Generative Models for Natural Science

Fedor Ratnikov

NRU Higher School of Economics,
Yandex School of Data Analysis

Generative Model. ML Perspective

- Generative models look very different from regression/classification models
  - actually they are not that different
- Consider set of objects each of which is described by a vector of parameters
  - we arbitrary split this vector into “features” x and “labels” y
- For classification/regression problem we search for deterministic function f which approximates dependency y from x: y=f(x)
  - in probabilistic approach we search for probability p(ylx)
- For generation problem we want to sample objects for a given label
  - we search for probability p(xly)
    - y for generative model is called “condition”
    - condition may be absent - unconditional generative model

Generative Model. ML Perspective

- In both discriminative model and generative model we want to get probability for subset of object parameters conditioned by another subset of object parameters
- Discriminative models:
  - evaluate distributions for few, usually redundant, parameters conditioned by many features
    - can discriminate basing on this parameters
- Generative models:
  - evaluate many features conditioned by few parameters (conditions)
    - can sample these features
- NB: logistic regression + binomial distribution = generative model
  - for the binary objects

Approaches

Generative models [edit]

Types of generative models are:

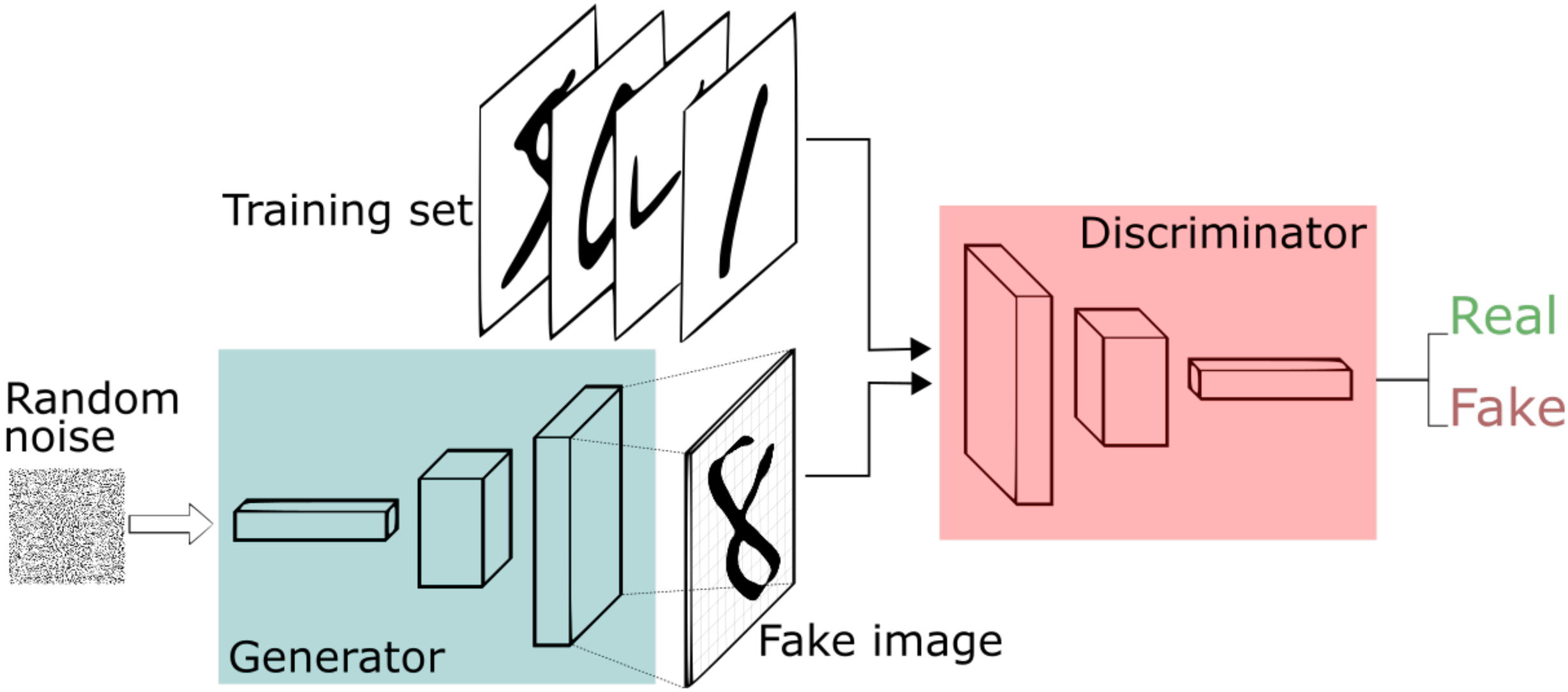
- Gaussian mixture model (and other types of mixture model)
- Hidden Markov model
- Probabilistic context-free grammar
- Bayesian network (e.g. Naive bayes, Autoregressive model)
- Averaged one-dependence estimators
- Latent Dirichlet allocation
- Boltzmann machine (e.g. Restricted Boltzmann machine, Deep belief network)
- Variational autoencoder
- Generative adversarial network
- Flow-based generative model

WIKIPEDIA
The Free Encyclopedia

◇ GAN and VAE are mostly used nowadays for generating complicated objects

GAN

https://medium.freecodecamp.org/an-intuitive-introduction-to-generative-adversarial-networks-gans-7a2264a81394



- Implicit p(xly), sampling only

JS GAN, WGAN

- GAN minimizes Jensen-Shannon divergence between real and generated objects

$$\max_D \mathbb{E}_{\hat{y}\sim p(\hat{y})}(1 - \log D(\hat{y})) + \mathbb{E}_{y\sim p(y)} \log D(y)$$
$$\min_G \mathbb{E}_{\epsilon\sim p(\epsilon)}[-\log D(G(\epsilon))]$$

arXiv:1406.2661

  - vanishing gradients for too strong discriminator
  - mode collapse

- WGAN minimizes Wasserstein distance between distributions

$$W(\mathbb{P}_r, \mathbb{P}_g) = \inf_{\gamma\in\Pi(\mathbb{P}_r,\mathbb{P}_g)} \mathbb{E}_{(x,y)\sim\gamma}\left[\, \|x - y\| \,\right]$$

arXiv:1701.07875

  - that may be converted into search for the discriminator function f(x)

$$W(p_r, p_g) = \frac{1}{K} \sup_{\|f\|_L\leq K} \mathbb{E}_{x\sim p_r}[f(x)] - \mathbb{E}_{x\sim p_g}[f(x)]$$

$$\max_D \; \mathbb{E}_{\hat{y}\sim p(\hat{y})}[D(\hat{y})) - \mathbb{E}_{y\sim p(y)}[D(y)] + \lambda\mathbb{E}_{\tilde{y}\sim p(\tilde{y})}[(||\nabla_{\tilde{y}} D(\tilde{y})||_2 - 1)^2]$$
$$\min_G \; \mathbb{E}_{\epsilon\sim p(\epsilon)}[-D(G(\epsilon))]$$

CramerGAN

- WGAN produces biased gradients, that makes converging slower, sometimes never reaching optimum
- CramerGAN tunes critic (discriminator) as inspired by the Cramer Metric:

arXiv:1705.10743

$$l_2^2(P, Q) := \int_{-\infty}^{\infty} (F_P(x) - F_Q(x))^2 \mathrm{d}x$$

  - in multidimensional case it is expressed as an energy distance:

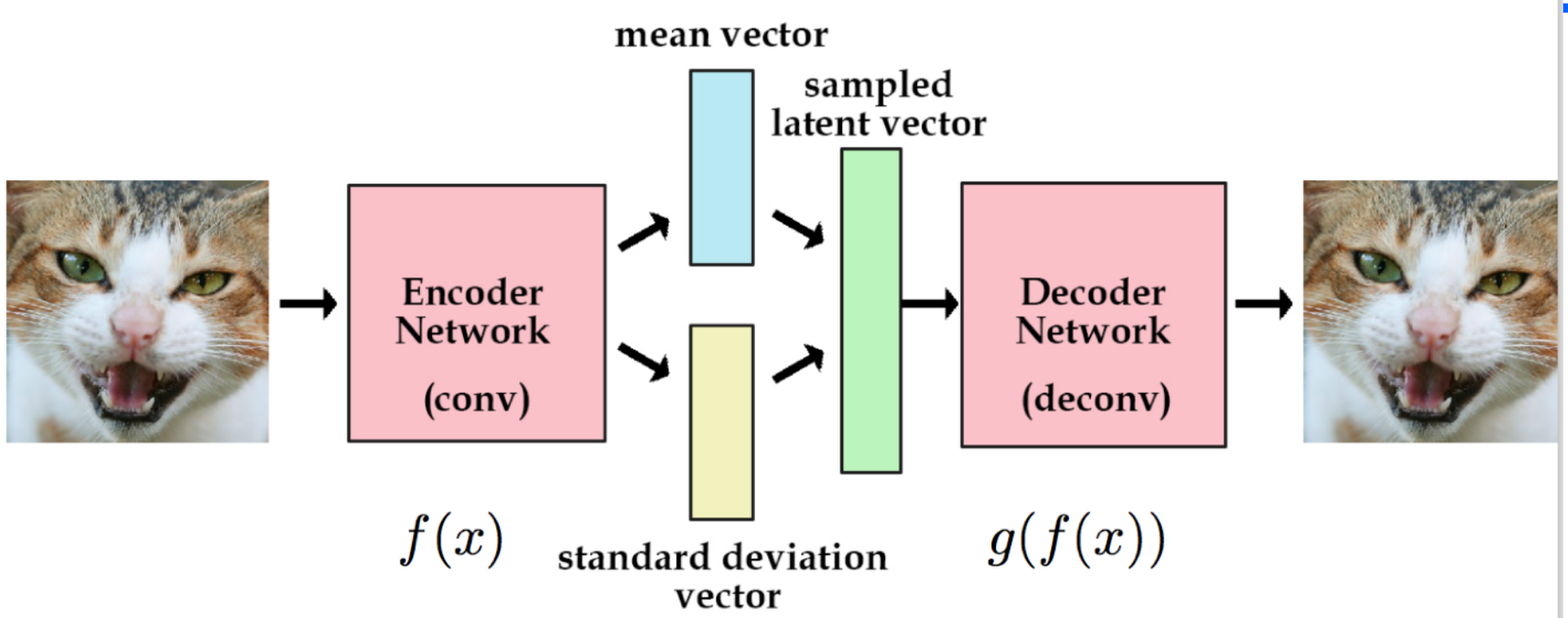
$$\mathcal{E}(X, Y) := 2\,\mathbb{E}\,\|X - Y\|_2 - \mathbb{E}\,\|X - X'\|_2 - \mathbb{E}\,\|Y - Y'\|_2$$

  - where X, X', Y, Y' are statistically independent samples from two distributions
    - generator loss is therefore more complicated:

$$L_g = \|h(x_r) - h(x_g)\|_2 + \|h(x_r) - h(x'_g)\|_2 - \|h(x_g) - h(x'_g)\|_2$$

