BT error modes

Recap

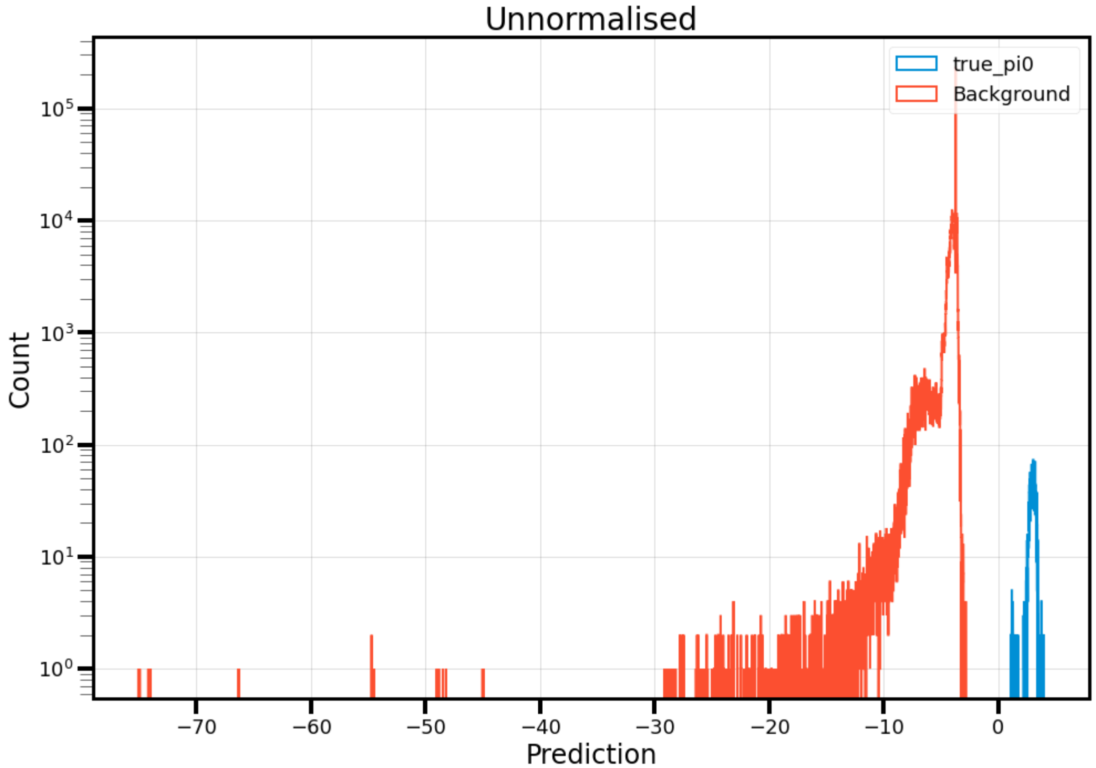
- Last week:
  - Running on MC true particle data.
  - Graph was struggling to identify pi0s based on edge pairings.
- Todo:
  - Fix edge pairings (plan was to separate shower and track like objects)
  - Run on reconstructed PFO, using back-tracked data (“BT” type graph)

Improving pi0 identification

- Laid groundwork for creating different nodes types for shower and track like PFOs.
- During testing discovered existing issues:
  - Standard ML practice – normalise inputs to mean: 0, standard deviation: 1
  - Normalisation was being applied to labels (values 0/1).
  - Shared mother gave little information – beam particle was a valid shared mother.
  - Potential issues with normalisation calculation.
  - Closest approach could go negative.

