Machine Learning for Simulation of Collision Events

R. Cobe, J. Fialho, B. Orzari, T. Tomei

RENAFAE WORKSHOP 2022

Advanced Institute for Artificial Intelligence

SPRACE-Unesp

Motivation

LHC collision events always present hadronic jets.

- State of the art: ME generation + hadronisation + simulation.
- Full simulation: based on GEANT.
- Parameterised sims: Delphes, FastSim, ...

Machine learning-based alternative.

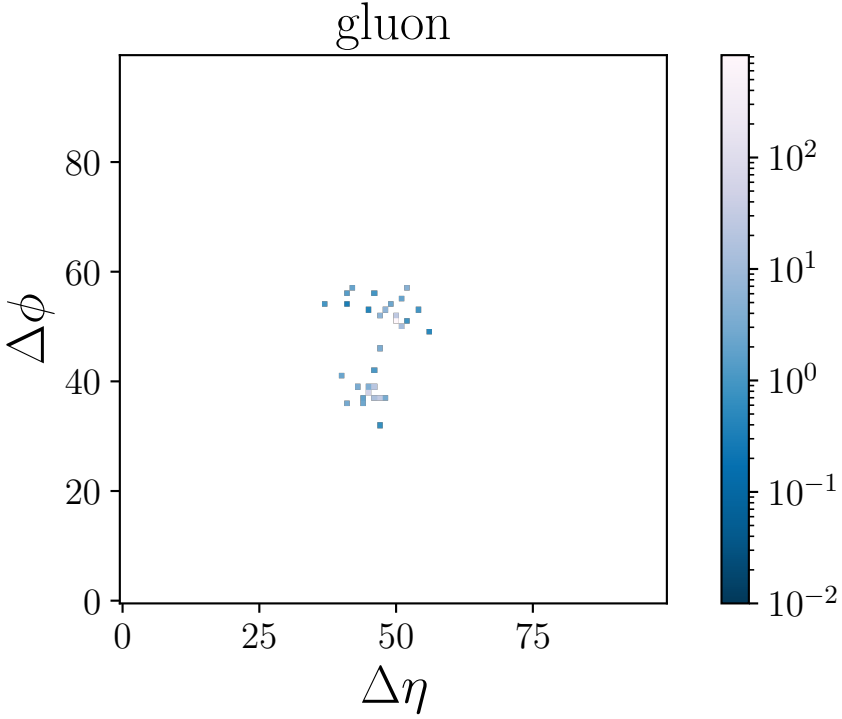
- Hadronic jets generation using a generative neural network.
- Possible applications:
  - Train on simulated jets: better simulation (substitute FastSim).
  - Train on real reconstructed jets: better hadronization + simulation.

Jets

Sparse sets of particles that are intrinsically unordered

- Each particle described by set of features: p_x, p_y, p_z, p_T, η, ϕ, ...
- Image representation: useful, but not fundamental.

Even though an ordering might be attributed to the data, it is important to preserve its permutation invariance.



Graphical representation of a gluon jet (arXiv:1908:05318)

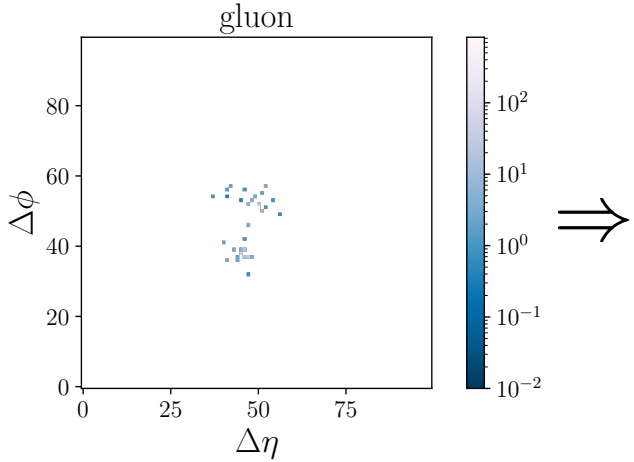
The JetNet Dataset

https://zenodo.org/record/6302454

- High momentum jets originating from gluons, light quarks, Z and W bosons, and top quarks.
- For now, only the gluon jets dataset is being used (~170k jets).

Each jet is represented as a list of 30 particles with 3 features.

- p_x, p_y, p_z showed better results.



Jet particles		
p_x	**p_y**	**p_z**
p_{x1}	p_{y1}	p_{z1}
p_{x2}	p_{y2}	p_{z2}
p_{x3}	p_{y3}	p_{z3}
	...	
p_{x30}	p_{y30}	p_{z30}

Variational Autoencoders

- "Unsupervised" learning algorithm that applies backpropagation, setting the target values to be equal to the inputs.
- Consists of two parts:
  - Encoder function $h = f(x)$, and
  - Decoder function (reconstruction) $r = g(h)$.
- Learn the parameters of a distribution instead of range of values at the latent space.
- Sample from the learnt distribution to *generate* examples from the data distribution.
- Usually depends on the *Kullback–Leibler Divergence* to train.

