Weekly Update

May 30, 2025

Leo Zhu, Denaisha Kraft

Investigating Preprocessing Issue

- Some events had voxels with higher energies than the incident energy

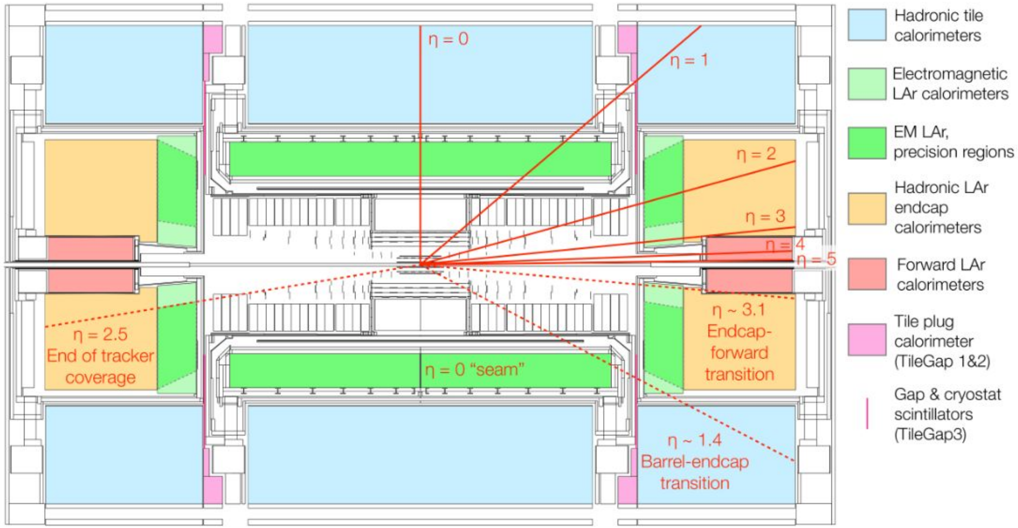
- From eta=0.20 to eta=0.95, each file had between 2 to 9 cases of this
  - eta=0.20 and eta=0.30 had no cases

- Checked which layers this was for:
  - One case in layer 1 with an incident energy of 131072 MeV (eta=0.60)
  - All other cases in layer 0 with an incident energy of 256 MeV

ATLAS layer dictionary

https://link.springer.com/article/10.1140/epjc/s10052-021-09402-3

- The number of the layer points to a specific region in ATLAS and a specific type of detector.



#	Layer	#	Layer
0	PreSamplerB	12	TileBar0
1	EMB1	13	TileBar1
2	EMB2	14	TileBar2
3	EMB3	15	TileGap1
4	PreSamplerE	16	TileGap2
5	EME1	17	TileGap3
6	EME2	18	TileExt0
7	EME3	19	TileExt1
8	HEC0	20	TileExt2
9	HEC1	21	FCal0
10	HEC2	22	FCal1
11	HEC3	23	FCal2

