04/26/2024 GELATO Weekly

Max Cohen

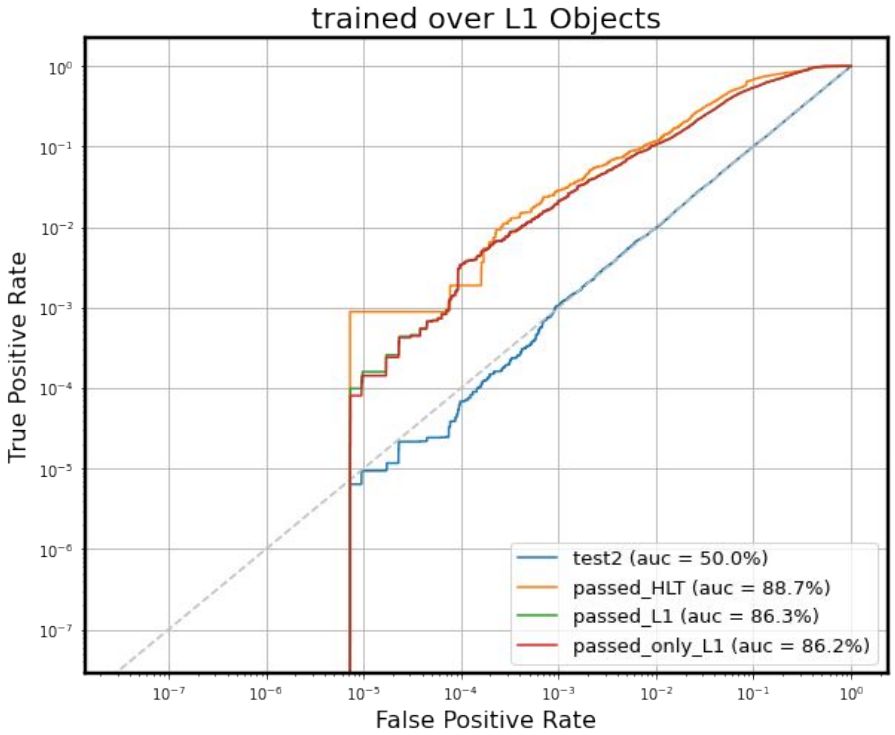
Updates from this week

- I no longer think my code has bugs
  - I ran the 40MHz data through my code and got good results
  - Reproduced similar results to Liam by training over the L1 data

- I found some MC that includes L1 objects, tested the L1 model
  - jj JZ2 and JZ4
  - HHbbtthadhad

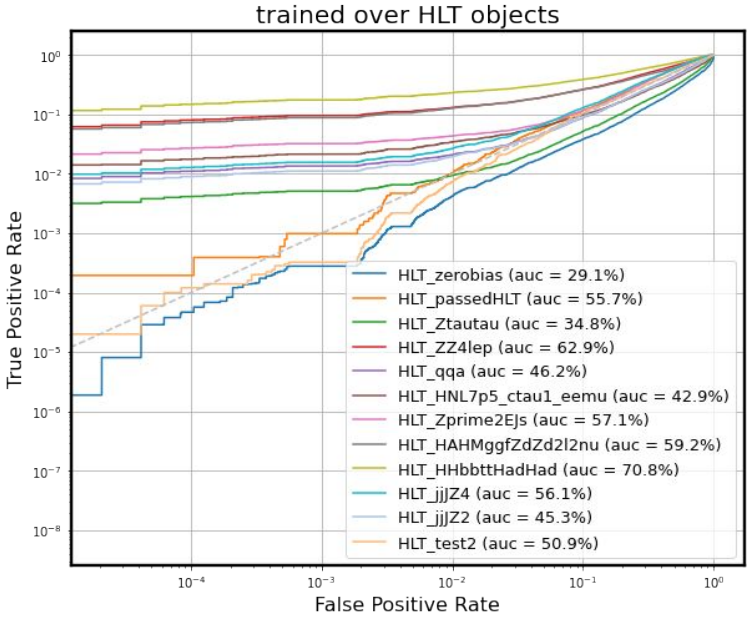
- Ran preliminary study of Compression Factor^{TM} as an AD benchmarking metric with 40MHz data

