Can ML Accelerate Particle Simulations?

Chitrakshee Yede

Mentor: Prof. Sourabh Dube, Arnab Laha

Lagrangian To Lasers(L2L)

April 11th, 2024

Simulations in HEP

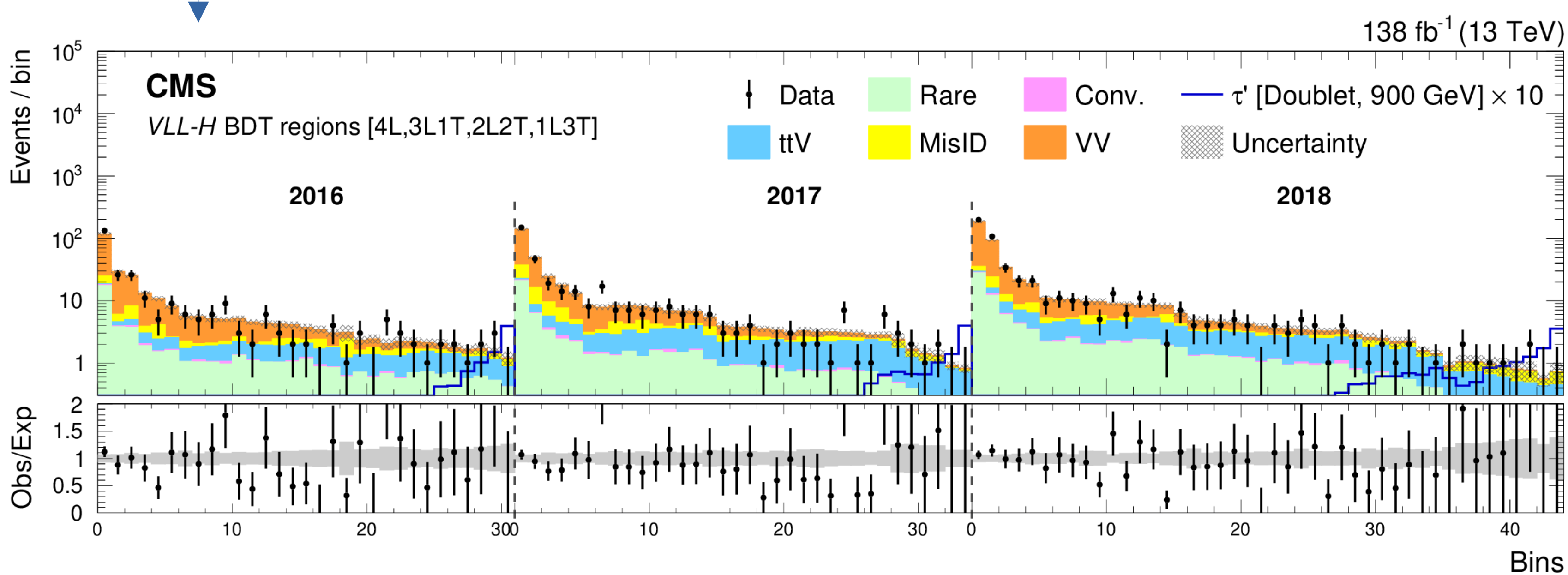
Simulation is crucial in HEP!

- Particle Interaction Modeling
- Detector performance and design
- Event Generation
- Background Estimation
- Signal and background discrimination

Inclusive nonresonant multilepton probes of new phenomena at √s= 13 TeV
Phys. Rev. D 105 (2022) 112007)

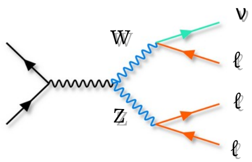
Used BDTs to discriminate between the signal and the background

BDTs (Boosted Decision Trees) are trained using large simulation samples



(Fig.19, Phys. Rev. D 105 (2022) 112007)

The Problem



Lets look at some numbers !!

	$N_{generated}$	$N_{training}$
2016	11.9M	124k
2017	10.8M	132k
2018	22M	233k

$N_{generated}$ - total number of WZ events generated, Madgraph + Pythia + Geant4 (CMSSW)
$N_{training}$ - number of WZ events used to train BDTs

~ 1% of the generated events used for training

Event selection optimizes signal-to-background ratio, excluding a significant portion of background events, particularly when training a signal vs. background classifier.

Higher statistics leads to better training performance!

How can we achieve that?

