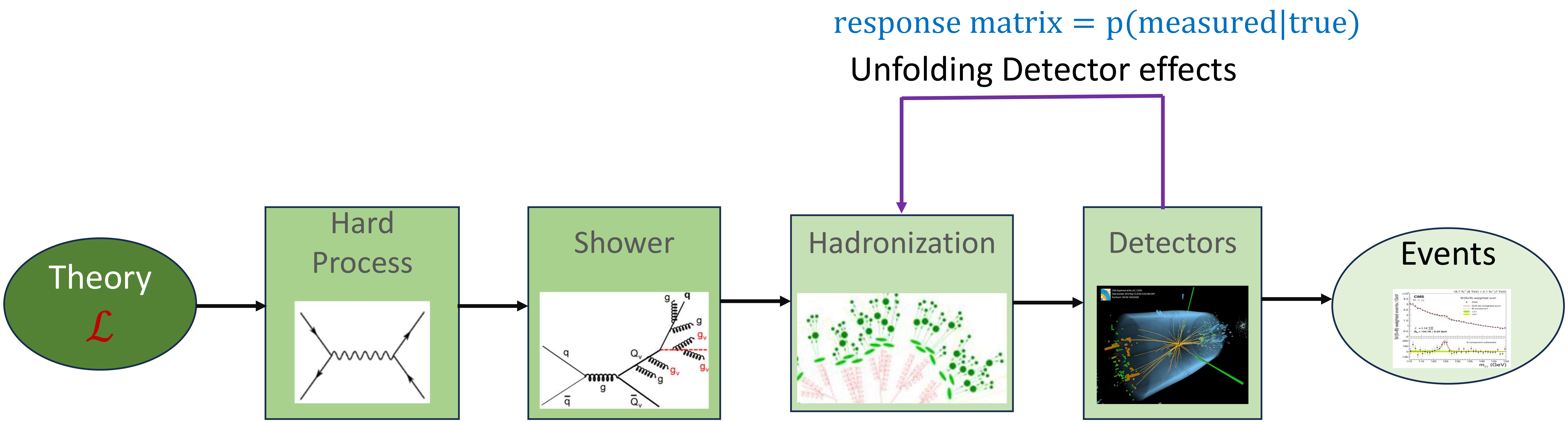
Claudius Krause and David Shih, 2106.05285



Average shower shapes for γ. Columns are calorimeter layers 0 to 2

Claudius Krause and David Shih, 2106.05285

Unfolding at the LHC



- Classifier based approach

OmniFold [1911.09107], Profiled Unfolding [2302.05390]

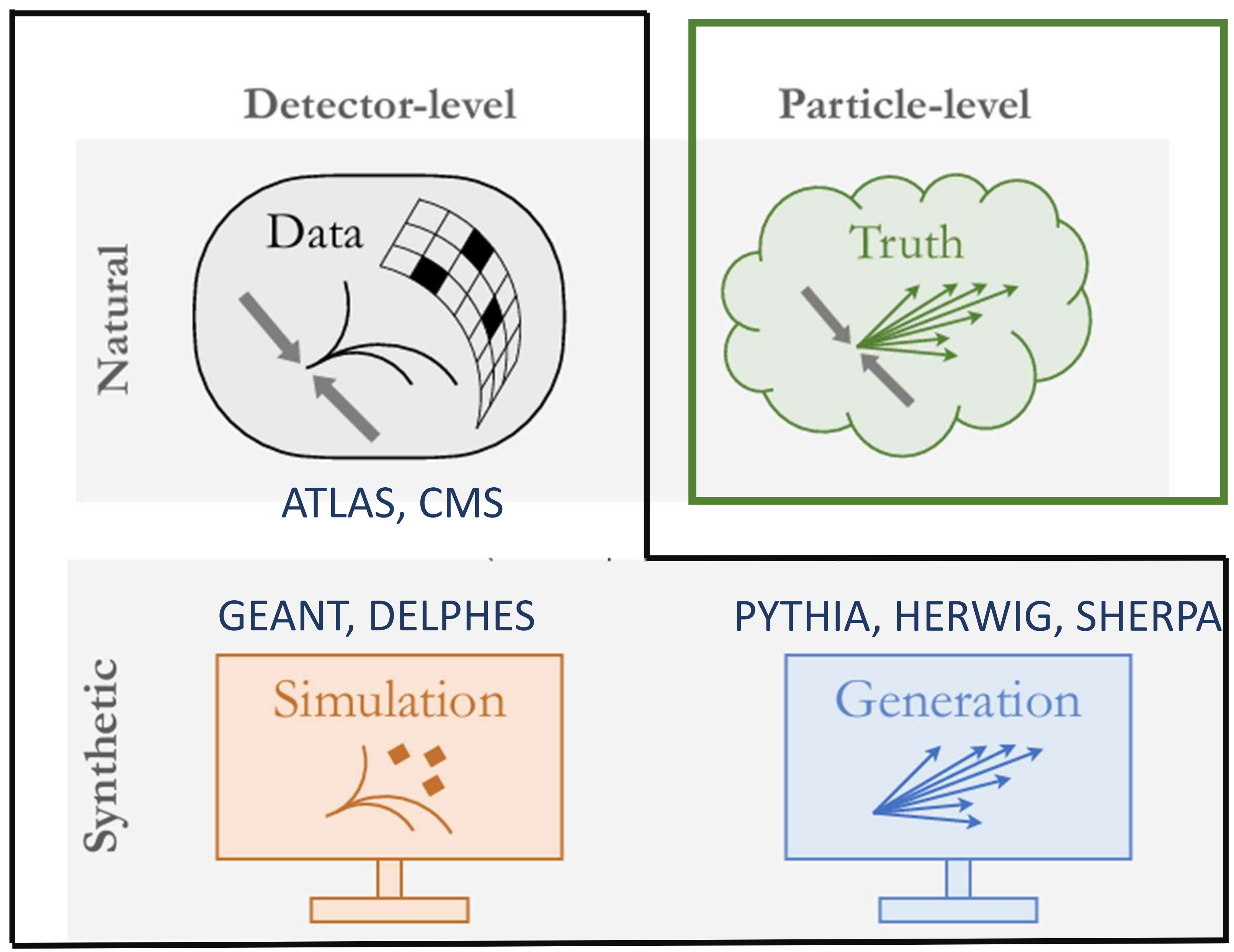
- Density based approach

FCGAN [1912.00477], cINN [2006.06685], IcINN [2212.08674], OTUS [2101.08944]

Unfolding Setup

- Measurements are affected by detector effects of finite resolution and limited acceptance.

