Calorimeter Fast Simulation Using ML Approaches

2nd IML Workshop
10/04/2018

speaker: Egor Zakharov, on behalf of the team

NRU Higher School of Economics,
Skolkovo Institute of Science and Technology

Problem

- We want to speed up calorimeter simulation (calorimeter showers) while keeping reasonable simulation accuracy (correctly reproducing simulation behavior)
    - consider LHCb ECAL as a practical goal

Problem

- ➢ We want to speed up calorimeter simulation (calorimeter showers) while keeping reasonable simulation accuracy (correctly reproducing simulation behavior)
    - consider LHCb ECAL as a practical goal
- ➢ *IDEAL* ML problem formulation (fully observed model):

Problem

- We want to speed up calorimeter simulation (calorimeter showers) while keeping reasonable simulation accuracy (correctly reproducing simulation behavior)
    - consider LHCb ECAL as a practical goal
- *IDEAL* ML problem formulation (fully observed model):

x input

K variables:

px, py, pz, …

particle type, etc

Problem

- ➢ We want to speed up calorimeter simulation (calorimeter showers) while keeping reasonable simulation accuracy (correctly reproducing simulation behavior)
  - ○ consider LHCb ECAL as a practical goal
- ➢ *IDEAL* ML problem formulation (fully observed model):

x input

K variables:

px, py, pz, …

particle type, etc

Function

(ex. some DNN)

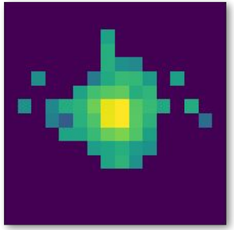
Problem

- ➢ We want to speed up calorimeter simulation (calorimeter showers) while keeping reasonable simulation accuracy (correctly reproducing simulation behavior)
  - ○ consider LHCb ECAL as a practical goal
- ➢ *IDEAL* ML problem formulation (fully observed model):

x **input**
K variables:
px, py, pz, …
particle type, etc

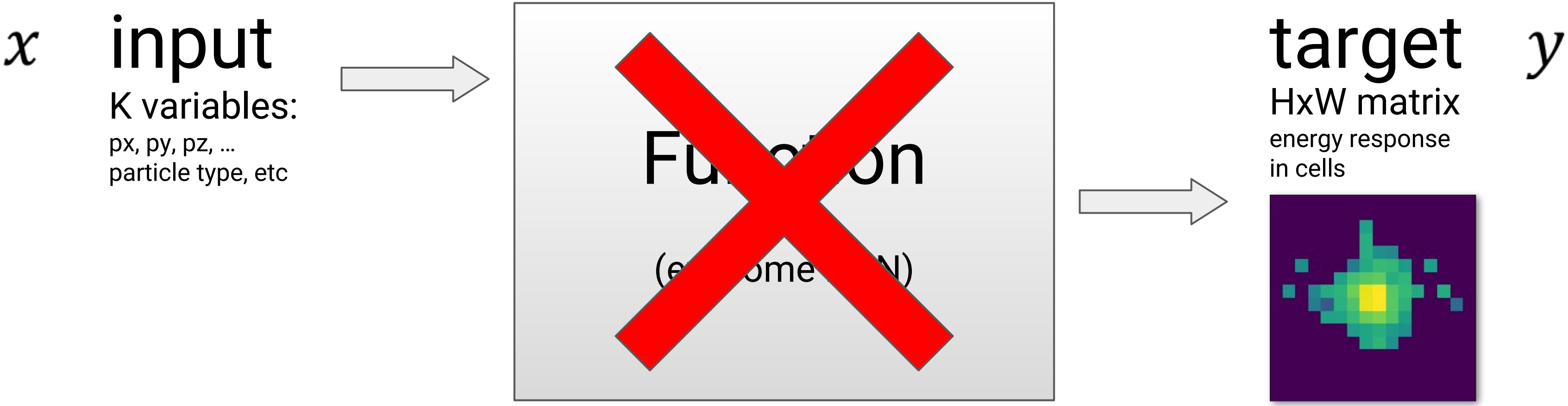
Function
(ex. some DNN)

target y
HxW matrix
energy response
in cells



Problem

- We want to speed up calorimeter simulation (calorimeter showers) while keeping reasonable simulation accuracy (correctly reproducing simulation behavior)
    - consider LHCb ECAL as a practical goal
- *IDEAL* ML problem formulation (fully observed model):



Problem

- We want to speed up calorimeter simulation (calorimeter showers) while keeping reasonable simulation accuracy (correctly reproducing simulation behavior)
    - consider LHCb ECAL as a practical goal
- Our ML problem formulation (hidden variables model):

x input

K variables:

px, py, pz, …

particle type, etc

Problem

- We want to speed up calorimeter simulation (calorimeter showers) while keeping reasonable simulation accuracy (correctly reproducing simulation behavior)
  - consider LHCb ECAL as a practical goal
- Our ML problem formulation (hidden variables model):

x input

K variables:
px, py, pz, …
particle type, etc

ε noise

L variables:
"hidden variables"

Problem

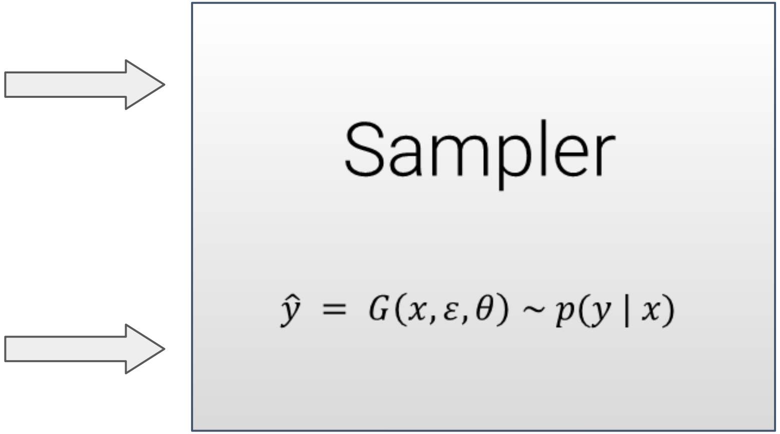
- ➢ We want to speed up calorimeter simulation (calorimeter showers) while keeping reasonable simulation accuracy (correctly reproducing simulation behavior)
    - ○ consider LHCb ECAL as a practical goal
- ➢ Our ML problem formulation (hidden variables model):

x **input**

K variables:
px, py, pz, …
particle type, etc

ε **noise**

L variables:
“hidden variables”



Problem

- We want to speed up calorimeter simulation (calorimeter showers) while keeping reasonable simulation accuracy (correctly reproducing simulation behavior)
    - consider LHCb ECAL as a practical goal
- Our ML problem formulation (hidden variables model):

x **input**

K variables:
px, py, pz, …
particle type, etc

ε **noise**

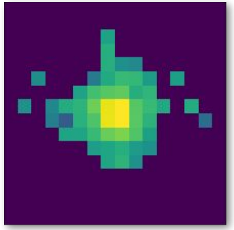
L variables:
“hidden variables”