Variational Autoencoders

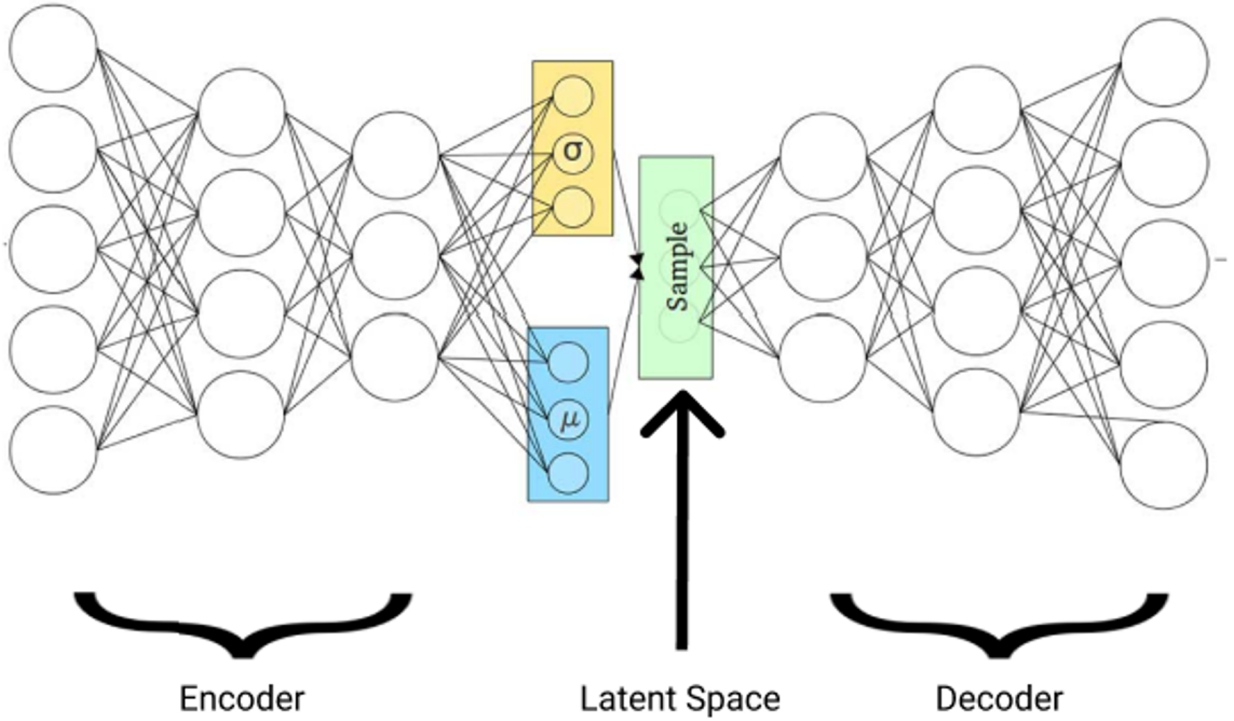


Image from [Variational Autoencoders Simply Explained]

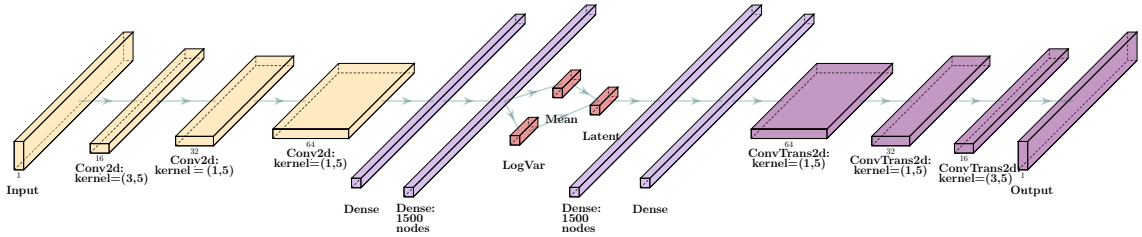
VAE for Sparse Data Generation

https://arxiv.org/abs/2109.15197

Training step:

Jet		
p_x	**p_y**	**p_z**
p_{x1}	p_{y1}	p_{z1}
p_{x2}	p_{y2}	p_{z2}
p_{x3}	p_{y3}	p_{z3}
	...	
p_{x30}	p_{y30}	p_{z30}

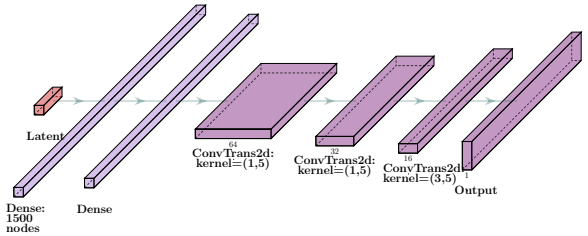
→



→

Jet		
p_x	**p_y**	**p_z**
p_{x1}	p_{y1}	p_{z1}
p_{x2}	p_{y2}	p_{z2}
p_{x3}	p_{y3}	p_{z3}
	...	
p_{x30}	p_{y30}	p_{z30}

Generation step:



→

Jet		
p'_x	**p'_y**	**p'_z**
p'_{x1}	p'_{y1}	p'_{z1}
p'_{x2}	p'_{y2}	p'_{z2}
p'_{x3}	p'_{y3}	p'_{z3}
	...	
p'_{x30}	p'_{y30}	p'_{z30}

Loss Function (1)

The error function L_{VAE} is built as

$$L_{\mathrm{VAE}} = (1-\beta)L_{\mathrm{rec}} + \beta D_{\mathrm{KL}}$$

where D_{KL} (L_{rec}) is the generation (reconstruction) error. The L_{rec} term is divided as

$$L_{\mathrm{rec}} = L^{\mathrm{NND}} + L^{\mathrm{J}}$$

where L^{NND} is a permutation-invariant term in the particles' properties:

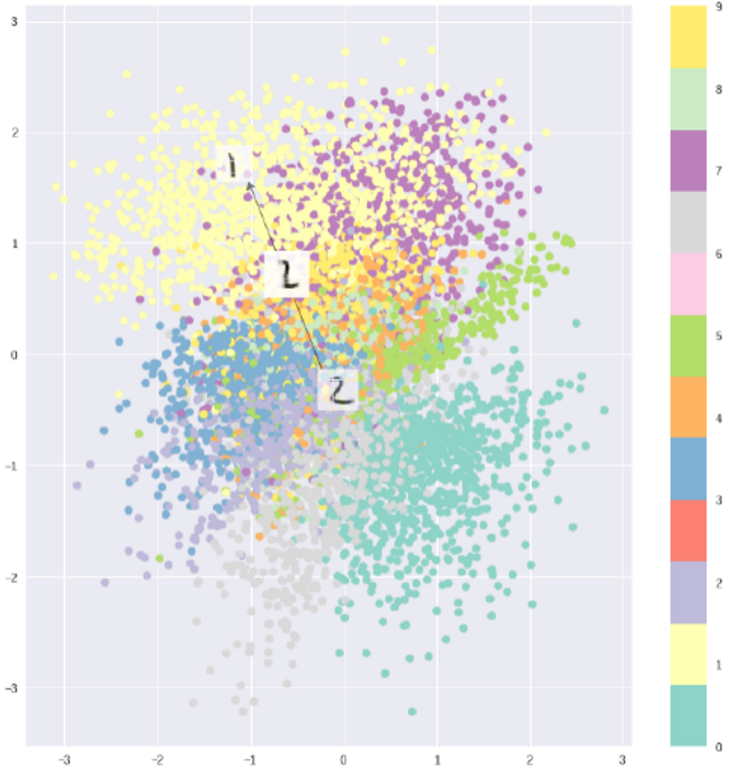
$$L^{\mathrm{NND}} = \sum_k \left[\sum_{i \in \mathcal{J}_k} \min_{j \in \hat{\mathcal{J}}_k} D\left(\vec{p}_i, \vec{\hat{p}}_j\right) + \sum_{j \in \hat{\mathcal{J}}_k} \min_{i \in \mathcal{J}_k} D\left(\vec{p}_i, \vec{\hat{p}}_j\right) \right]$$