- CramerGAN demonstrates better convergency indeed

Variational Autoencoder



- We want to sample from latent space
- Split into mean and standard deviation
- Add penalty term (Kullback-Leibler divergence) so mean/std are close to unit Gaussian

kvfrans
towardsdatascience.com

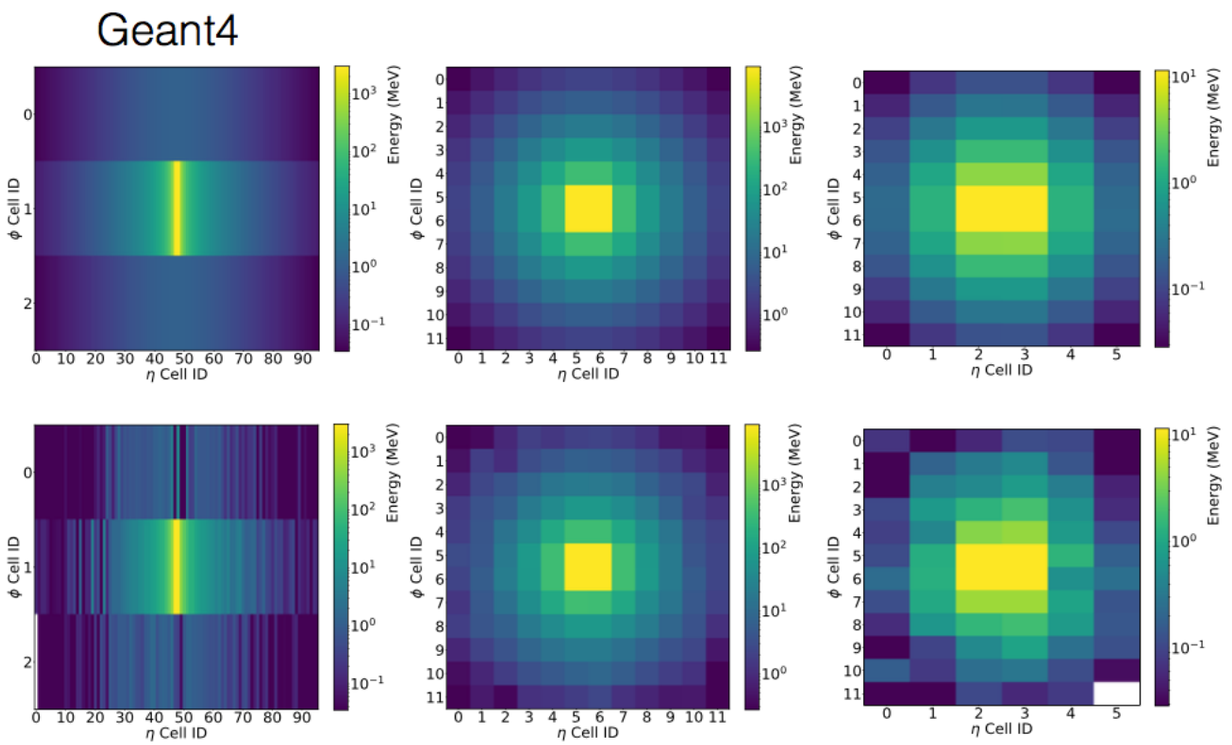
79

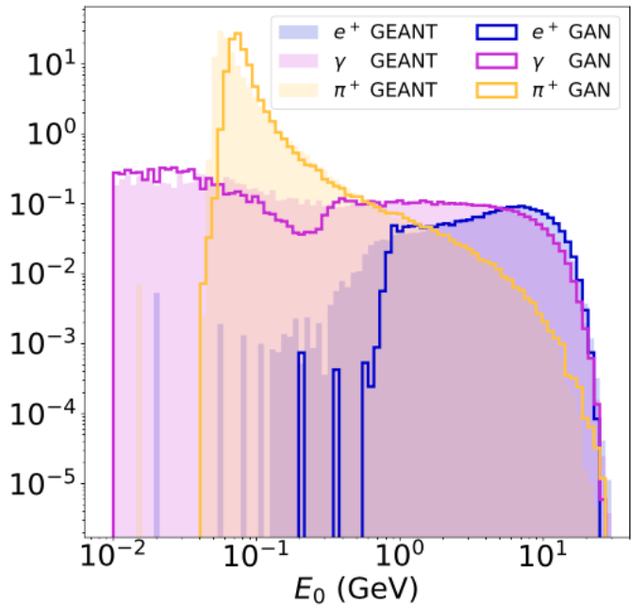
- VAE allows calculate p(x|y)
  - NB: GAN only allows sampling from p(x|y)
- … but smaller size of latent dimensions
  - blurry objects

Library Approach

- We have train sample for the generative model
  - consistency with this train sample is a figure of merit for the generative model
- Objects of the train sample may be used for generation directly
  - remember KNN classification algorithm
    - k=1 - straightforward
      - the only drawback - search for the object with appropriate conditions in the (presumably huge) data library
    - k>1 - problem to interpolate between objects
      - short distance objects interpolation, more robust than global generation
- NB: this approach by construction uses full information which is contained in the training sample

CaloGAN

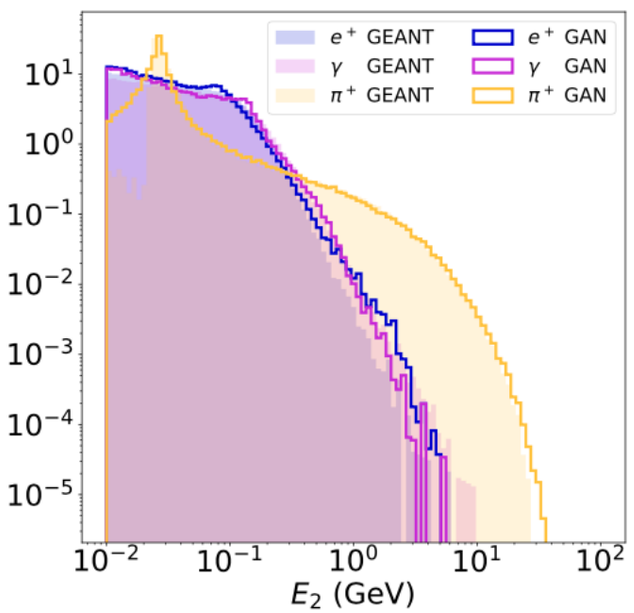




CaloGAN

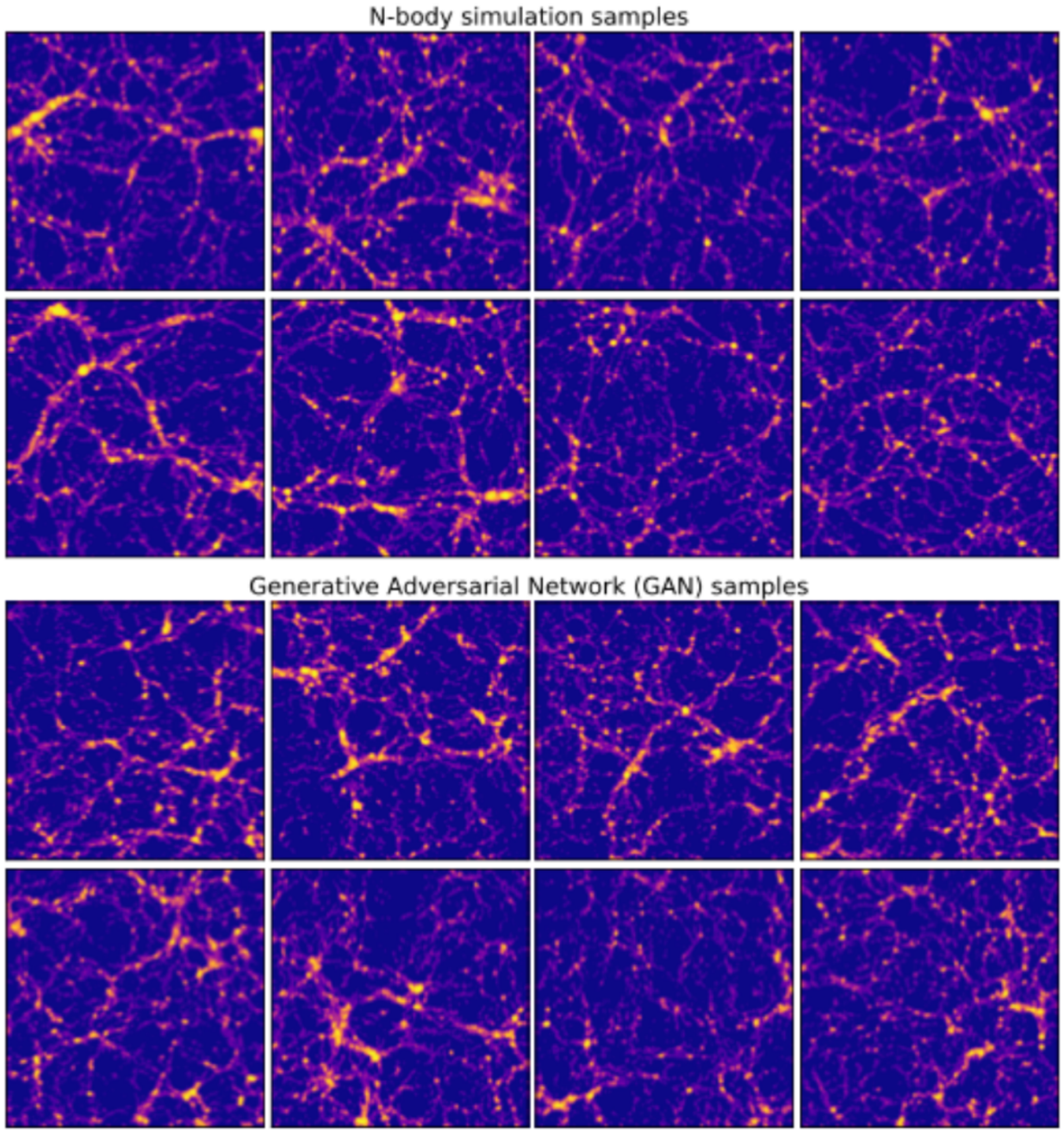
Generation Method	Hardware	Batch Size	milliseconds/shower
GEANT4	CPU	N/A	1772 ←
CALOGAN	CPU	1	13.1
		10	5.11
		128	2.19
		1024	2.03
	GPU	1	14.5
		4	3.68
		128	0.021
		512	0.014
		1024	0.012 ←

94



- Distributions look similar.
  - Are they similar enough? What is “enough”?