3. Goal of unfolding is to learn a generative particle-level model that reproduces the data



2. Truth-level measurements can be compared across experiments and to theoretical calculations

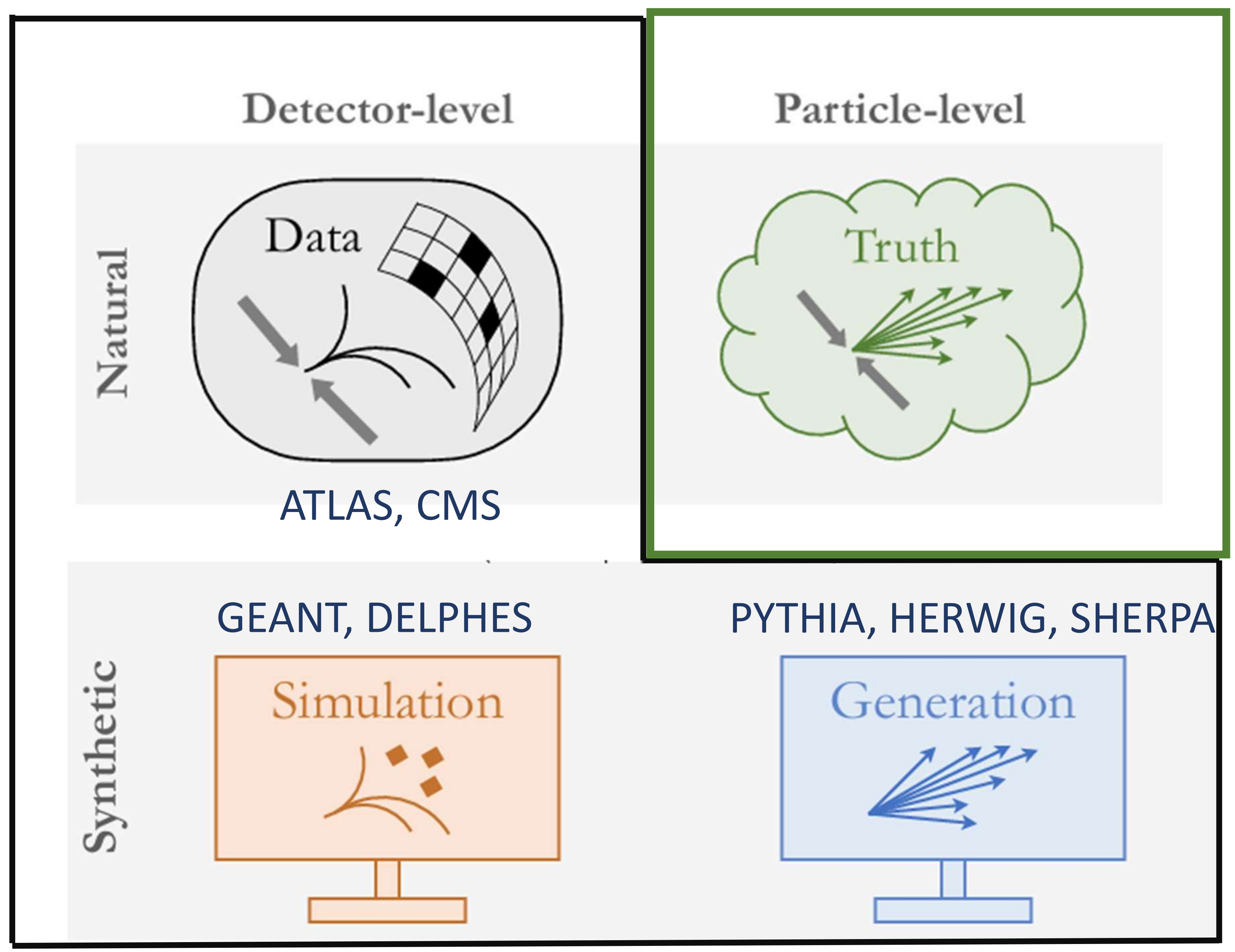
1. Learn detector response from trustable simulation

Andreassen, Komiske, Metodiev, Nachman, Thaler, 1911.09107

Unfolding Setup

Dataset A with points sampled from $p(x)$

Dataset B with points sampled from $q(x)$



Weights $\boldsymbol{w}(\boldsymbol{x}) = \boldsymbol{q}(\boldsymbol{x})/\boldsymbol{p}(\boldsymbol{x})$ so that when dataset A is weighted by $\boldsymbol{w}$, it is statistically identical to dataset B.