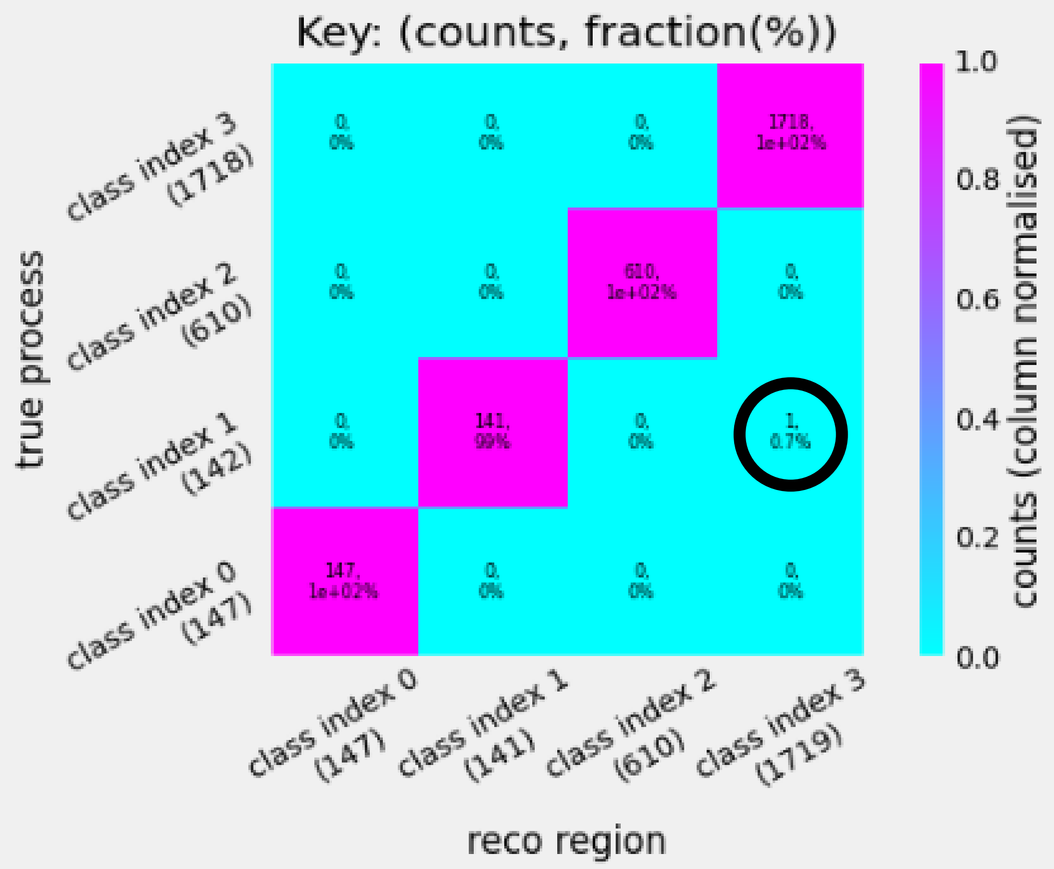
Improving pi0 identification

- After fixing the above issues, perfect pi0 identification could be recovered for the MC graphs
  - Skipped over finishing the track/shower node splitting