Trained over all L1 events (including zerobias)

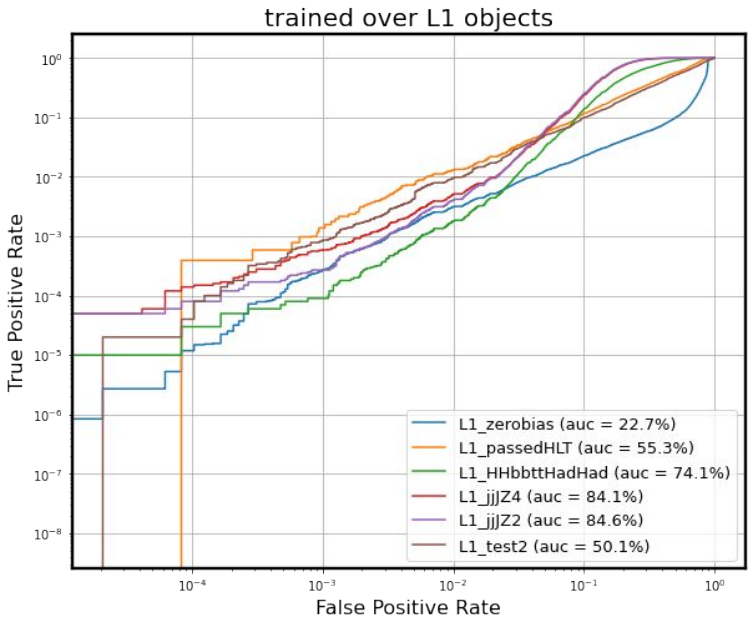


So I don’t think my code has a bug

So I put the L1 MC through the HLT model trained over L1 objects

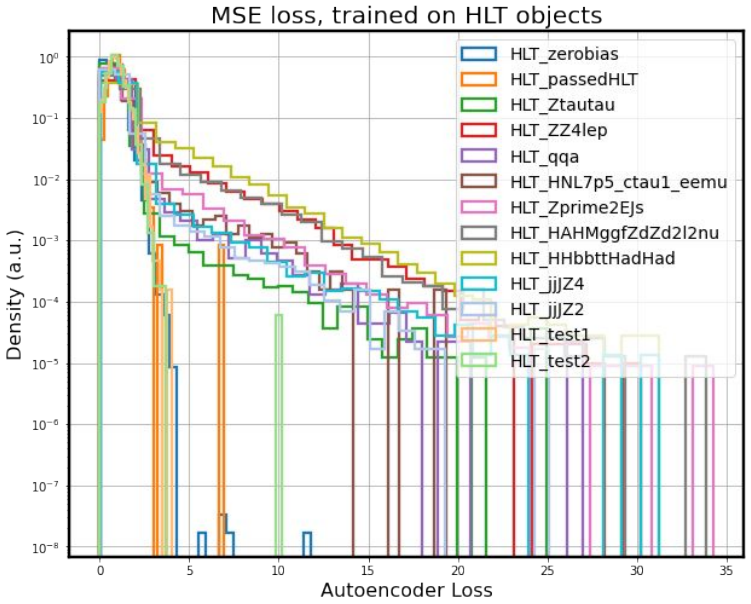


Trained over HLT events with HLT objects

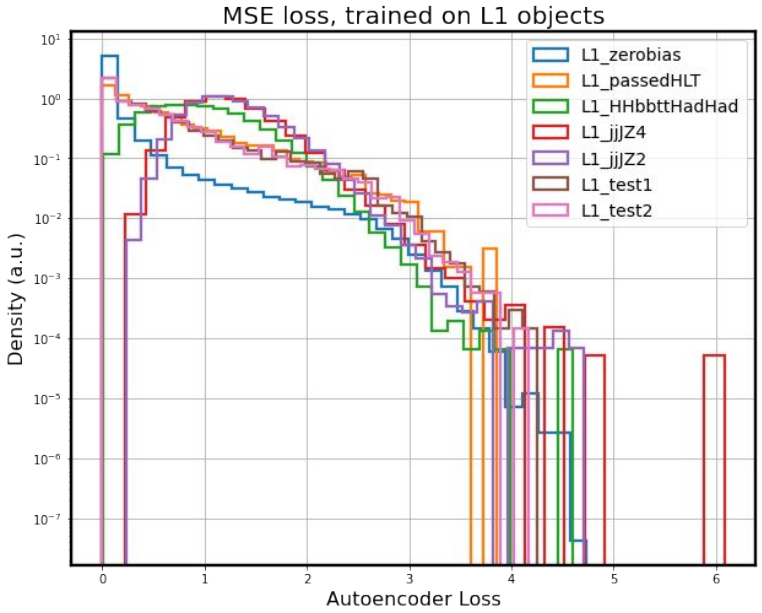


Trained over HLT events with L1 objects

So I put the L1 MC through the HLT model trained over L1 objects



Trained over HLT events with HLT objects



Trained over HLT events with L1 objects

So I put the L1 MC through the HLT model trained over L1 objects

In general, these networks (especially the HLT objects network) are very volatile between trainings.

Is it possible the MC is not a good approximation of this EB run because of some of the tags being used?

How good is the network at reconstructing each variable?

HLT event - reconstruction mean for each entry::

-0.002 +- 0.083	0.027 +- 1.308	0.009 +- 1.518
-0.002 +- 0.053	-0.035 +- 1.791	0.015 +- 1.734
-0.002 +- 0.037	0.045 +- 0.832	-0.010 +- 1.770
-0.001 +- 0.032	-0.014 +- 0.921	0.006 +- 1.769
-0.003 +- 0.030	-0.001 +- 1.075	-0.005 +- 1.754
-0.005 +- 0.027	-0.073 +- 2.179	-0.013 +- 1.726
-0.007 +- 0.027	-0.168 +- 1.864	0.007 +- 1.705
-0.008 +- 0.025	-0.047 +- 1.951	0.002 +- 1.659
-0.009 +- 0.024	-0.057 +- 1.845	-0.011 +- 1.617
-0.009 +- 0.022	-0.032 +- 1.734	0.016 +- 1.574
-0.092 +- 0.018	0.008 +- 0.955	-0.076 +- 1.324