Can generate more events! (*Time-Consuming, computationally intensive*) : (

or

Can generate partial events (only the required variables) ? :)

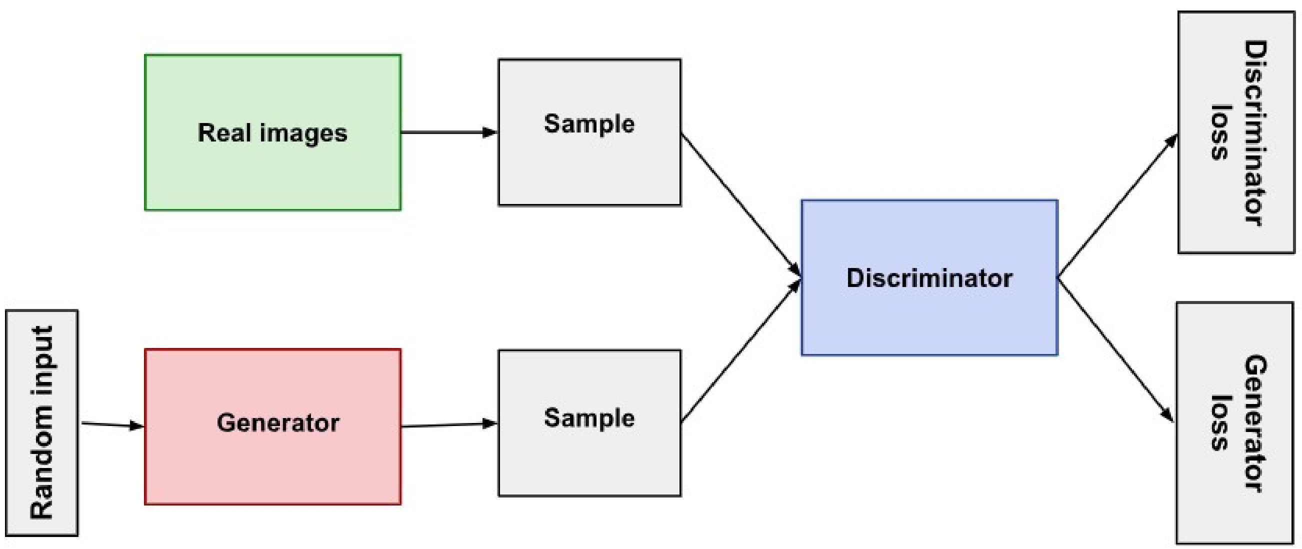
Using ML!!

Generative Models

Generative Adversarial Networks (GANs)

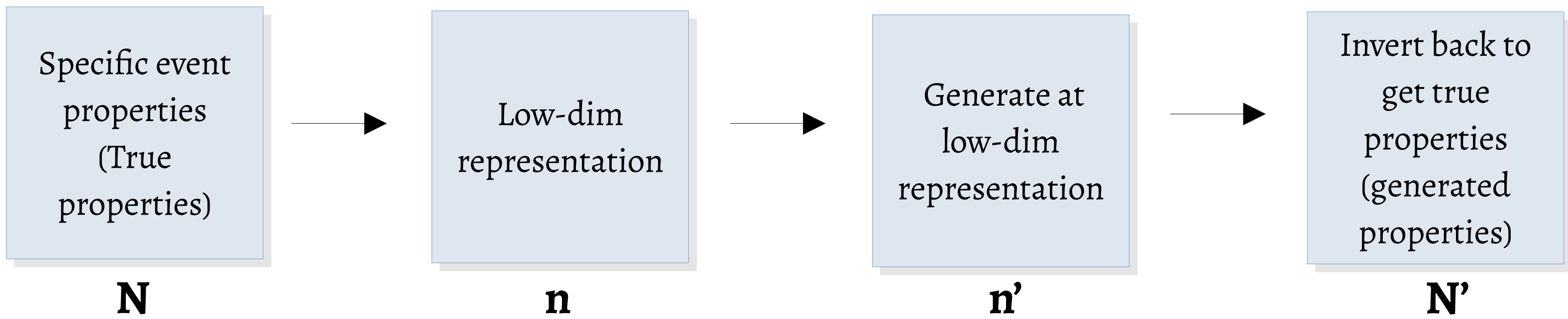
- Two sub-models – Generator and Discriminator that competes with each other!
- Generator : Generates pseudo data
- Discriminator : Distinguishes between real and pseudo data
- This iteration continues until generator succeeds to fool the discriminator

Generates a distribution by sampling from latent space and learning the correlation of features space



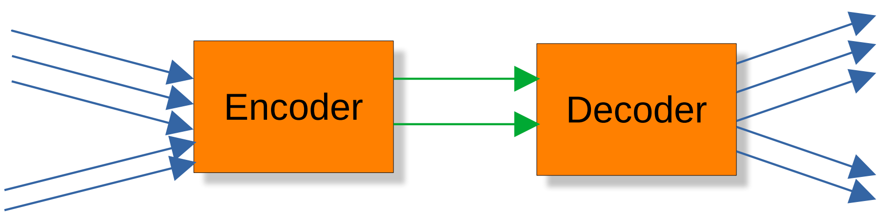
Complication: Slower and limited performance as number of features increases