and L^{J} is a sum of two mean-squared errors (MSE) on jets' properties

$$L^{\mathrm{J}} = \sum_k \left[\gamma_{p_{\mathrm{T}}} \cdot \mathrm{MSE}\left(p_{\mathrm{T}k}, \hat{p}_{\mathrm{T}k}\right) + \gamma_m \cdot \mathrm{MSE}\left(m_k, \hat{m}_k\right) \right]$$

Loss Function (2)

- The penalization term is the KL divergence (relative entropy).
  - Distance between a Gaussian with mean μ and std deviation σ and the standard normal distribution.
- Keeps the latent space variables centered around 0 avoiding the scatter.
- The β controls the influence of the penalization term.



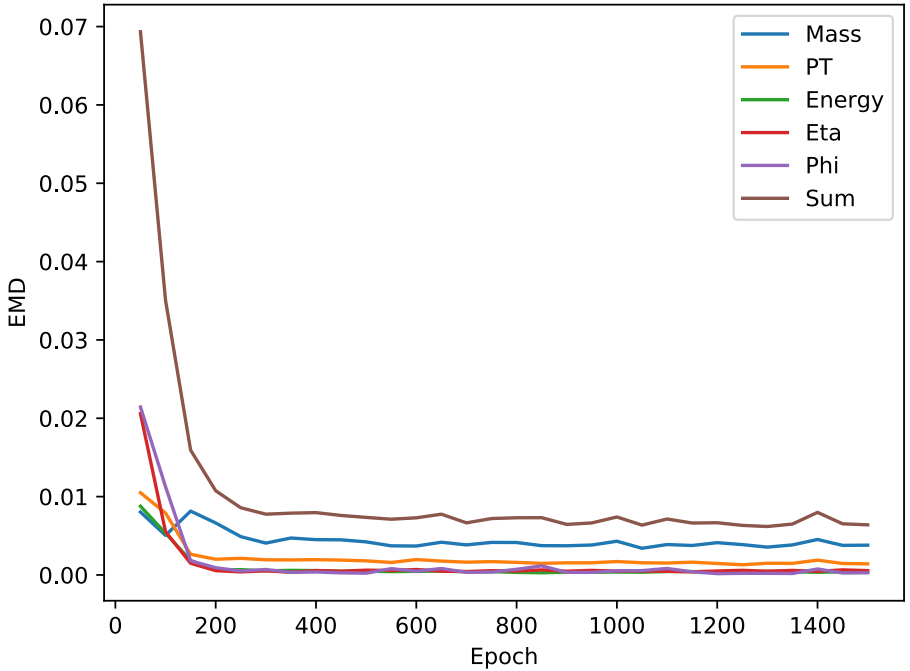
Generation Capacity Measurement

Jets features being compared:

- Mass, energy, p_T, η, ϕ

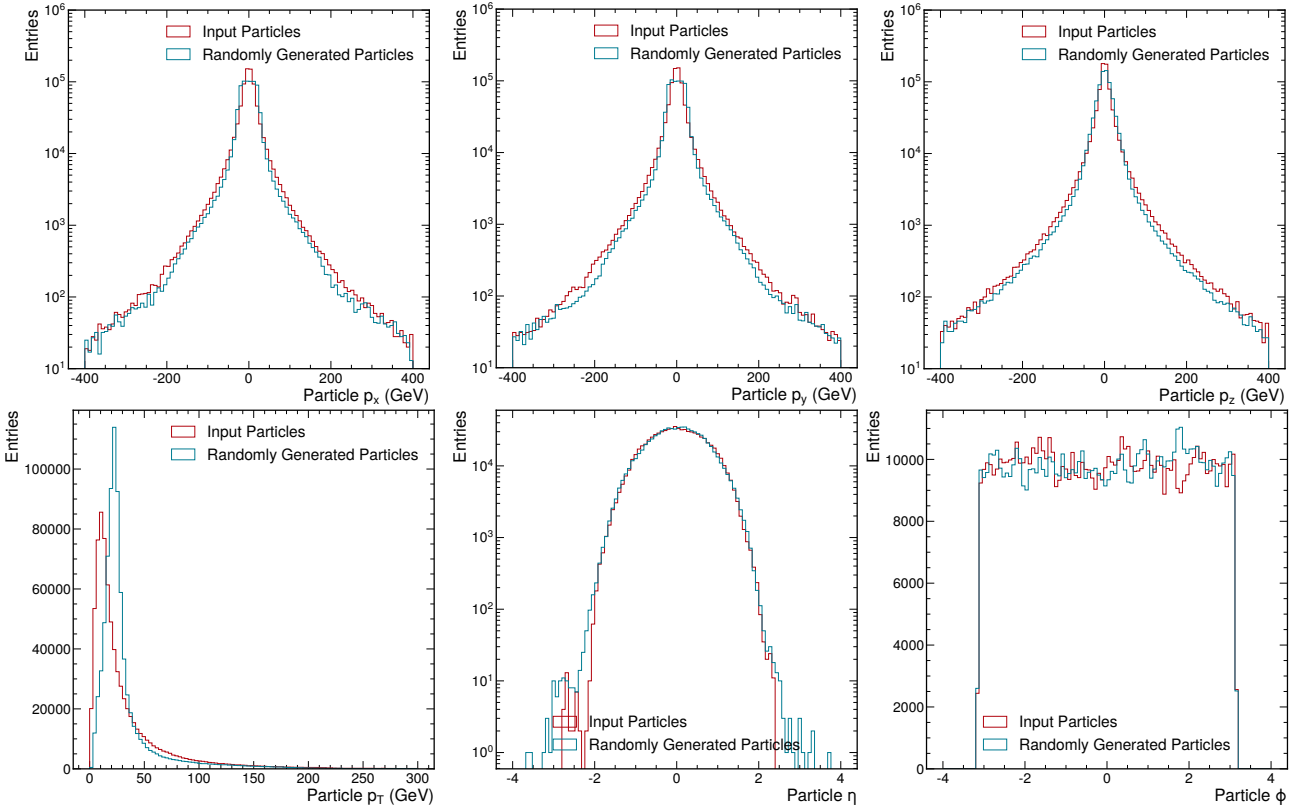
Quantitative measurement:

- Sum of “earth mover’s” (Wasserstein’s) distance (EMD) between histograms of input and generated jets features.



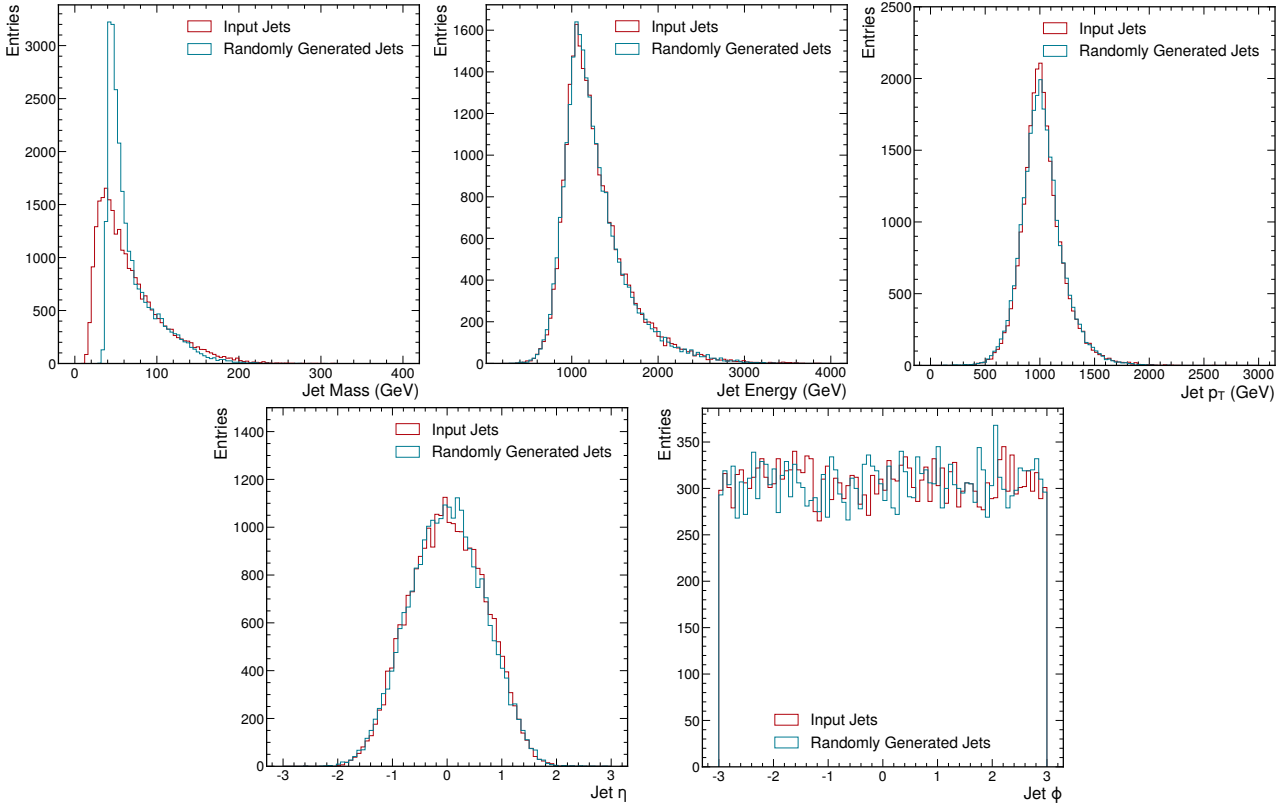
Results 1: Generated Particles Properties