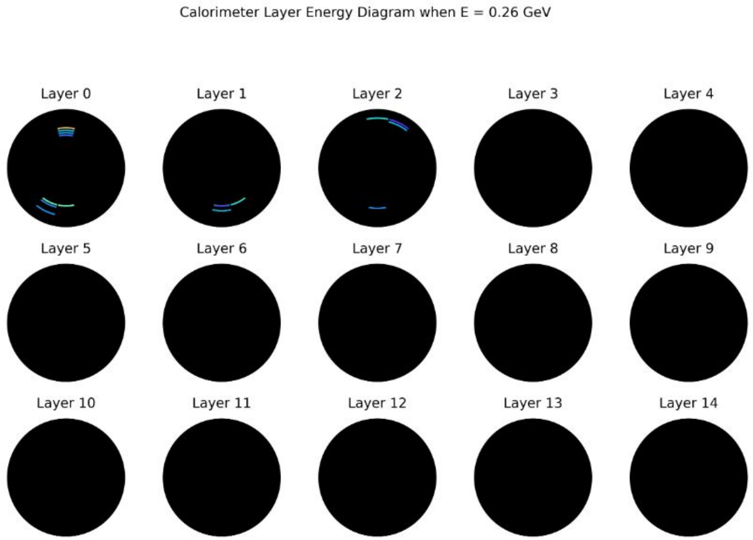
Event Displays

- Examples for eta=0.60

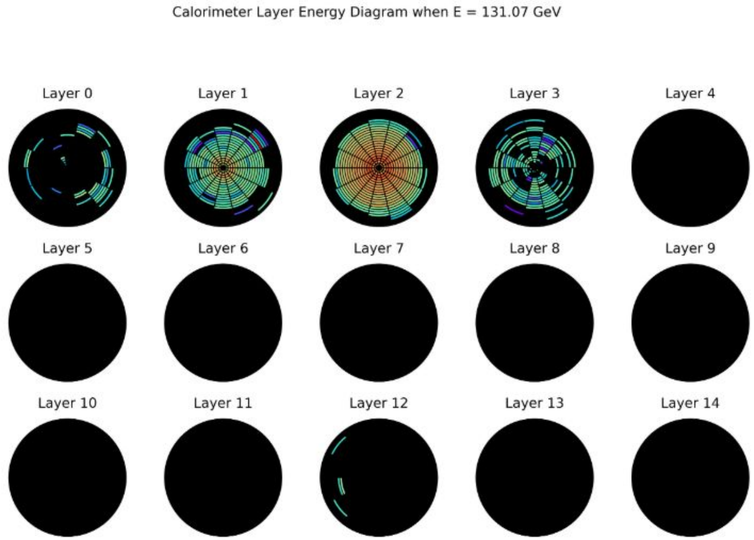
These events were removed from the dataset

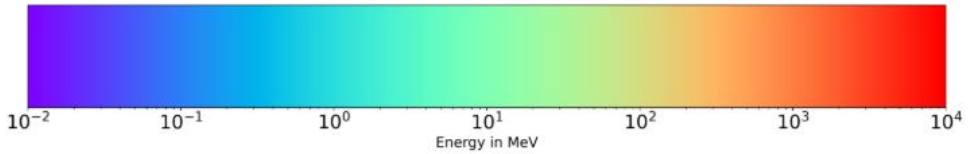
```
Event 94162, voxel 575,  LAYER = 1
Incident energy = 131072.0
Voxel energy = 3568258829516800.00
Voxel / Incident = 27223654400.0000
```

Issue in layer 0



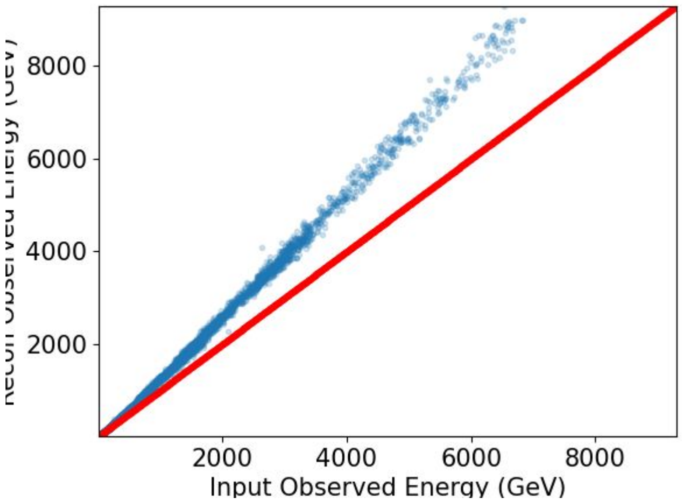
Issue in layer 1

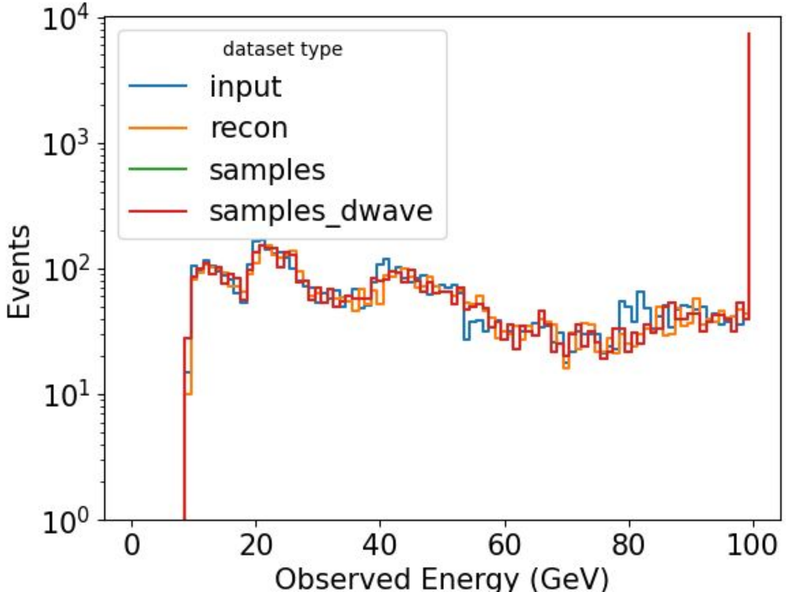




Smearing

```
noise = torch.empty_like(incident_energies).uniform_(-0.5, 0.5)
perturbed_energies = torch.exp(torch.log(incident_energies) + noise)
scale = perturbed_energies/incident_energies
```