Sampler

$$\hat{y} = G(x, \varepsilon, \theta) \sim p(y \mid x)$$

target y

HxW matrix
energy response
in cells



Goal

- For the particle of a given type with given momentum and position on the face of the calorimeter generate reasonable response in calorimeter cells

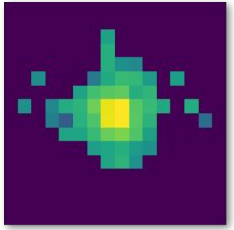
- Metrics we desire to match between simulated data and our samples:

    - cluster mean energy and shape
    - total energy resolution
    - cluster shape fluctuation
    - correlations between different cells of the cluster

target y

HxW matrix

energy response in cells



Data

- Dataset of (X, Y) is produced with simple GEANT simulation of LHCb-like ECAL
  - 66 layers 2mm absorber + 4mm scintillator
  - Block 5x5 big modules
  - Each module is split 6x6
  - Single particle on the entrance (currently electron)
- Information about every event:
  - 3-momentum, 2-position, particle type (X)
  - Full energy lost in absorver and deposited in scintillator
  - 30x30 matrix of energies deposited in scintillator for every cell tower (Y)

Approach

- Consider an unconditional sampler $G(\varepsilon, \theta)$ to be a neural network

- Consider loss function for G to be a neural network D. We want this loss to measure how "distant" are real samples y from samples $\hat{y}$ produced by our model, i.e. distance between distributions $p(\hat{y})$ and $p(y)$

- This is accomplished by a zoo of "adversarial" objective functions:

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

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

- Still, we need to sample from $p(y \mid x)$, not just $p(y)$, i.e. we need conditional model

Conditional WGAN

x

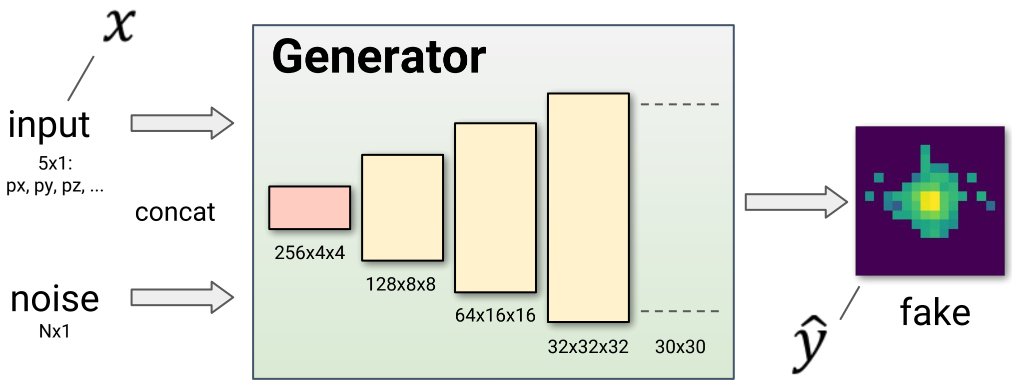
input

5x1:
px, py, pz, ...