Use in Cosmology



Rodríguez et al,

arXiv:1801.09070

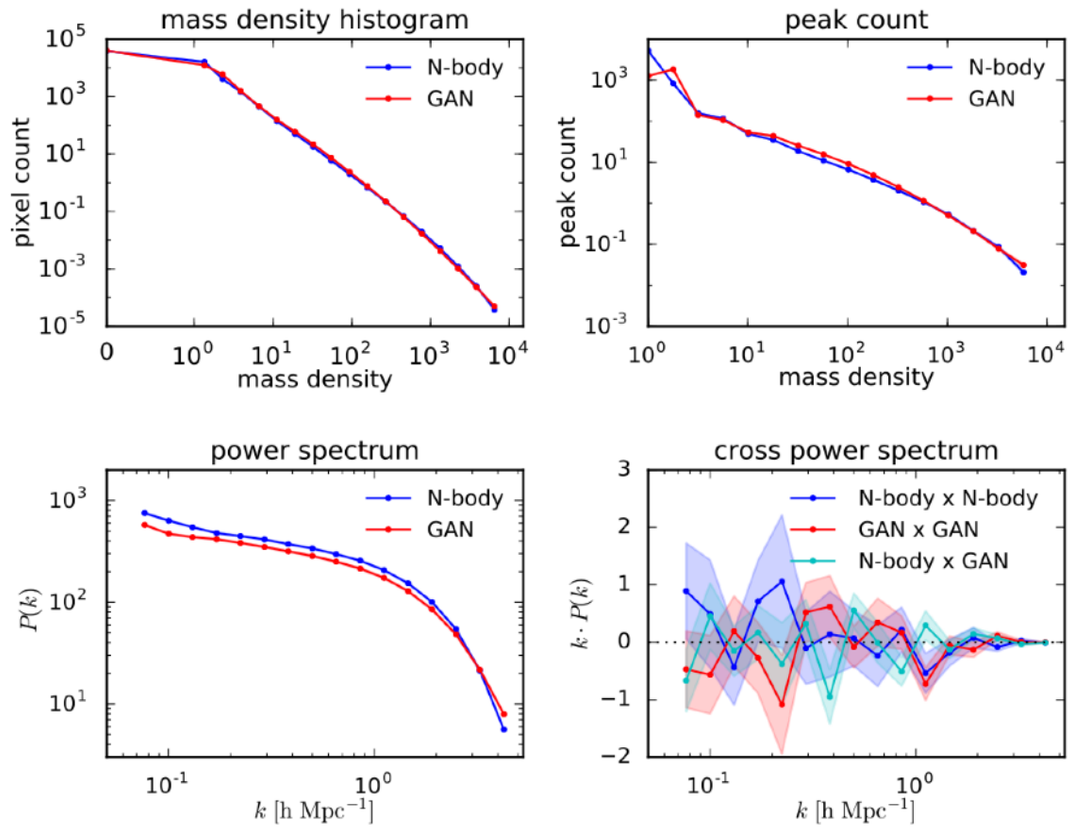
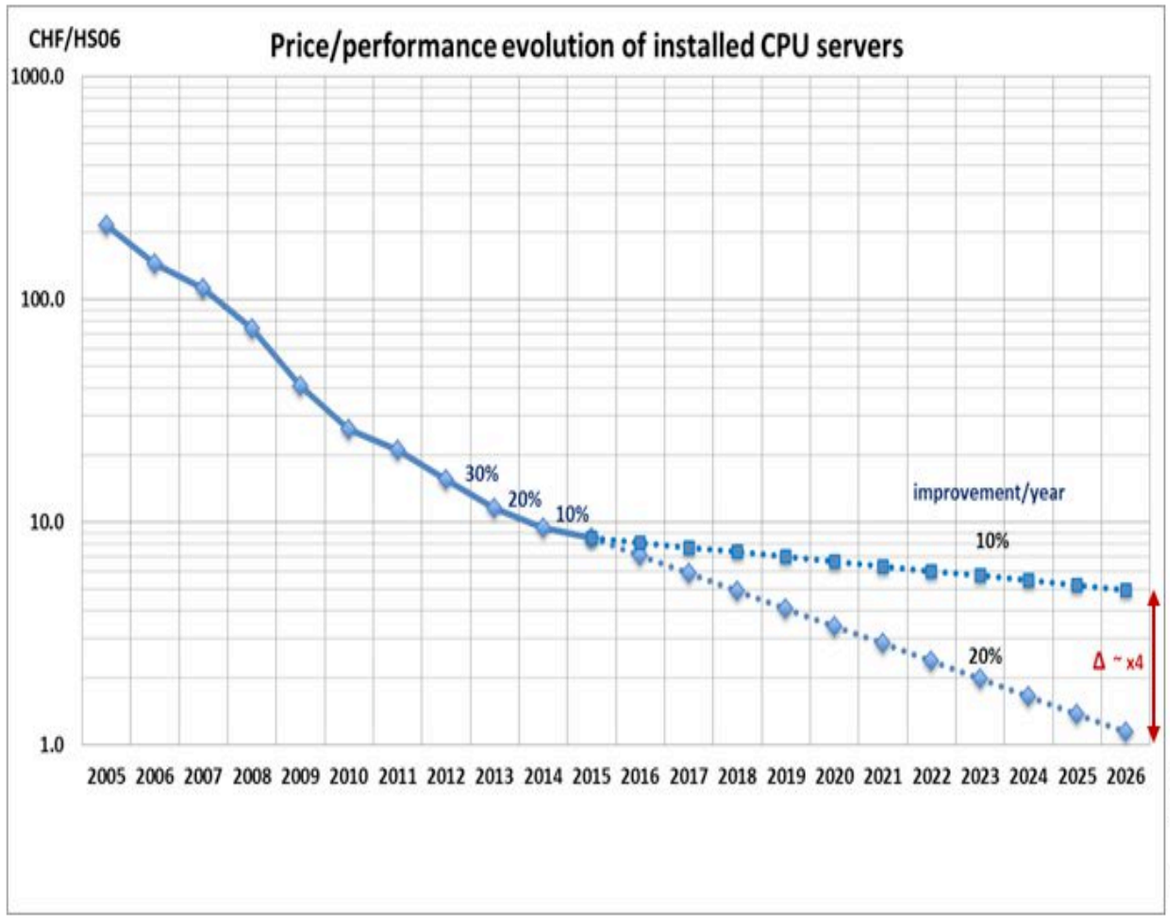


Figure 3: Samples from N-body simulation (top two rows) and from GAN (bottom two rows) for the box size of 100 Mpc. In this figure, transformation S1 with $k = 7$ was applied.

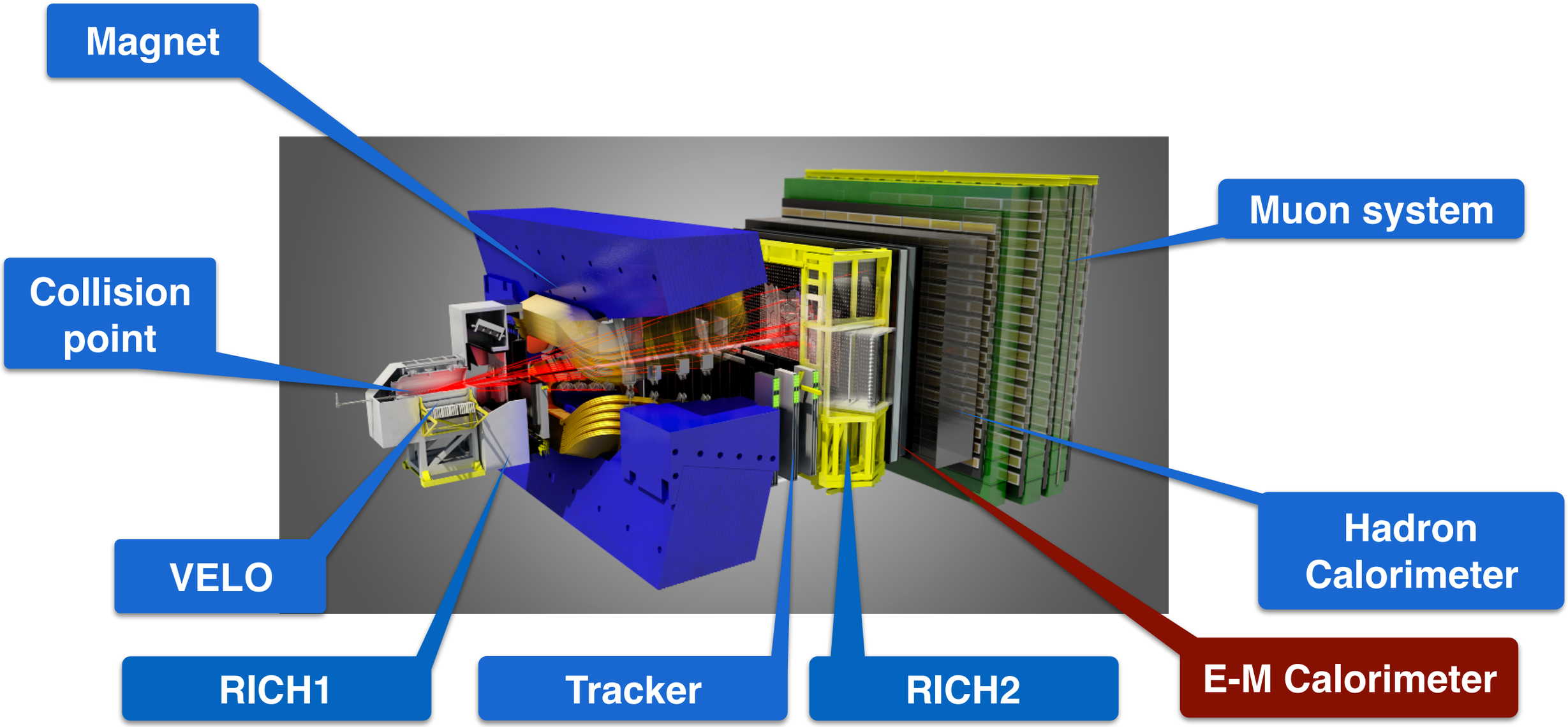
- What are systematics in physics results due to the generator imperfection?