The Idea

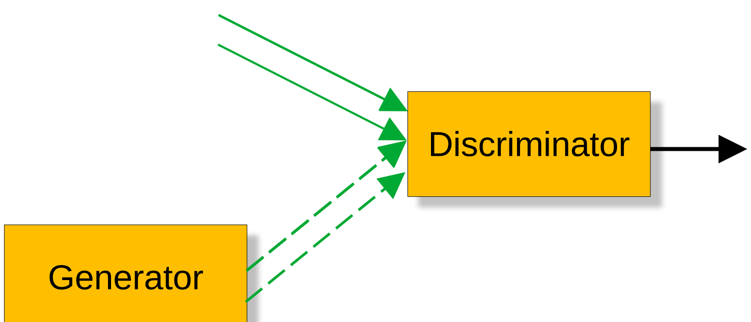


Dimensionality Reduced Generative Adversarial Network

Training

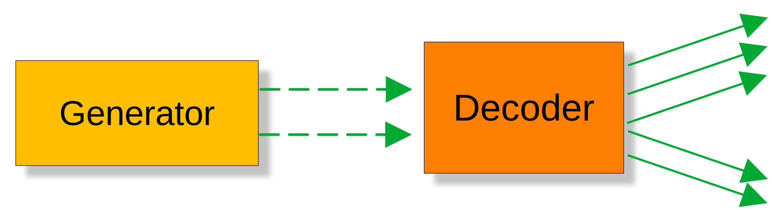


Train an Autoencoder to create a representation in lower dimension (**N→n**)



Train a GAN to generate **n’** variables indistinquishable from original **n**

Implementation



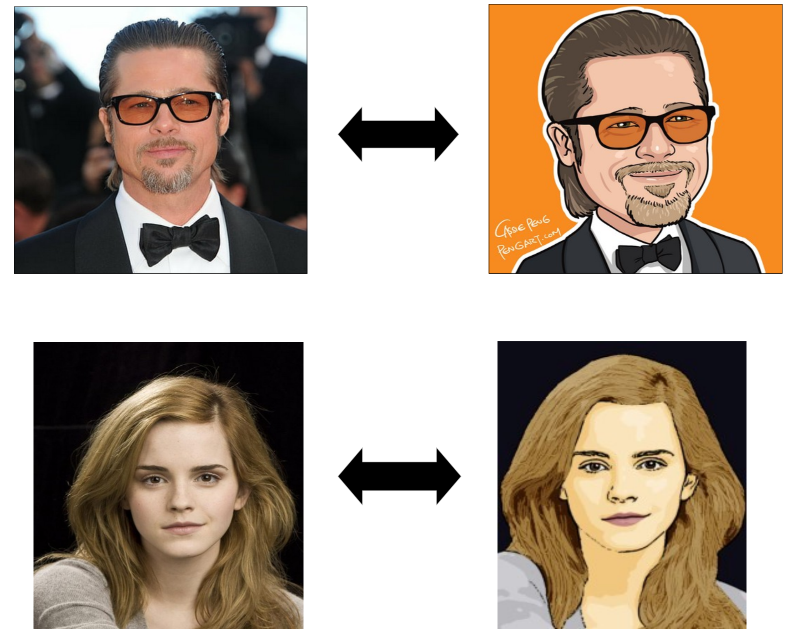
The Generator generate **n’** variables for as many events as needed.
The decoder decodes **n’** variables back to original higher dimension **N’** to be used in analysis.

The idea is that generated **N’** is comparable to original **N** .

Dimensionality Reduction - Autoencoders

Suppose we obtain a lower dimension representation of the data

Generating at lower dimension is easier!



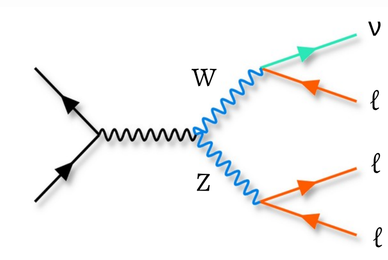
Goals:

- Should be faster
- To design a user-friendly pipeline
- Should be operable on personal workstations

Specific event properties (True properties) → Low-dim representation → Generate at low-dim representation → Invert back to get true properties (generated properties)