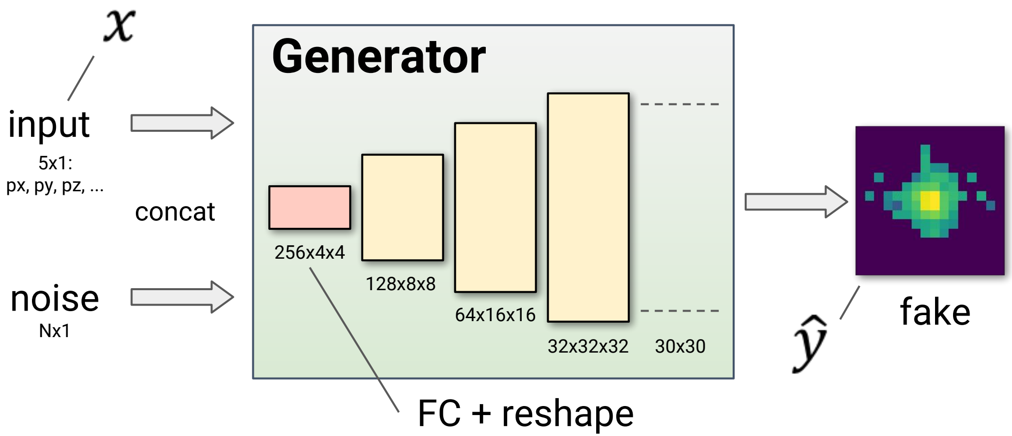
noise

Nx1

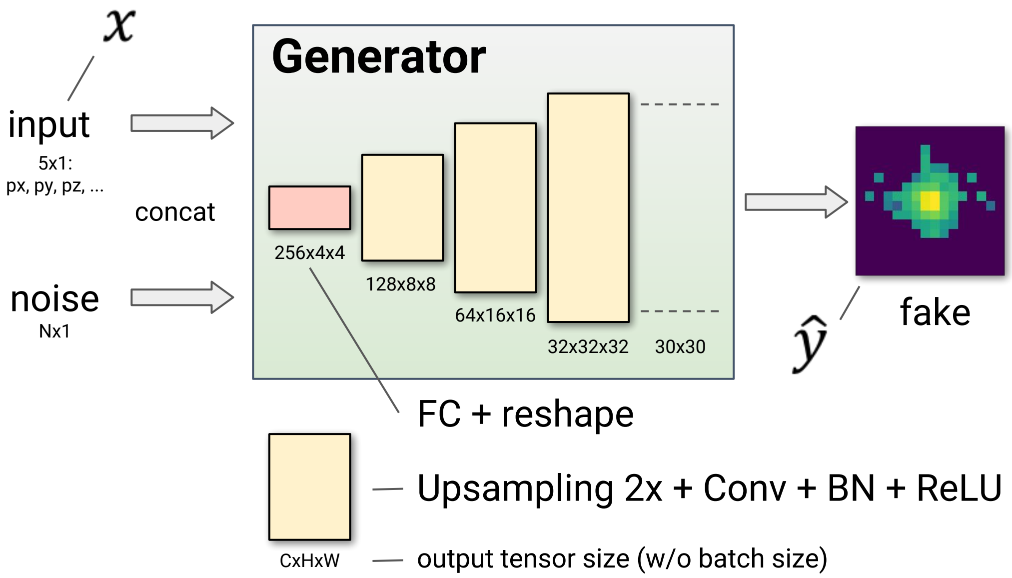
Conditional WGAN



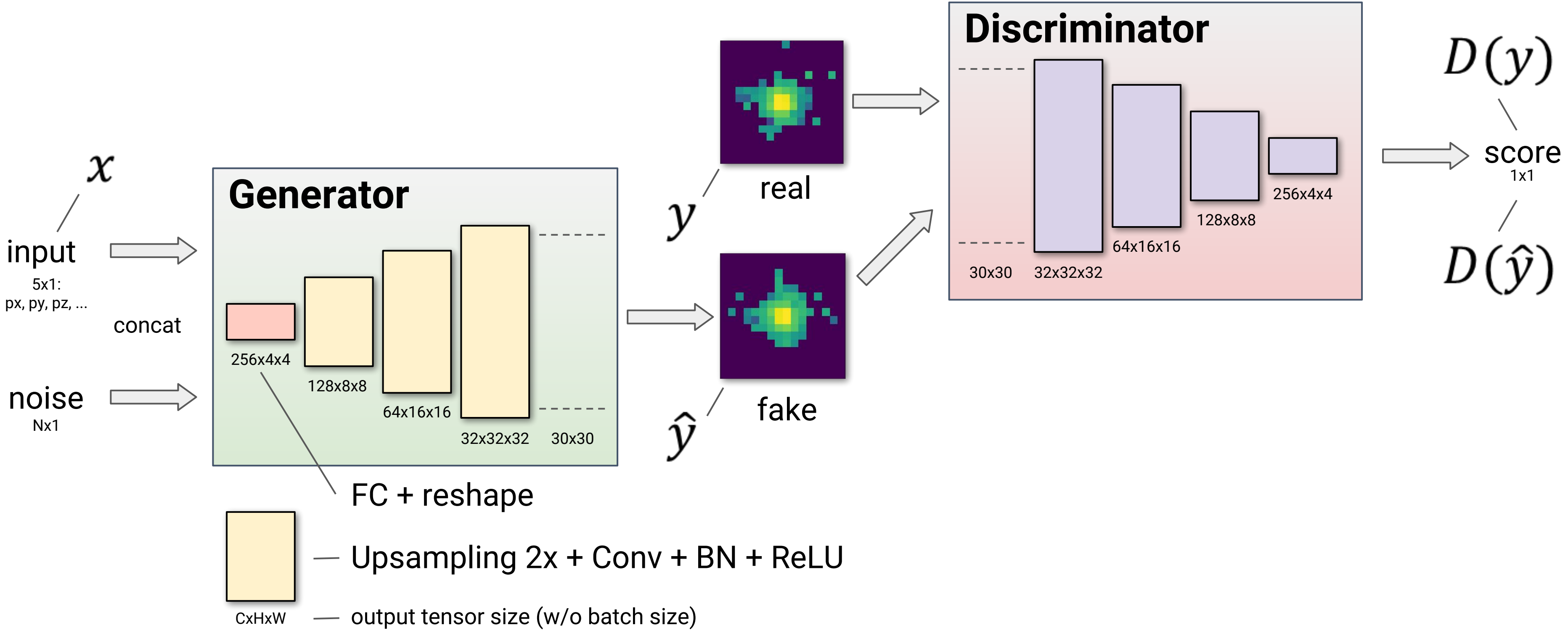
Conditional WGAN



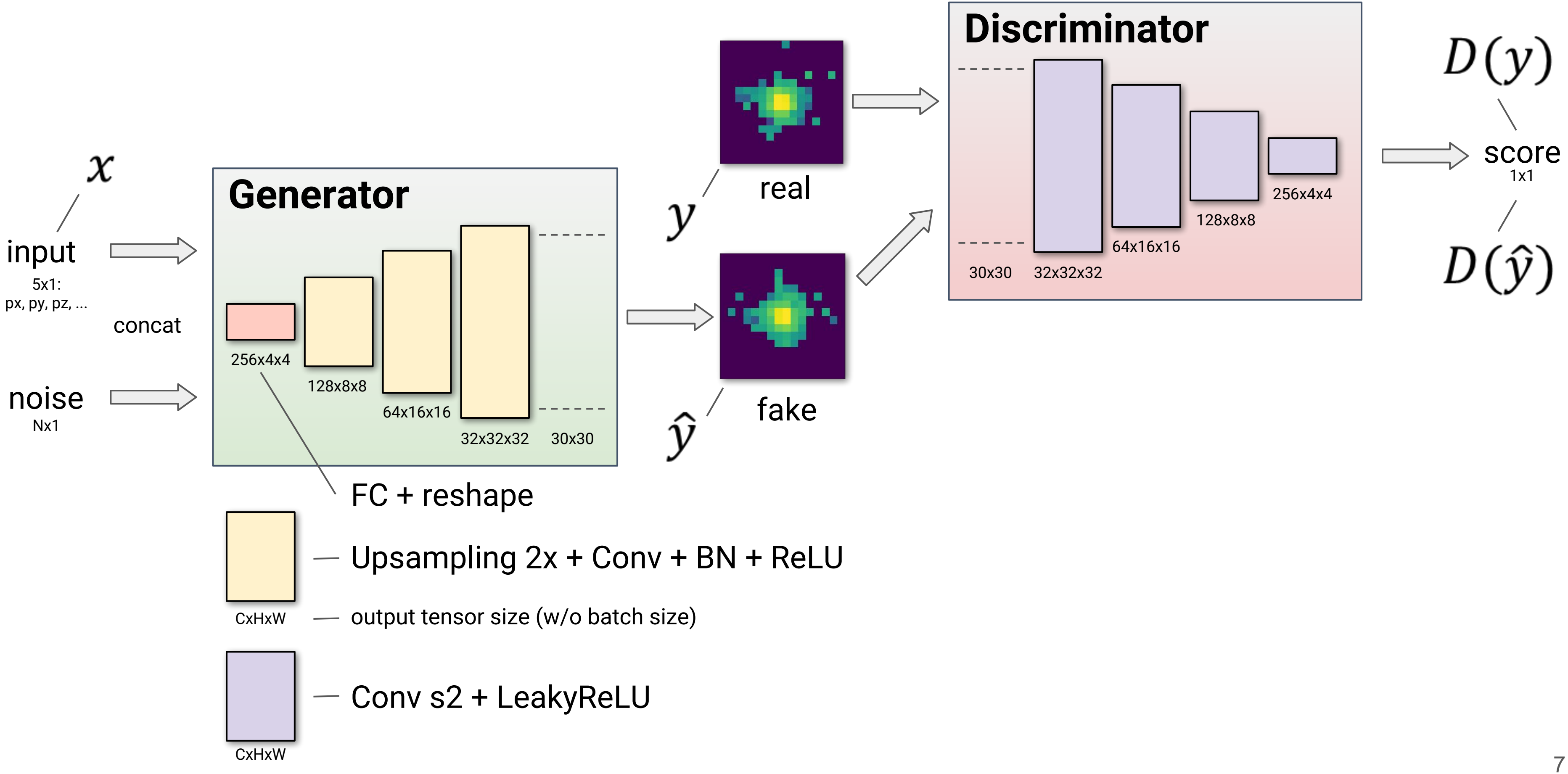
Conditional WGAN



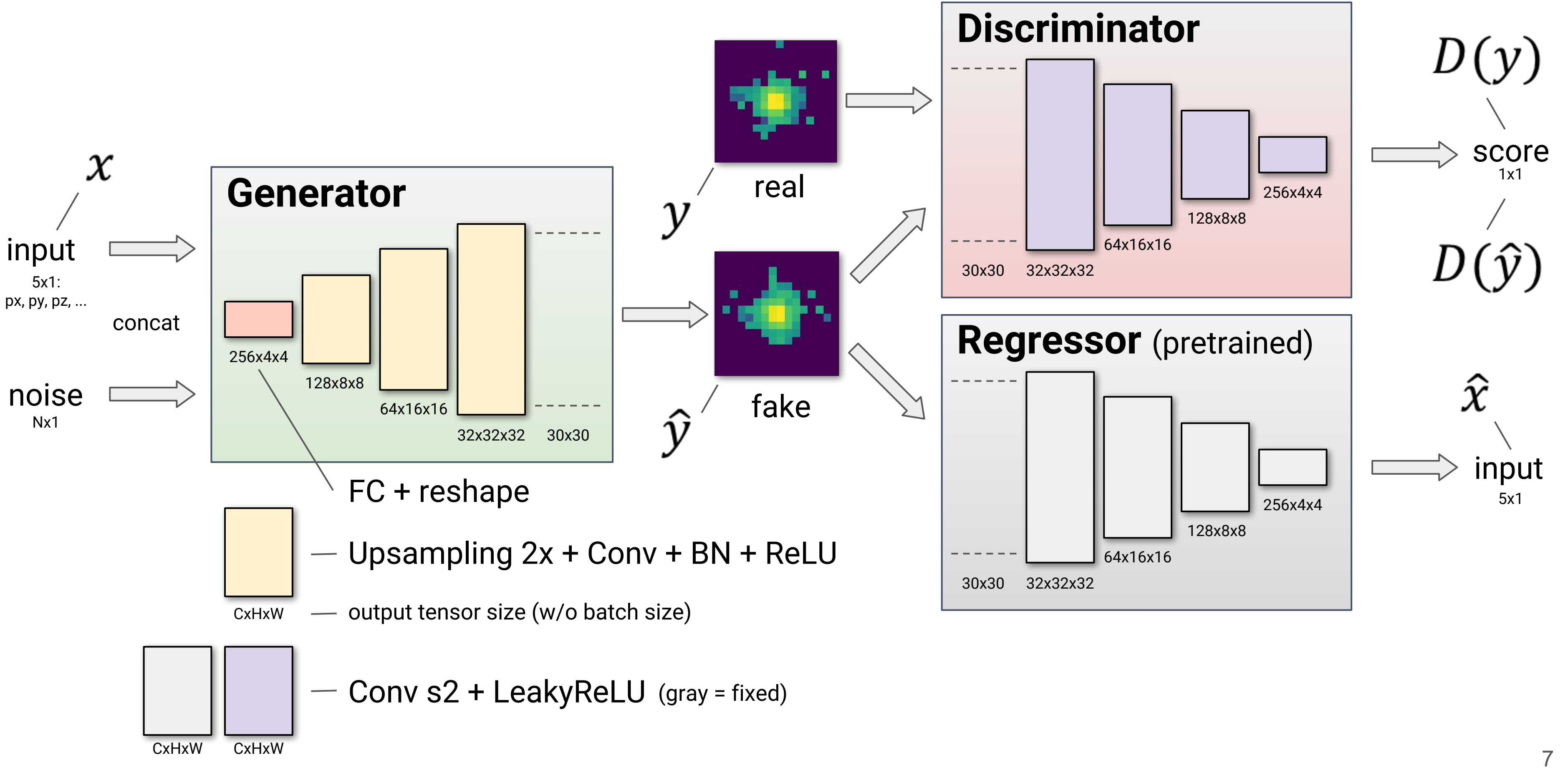
Conditional WGAN



Conditional WGAN

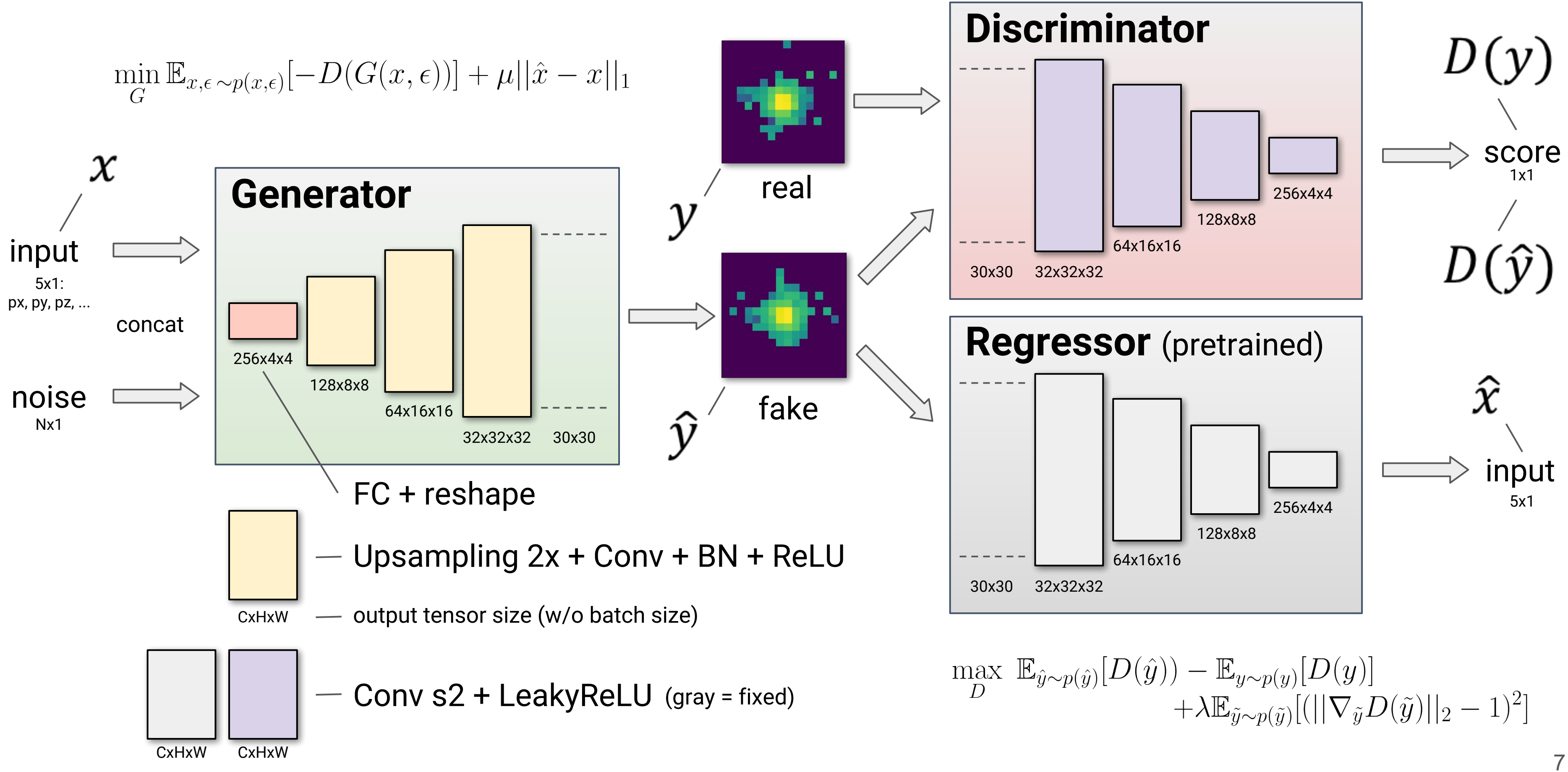


Conditional WGAN



Conditional WGAN

$$\min_G \mathbb{E}_{x,\epsilon \sim p(x,\epsilon)}[-D(G(x,\epsilon))] + \mu||\hat{x} - x||_1$$



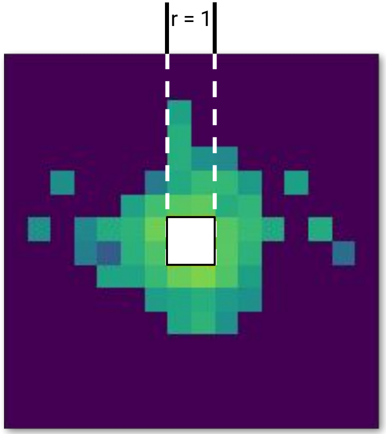