The min(EMD_{sum}) over all trained models was 0.0061



Results 2: Generated Jets Properties

The min(EMD_{sum}) over all trained models was 0.0061

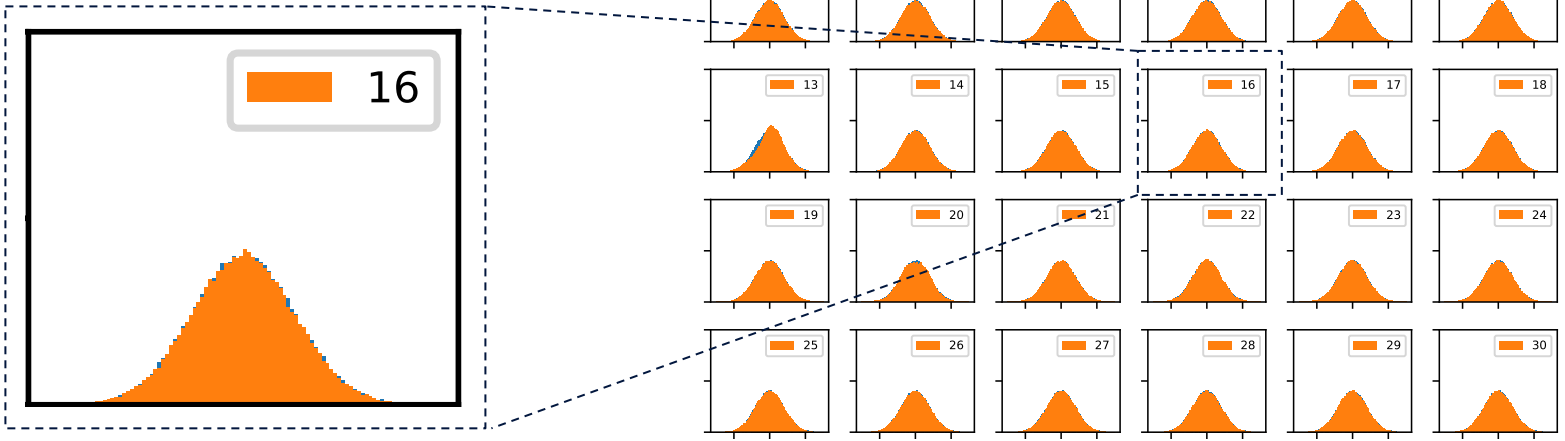


Alternative Approach to VAE Jet Generation

Generated jets mass does not match the input jets mass.

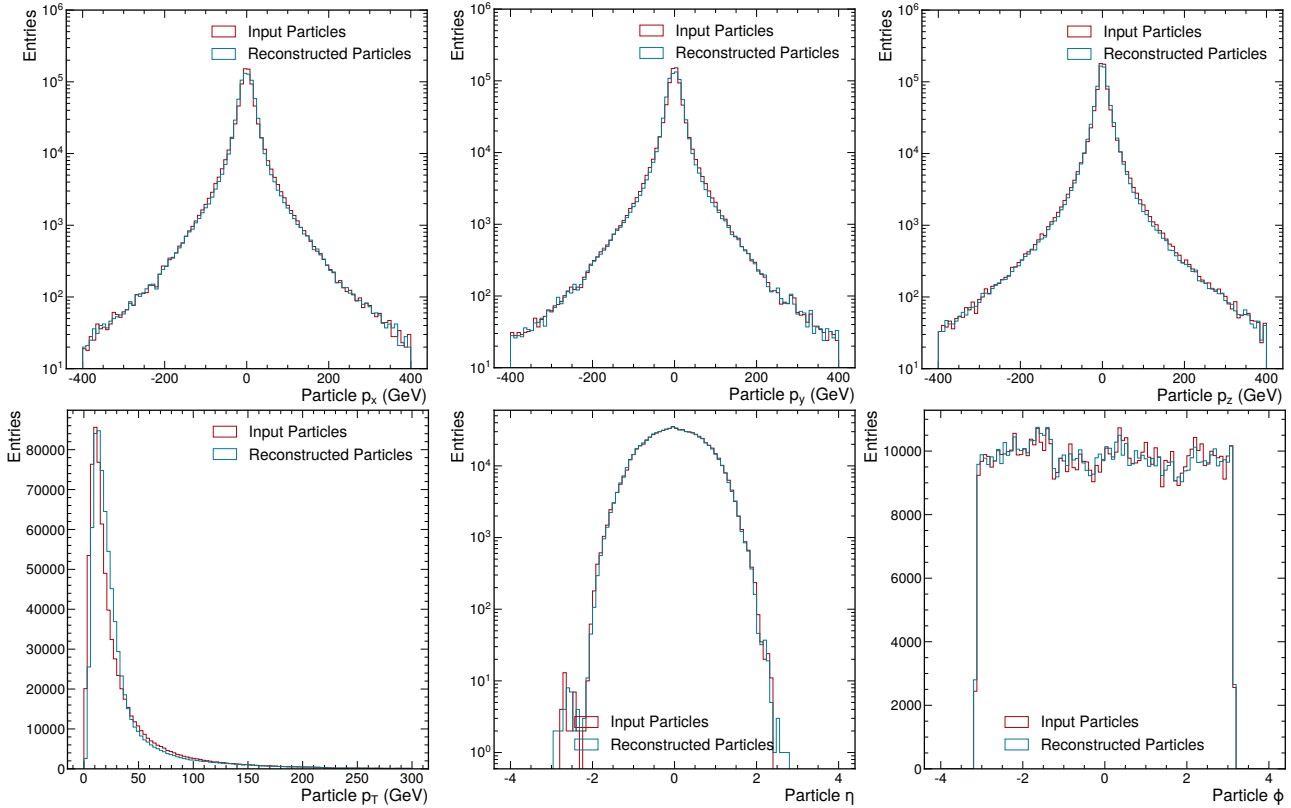
- Network encodes data into standard gaussians →
  might not be the optimal distribution for latent vector elements (as shown in next slides)

Blue distribution: $\mathcal{N}(0,1)$
Orange distribution: $p_Z(\mathbf{z})$
in generation case