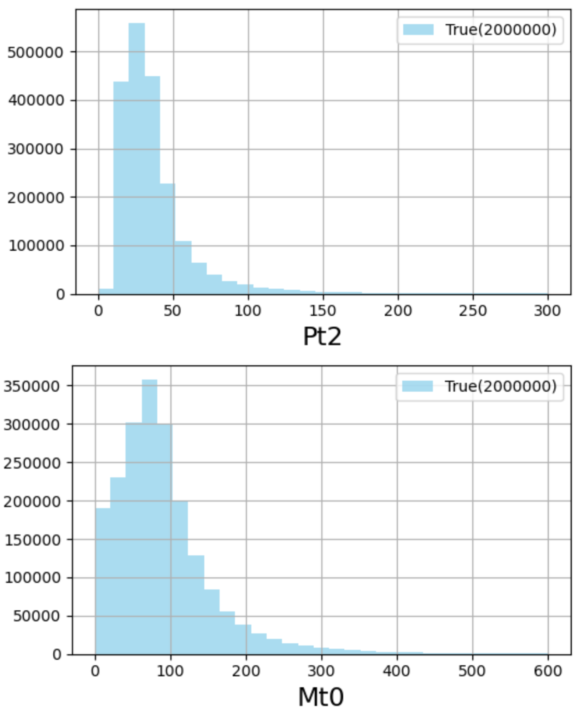
N → n → n' → N'

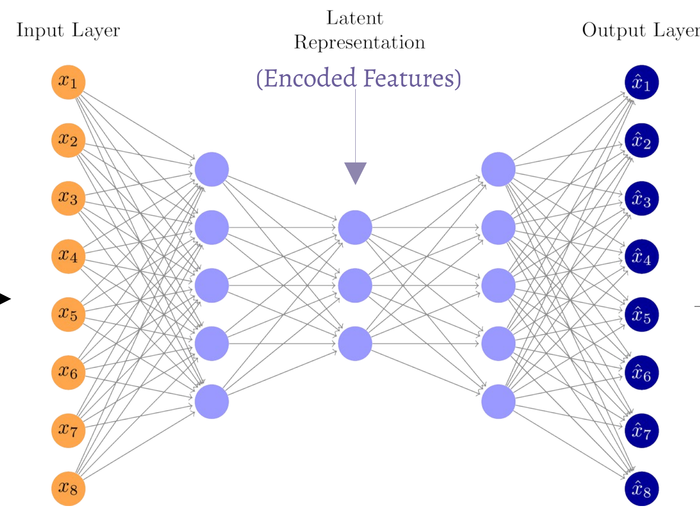
Autoencoders

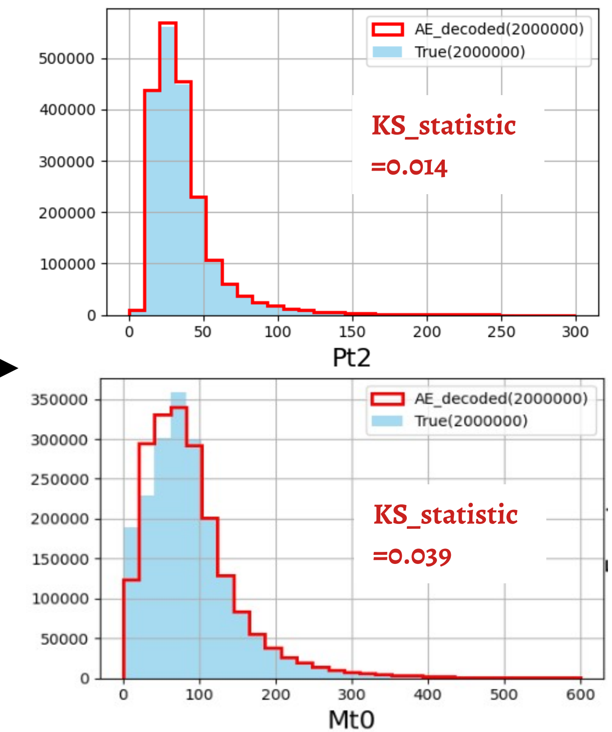


WZ process is a irreducible background for multilepton searches

Selected event properties: Particle momenta, Missing pT, Transverse Masses, and Angular information, HT, ST, Dijet mass







Compared using KS Test
(Similar distribution → KS statistic = 0)

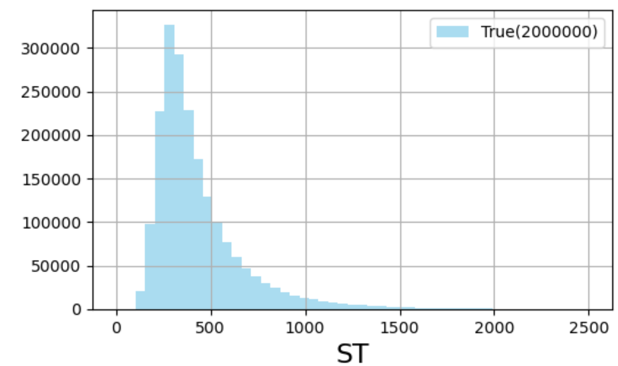
- Employed the Kolmogorov-Smirnov (KS) test to assess the match between true and reconstructed distributions.
- Utilized the average KS statistic for all variables as a quantitative metric.