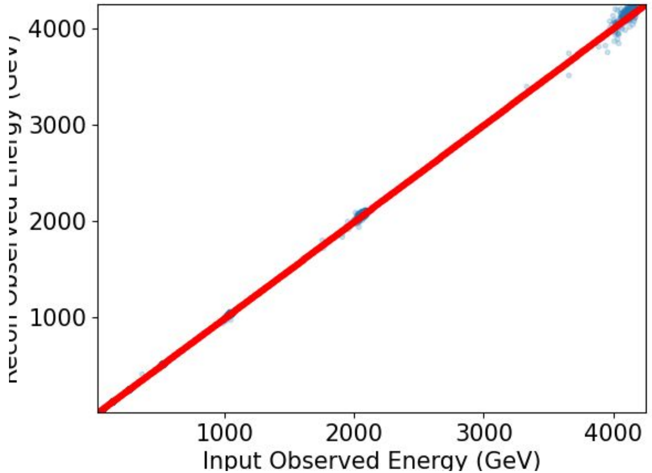


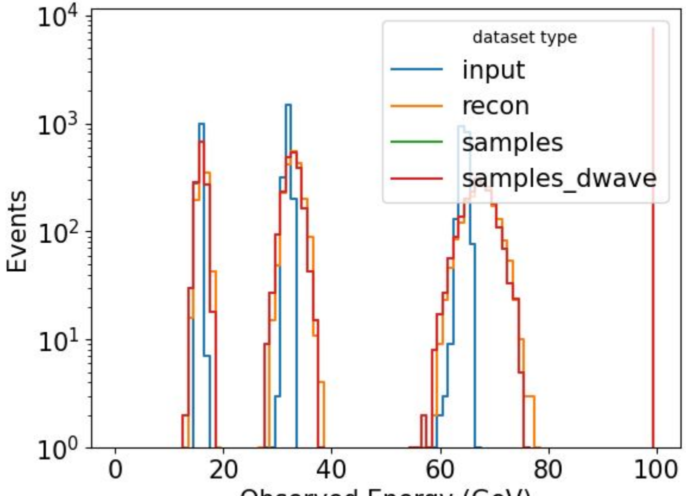
Adapting Ian’s Model

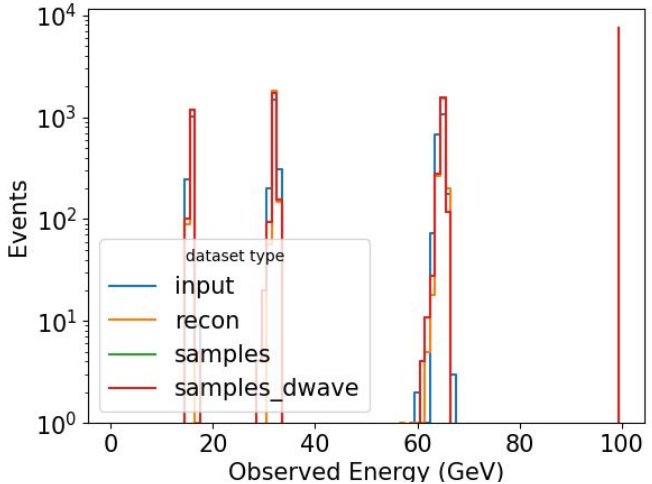
- What’s New?
  - New decoder where **skip connections** send information from slices of the latent space to subdecoders
- Adapting to the Atlas dataset
  - Calo dataset: z = 45, r = 9, phi = 16
  - Atlas dataset: z=7, r = 24, phi = 14
- Updated encoder and decoder convolutions to match to new target dimensions
  - Requires trial and error
  - Slight flaw: cropping required for phi
  - Potentially problematic since z changes for different eta