-0.011 +- 0.019	0.039 +- 1.094	-0.039 +- 1.199
0.063 +- 0.194	0.066 +- 0.987	-1.147 +- 1.088
-0.052 +- 0.036	-0.003 +- 0.651	-0.041 +- 0.841
-0.038 +- 0.010	0.043 +- 0.377	-0.051 +- 0.320
-0.030 +- 0.011	0.000 +- 0.496	0.001 +- 0.479
-0.013 +- 0.039	0.017 +- 1.023	-0.007 +- 1.169
-0.015 +- 0.038	0.021 +- 0.994	-0.014 +- 1.191
-0.014 +- 0.026	0.021 +- 0.997	0.001 +- 1.470
-0.059 +- 0.043	0.617 +- 0.548	-0.012 +- 0.435

L1 event - reconstruction mean for each entry::

-0.077 +- 0.171	-0.373 +- 1.597	0.011 +- 1.143
-0.059 +- 0.101	-0.292 +- 2.691	0.042 +- 1.454
-0.047 +- 0.065	-0.250 +- 3.054	0.156 +- 1.420
-0.045 +- 0.047	-0.227 +- 3.134	0.113 +- 1.279
-0.042 +- 0.035	-0.212 +- 3.089	0.099 +- 1.104
-0.043 +- 0.030	-0.132 +- 3.265	0.158 +- 0.965
-0.041 +- 0.025	-0.088 +- 3.305	0.180 +- 0.874
-0.042 +- 0.023	-0.056 +- 4.545	0.120 +- 0.805
-0.039 +- 0.019	-0.137 +- 3.301	0.125 +- 0.690
-0.040 +- 0.017	-0.273 +- 3.355	0.102 +- 0.681
-0.030 +- 0.085	-0.040 +- 0.988	0.002 +- 1.073
-0.025 +- 0.043	-0.067 +- 1.006	-0.021 +- 1.364
-0.015 +- 0.021	-0.076 +- 0.966	0.049 +- 1.352
-0.000 +- 0.002	-0.057 +- 0.798	-0.070 +- 1.147
-0.000 +- 0.001	-0.063 +- 0.690	-0.102 +- 0.962
0.000 +- 0.003	-0.067 +- 0.693	-0.103 +- 0.894
-0.048 +- 0.099	-0.062 +- 0.971	-0.026 +- 0.942
-0.026 +- 0.040	-0.122 +- 1.092	-0.005 +- 1.333
-0.013 +- 0.017	-0.013 +- 1.004	0.033 +- 1.230
0.000 +- 0.180	0.165 +- 0.249	0.001 +- 0.484