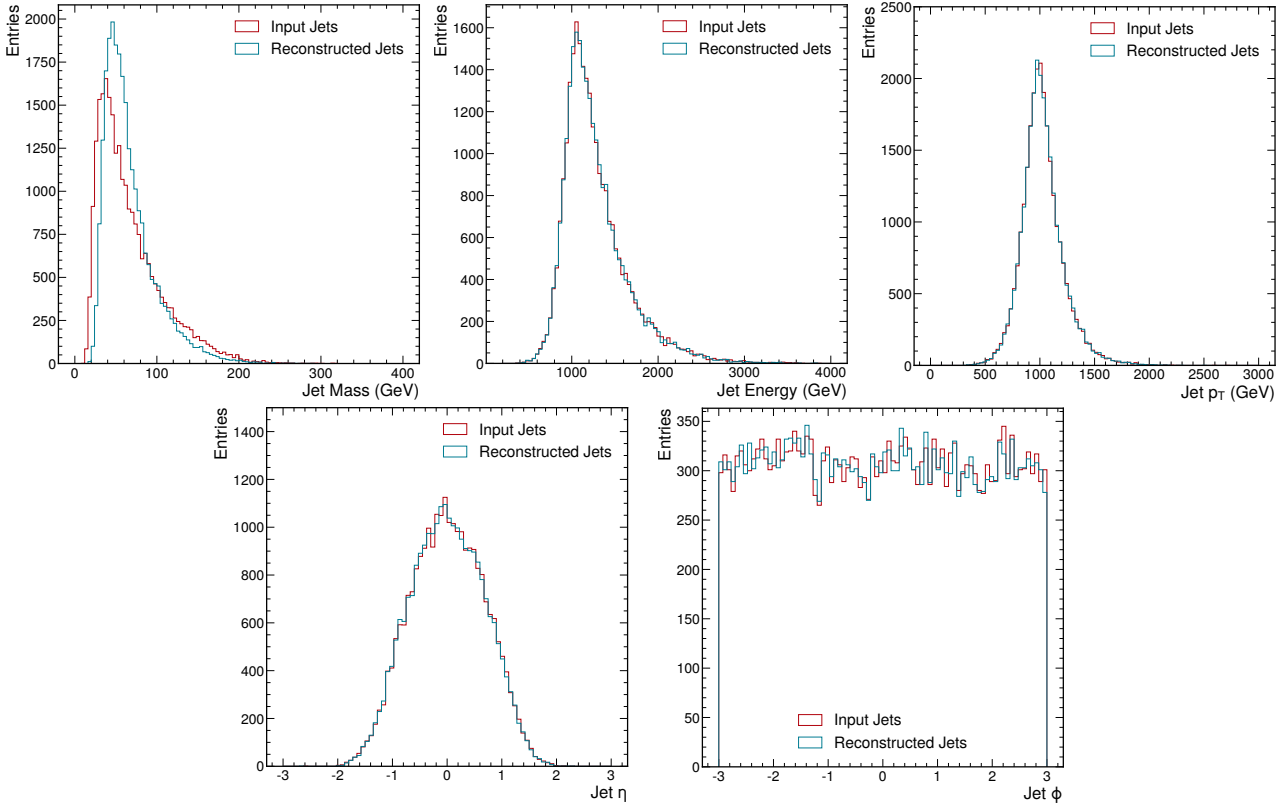
Results 3: Reconstructed Particles Properties

Network tuned for jets reconstruction ($\beta = 0$): $\min(EMD_{sum}) = 0.0027$



Results 4: Reconstructed Jets Properties

Network tuned for jets reconstruction ($\beta = 0$): $\min(\mathrm{EMD}_{\mathrm{sum}}) = 0.0027$

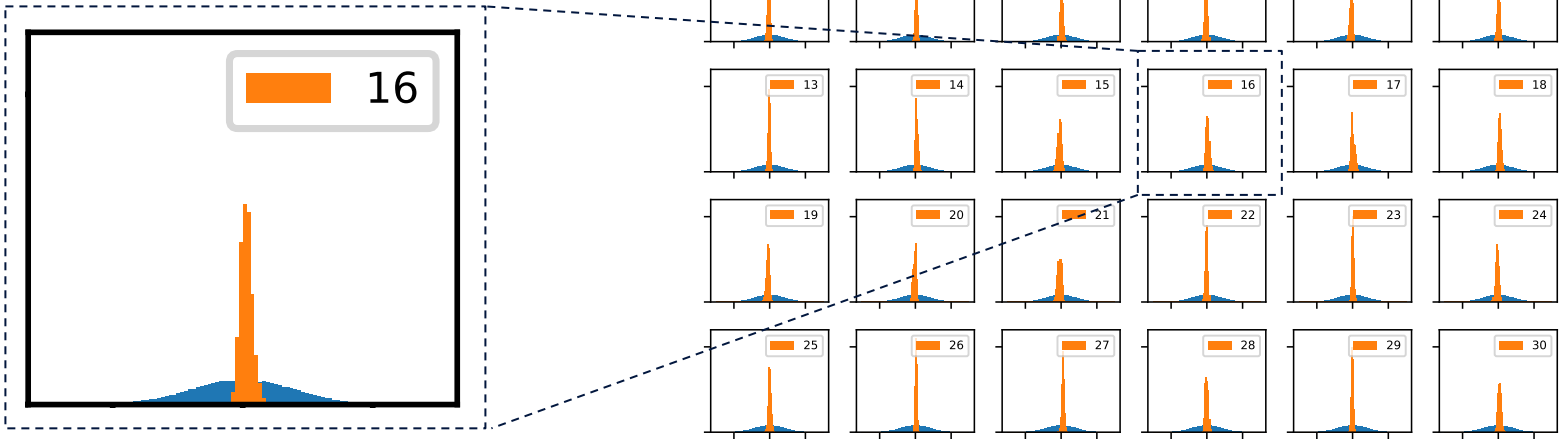


VAE for Jets Reconstruction

More distinct distributions of latent variables in comparison to standard gaussians.

- Might provide better jets if sampling is made from p(**z**)

Blue distribution: $\mathcal{N}(0,1)$
Orange distribution: $p_Z(\mathbf{z})$
in reconstruction case



Normalizing Flows

Technique to find the best distribution of the latent data.

- Learn the parameters of a transformation $f(\mathbf{z})$ taking distribution $p_Z(\mathbf{z})$ to $p_X(\mathbf{x})$.
- $f(\mathbf{z})$ is the composition of several simpler transformations:

$$\mathbf{x} = f(\mathbf{z}) = f_n \circ \cdots \circ f_2 \circ f_1(\mathbf{z})$$

Analytical expression of $p_Z(\mathbf{z})$ is unknown.

- Learn the transformation taking $p_Z(\mathbf{z})$ → $p_X(\mathbf{x}) = \mathcal{N}(0,1)$ and invert it.

$$g(\mathbf{x}) = f^{-1}(\mathbf{x}) = g_n \circ \cdots \circ g_2 \circ g_1(\mathbf{x}) = f_1^{-1} \circ f_2^{-1} \circ \cdots \circ f_n^{-1}(\mathbf{x}) = \mathbf{z}$$

- Sample values from $\mathcal{N}(0,1)$, apply the inverse transformation, obtain values that follow $p_Z(\mathbf{z})$.