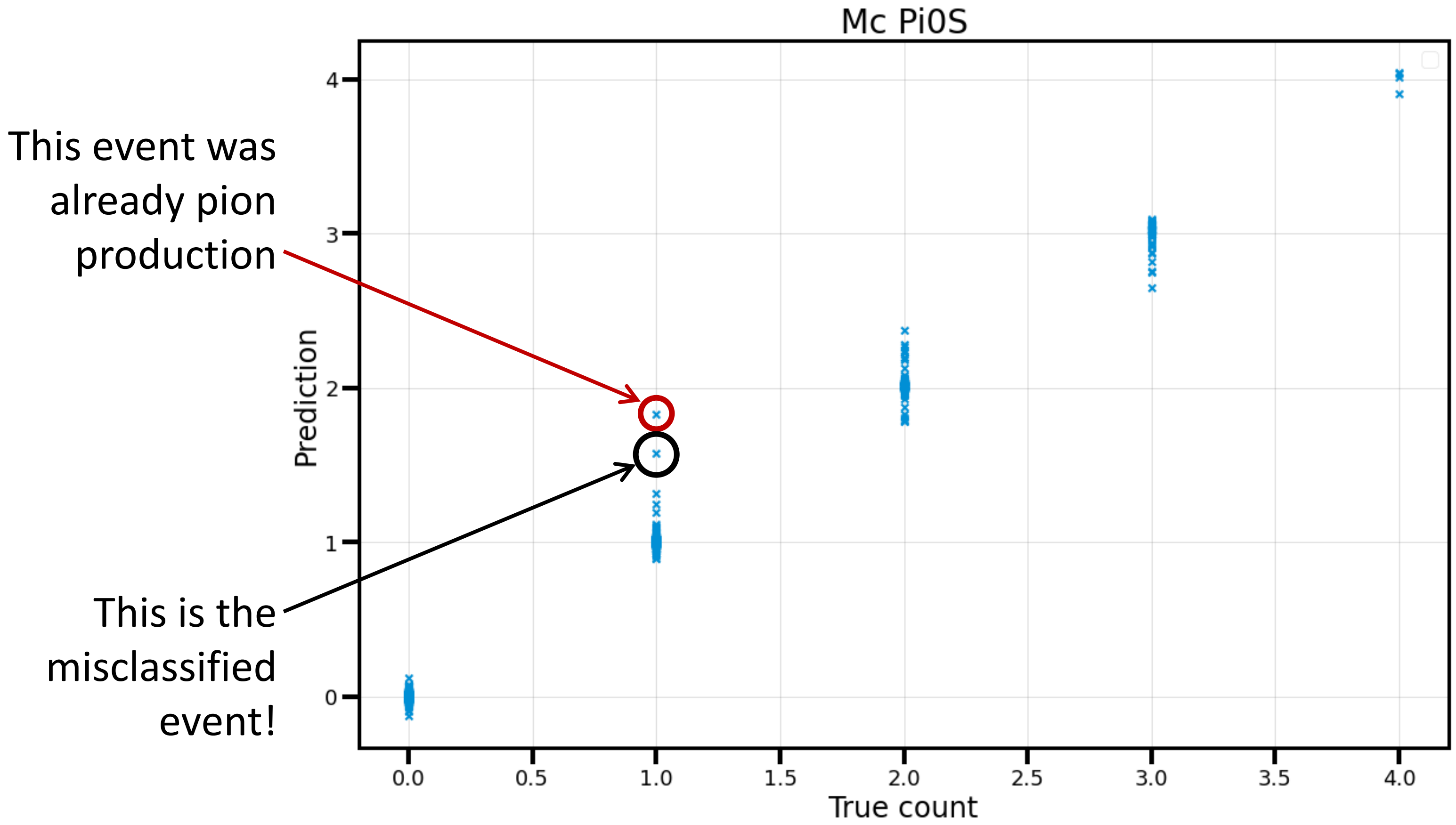
Understanding errors

- Final model train on MC had 1 misclassified event
- Can we use the extra losses to understand this?



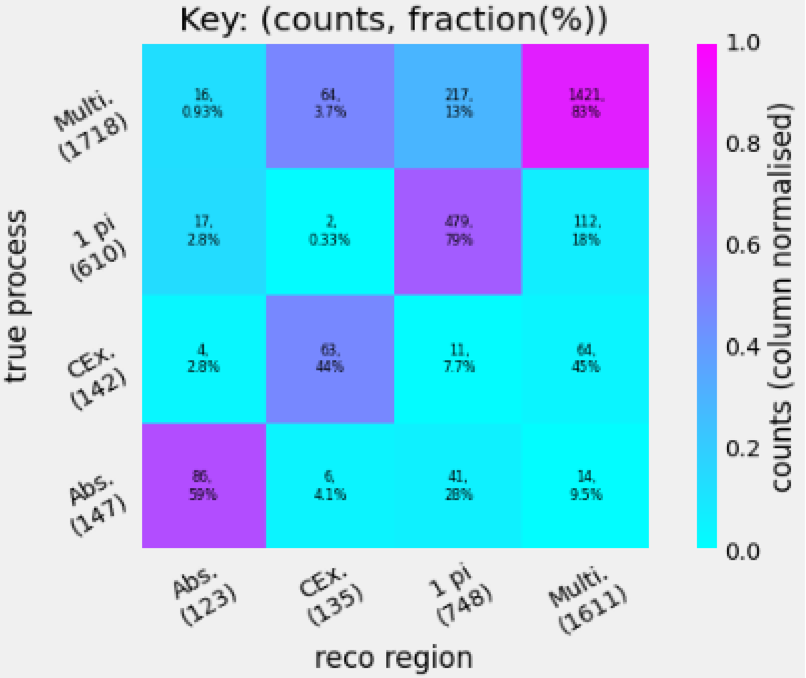
Understanding errors

- Looking at the pi0 counts predicted by the network:



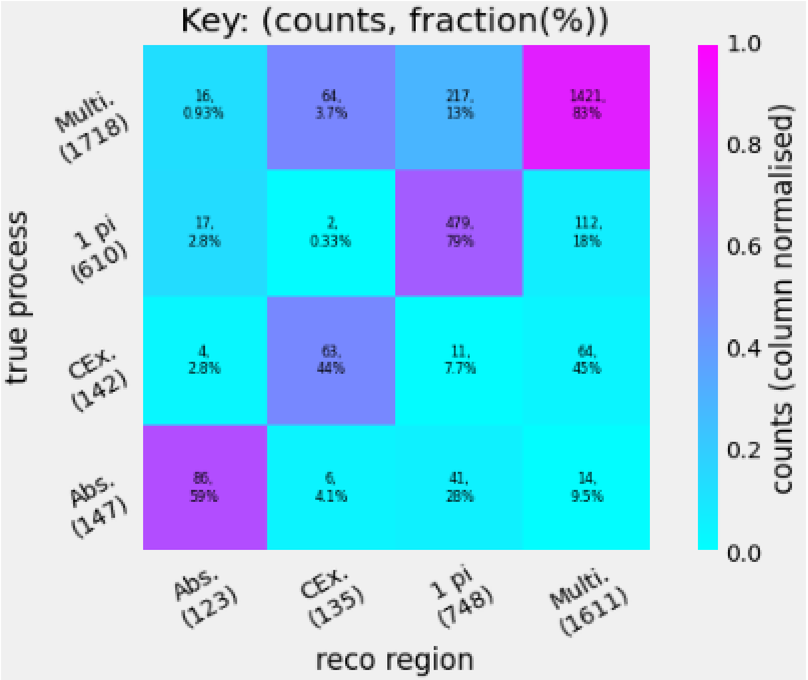
Moving to BT data

- Ran using the same model as MC, using the classification weightings of the last run BT model, and slightly higher importance on particle count predictions:
  - Abs.: 0.275
  - CEx.: 0.5
  - 1 pi: 0.2625
  - Multi. pi: 0.1875



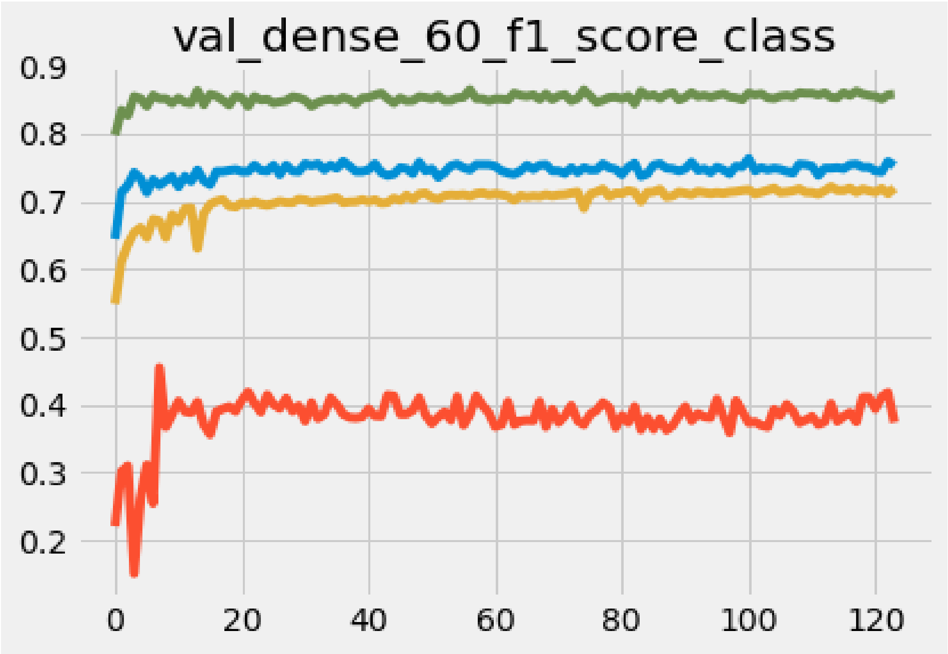
Moving to BT data

- For Abs.: 58.50% efficiency, 74.14% purity (product: 43.37%).
- For CEx.: 48.59% efficiency, 40.59% purity (product: 19.72%).
- For 1 pi: 79.18% efficiency, 65.80% purity (product: 52.10%).
- For Multi.: 82.54% eff., 88.79% purity (product: 73.29%).