Generative Models at LHC

- About 80% of computing resources are used for MC simulation in HEP experiments
  - Calorimeter simulation is one of bottlenecks
  - RICH is the next in the row for LHCb detector
    - > 85% of simulation is taken by these
- Can not expect exponential rise of CPU performance
- Need work around for Run3 and HL-LHC
- Generative models trained on the detailed GEANT simulation may be a solution



Example: Fast Simulation of the ECAL Response

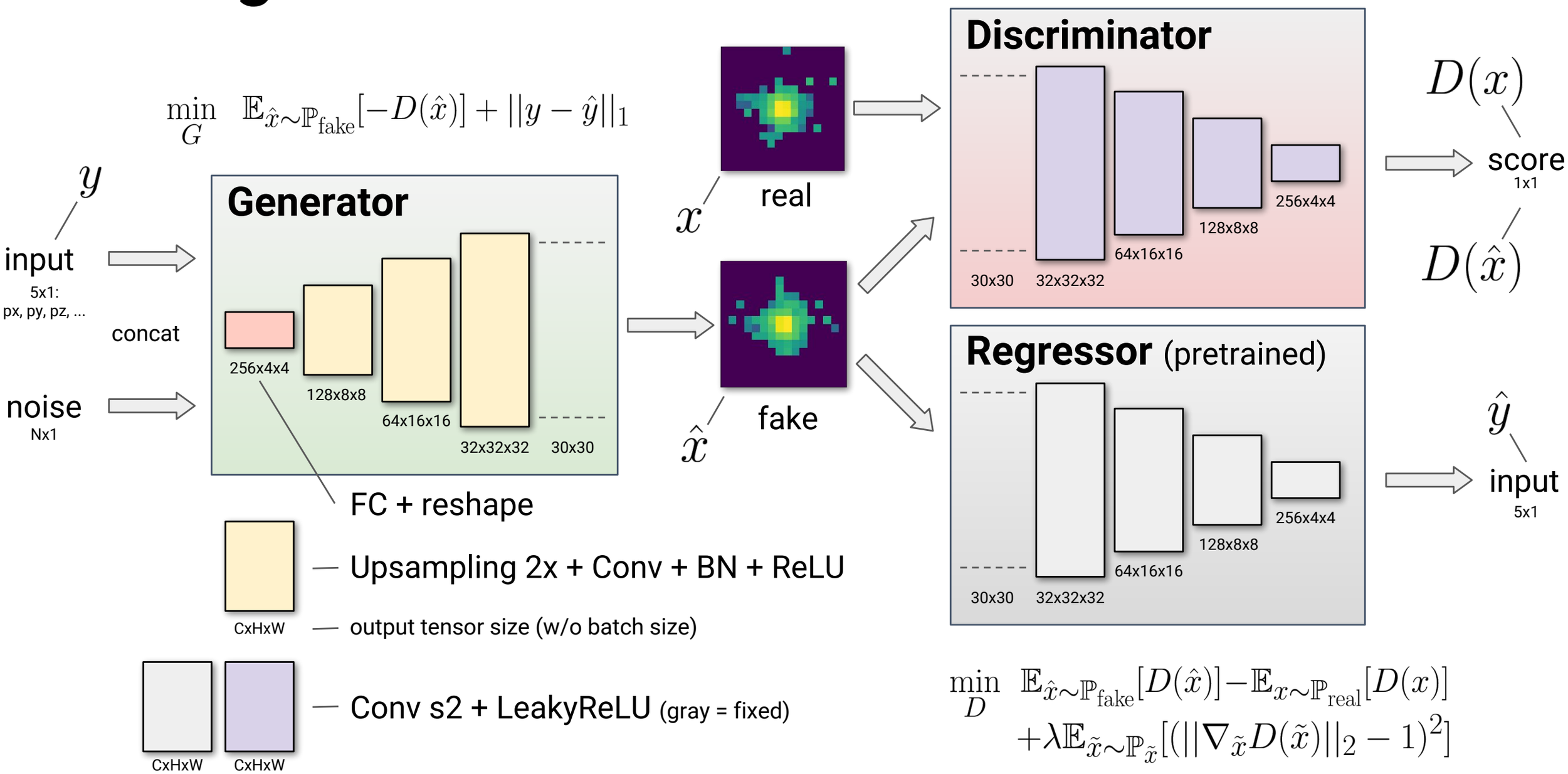


- ECAL takes the most time in the LHCb event simulation

Example: ECAL Fast Simulation

Training scheme

$$\min_G \quad \mathbb{E}_{\hat{x}\sim\mathbb{P}_{\text{fake}}}[-D(\hat{x})] + ||y - \hat{y}||_1$$



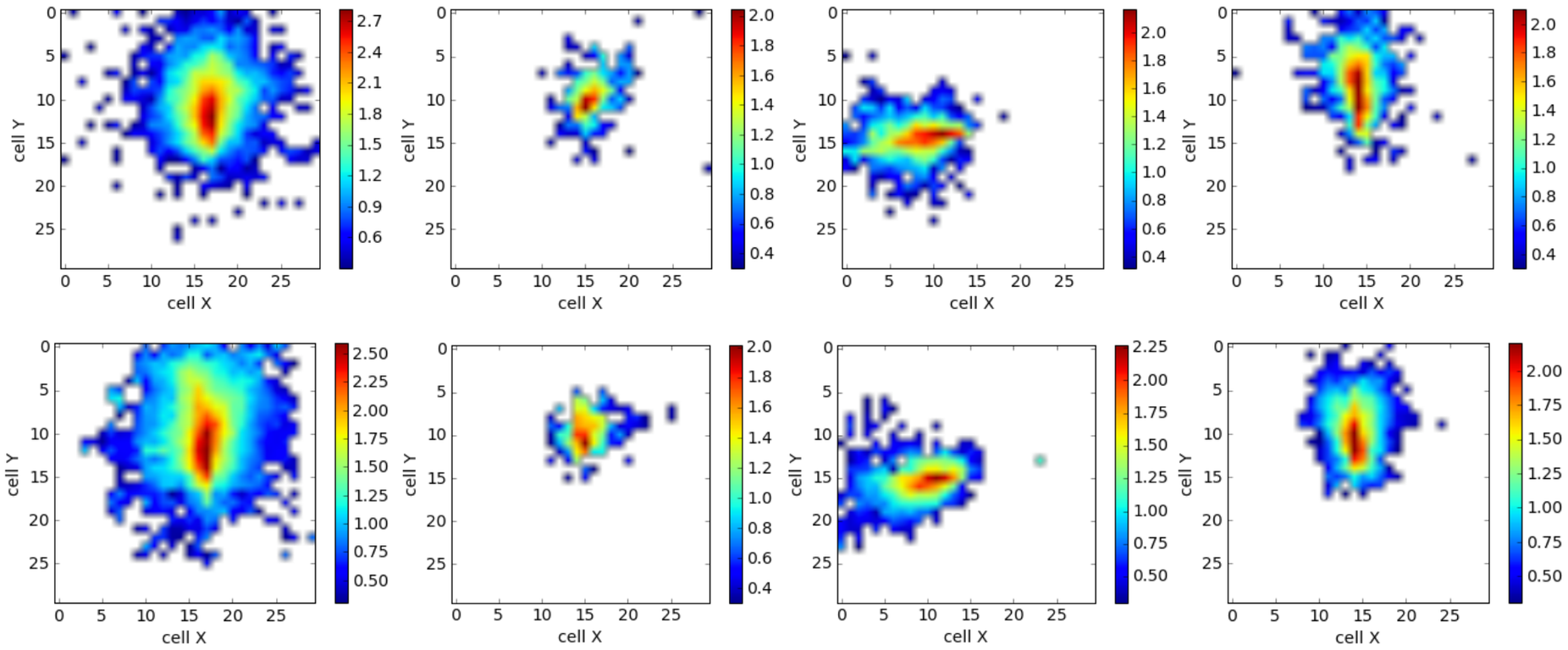
$$\min_D \quad \mathbb{E}_{\hat{x}\sim\mathbb{P}_{\text{fake}}}[D(\hat{x})] - \mathbb{E}_{x\sim\mathbb{P}_{\text{real}}}[D(x)] + \lambda\mathbb{E}_{\tilde{x}\sim\mathbb{P}_{\tilde{x}}}[(||\nabla_{\tilde{x}}D(\tilde{x})||_2 - 1)^2]$$

LHCb ECAL Simulation

GEANT Simulated

$\log_{10}$(cell energy)

