HEP Data

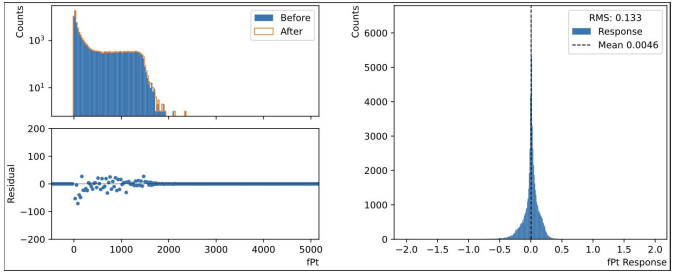
Improvements in Baler

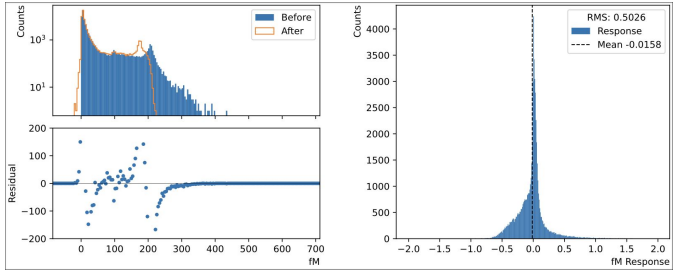
Changes made

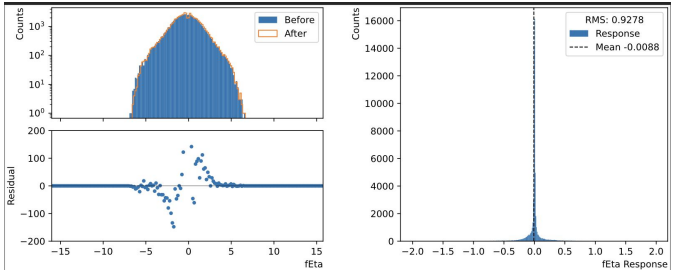
- Model
  - Added skip connections in encoder and decoder for easier flow of gradients, smoother optimization landscape
  - Multiple decoder heads as an ensembling technique, resulting in better overall performance and robustness
- Trainer
  - Simple L1 loss suffices, no explicit regularization
  - For compression/decompression, use model that achieved best performance on val set, not latest model
  - Use StepLR scheduler that decays learning rate by half every 25 epochs

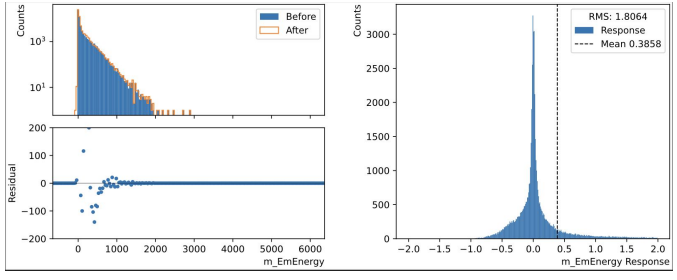
The code for all the changes has been pushed to the fork

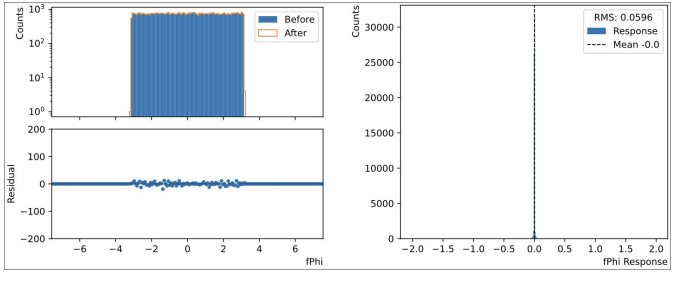
Results

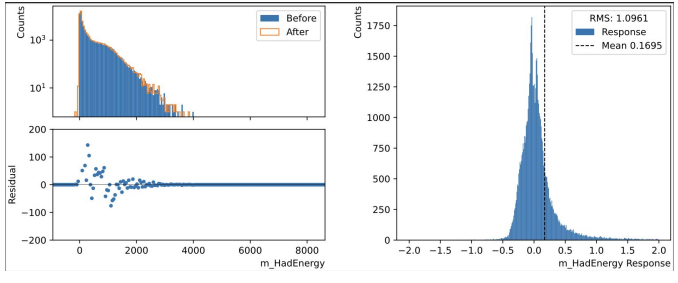

Counts
Before
After
Residual
fPt
RMS: 0.133
Response
Mean 0.0046
fPt Response


Counts
Before
After
Residual
fM
RMS: 0.5026
Response
Mean -0.0158
fM Response

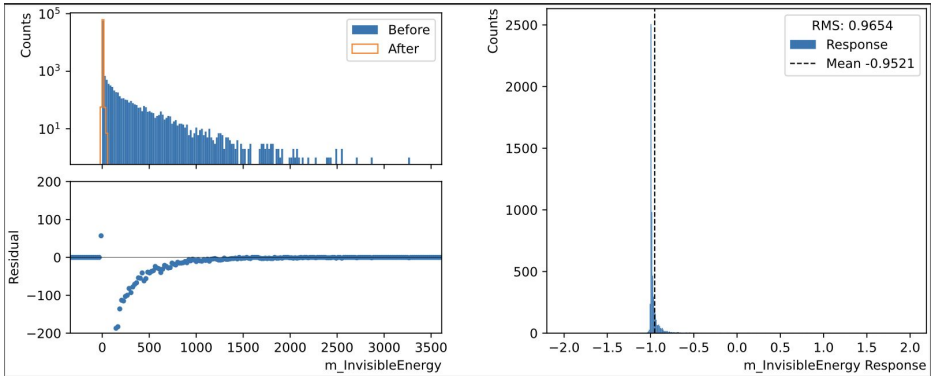

Counts
Before
After
Residual
fEta
RMS: 0.9278
Response
Mean -0.0088
fEta Response