$$\max_D \mathbb{E}_{\hat{y}\sim p(\hat{y})}[D(\hat{y})) - \mathbb{E}_{y\sim p(y)}[D(y)]$$
$$+\lambda \mathbb{E}_{\tilde{y}\sim p(\tilde{y})}[(||\nabla_{\tilde{y}} D(\tilde{y})||_2 - 1)^2]$$

Qualitative evaluation (input = pz)

Distributions inside calorimeter regions

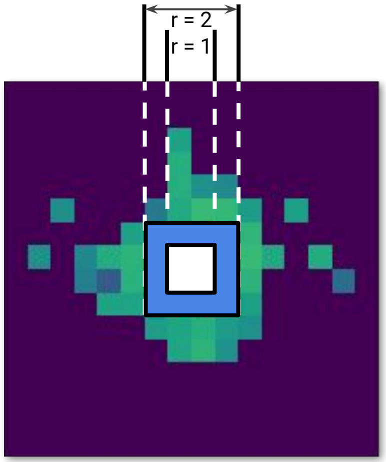
Qualitative evaluation (input = pz)

Distributions inside calorimeter regions



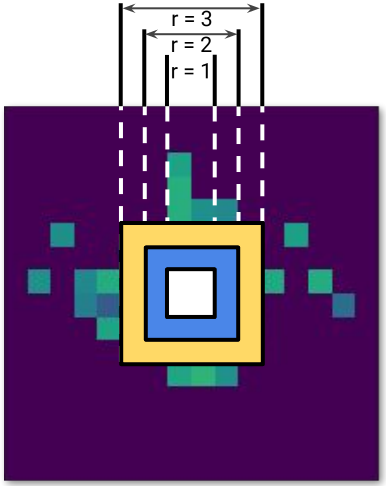
Qualitative evaluation (input = pz)

Distributions inside calorimeter regions



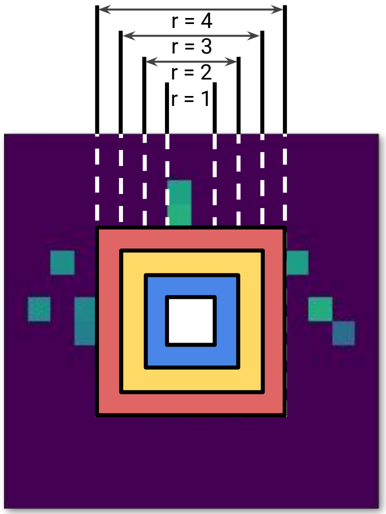