Performance of the New Model

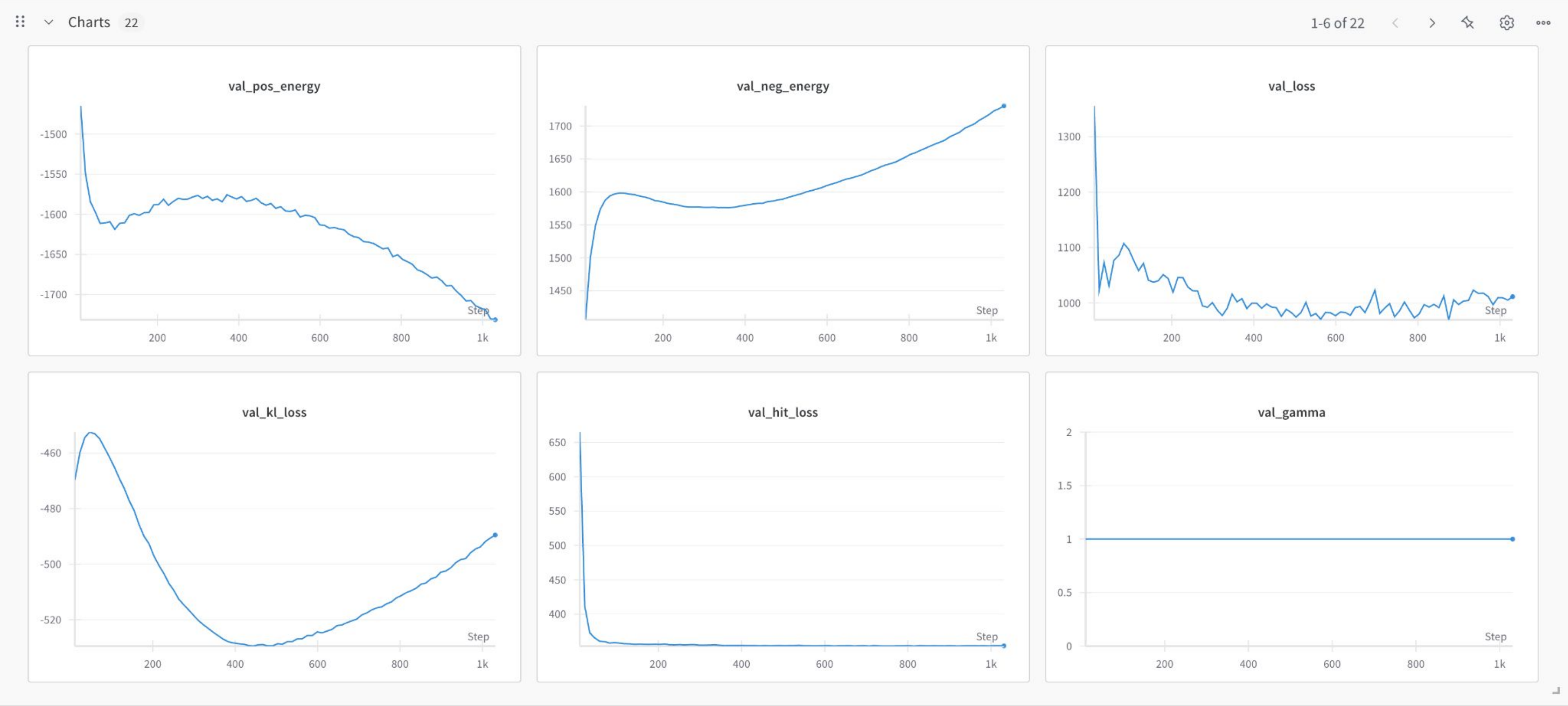
- Much better energy reconstruction
  - Even with discrete energies!
- Old model struggled learning discrete energies