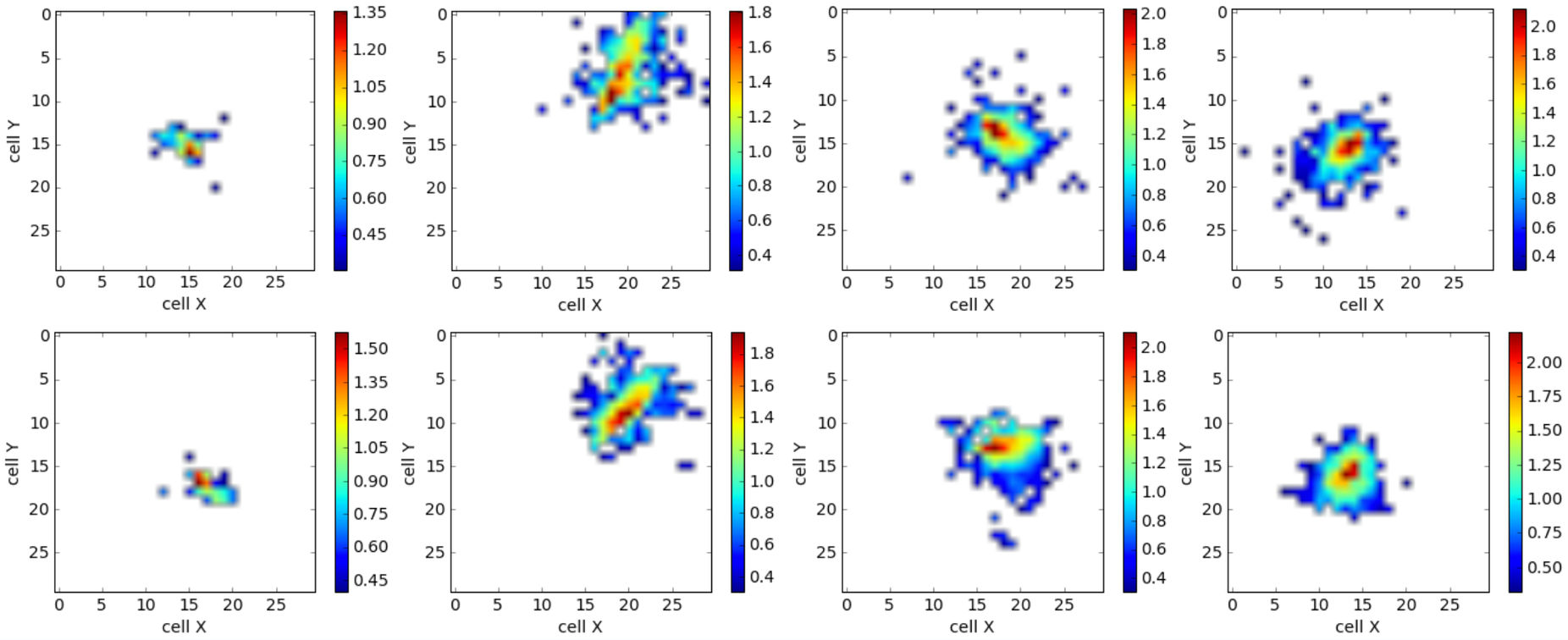
GAN Generated



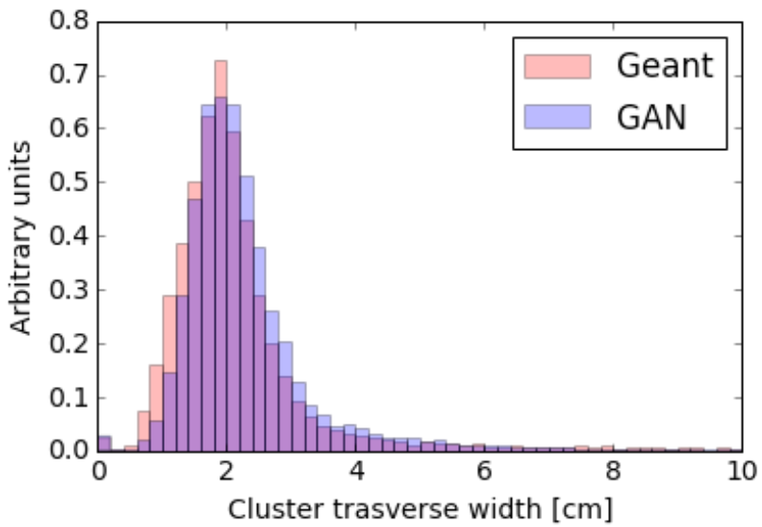
GEANT Simulated

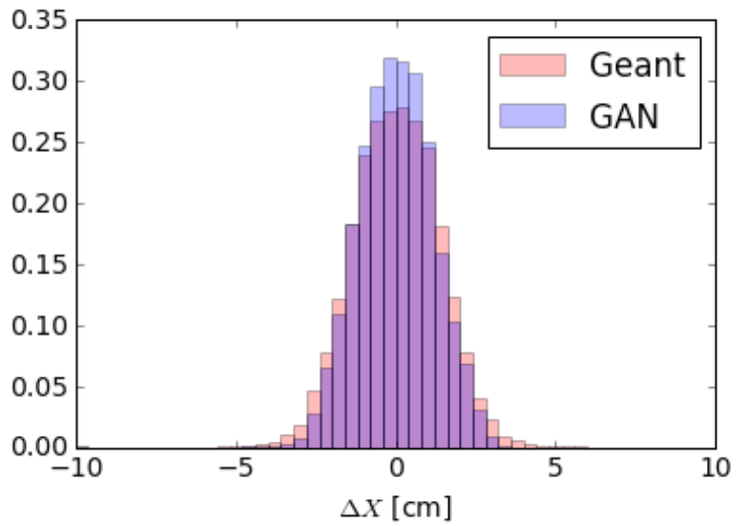
$\log_{10}$(cell energy)

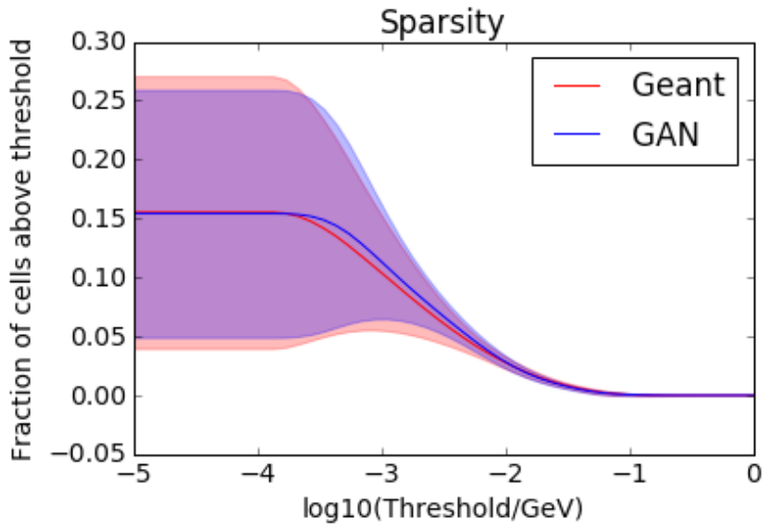
GAN Generated

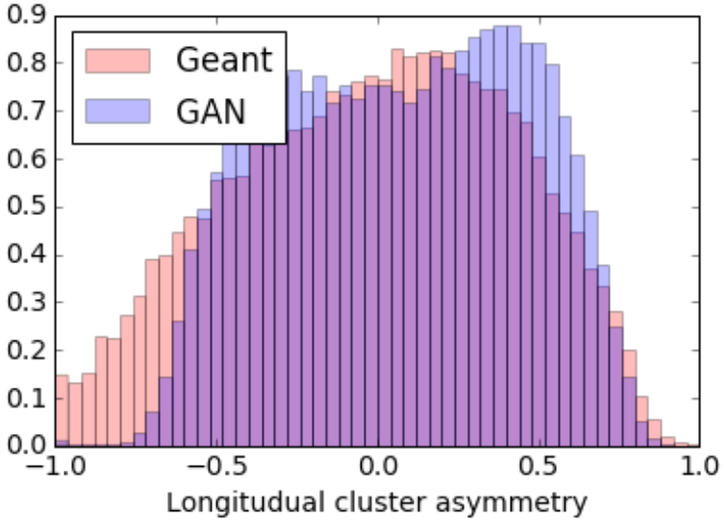


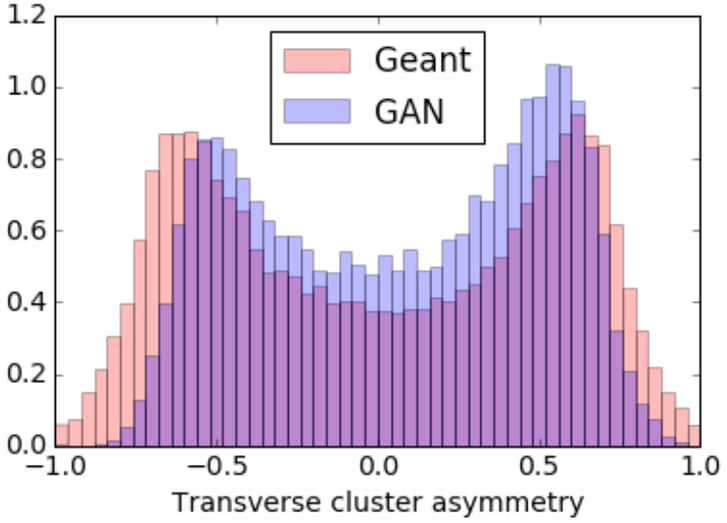
Primary and Marginal Distributions











- Is hard to fit marginal distributions
  - unless the model is aware that those are important for us

Natural Requirements

- For image generation we are usually happy if the result looks like it is desired
- In science we need the result to reasonably well match the given set of requirements. This target set is driven by scientific considerations to reach the ultimate scientific goal
  - e.g. we could want $E^2-p^2=m^2$ for generated particles
- Explicit control to satisfy requirements is preferable
  - e.g. exclude E from generated features, set it explicitly from generated p

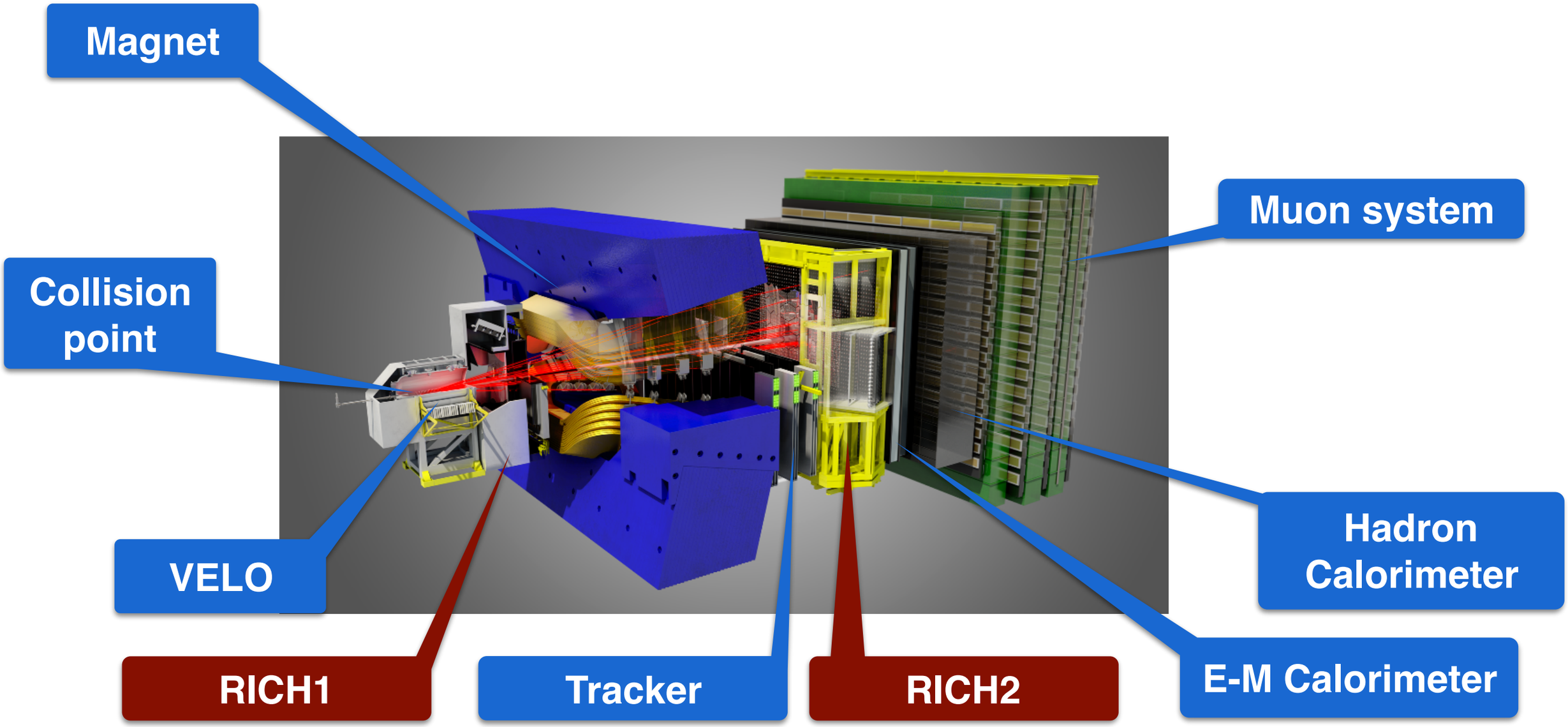
Enforcing Important Statistics

- No generative model is ideal
  - some deviations from the original distribution remain
- Model tends to learn primary statistics of generated objects
- In physics applications we mostly need our model to learn some particular statistics which may be marginal to the generated object
  - e.g. cluster shape fluctuations for fast calorimeter simulation
- Can enforce these statistics by explicit adding them to the los
  - can’t we?

