Weekly Update

May 23, 2025

Leo Zhu, Denaisha Kraft

Files

- Using new ATLAS dataset
- 14 different files, each for a different eta
- [0.20,0.25,0.30,0.35,0.40,0.45,0.50,0.55,0.60,0.65,0.70,0.85,0.90,0.95]

Made concatenated files using the layers with nonzero events and the incident energies for each eta

Example:

```
using these layers: [0, 1, 2, 3, 12, 13, 14]
/fast_scratch_1/caloqvae/data/atlas_regular_cat/dataset_eta_020_positive_cat.hdf5

using these layers: [0, 1, 2, 3, 12, 13, 14, 15, 16, 17, 18, 19, 20]
/fast_scratch_1/caloqvae/data/atlas_regular_cat/dataset_eta_085_positive_cat.hdf5
```

Training on ATLAS Data: Challenges

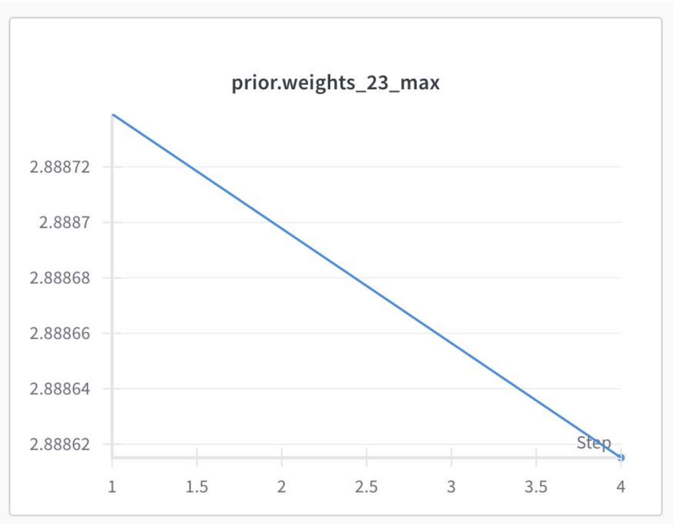
- Eta = 0.20, 0.30 train successfully, but all other ones fail
- Problem: p_state in RBM is filled with NaNs!
- No NaNs in data for any eta…

```
  File "/home/leozhu/CaloQVAE/models/samplers/pgbs.py", line 62, in _p_state
    raise RuntimeError("p_activations contains invalid values")
RuntimeError: p_activations contains invalid values
NaNs/Infs detected in p_activations!
p_activations has 38656 NaNs
```

Some Hypotheses

Debugging the RBM

- Idea: Weight explosion?
    - Reduced learning rate
    - Weight decay term in loss function
    - Weight initialization
- Tracking weights in WandB
    - No explosion!
- Problem must be in encoder output



Culprit: Processed Data

- Manually ran forward passes in notebook to replicate NaNs
- Always the same few data points that cause problems
- Problematic step: data reduction/normalization
- When voxel energy is greater than incident energy, end up taking the log of a negative value

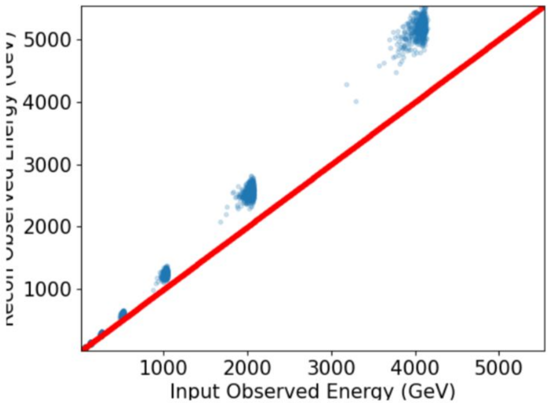
```
for batch_idx, (inputs, labels) in enumerate(train_loader):
    in_data_og, true_energy, in_data_flat = engine._preprocess(inputs, labels)
    in_data = engine._reduce(in_data_og, true_energy)
    if torch.isnan(in_data).any():
        nan_index = torch.nonzero(torch.isnan(in_data), as_tuple=False)[0]
        print("bad batch: ", batch_idx)
        print(nan_index)
        print(in_data[nan_index[0]][nan_index[1]-5:nan_index[1]+5])
        print(in_data_og[nan_index[0]][nan_index[1]-5:nan_index[1]+5])
        print(true_energy[nan_index[0]])
print("done")
```

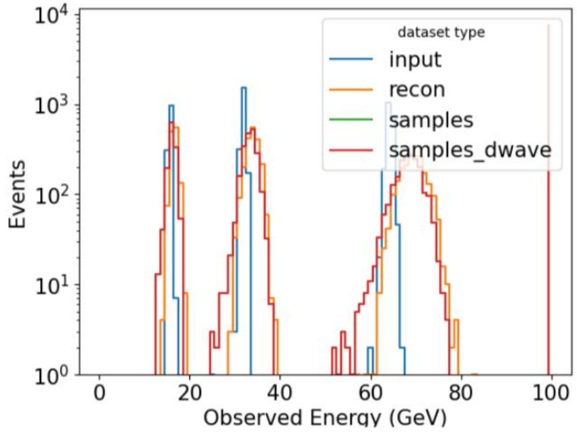
```
def _reduce(self, in_data, true_energy, R=1e-7):
    """
    CaloDiff Transformation Scheme
    """
    scaling_energy = max(torch.max(in_data), torch.max(true_energy))
    ϵ = in_data/true_energy #*self.e_scale
    x = R + (1-2*R)*ϵ
    u = torch.log(x*(1-R)/(R*(1-x)))/self._std
    return u
```