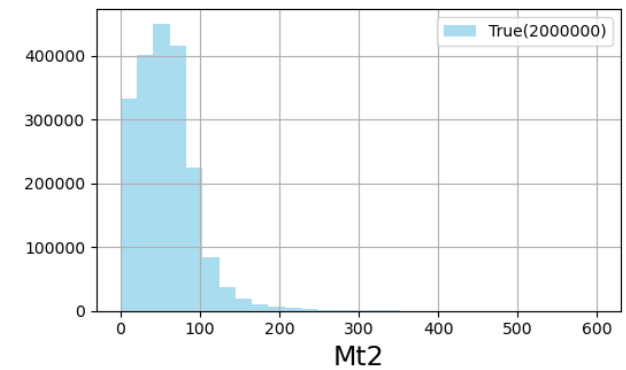
Encoding

- Trained the autoencoder
  (Encoder + Decoder model)
- Use the encoder model to get the lower dimension representation of the true variables.

<u>AE Configurations</u>

- Input Variables – StandardScalar to normalize
- NN architecture ~ 20k
- Activation function- LeakyReLU
- Epochs – 200
- Batch_Size- 2000
- Loss Function – MSE
- Trained on – 500k events
- Traning Time ~1.5 hours

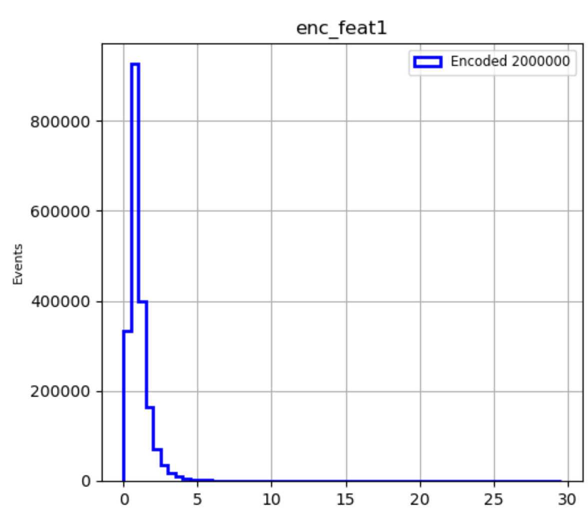


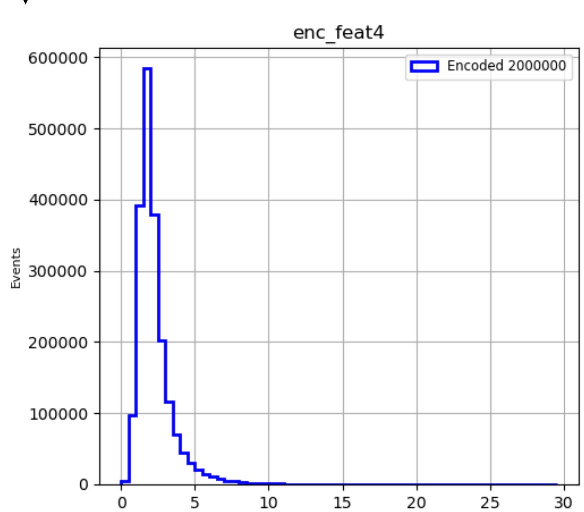


+7 more

True variables

Encoder