Qualitative evaluation (input = pz)

Distributions inside calorimeter regions



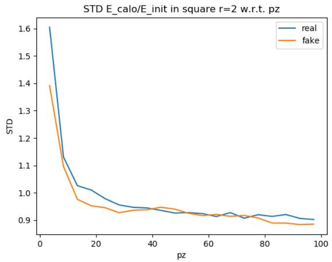
Qualitative evaluation (input = pz)

Distributions inside calorimeter regions

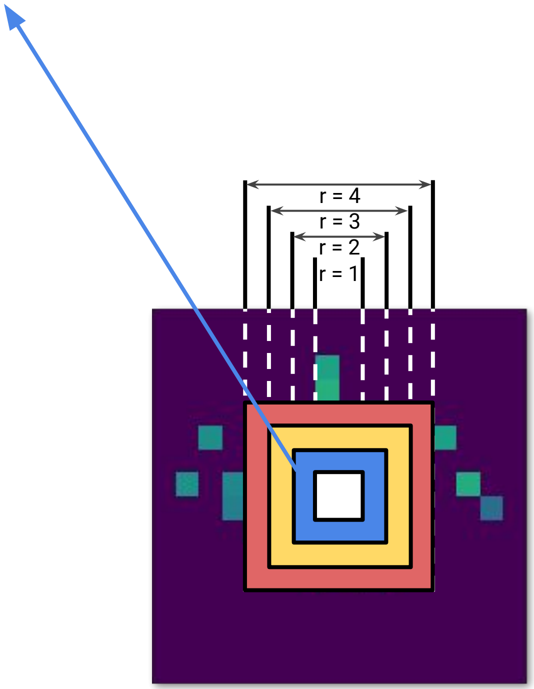


Qualitative evaluation (input = pz)

Distributions inside calorimeter regions (bins represent different energy levels)

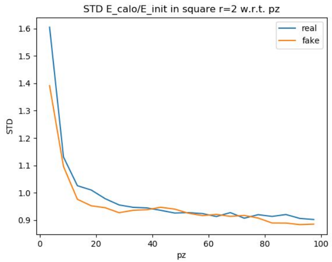


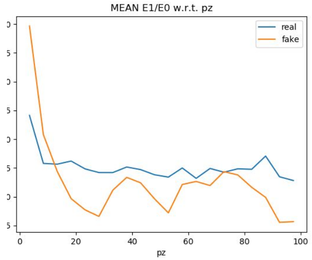
Standard deviation of sum of energies inside the square normalized by the initial energy