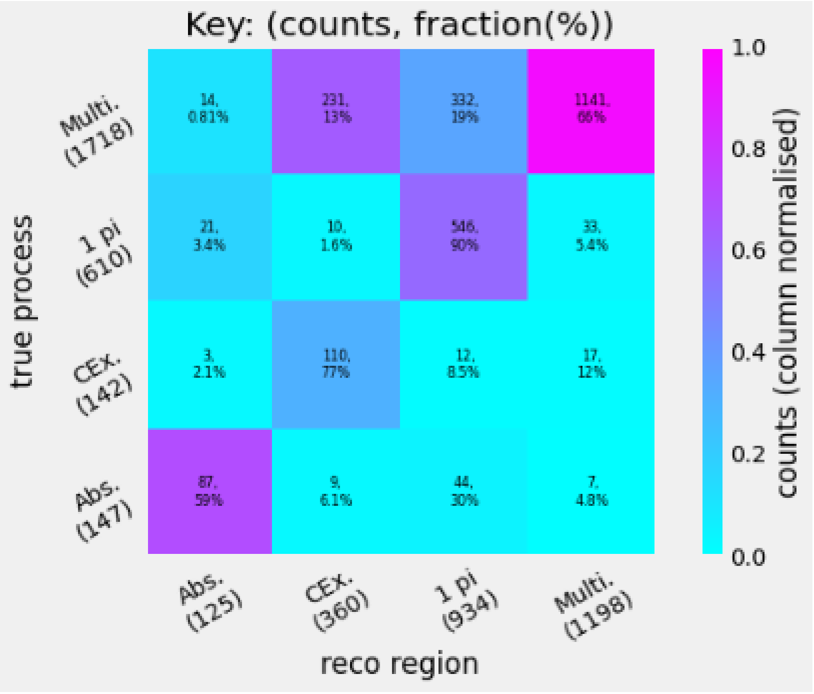
Exploring BT results

- Same model as above, but changed the relative weighting importances:
  - Abs.: 0.275 -> 0.27
  - CEx.: 0.5 -> 0.7
  - 1 pi: 0.2625 -> 0.27
  - Multi. pi: 0.1875 -> 0.13



Exploring BT results

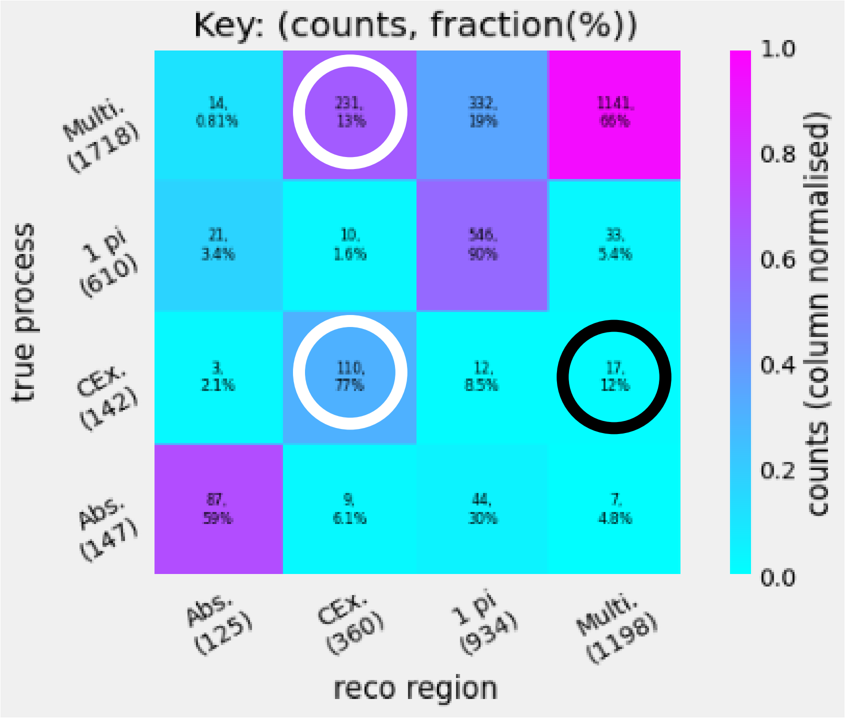
- For Abs.: 59.18% efficiency, 69.60% purity (product: 41.19%).
- For CEx.: 77.46% efficiency, 30.56% purity (product: 23.67%).
- For 1 pi: 89.51% efficiency, 58.46% purity (product: 52.32%).
- For Multi.: 66.41% efficiency, 95.24% purity (product: 63.25%).