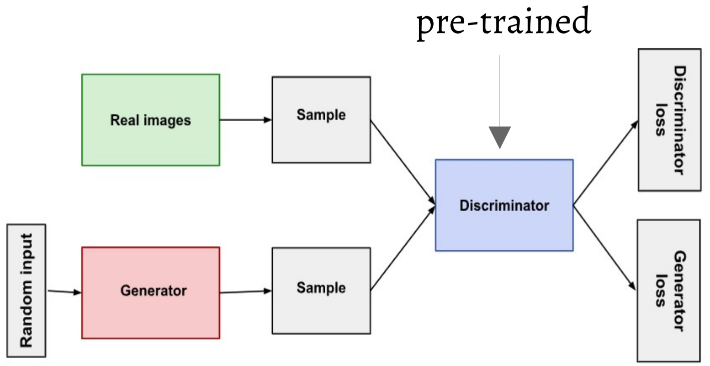


Lower dimension representation

+3 more

Training a GAN

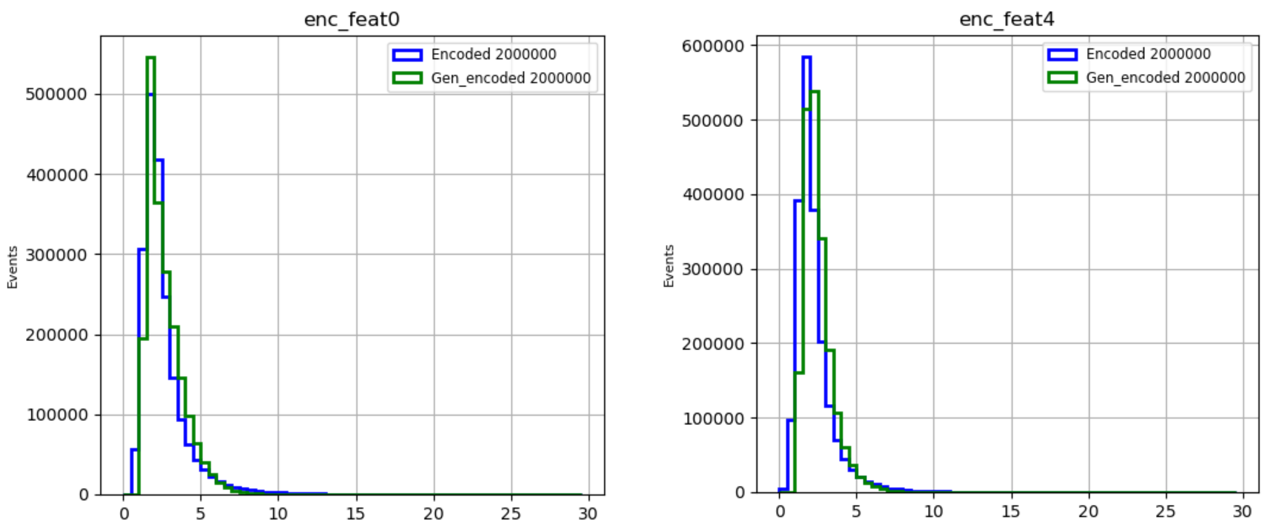
- Trained the GAN network using low-dim features.
- Used a pre-trained discriminator
- Generate the features using trained Generator.



Configurations

- Input - 5 low-dim encoded features
- NN architecture complexity ~ 10k (D and G)
- Activation function- LeakyReLU
- Latent_dimension -5
- Epochs – 20k
- Batch_Size- 5k
- Tranied on - 2M events
- Time Taken to train ~ 6 hours

GAN Output



Generated feature(Green) Vs. Encoded feature (Blue)