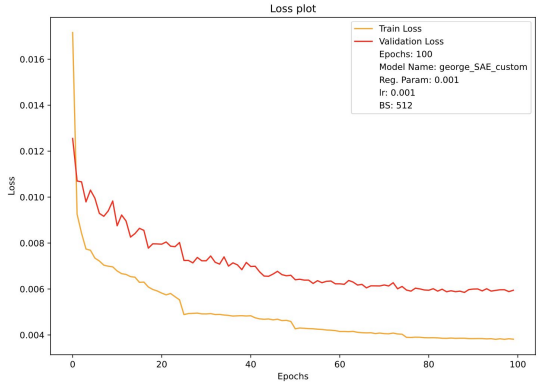

Counts
Before
After
Residual
m_EmEnergy
RMS: 1.8064
Response
Mean 0.3858
m_EmEnergy Response

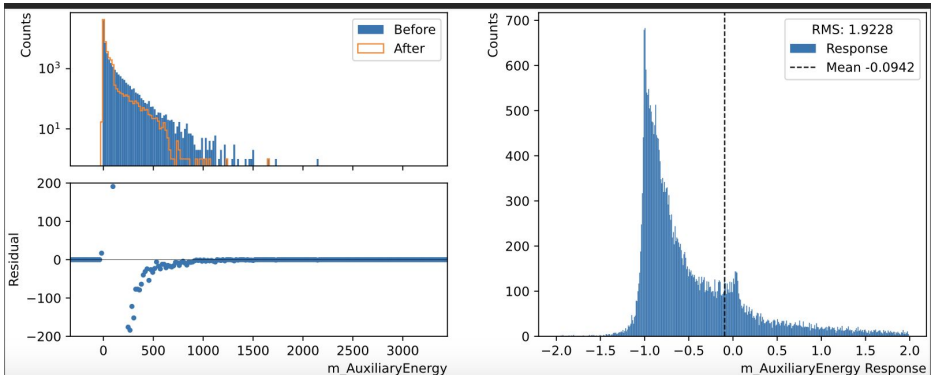

Counts
Before
After
Residual
fPhi
RMS: 0.0596
Response
Mean -0.0
fPhi Response

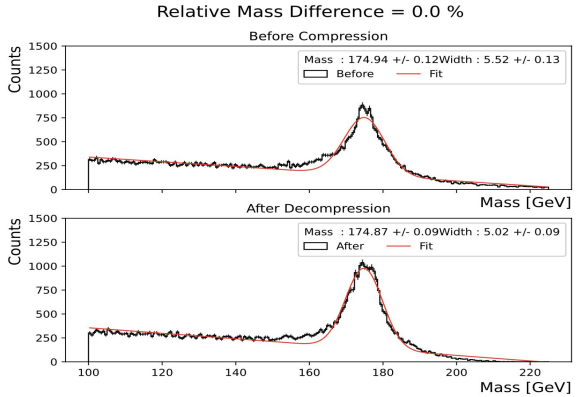

Counts
Before
After
Residual
m_HadEnergy
RMS: 1.0961
Response
Mean 0.1695
m_HadEnergy Response

Results (contd.)









Discussion

- With the described changes, the model was able to achieve a relative mass difference of 0.0%; the distributions before and after compression are very similar (with the default settings, it was 2.3%)
- Skip connections allow the network to learn the identity function and prevents feature degradation. The loss landscape becomes smoother; it is easier for an optimization algorithm to find a better minimum and converge.
- Ensembling has long been known to improve performance by aggregating the predictions across multiple models - 5 decoder heads are used and the outputs are then averaged, exploiting statistical diversity.
- A hyperparameter search suggested that using the L1 loss along with a StepLR that decays the optimizer's learning rate by half every 25 epochs performed best. Details are omitted for brevity.
- Checkpoints where the model achieved the best performance on the validation set are saved and used for compression and decompression.

Discussion (contd.)

- Further improvements would involve a systematic study of how model performance changes with architecture (number of layers, dropout, self attention etc.), loss functions and preprocessing strategies, to name a few.
- Arguably, a good overall evaluation across all the variables would be better than choosing a model that might be better by random chance at reconstructing a single variable. This can be observed in slides 3 and 4 - the distributions match better for some variables than others.
- Some shortcomings I found in Baler are as follows: the experiments are not fully reproducible, giving different results every time; there is no checkpointing strategy, which would be important with larger datasets; many design choices are hardcoded (architecture, loss, scheduler) - it would be better to use a JSON based config file; support for non-tabular data would be a useful addition.