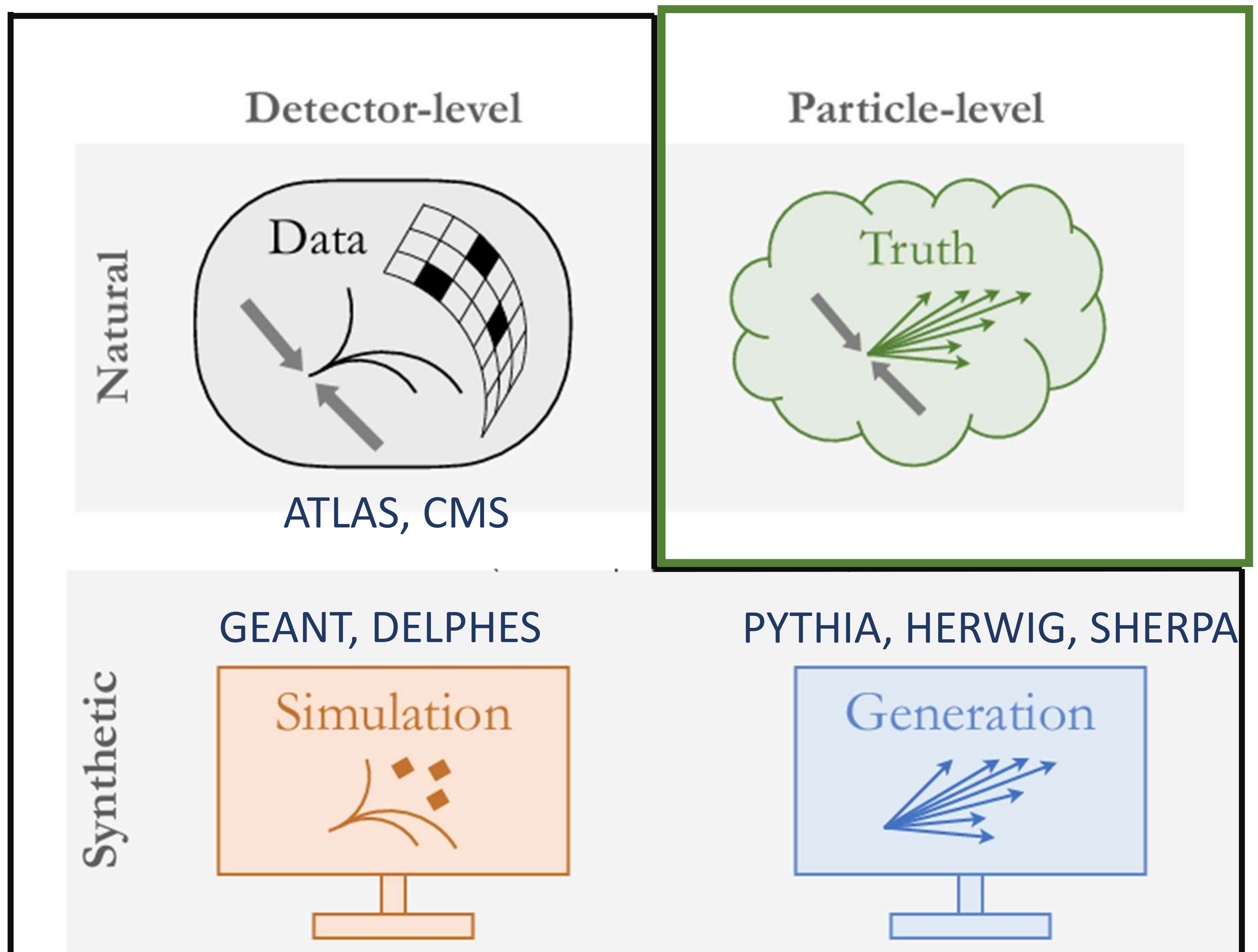
Andreassen, Komiske, Metodiev, Nachman, Thaler, 1911.09107

Unfolding Setup

- What if we don't (and can't easily) know $q(x)$ and $p(x)$?

Dataset A with points sampled from $p(x)$

Dataset B with points sampled from $q(x)$



Weights $\boldsymbol{w}(\boldsymbol{x}) = \boldsymbol{q}(\boldsymbol{x})/\boldsymbol{p}(\boldsymbol{x})$ so that when dataset A is weighted by $\boldsymbol{w}$, it is statistically identical to dataset B.

Andreassen, Komiske, Metodiev, Nachman, Thaler, 1911.09107

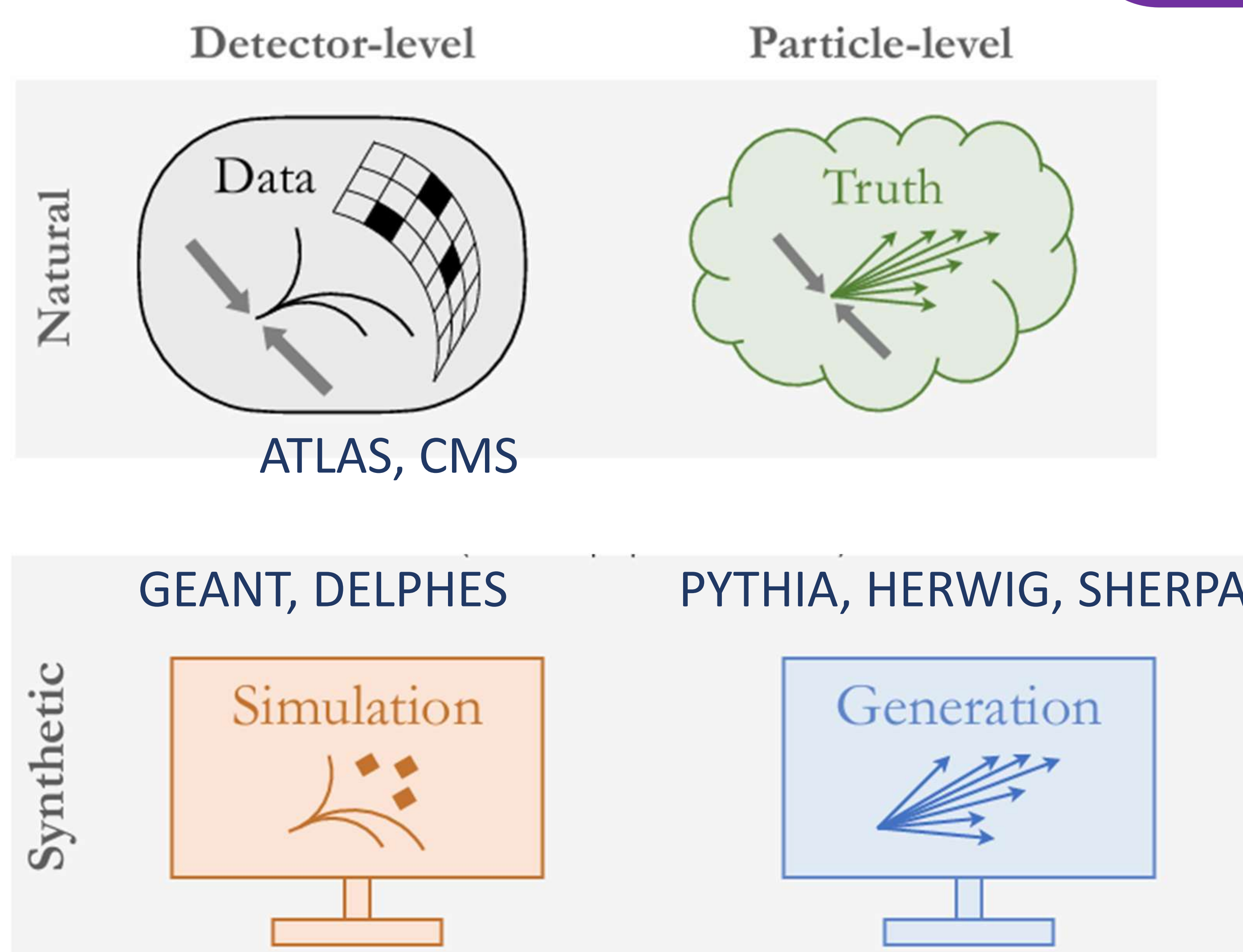
Unfolding Setup

- What if we don't (and can't easily) know $q(x)$ and $p(x)$?