Enforcing Important statistics

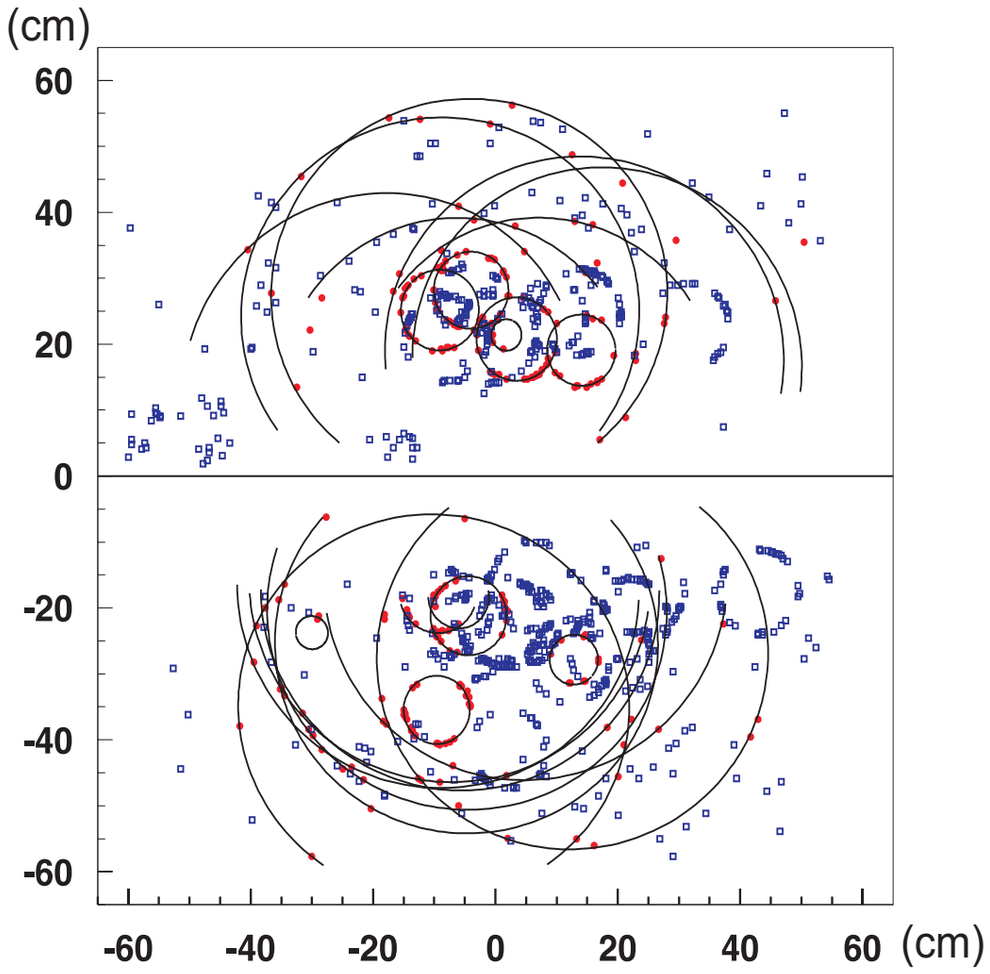
- Can enforce statistics by explicit adding them to the los
  - can’t we?
- By adding statistics into the loss we do enforce match for these statistics
  - most likely by the price of overtraining these particular statistics
    - … and we lose handle to validate quality of generator on this statistics
- Still can remove those statistics from loss, and see how far they would deviate
  - figure of merit for generating this statistics

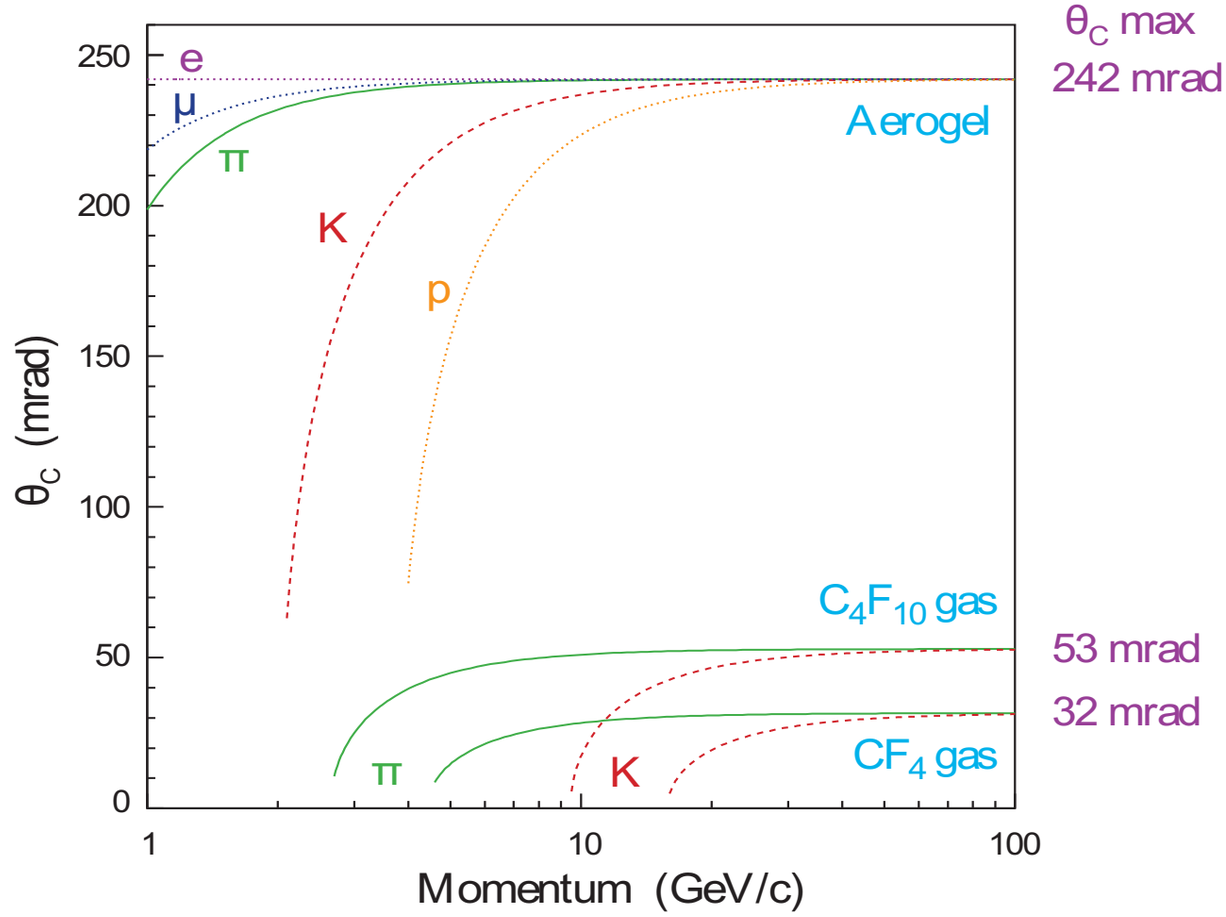
Example: RICH-based particle ID



- Ring Image CHerenkov detector

RICH Basics





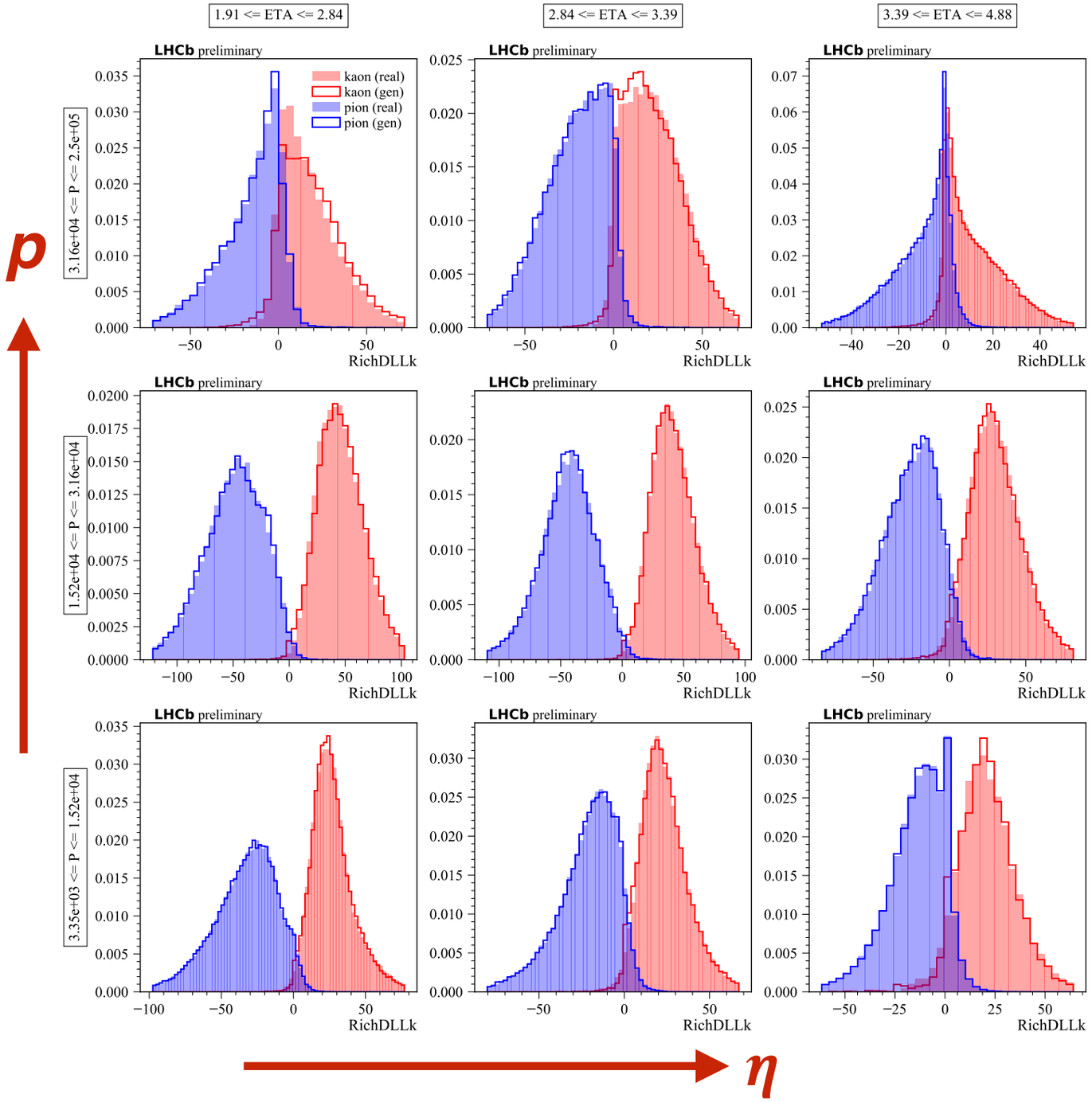
- RICH response is used to identify particles
  - e.g. separate pions and kaons