Models Currently Training

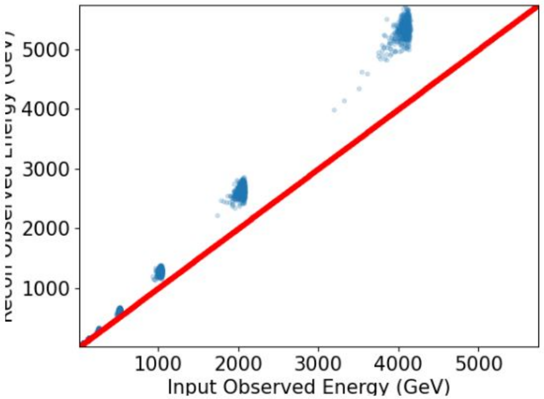
Currently training eta=0.2 and eta=0.3

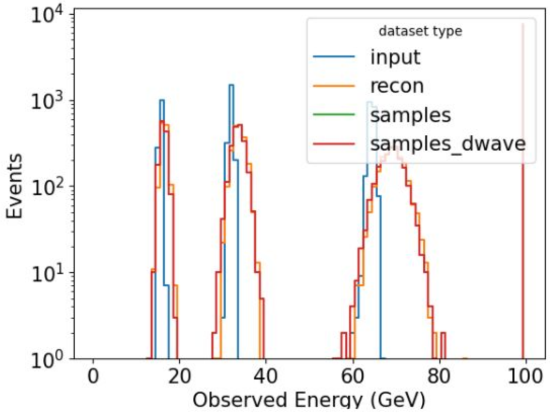
For eta=0.2:





For eta=0.3:



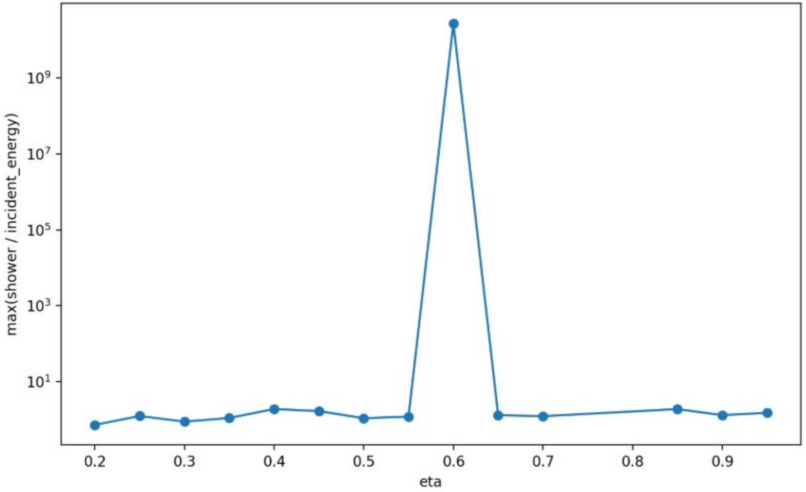


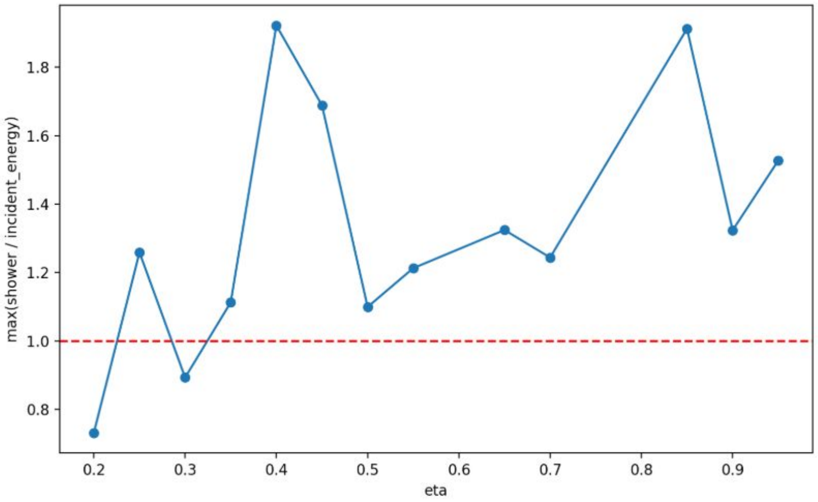
Investigating Individual Files

Looking at voxels divided by the incident energy and taking the max value

Very large value for eta=0.6

- Omitting eta=0.6
- Only eta=0.2 and eta=0.3 were able to train
- Every other file has this max greater than 1