Use classification to train a neural network to distinguish the two datasets, NN learn to approximate the likelihood ratio $\mathbf{q(x)/p(x)}$

Dataset A with points sampled from $\mathrm{p(x)}$

Dataset B with points sampled from $\mathrm{q}(x)$



Weights $\mathbf{w(x)} = \mathbf{q(x)/p(x)}$ so that when dataset A is weighted by $\boldsymbol{w}$, it is statistically identical to dataset B.

Andreassen, Nachman, PRD RC 101 (2020) 091901

OmniFold Equations

Inputs

(t, m) – pairs of Gen and Sim events
$\nu_0(t)$– initial particle-level weights for Gen – Data

$\nu_n(t)$– particle-level weights for Gen, n[th] iteration
$\omega_n(m)$– detector-level weights for Sim, n[th] iteration

$\omega_n^{pull}(t) = \omega_n(m)$ – pulling ω_n back to particle-level
$\nu_n^{push}(m) = \nu_n(t)$ – pushing ν_n to detector-level

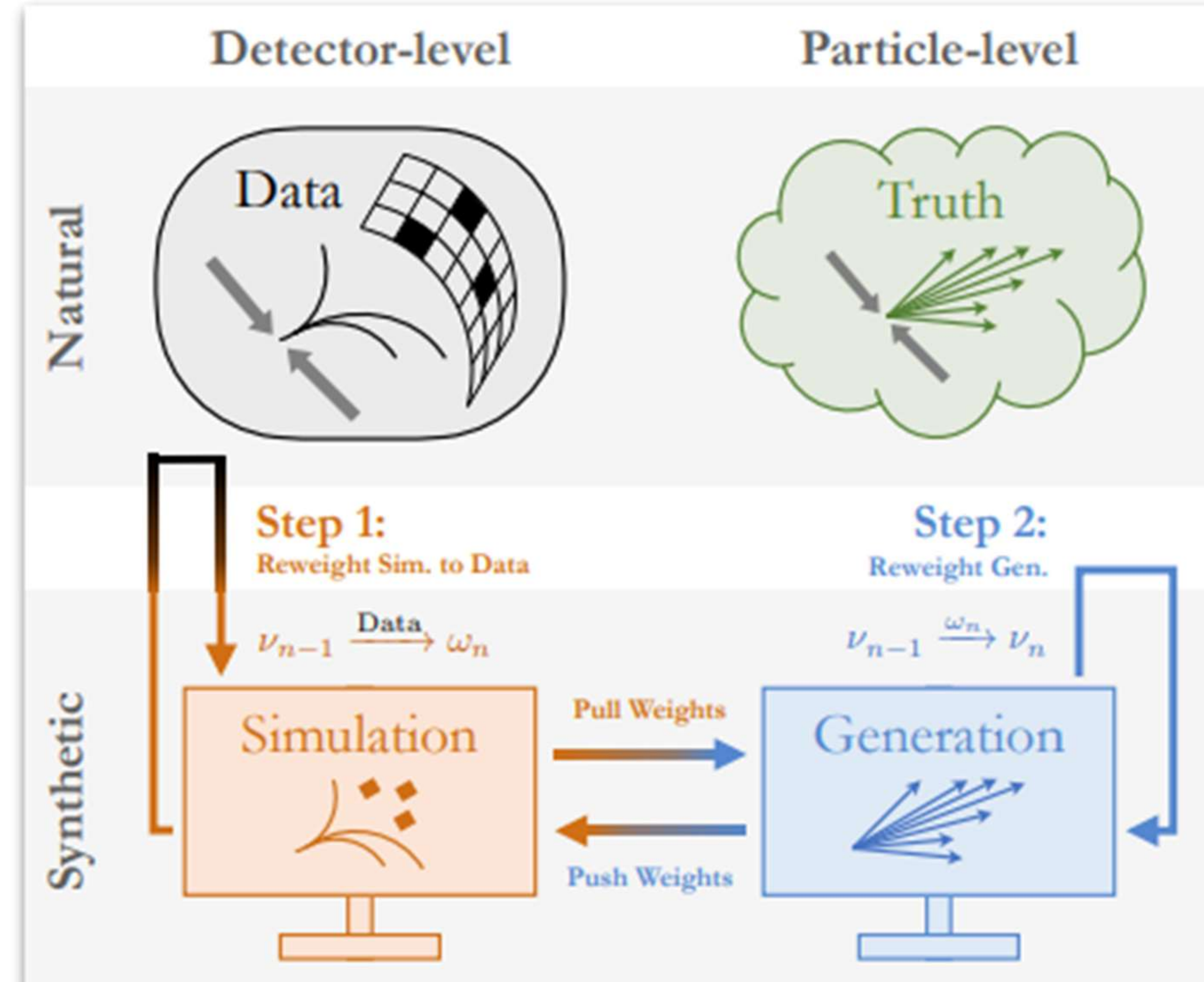
OmniFold

Step 1 $-\ \omega_n(m) = \nu_{n-1}^{\text{push}}(m) \times \mathrm{L}\left[(1, \text{Data}), \left(\nu_{n-1}^{\text{push}}, \text{Sim}\right)\right](m)$

Step 2 $-\ \nu_n(t) = \nu_{n-1}(t) \times \mathrm{L}\left[\left(\omega_n^{\text{pull}}, \text{Gen}\right), (\nu_{n-1}, \text{Gen})\right](t)$