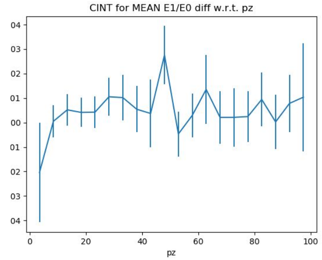


Qualitative evaluation (input = pz)

Distributions inside calorimeter regions (bins represent different energy levels)

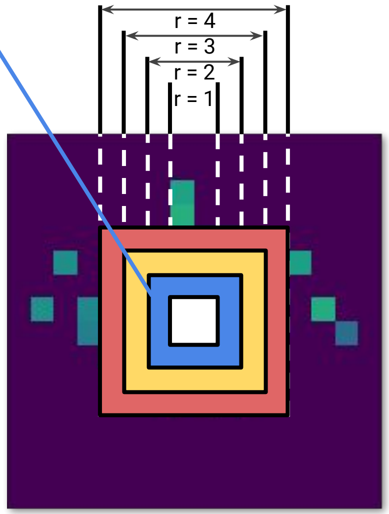






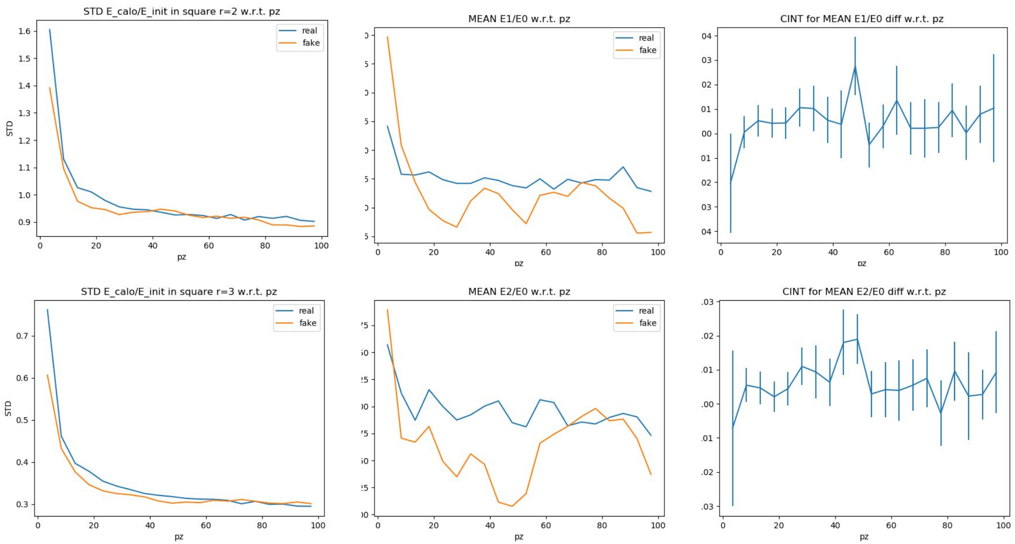
Mean of Ek/E0 and conf. int. for difference between real and fake means

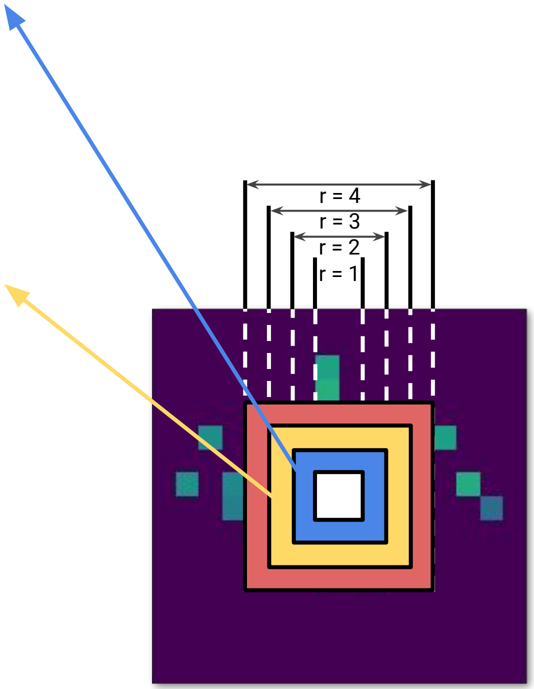
Standard deviation of sum of energies inside the square normalized by the initial energy



Qualitative evaluation (input = pz)

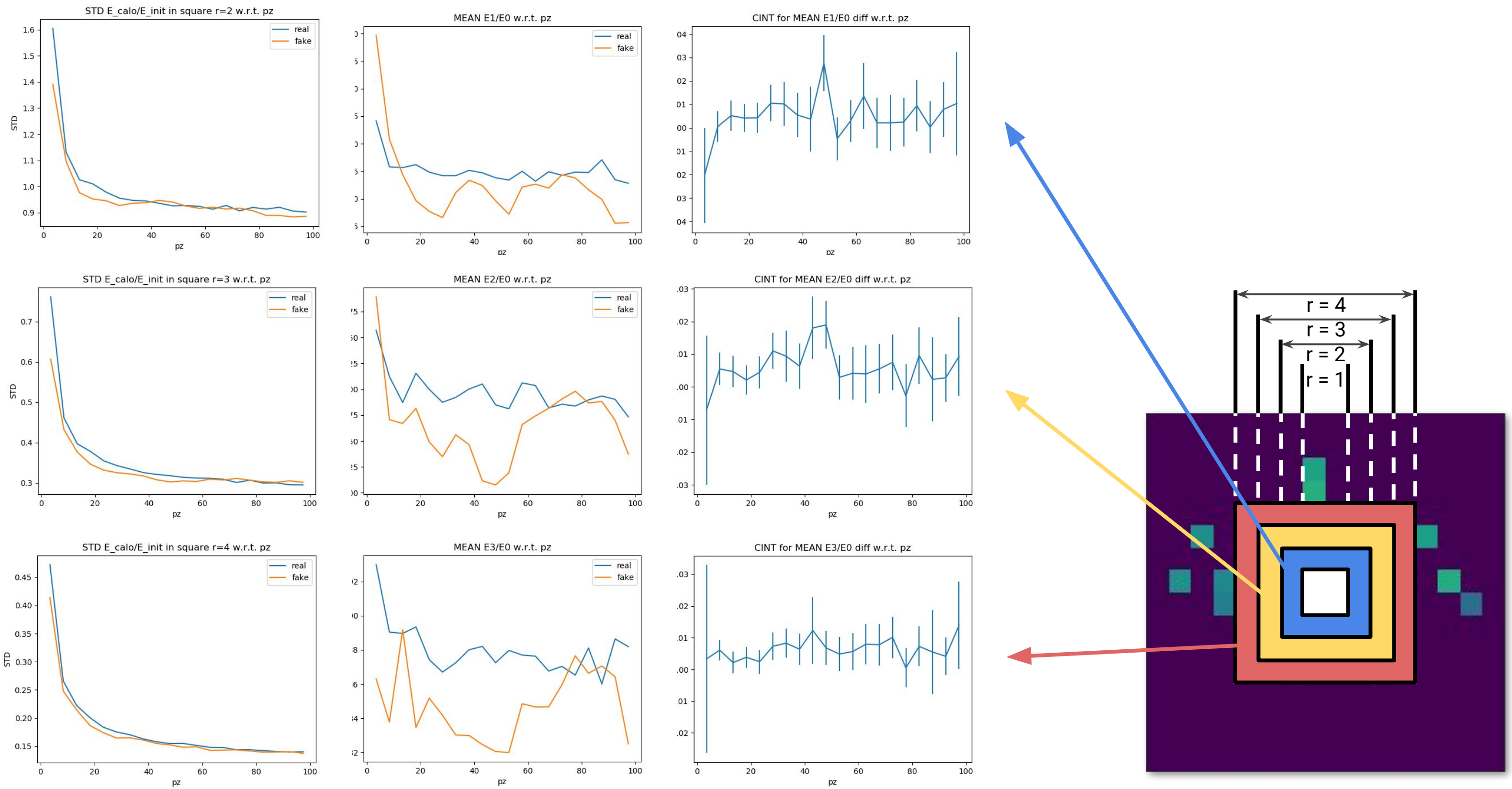
Distributions inside calorimeter regions (bins represent different energy levels)