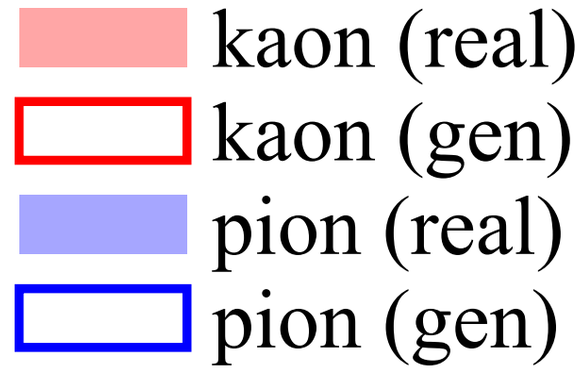
RICH ID Simulation

- Accurate RICH simulation involves:
  - tracing the particles through the radiators
  - Cherenkov light generation
  - photon propagation, reflection, refraction and scattering
  - Hybrid Photon Detector (photo-cathode + silicon pixel) simulation
- These require significant computing resources

- Besides:
  - quality of obtained simulated ID variables is not satisfactory when comparing to calibration data samples
- Let's use ML:
  - train generative model to directly convert track kinematics into ID variables
    - can train directly on calibration data samples

RICH ID Separation

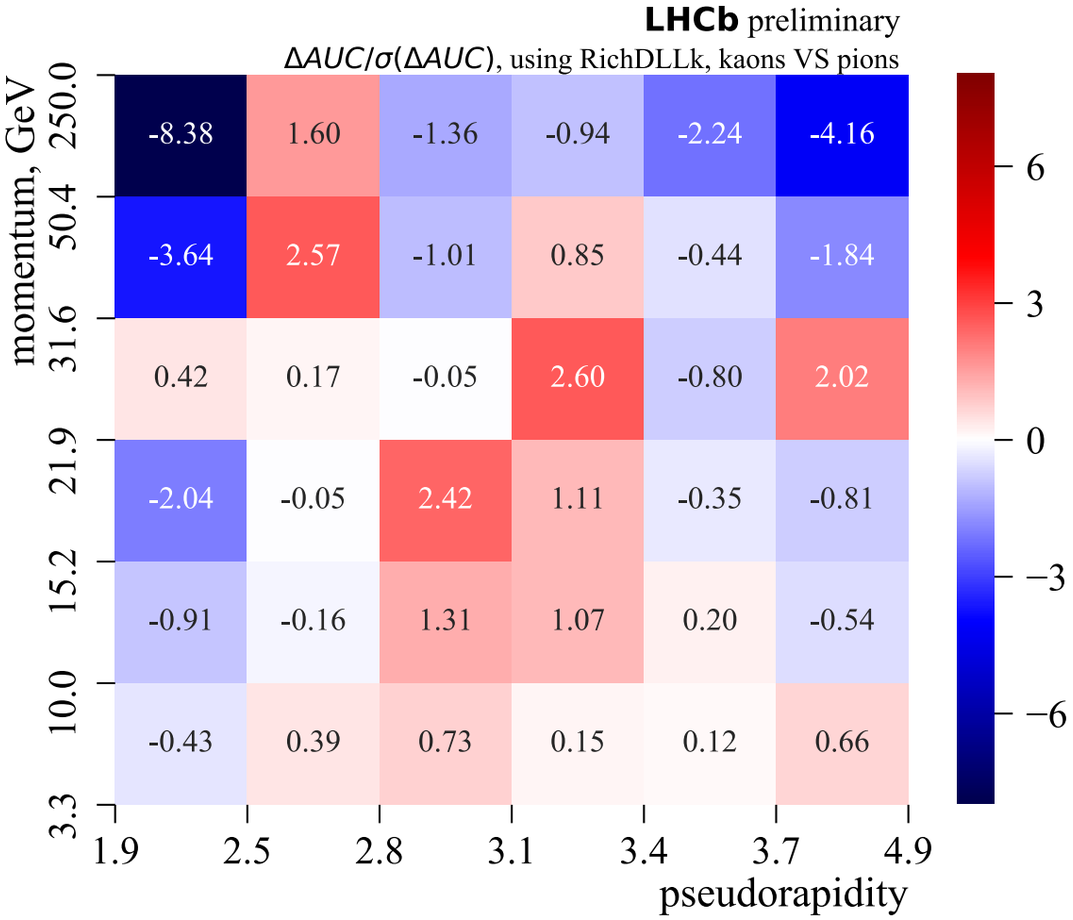




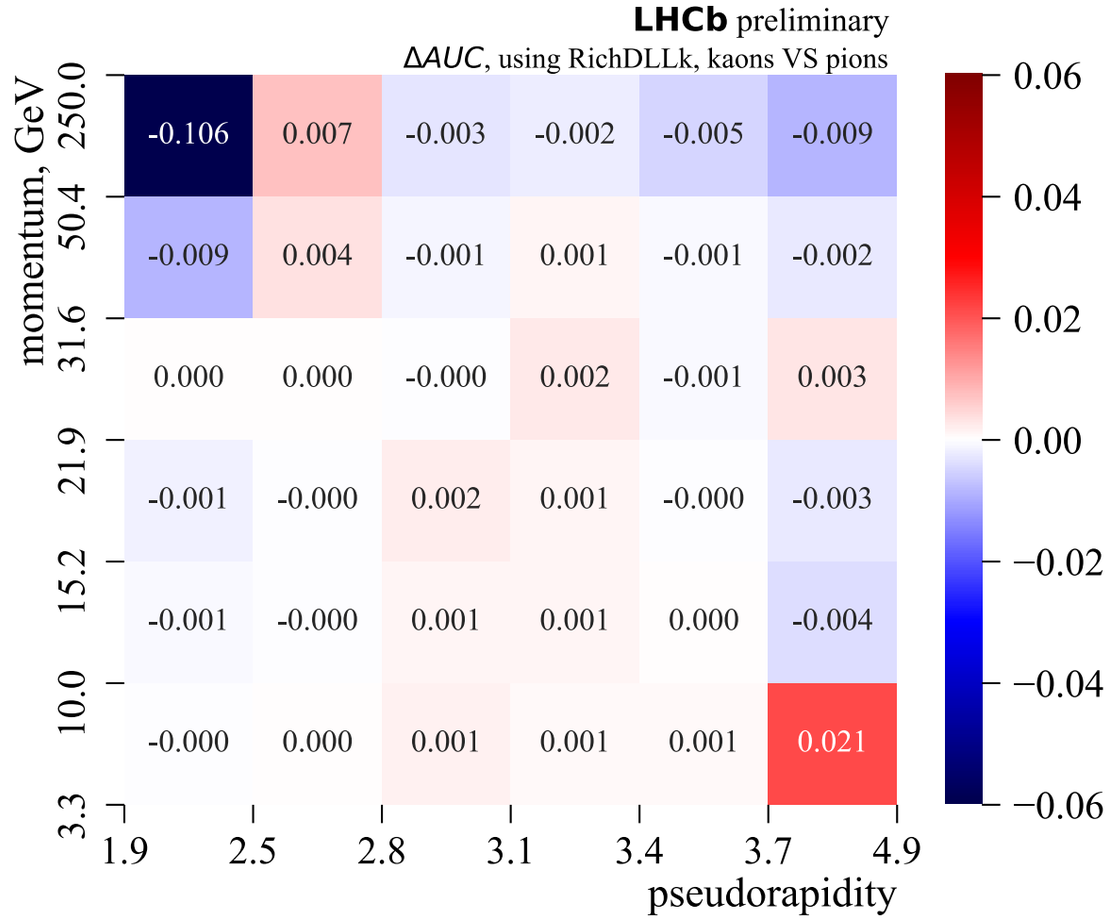
- Is this generation quality good or bad?
  - depends on what it is used for...

Comparing Separation Power

$(AUC_{REAL}\text{-}AUC_{GEN})/\sigma_{AUC}$



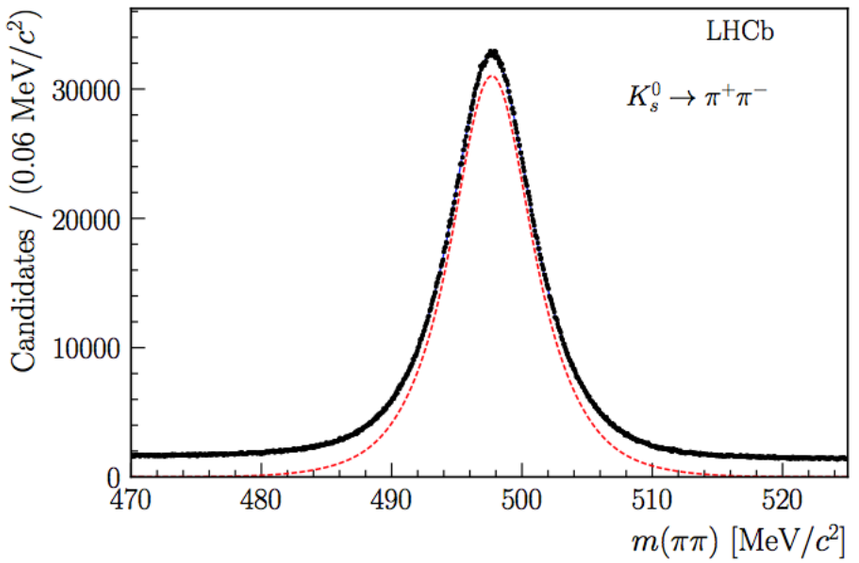
$AUC_{REAL}\text{-}AUC_{GEN}$



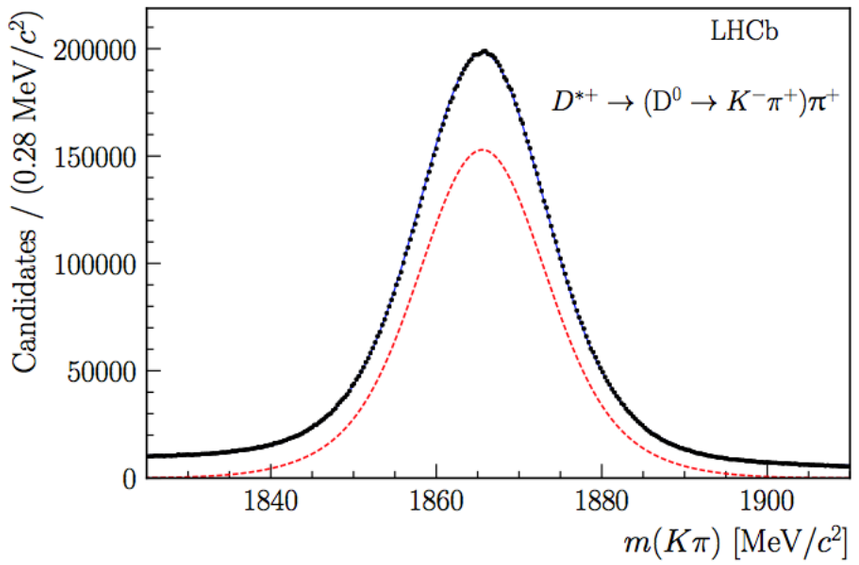
- The final figure of merit is driven by particular physics analysis using this generated ID
  - FOM is as accurate as can be evaluated from available calibration statistics
    - fundamental limitation