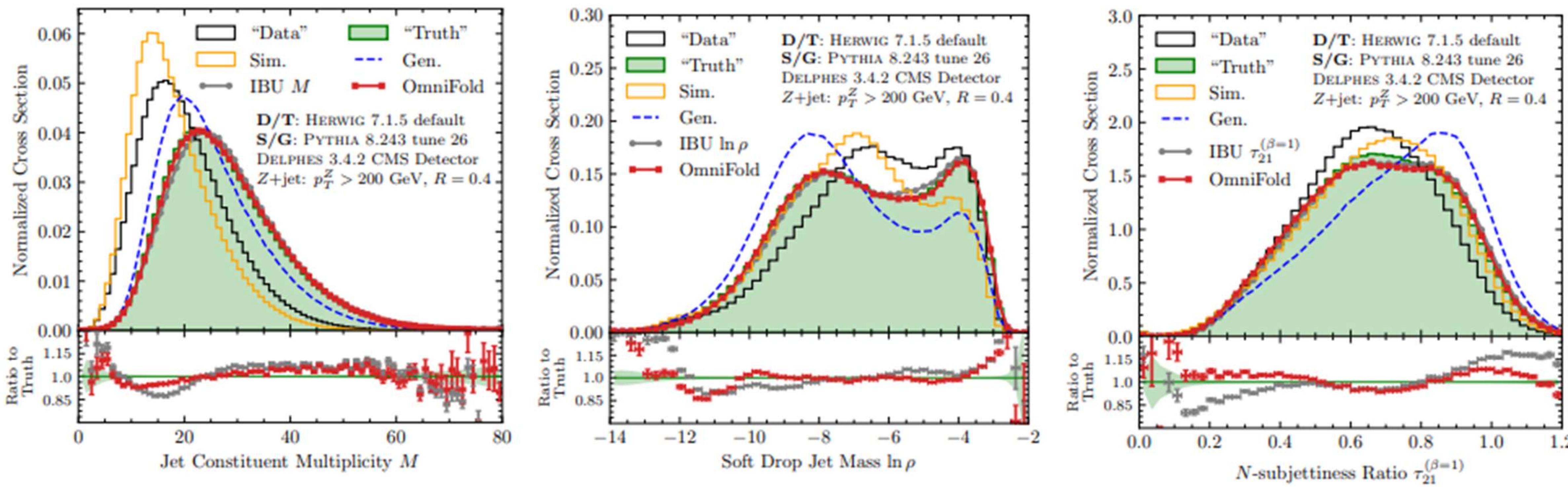
Unfold any observable $p_{Gen}(t)$ using universal weights $\nu_n(t)$

$$\boxed{p_{\text{unfolded}}^{(n)}(t) = \nu_n(t) \times p_{\text{Gen}}(t)}$$



Andreassen, Komiske, Metodiev, Nachman, Thaler, 1911.09107

OmniFolding Jet Substructure Observables



$Z(\rightarrow \mu^+\mu^-)$ + Jet Events

"Data" – Herwig 7.1.5

MC – Pythia 8.243, tune 26

1.6 million events each after cuts

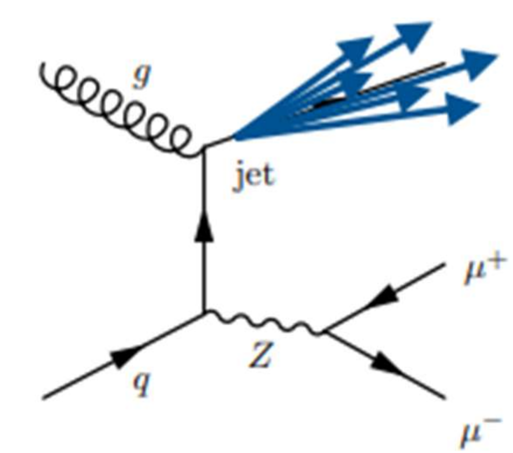
Detector Simulation

CMS-like detector – Delphes 3.4.2

Jets

Anti-k_T, $R = 0.4$ – FastJet 3.3.2

$p_T^Z > 200$ GeV, assume excellent muon detector resolution



- Five unfolding iterations in all cases
- Statistical uncertainties on prior shown in ratio

Andreassen, Komiske, Metodiev, Nachman, Thaler, 1911.09107

Summary/Outlook

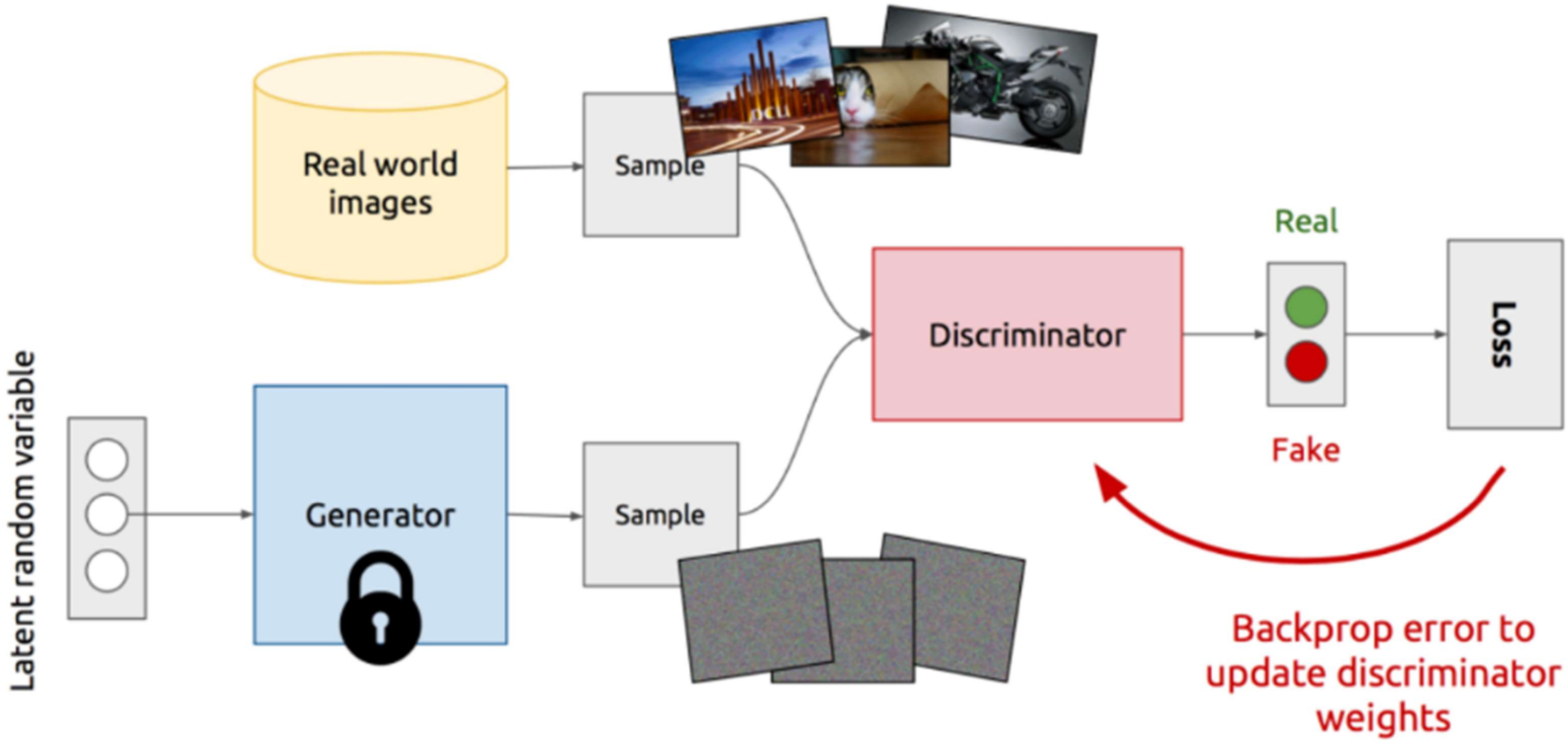
- There is a substantial ongoing work in model agnostic searches, and its exciting to see it starting to be used in experimental results.