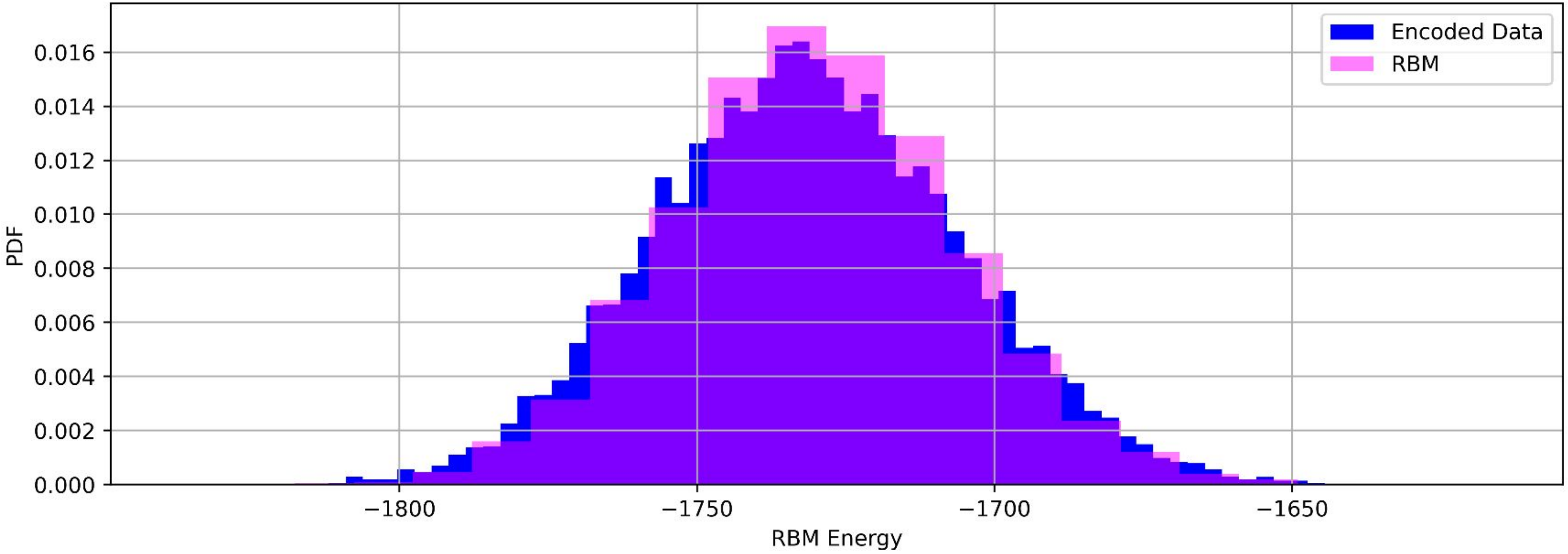




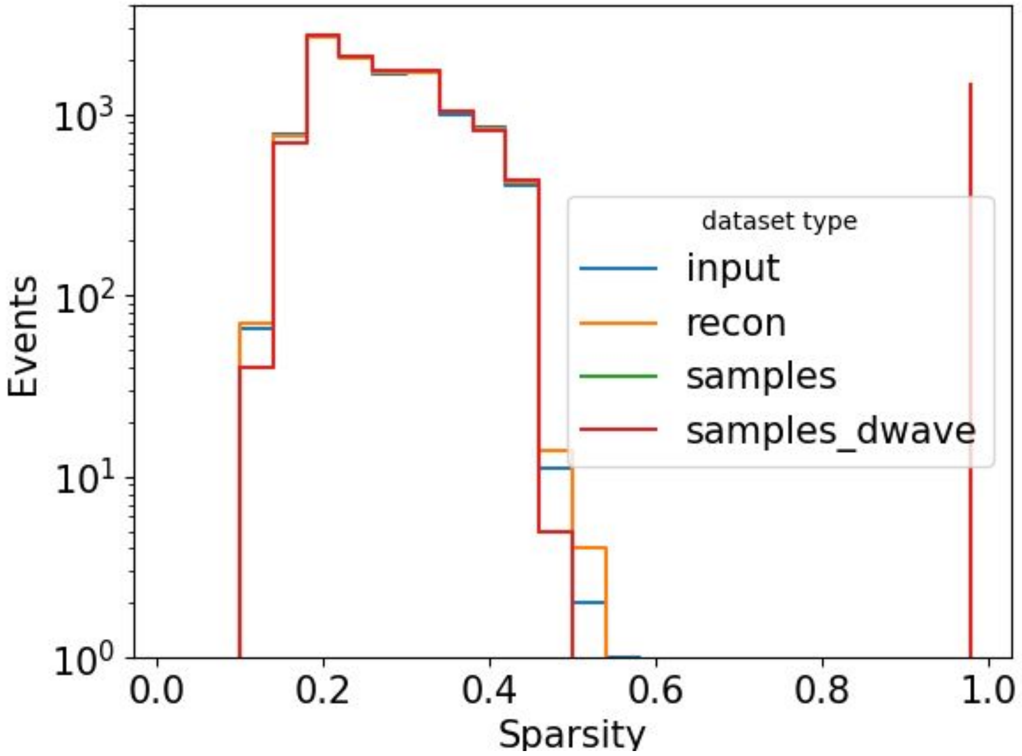
Performance of the New Model



Performance of the New Model

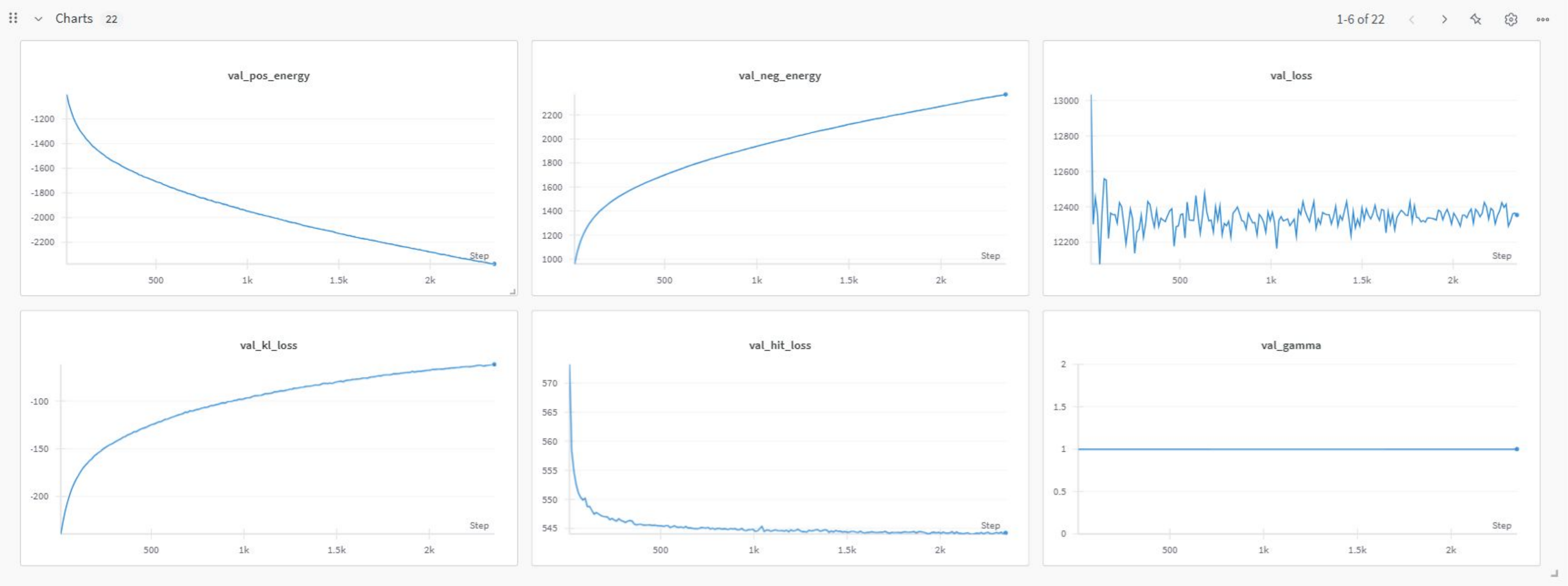


Performance of the New Model

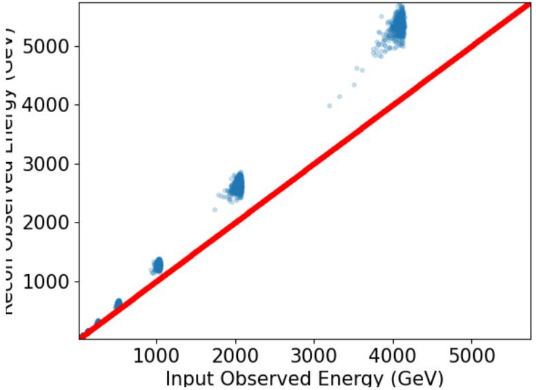


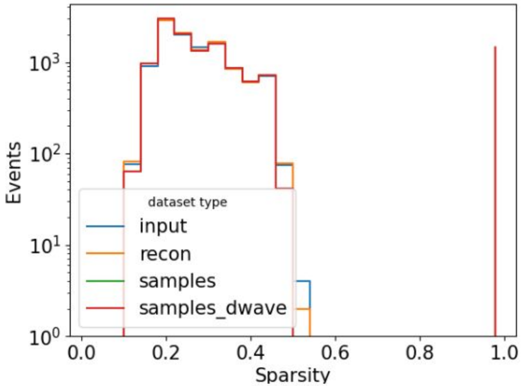
Performance of the Old Model

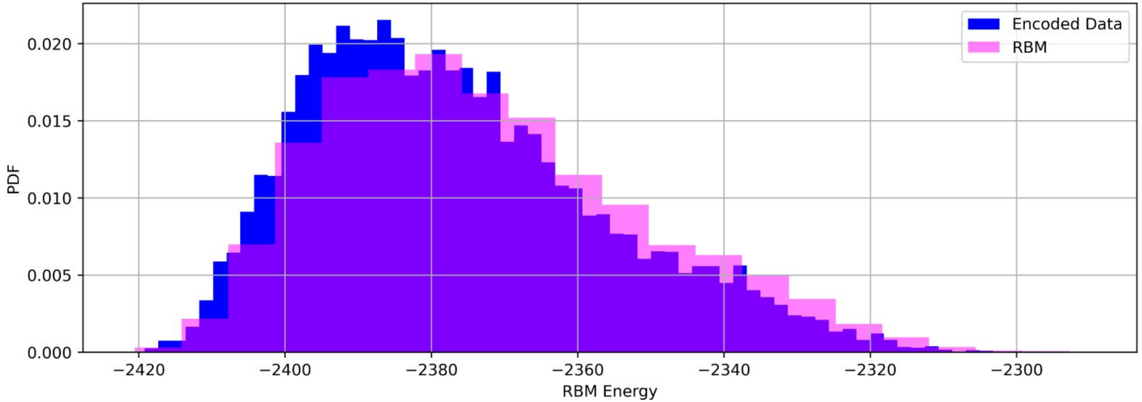
For eta=0.30



Performance of the Old Model







Comparisons to FastCaloSim

Things done by FastCaloSim:

- Compare the energy deposited in the layer vs the eta value
- Compare eta asymmetries

Our dataset is only positive eta values

- If we had negative eta files could look at asymmetries in voxel energy deposits across positive and negative eta values within a specific layer to compare GEANT4 and the model

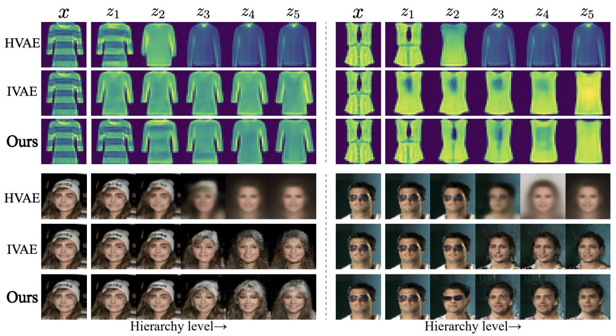