Generative Models Trained on Real Data

- Real data samples, even calibration, are never 100% clean
  - contamination from events with different labels/conditions

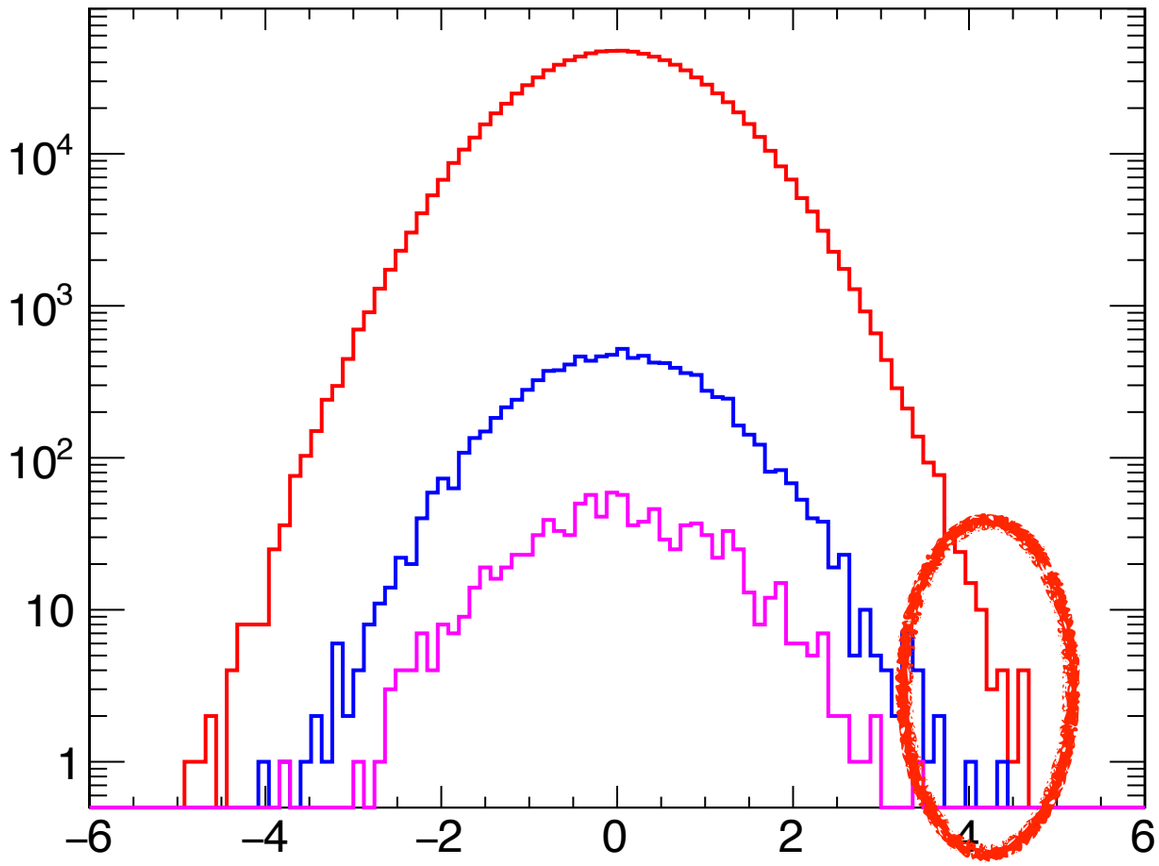


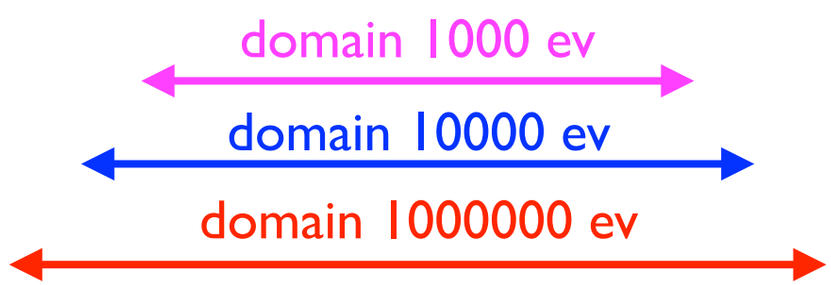
- Can not determine label of particular object uniquely
  - however can statistically determine fractions of different labels



- Can use weighted samples to train WGAN and CramerGAN

Completeness





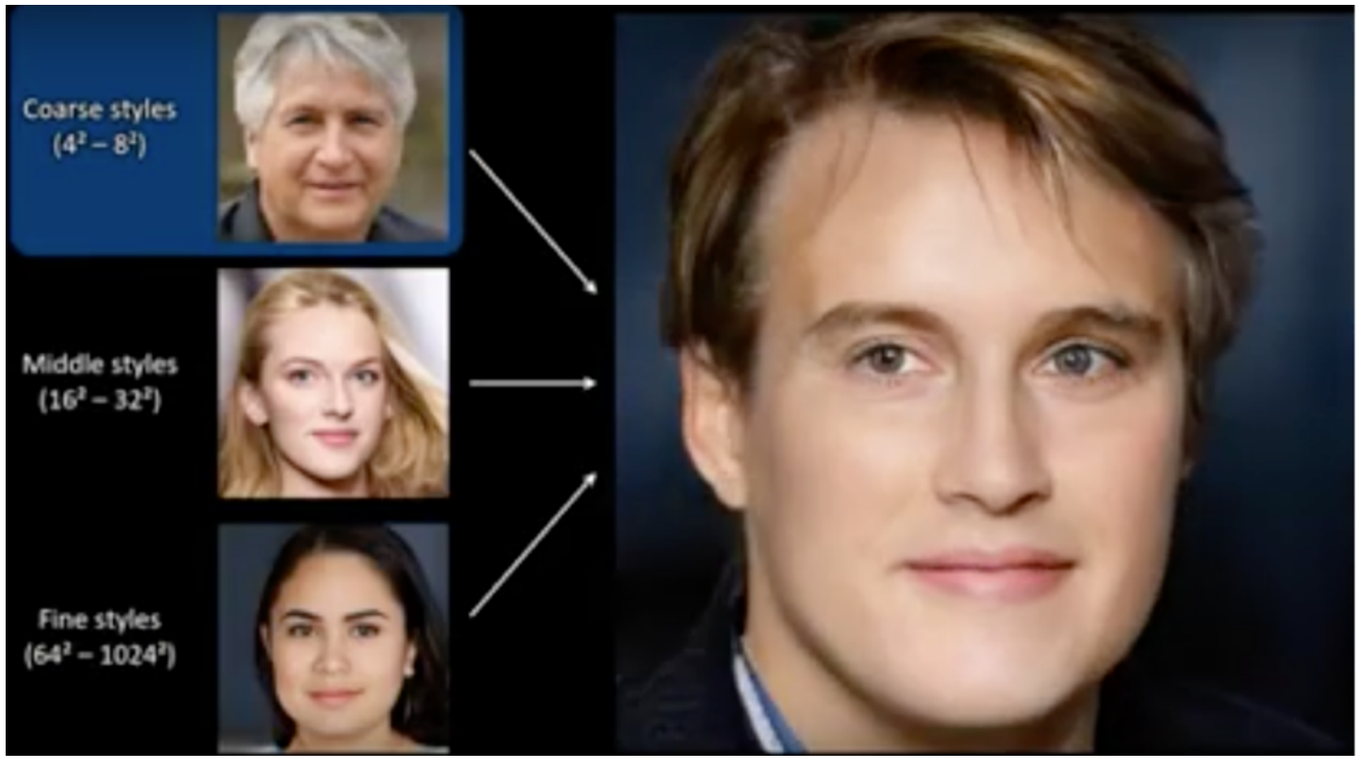
- Domain for the generative model is driven by the training sample
- model can not extend beyond the train domain even if produces high statistics
  - until explicitly set to behave beyond train domain

Decomposition

- Quality of the generative models is limited by the size of the train data sample
  - generative models may not give profit for producing statistically correct big data sets
    - no information beyond the train sample is available

Decomposition

- Quality of the generative models is limited by the size of the train data sample
  - generative models may not give profit for producing statistically correct big data sets
    - no information beyond the train sample is available



- Not quite if we can decompose generative model into separate components
  - random combinations of different components may drastically increase variativity

Decomposition

- Quality of the generative models is limited by the size of the train data sample
  - generative models may not give profit for producing statistically correct big data sets
    - no information beyond the train sample is available
- Not quite if we can decompose generative model into separate components
  - random combinations of different components may drastically increase variativity
- E.g. fast simulation of the calorimeter response
  - generator is trained on 10^6 incident particles
  - ~50 particles in the calorimeter per event
  - total variativity $\sim(10^6)^{50} = 10^{300}$!

Quality Metric

- No generative model is ideal
  - some deviations from the original distribution remain
- Minor deviations are not that important e.g. for image generation
- Minor deviations may be a big deal for physics generative models
  - e.g. we could want $E^2-p^2=m^2$ for generated particles to be precise
- Ultimate generative model quality metric is comparing final physics result obtained using generative model, and the one obtained using train data
  - accuracy is limited by the size of the train data

Enforcing Quality

- No generative model is ideal
  - some deviations from the original distribution remain
- Model tends to learn primary statistics of generated objects
- In physics applications we mostly need our model to learn some particular statistics which may be marginal to the generated object
  - e.g. cluster shape fluctuations for fast calorimeter simulation
- Can enforce these statistics by explicit adding them to the los
  - can’t we?

Enforcing Quality

- Can enforce statistics by explicit adding them to the los
  - can’t we?
- By adding statistics into the loss we do enforce match for these statistics
  - most likely by the price of overtraining these particular statistics
    - … and we lose handle to validate quality of generator on this statistics
- Still can remove those statistics from loss, and see how far they would deviate
  - figure of merit for generating this statistics

Conclusions

- Surrogate generative models demonstrate extraordinary progress in current years
- There are many applications for use in natural science research
- Generative models need attention ensure scientifically solid results
  - completeness of generated sample
  - satisfying boundary conditions, control of scientifically important but marginal statistics
  - evaluating quality of the model, propagate model imperfections to systematic uncertainties of the final scientific result