  [ATLAS: 2005.02983, ATLAS-CONF-2022-045, ATLAS-CONF-2023-022, CMS-DP-2022-021, CMS-DP-2022-043....]

- Techniques, like generative models, are paving the way for more efficient and realistic simulations, reducing computational costs and expanding the scope of theoretical investigations.

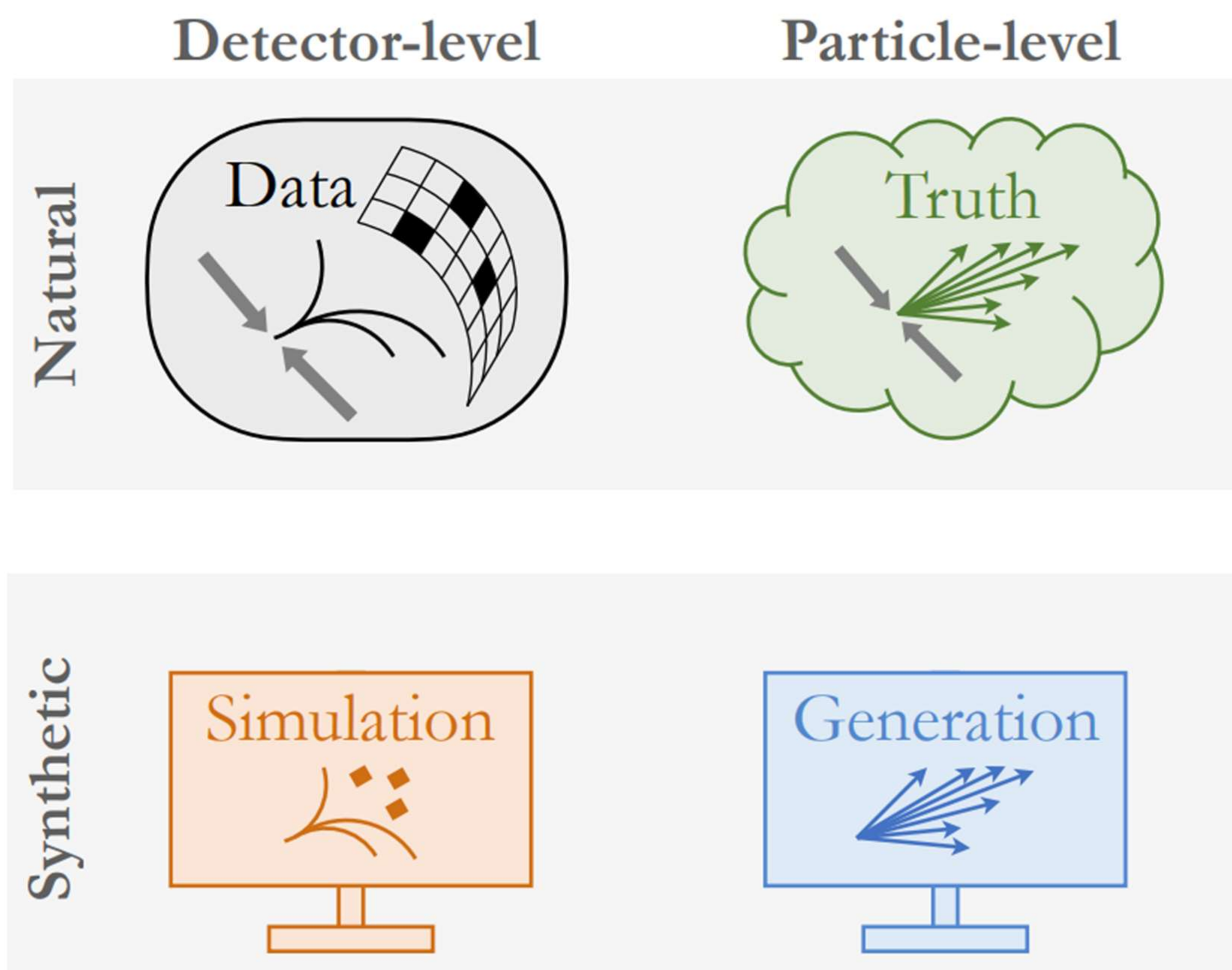
- Density estimation-based models like normalizing flows/diffusion models (that can efficiently map a simple distribution to a target one) are being used for detector simulation.