Training: maximize the expression

$$\log\left(p_Z(\mathbf{z})\right) = \log\left(p_X(f(\mathbf{z}))\right) + \log\left(\left|\det\left(\frac{\partial f(\mathbf{z})}{\partial \mathbf{z}}\right)\right|\right)$$

Choice of Normalizing Flows

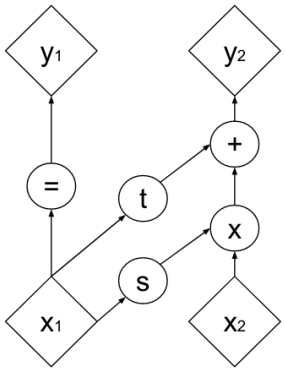
Options in the market:

- Planar flow,
- Sylvester flow
- More complicated options...

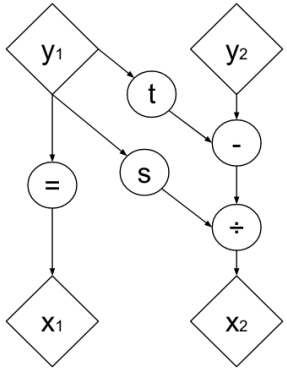
Our current option:

"real-valued non-volume preserving"
(i.e. RealNVP, arXiv:1605:08803)

- Simple analytic expression for transformations.
  - Still flexible since parameters can be given by any function.
  - Easily invertible.
- Retains representativity and is easily trainable.
  - Jacobian is given by triangular matrix.
  - Determinant calculation is linear in time.