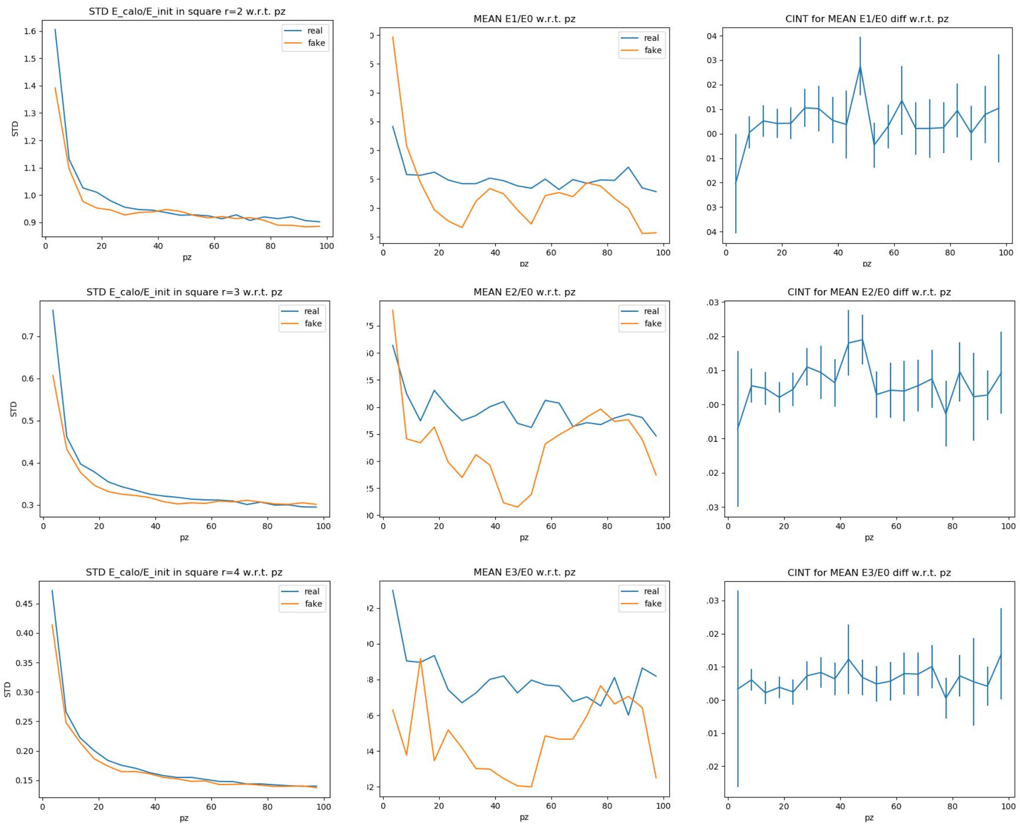
Qualitative evaluation (input = pz)

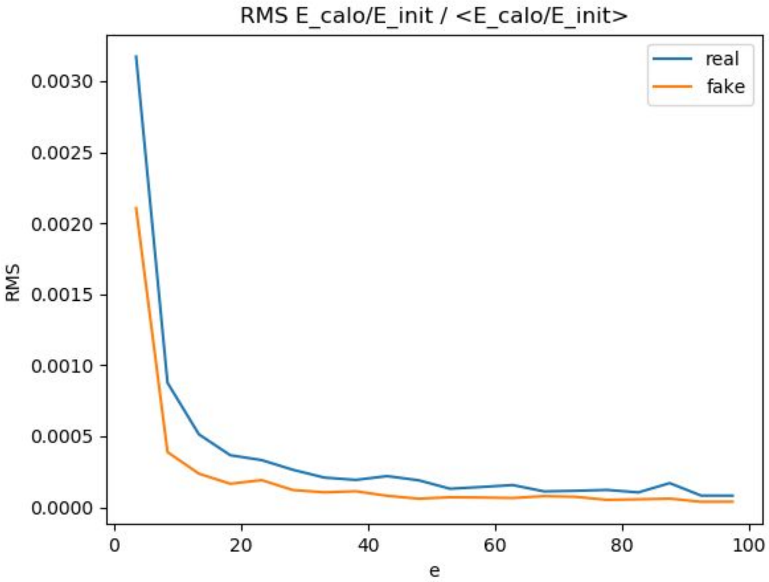
Distributions inside calorimeter regions (bins represent different energy levels)



Qualitative evaluation (input = pz)

Distributions inside calorimeter regions (bins represent different energy levels)