Training Discriminator

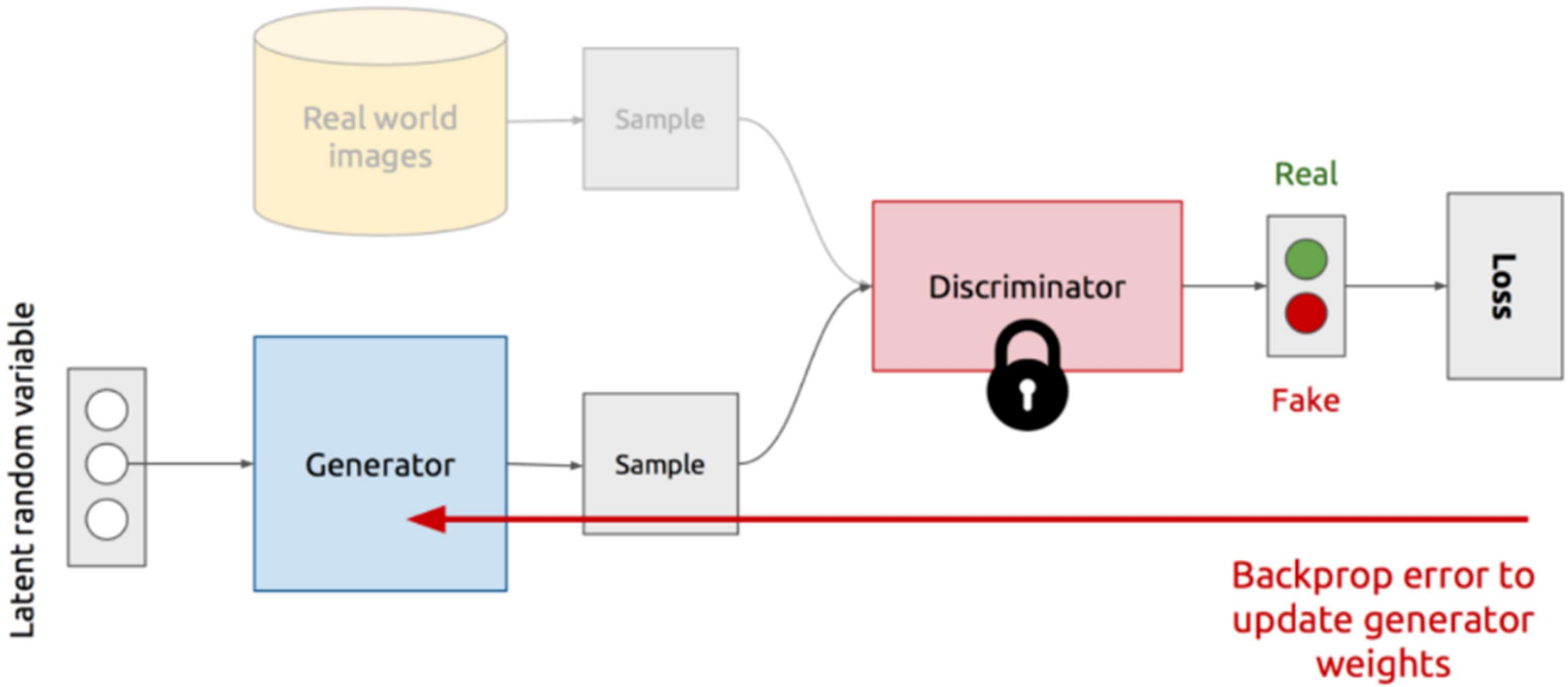


OmniFold - Schematic

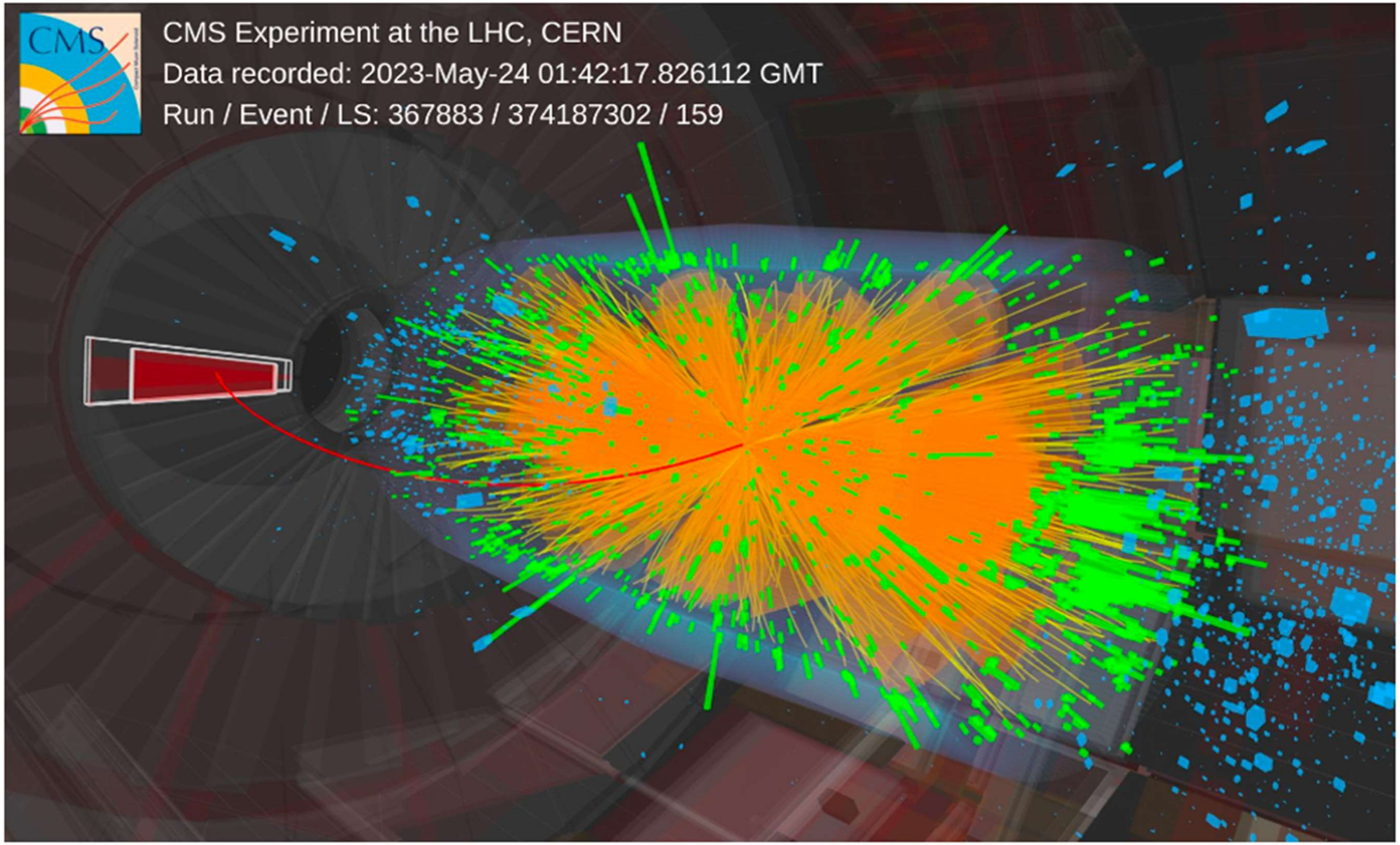
- OmniFold weights particle-level Gen to be consistent with Data once passed through the detector