I also tested this out on the L1 model trained on ALL events

```
L1 event - reconstruction mean for each entry::
-0.076 +- 0.168     0.516 +- 1.533     -0.375 +- 1.115
-0.098 +- 0.095     -0.096 +- 1.444     0.071 +- 1.192
-0.081 +- 0.059     -0.403 +- 0.935     0.038 +- 1.007
-0.070 +- 0.039     -0.042 +- 0.866     -0.003 +- 0.847
-0.061 +- 0.028     0.040 +- 1.184     0.050 +- 0.733
-0.054 +- 0.021     0.498 +- 3.119     -0.064 +- 0.645
-0.049 +- 0.017     0.103 +- 0.985     -0.022 +- 0.560
-0.044 +- 0.014     0.133 +- 0.889     0.017 +- 0.505
-0.040 +- 0.013     0.105 +- 0.825     0.040 +- 0.468
-0.037 +- 0.011     0.010 +- 0.780     -0.037 +- 0.413
-0.029 +- 0.091     0.191 +- 1.041     -0.456 +- 1.312
0.004 +- 0.038     0.207 +- 1.120     -0.141 +- 1.348
-0.167 +- 0.298     -0.538 +- 2.195     -0.659 +- 1.396
-0.072 +- 0.017     0.095 +- 0.722     -0.018 +- 0.873
-0.039 +- 0.011     0.088 +- 0.408     -0.088 +- 0.423
-0.035 +- 0.015     0.183 +- 0.506     -0.064 +- 0.383
0.008 +- 0.094     0.183 +- 0.888     -0.505 +- 1.148
-0.043 +- 0.034     0.188 +- 0.905     -0.508 +- 1.408
-0.035 +- 0.015     0.110 +- 0.726     -0.083 +- 0.819
0.417 +- 0.378     1.248 +- 0.344     0.716 +- 0.422
```

I also tested this out on the L1 model trained on ALL events

```
L1 event - reconstruction mean for each entry::
-0.076 +- 0.168     0.516 +- 1.533     -0.375 +- 1.115
-0.098 +- 0.095     -0.096 +- 1.444     0.071 +- 1.192
-0.081 +- 0.059     -0.403 +- 0.935     0.038 +- 1.007
-0.070 +- 0.039     -0.042 +- 0.866     -0.003 +- 0.847
-0.061 +- 0.028     0.040 +- 1.184     0.050 +- 0.733
-0.054 +- 0.021     0.498 +- 3.119     -0.064 +- 0.645
-0.049 +- 0.017     0.103 +- 0.985     -0.022 +- 0.560
-0.044 +- 0.014     0.133 +- 0.889     0.017 +- 0.505
-0.040 +- 0.013     0.105 +- 0.825     0.040 +- 0.468
-0.037 +- 0.011     0.010 +- 0.780     -0.037 +- 0.413
-0.029 +- 0.091     0.191 +- 1.041     -0.456 +- 1.312
0.004 +- 0.038     0.207 +- 1.120     -0.141 +- 1.348
-0.167 +- 0.298     -0.538 +- 2.195     -0.659 +- 1.396
-0.072 +- 0.017     0.095 +- 0.722     -0.018 +- 0.873
-0.039 +- 0.011     0.088 +- 0.408     -0.088 +- 0.423
-0.035 +- 0.015     0.183 +- 0.506     -0.064 +- 0.383
0.008 +- 0.094     0.183 +- 0.888     -0.505 +- 1.148
-0.043 +- 0.034     0.188 +- 0.905     -0.508 +- 1.408
-0.033 +- 0.015     0.110 +- 0.726     -0.083 +- 0.819
0.417 +- 0.378     1.248 +- 0.344     0.716 +- 0.422
```

Is the L1 network just a MET detector?