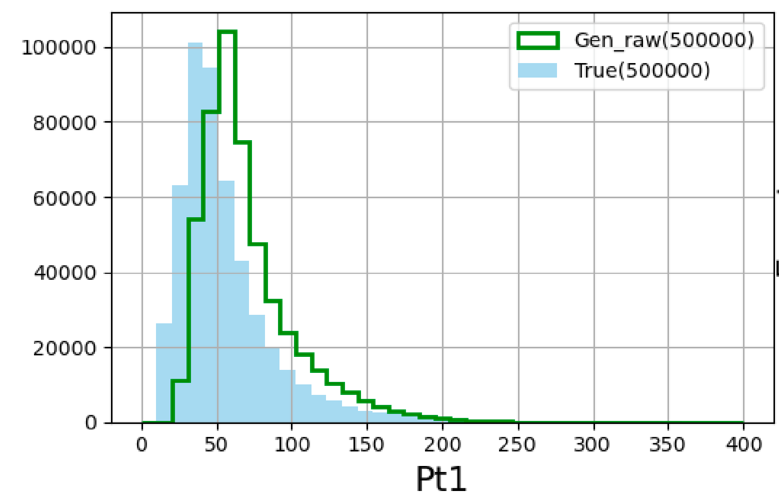
+3 more

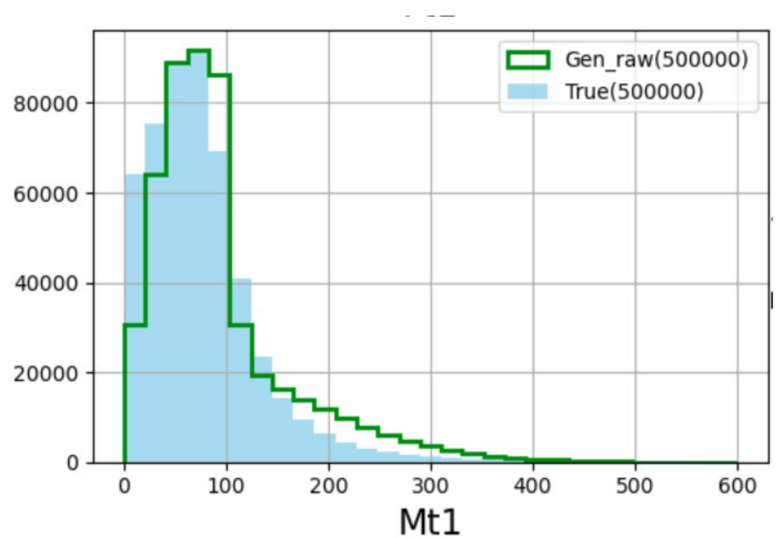
Decoding

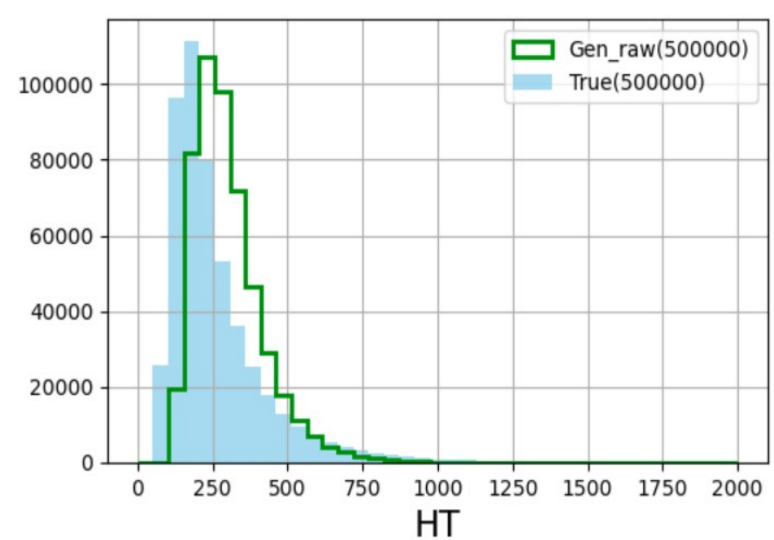
- Using decoder model from previously trained autoencoder, decode to get high dimension generated variables.
- Scaled them back using the inverse scalar transform.
- Compared with the original variables.

- **Currently, refining the models to enhance performance and improving these results.**
- **Aiming to validate the relevance and effectiveness of generated variables in preserving essential physics information.**

Tested on Independent 500k events







+6 more

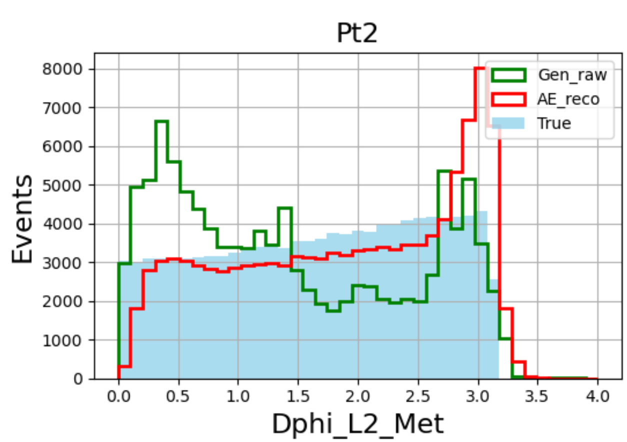
Explored NN architecture and Hyperparameter Tuning

For both AE and GAN architectures, Experimented with:

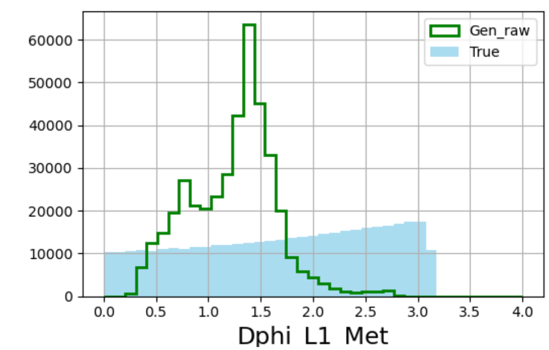
- NN Complexity
- Latent dimensions in GAN
- GAN with and without pre-trained discriminator
- Varying epochs, batch_size
- LeakyReLU instead of ReLU

AE Model (Encoding)	Complexity (trainable parameters)	KS test Score (mean)
7 to 2	~157k	0.074
7 to 4	~40k	0.048
7 to 4	~157k	0.026
10 to 5	~6k	0.075
10 to 5	~40k	0.063
10 to 5	~160k	0.054
9 to 5	~160k	0.027