Training Generator

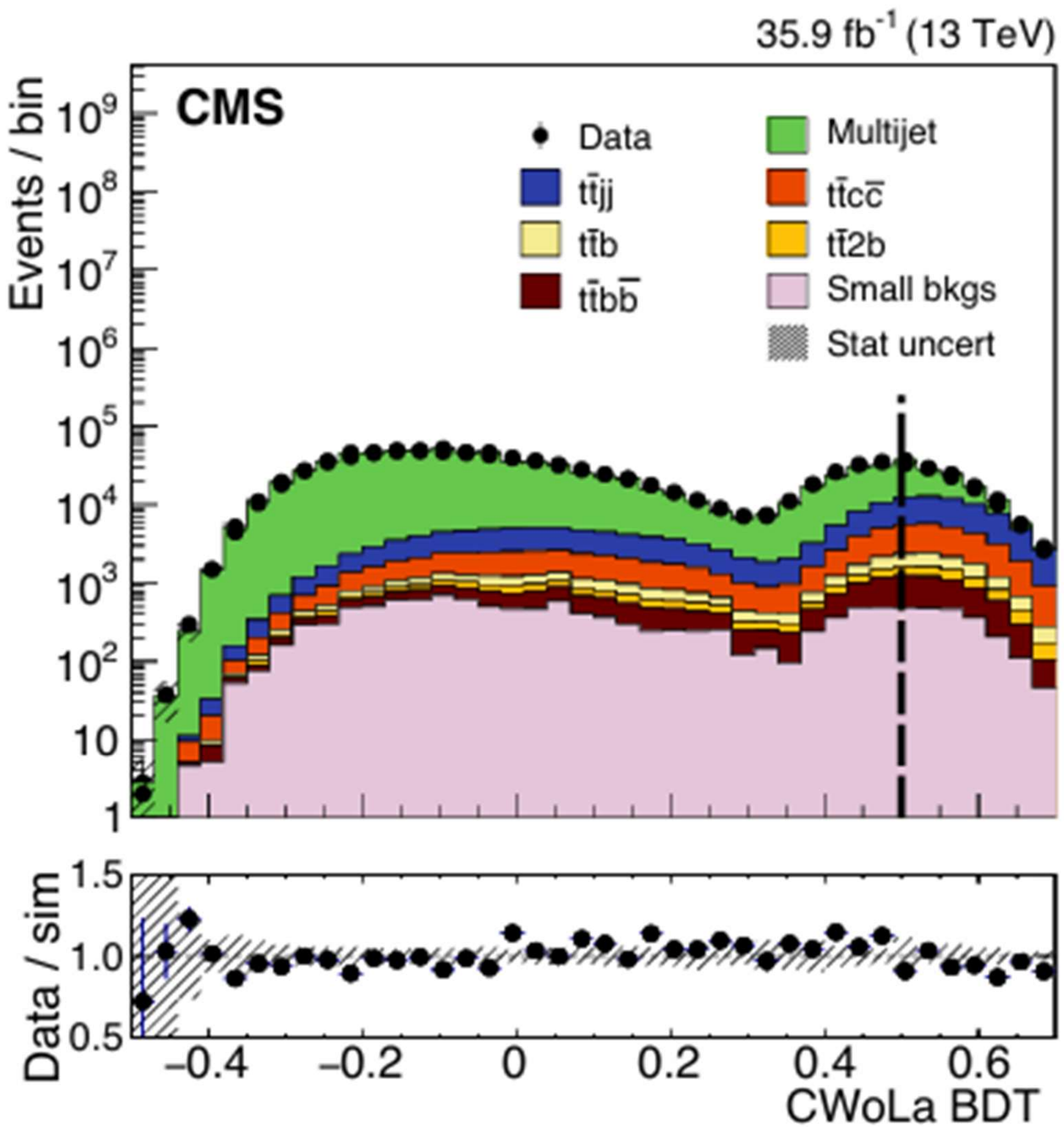


Real time Anomaly Detection



An event selected by an autoencoder-based anomaly detection hardware triggering algorithm in the CMS Experiment, [https://cds.cern.ch/record/2876546]

CWoLa in action: tt+bb



➢ Multijet background is hard to model – learn a classifier directly from data using jet substructure to make two samples and use jet kinematics to train the CWoLa classifier

Phys. Lett. B 803 (2020) 135285

Other Unsupervised Bump Hunts

- **ANODE**: interpolates probability densities from sidebands to the signal-region & constructs likelihood ratio [Nachman, Shih: 2001.04990]
- **CATHODE**: samples from the background model in signal region after interpolating and estimates likelihood ratio with classifier [Hallin et al: 2109.00546]
- **SALAD**: Reweight simulation to match sidebands, then interpolate into the signal region and use a second classifier to get the likelihood ratio [Andreassen et al: 2001.05001]
- **CURTAINS**: Train an invertible neural network conditioned on mass to map between sidebands [Raine et al: 2203.09470]
- **FETA**: Map simulation to data in sidebands, then compare to SR data [Golling et al: 2212.11285]