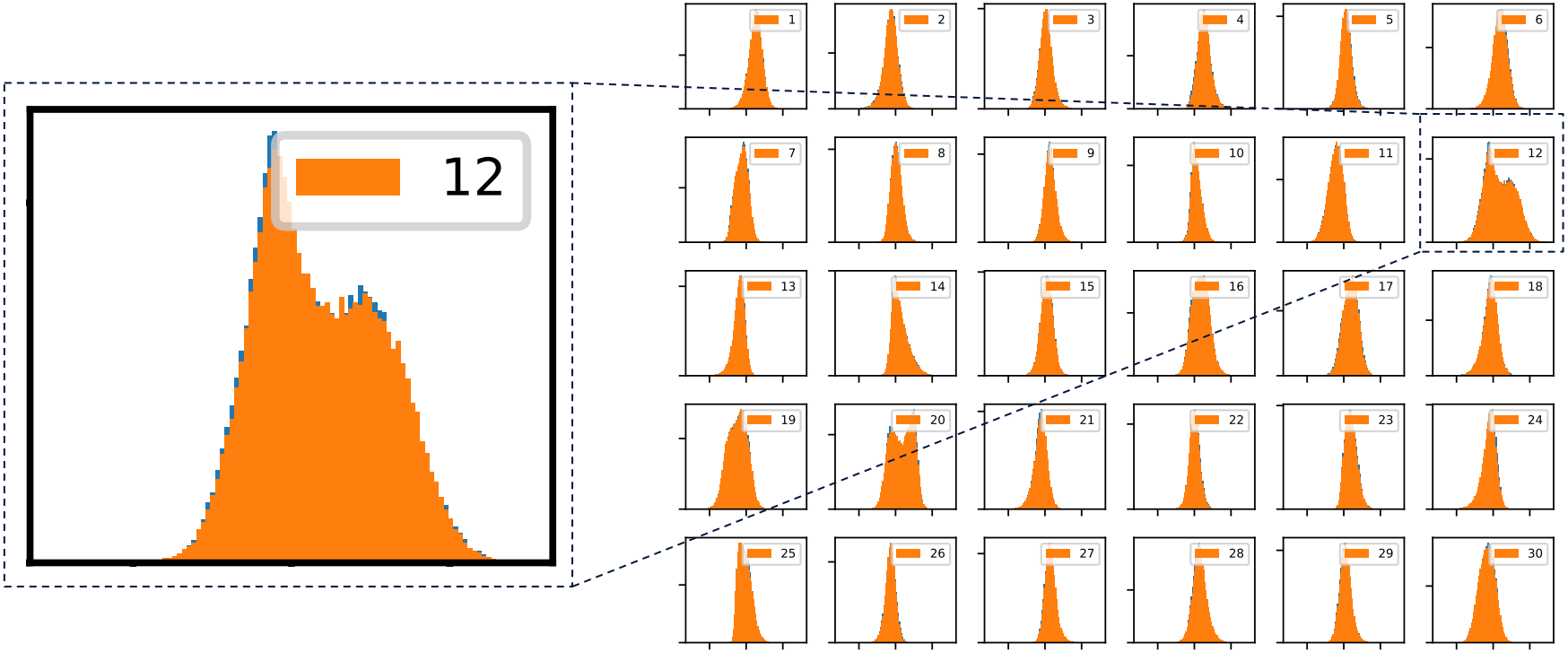


(a) Forward propagation

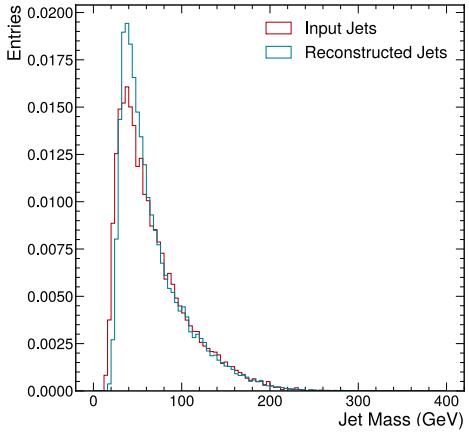


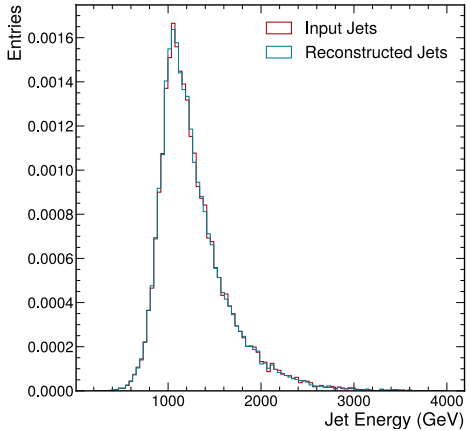
(b) Inverse propagation

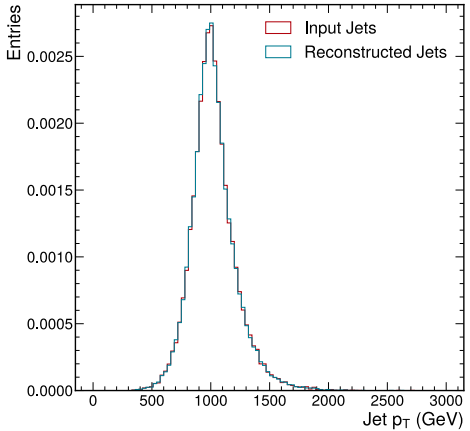
Transformation Output w/ RealNVP

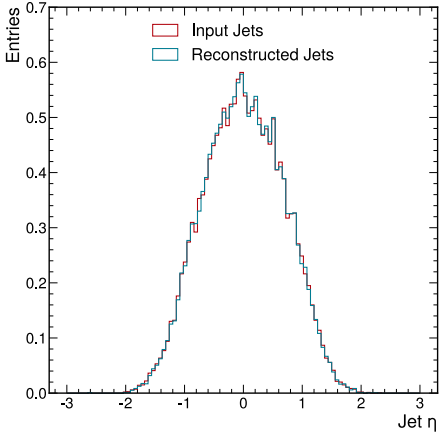


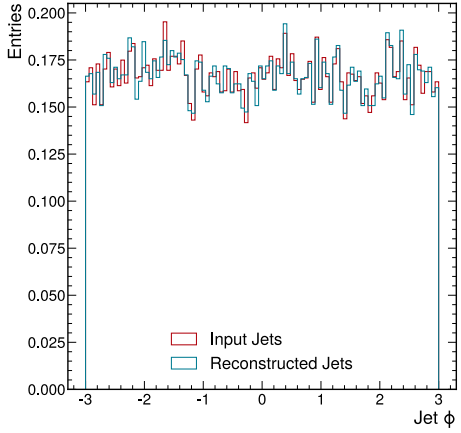
Preliminary Results: Better Reconstruction







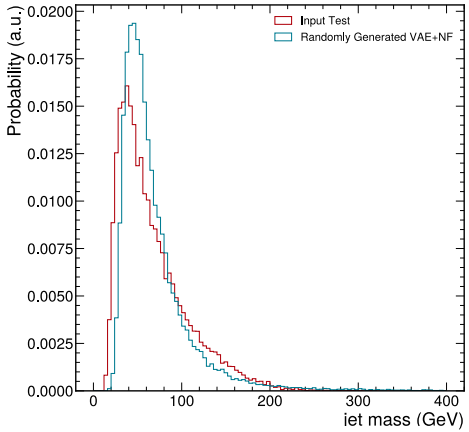


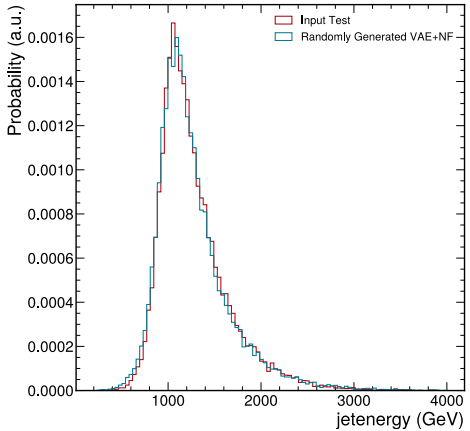


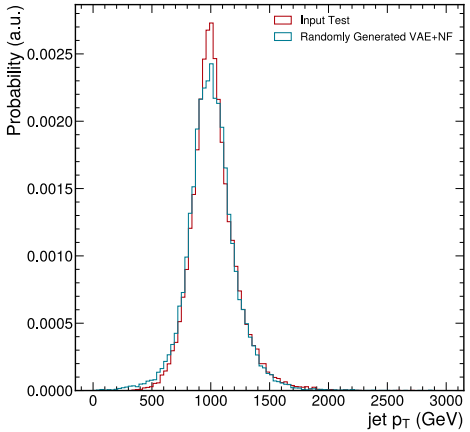
$EMD_{sum}=0.0018$

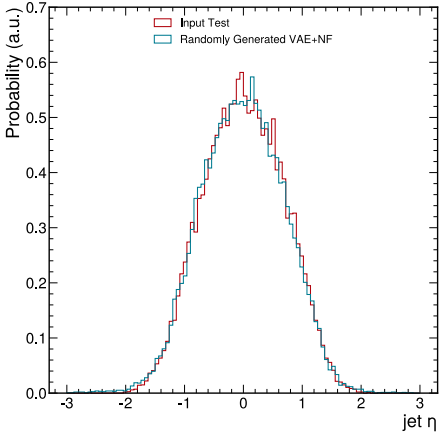
$EMD_{mass}=0.0013$

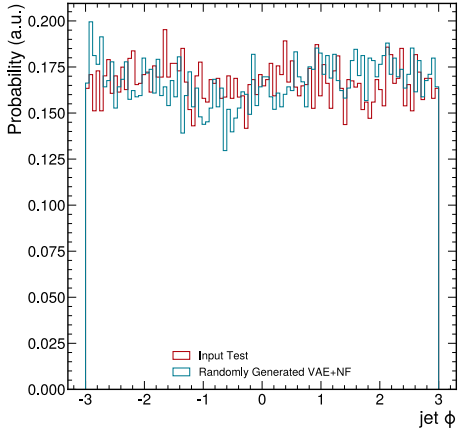
Preliminary Results w/ RealNVP: Generated Jets











$EMD_{sum}=0.0040$

$EMD_{mass}=0.0022$

Conclusions

Simulation of hadronic processes – and particularly jets – will still be a significant challenge for HL-LHC.

- Geant-based solutions may be too slow.
- Parametrized solutions may be too inaccurate.

Generative neural networks are a promising approach.

- Started with the “jet images” approach with CNNs.
- Moving on to graph-based neural nets (GNNs).
- Also considering GAN-based approaches.

Some of our other results:

- https://arxiv.org/abs/2012.00173
- https://arxiv.org/abs/2106.11535
- https://arxiv.org/abs/2203.00520