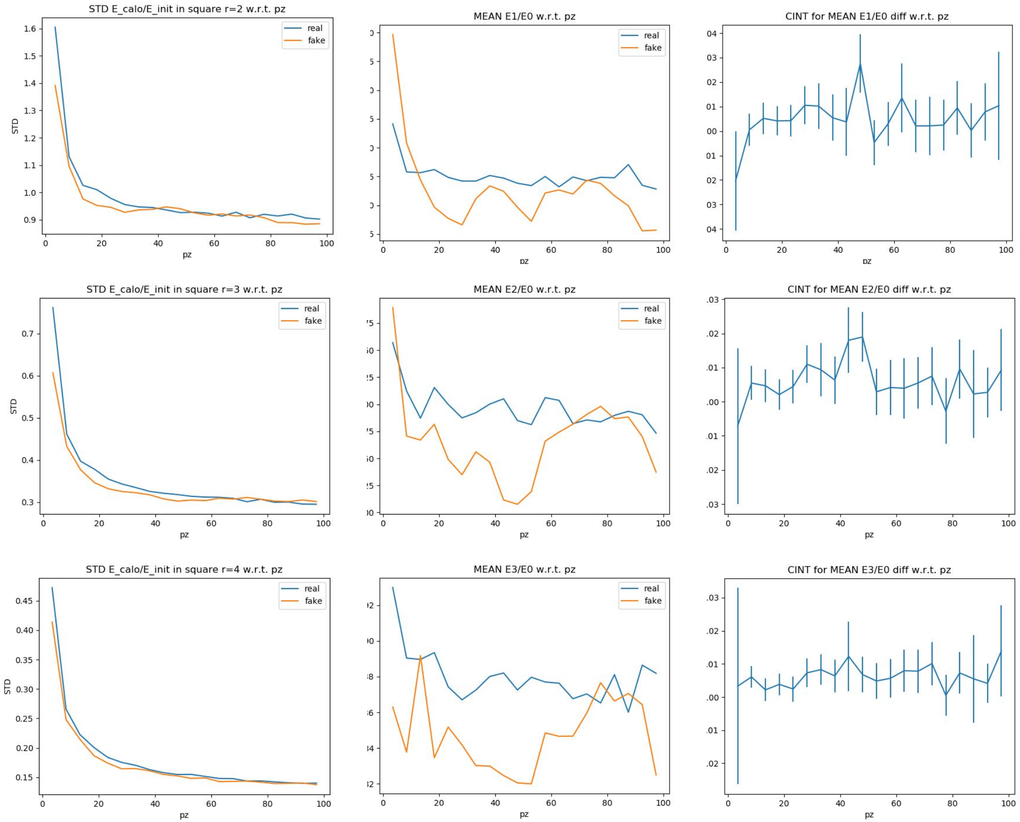


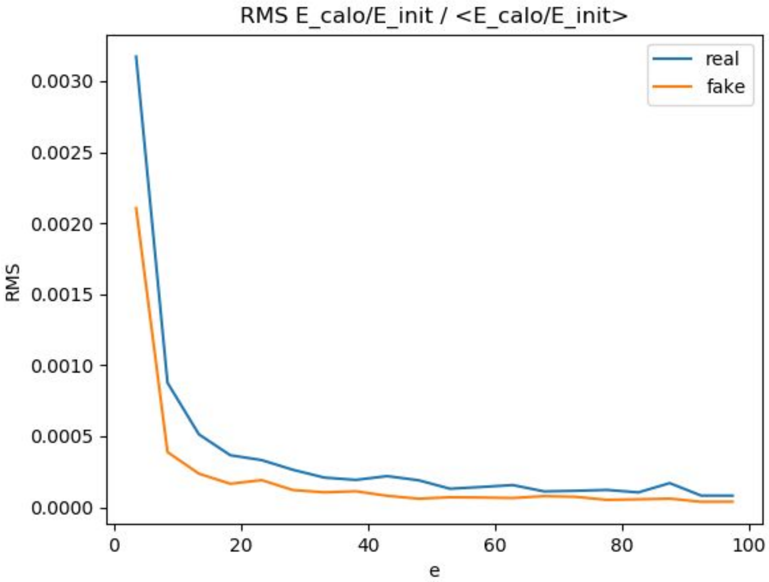


Energy resolution

Qualitative evaluation (input = pz)

Distributions inside calorimeter regions (bins represent different energy levels)





Energy resolution

➢ Observe good match for statistics inside the regions and decent match for resolution

Quantitative evaluation (input = pz)

- ➢ We perform classifier two sample test (C2ST) on other candidates for sampler model:
  - conditional WGAN/GAN
  - WGAN/GAN

Quantitative evaluation (input = pz)

- We perform classifier two sample test (C2ST) on other candidates for sampler model:
  - conditional WGAN/GAN
  - WGAN/GAN

Quantitative evaluation (input = pz)

➢ We perform classifier two sample test (C2ST) on other candidates for sampler model:

- conditional WGAN/GAN
- WGAN/GAN

➢ Training metric: binary cross entropy

➢ Val. metric: avg. error on the val. set

Quantitative evaluation (input = pz)

- ➢ We perform classifier two sample test (C2ST) on other candidates for sampler model:
  - conditional WGAN/GAN
  - WGAN/GAN

- ➢ Training metric: binary cross entropy
- ➢ Val. metric: avg. error on the val. set

- ➢ Goal is to compare distance between distributions of simulated data and different versions of our model (i.e. provide us a quality metric)

Quantitative evaluation (input = pz)

➢ We perform classifier two sample test (C2ST) on other candidates for sampler model:

  ○ conditional WGAN/GAN
  ○ WGAN/GAN

➢ Training metric: binary cross entropy

➢ Val. metric: avg. error on the val. set

➢ Goal is to compare distance between distributions of simulated data and different versions of our model (i.e. provide us a quality metric)

➢ Setup: ResNet18, 100 epochs, least error on validation

Quantitative evaluation (input = pz)

- We perform classifier two sample test (C2ST) on other candidates for sampler model:
    - conditional WGAN/GAN
    - WGAN/GAN

- Training metric: binary cross entropy
- Val. metric: avg. error on the val. set

- Goal is to compare distance between distributions of simulated data and different versions of our model (i.e. provide us a quality metric)

- Setup: ResNet18, 100 epochs, least error on validation

	Cond. WGAN	Cond. GAN	WGAN	GAN
Score (0.5 — best)	**0,36**	0,08	0,12	0,10

Summary

- Convolutional generative adversarial models can produce calorimeter showers that match our desired realism criteria

Summary

- Convolutional generative adversarial models can produce calorimeter showers that match our desired realism criteria

- They achieve sampling rate of 0.04 ms per sample on GPU, 4.7 ms per sample on CPU (NVIDIA DGX-1, batch size = 64)

Summary

- Convolutional generative adversarial models can produce calorimeter showers that match our desired realism criteria

- They achieve sampling rate of 0.04 ms per sample on GPU, 4.7 ms per sample on CPU (NVIDIA DGX-1, batch size = 64)

- A lot of work ahead to bring these generative models to production quality for use in LHCb and in HEP: make statistics match uniform across all energy levels, incorporate support for different particle types and multiple inputs (px, py, …), etc.

Summary

- Convolutional generative adversarial models can produce calorimeter showers that match our desired realism criteria
- They achieve sampling rate of 0.04 ms per sample on GPU, 4.7 ms per sample on CPU (NVIDIA DGX-1, batch size = 64)
- A lot of work ahead to bring these generative models to production quality for use in LHCb and in HEP: make statistics match uniform across all energy levels, incorporate support for different particle types and multiple inputs (px, py, …), etc.
- Need to compare our model using proposed metrics with other existing models (ex., CaloGAN)