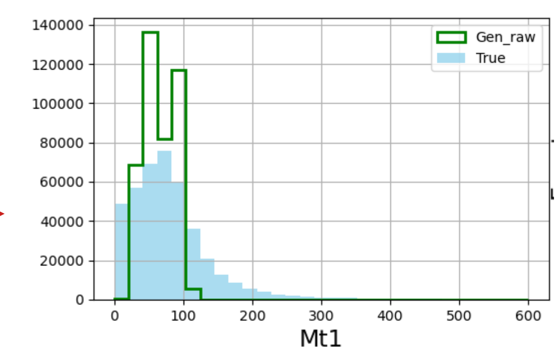
Angular Variable Reconstruction



Latent_dim = 7

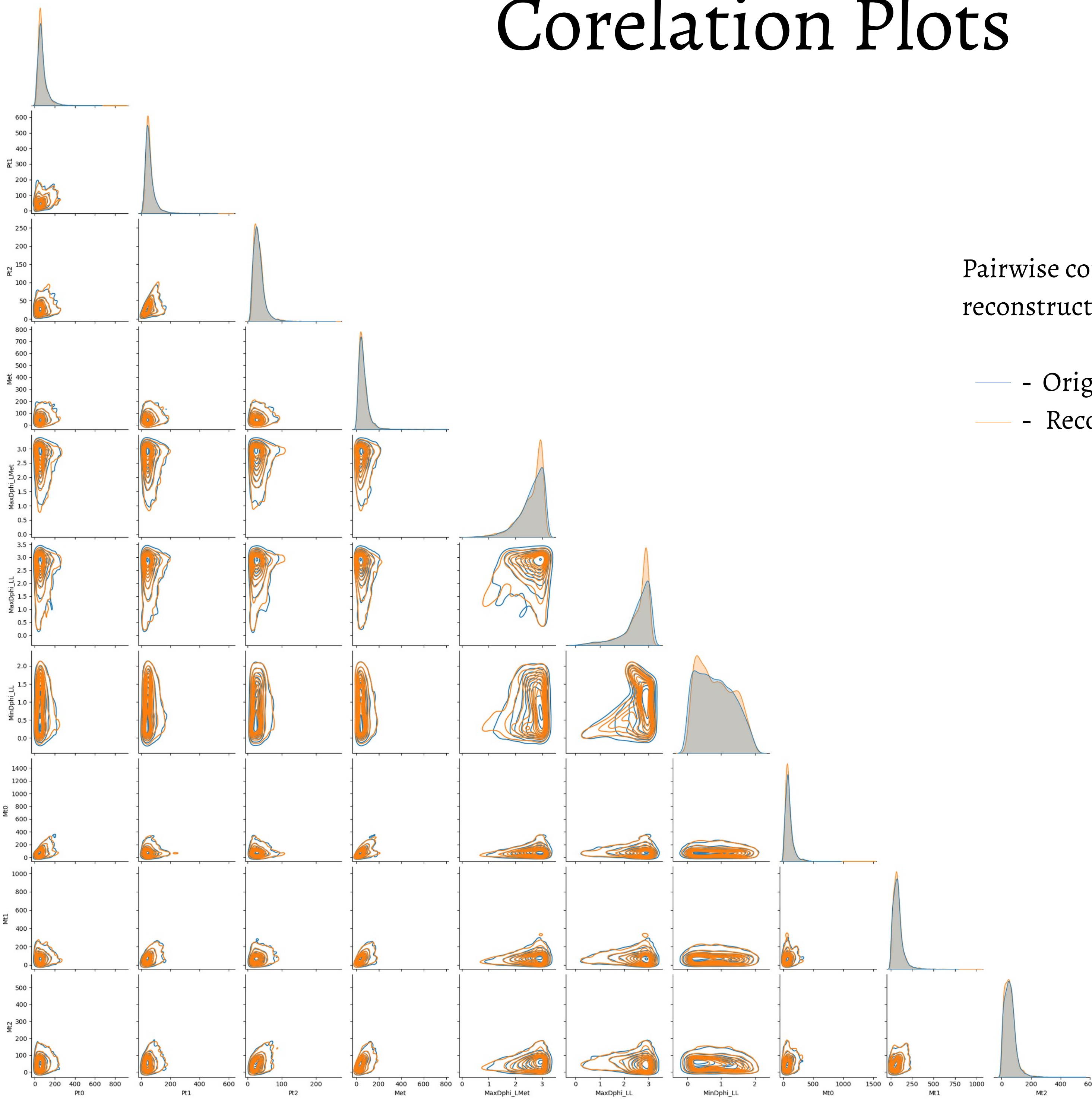


GAN complexity~170k



Thanks!

Corelation Plots



Pairwise correlation plot of true and reconstructed variables

- Original
- Reconstructed (AE)