Can make 2d hist of MET vs AD score, maybe Liam can do this

Study of compression factor as AD benchmark metric

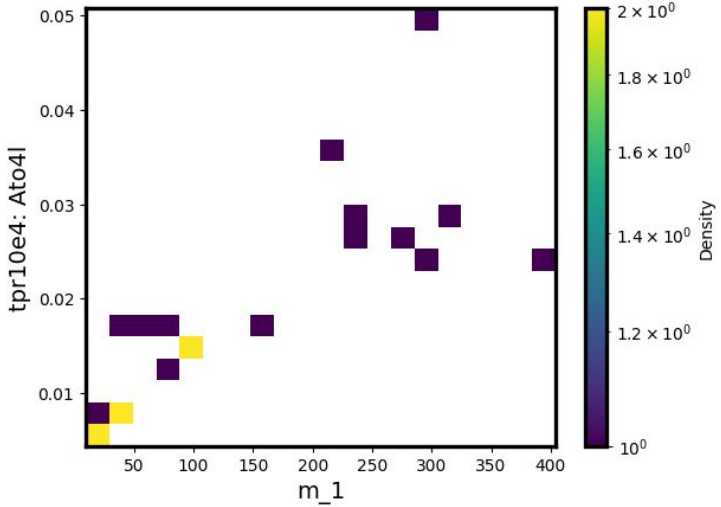
- I took a regular AE with MSE loss, and trained 20 different versions with a hyperparameter tuner

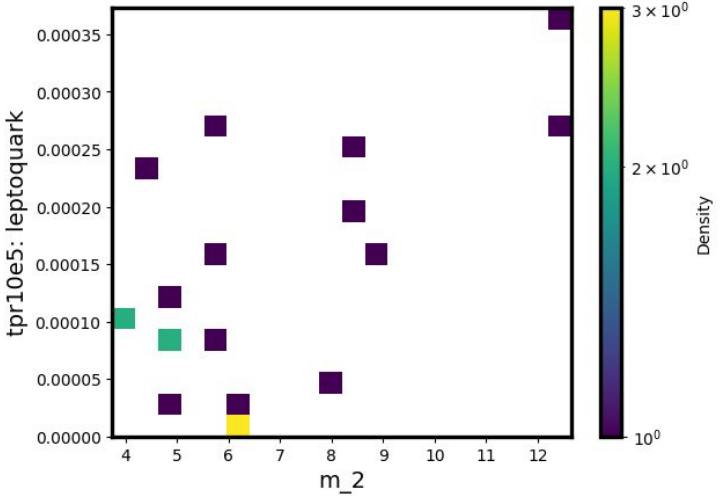
- For each version, looked at signal efficiency at fixed FPR, signal AUC, as well as 4 metrics:
  - m_1: compression_factor / mean AD score, where the mean AD score was computed over the test split of the background data
  - m_2: compression_factor / (1 + mean AD score)
  - m_3: (compression_factor / mean AD score) * variance AD score
  - m_4: (compression_factor / (1 + mean AD score)) * variance AD score

- Where compression_factor = input dimension / latent dimension

Study of compression factor as AD benchmark metric

- I then made 2d histograms comparing each variable, here are a few examples





- Where the y axes are signal efficiency at 10e-4 / 10e-5

Study of compression factor as AD benchmark metric

- Some of these look *a bit* correlated, and others look completely random / uncorrelated
- Hard to tell with only 20 points
- Also can use more extreme hyperparameter values which will intentionally tank performance to see how these metrics react

Study of compression factor as AD benchmark metric

There's also some other metrics I'd like to test out:

- Some kind of latent space entropy, e.g.
  - calculate variances along each latent direction, then normalize. Call these norm_vars
  - Metric = -sum(norm_vars * log(norm_vars))
- Stability of latent representations
  - compute latent representations for multiple subsets of background data, then calculate average cosine similarity or euclidean distance between each subset. Higher stability (higher cosine similarity of lower distance) indicates the network is reliable at encoding typical data
- Latent space density
  - use kernel density estimation (KDE) or nearest-neighbor distances in latent space to estimate the density of data. More uniform density might indicate the network is not overfitting to specific features of the data, resulting in better generalization