Data Clean Up

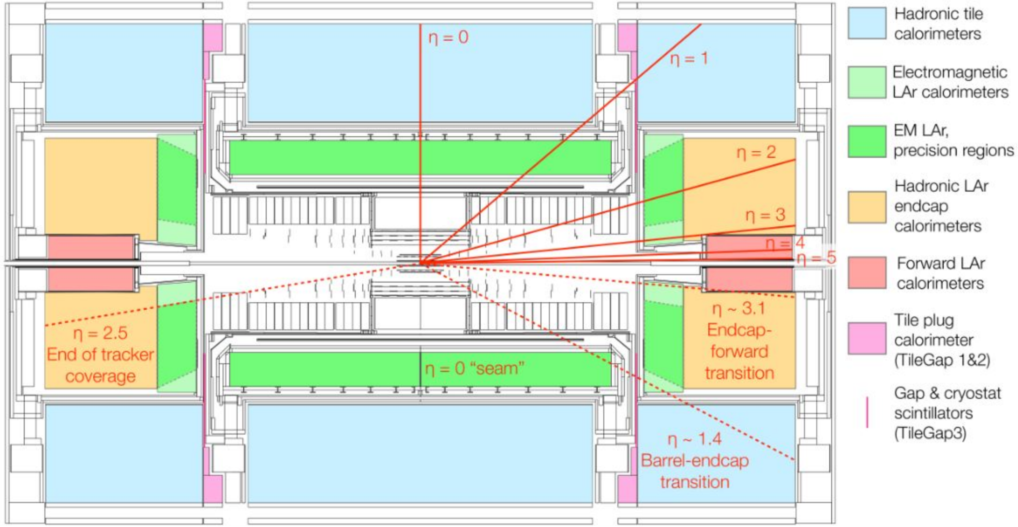
- Other than eta = 0.6, all other problematic energies are 256 MeV
- No negative voxels in any eta slices
- New dataset created without problematic entries: plans to start training ASAP

```
Eta: 025, incident energy, tensor([256.], dtype=torch.float64)
Eta: 025, incident energy, tensor([256.], dtype=torch.float64)
Eta: 035, incident energy, tensor([256.], dtype=torch.float64)
Eta: 035, incident energy, tensor([256.], dtype=torch.float64)
Eta: 040, incident energy, tensor([256.], dtype=torch.float64)
Eta: 040, incident energy, tensor([256.], dtype=torch.float64)
Eta: 040, incident energy, tensor([256.], dtype=torch.float64)
Eta: 045, incident energy, tensor([256.], dtype=torch.float64)
Eta: 045, incident energy, tensor([256.], dtype=torch.float64)
Eta: 050, incident energy, tensor([256.], dtype=torch.float64)
Eta: 050, incident energy, tensor([256.], dtype=torch.float64)
Eta: 055, incident energy, tensor([256.], dtype=torch.float64)
Eta: 055, incident energy, tensor([256.], dtype=torch.float64)
Eta: 060, incident energy, tensor([256.], dtype=torch.float64)
Eta: 060, incident energy, tensor([256.], dtype=torch.float64)
Eta: 060, incident energy, tensor([131072.], dtype=torch.float64)
Eta: 060, incident energy, tensor([256.], dtype=torch.float64)
Eta: 065, incident energy, tensor([256.], dtype=torch.float64)
Eta: 065, incident energy, tensor([256.], dtype=torch.float64)
Eta: 065, incident energy, tensor([256.], dtype=torch.float64)
Eta: 065, incident energy, tensor([256.], dtype=torch.float64)
Eta: 065, incident energy, tensor([256.], dtype=torch.float64)
Eta: 070, incident energy, tensor([256.], dtype=torch.float64)
Eta: 070, incident energy, tensor([256.], dtype=torch.float64)
Eta: 085, incident energy, tensor([256.], dtype=torch.float64)
Eta: 085, incident energy, tensor([256.], dtype=torch.float64)
Eta: 085, incident energy, tensor([256.], dtype=torch.float64)
Eta: 085, incident energy, tensor([256.], dtype=torch.float64)
Eta: 085, incident energy, tensor([256.], dtype=torch.float64)
Eta: 090, incident energy, tensor([256.], dtype=torch.float64)
Eta: 090, incident energy, tensor([256.], dtype=torch.float64)
Eta: 090, incident energy, tensor([256.], dtype=torch.float64)
Eta: 095, incident energy, tensor([256.], dtype=torch.float64)
Eta: 095, incident energy, tensor([256.], dtype=torch.float64)
Eta: 095, incident energy, tensor([256.], dtype=torch.float64)
Eta: 095, incident energy, tensor([256.], dtype=torch.float64)
Eta: 095, incident energy, tensor([256.], dtype=torch.float64)
Eta: 095, incident energy, tensor([256.], dtype=torch.float64)
Eta: 095, incident energy, tensor([256.], dtype=torch.float64)
Eta: 095, incident energy, tensor([256.], dtype=torch.float64)
Eta: 095, incident energy, tensor([256.], dtype=torch.float64)
```

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