Other Comparisons

Things done by FastCaloSim:

- Chi squared test statistic to compare GEANT4 and the model
    - Using both the formula from ROOT for unweighted histograms and the formula from FastCaloChallenge
- KS-statistic (scipy.stats.ks_2samp)
    - Compares underlying distributions
- Wasserstein distance (scipy.stats.wasserstein_distance)
    - Similarity metric between two probability distributions
- Ratios of the means and standard deviations
- Ratio of the standard error of the mean (SEM)

Proposal for Future Models: Reinforcement Learning

- Motivation
  - Is our VAE really adding new meaning in each hierarchy? How do we prevent posterior collapse?
  - Reinforcement learning has been wildly successful in generative models (LLMs, Chain of Thought)
- Brief RL summary
  - Train an agent to make good choices (winning at Go, generating meaningful latent codes)
  - Agent is always in a state associated with a reward, and can take actions to enter different states
  - Over many iterations, agent learns to take actions that maximize reward



Implementing RL

- Technique from literature:
    - Paper name: Improving Unsupervised Hierarchical Representation with Reinforcement Learning
    - Reward function:

$$R(T) = \log p(x|z_{\geq t}) - KL(q(z_t|z_{<t}, x)||p(z_t))$$

        - How good is the decoder considering only the latent variables outputted this layer onward?
        - How **different** is the distribution q compared to the generic posterior p(x)
- Some advantages
    - Very successful in paper mentioned above: possibility of learning richer latent features
    - No change at all to inference time, no increase in parameter size