Exploring BT results

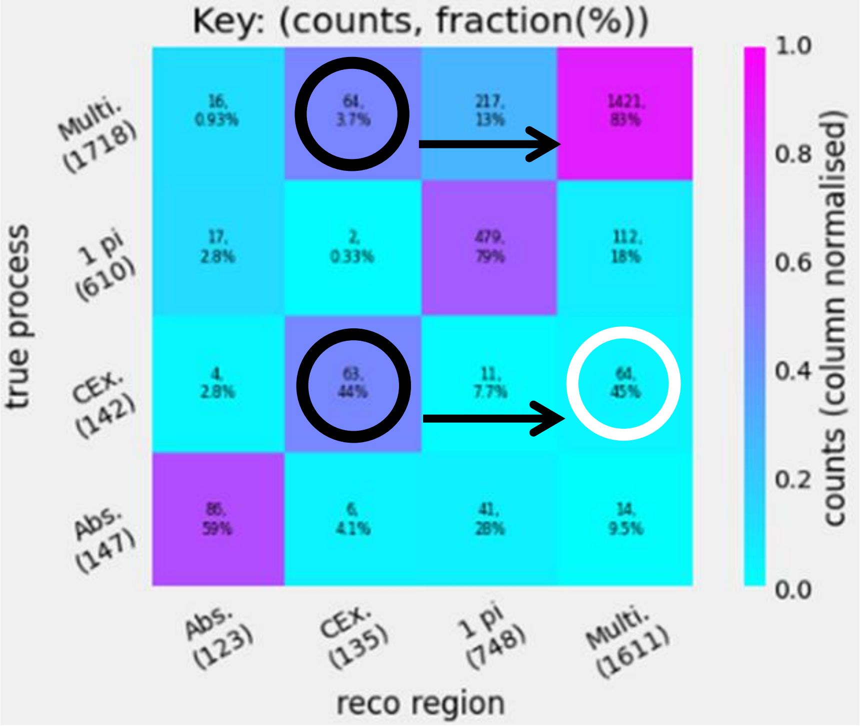
Left

CEx. : Multi. = 0.7:0.13



Right

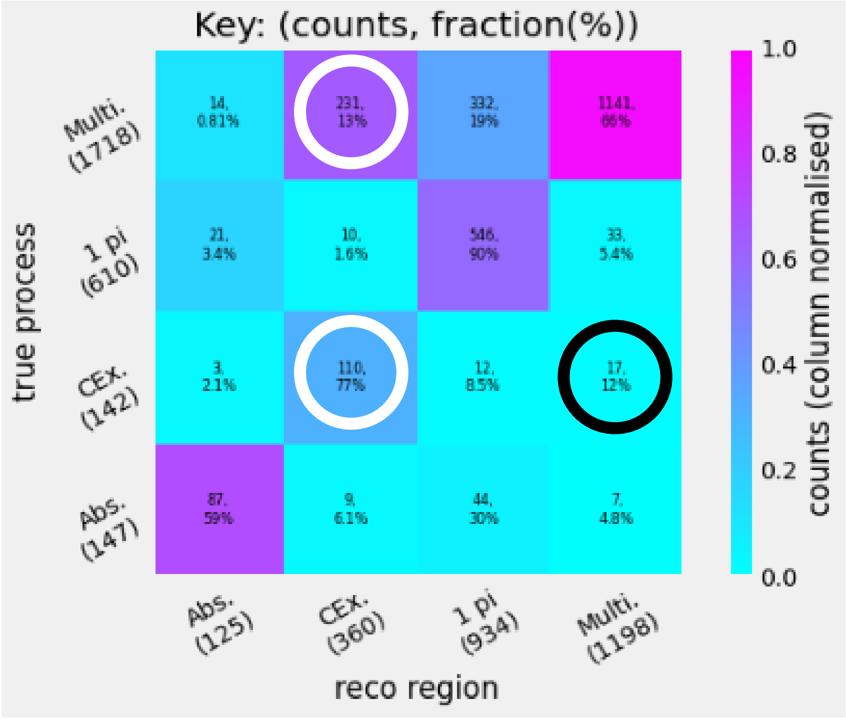
CEx. : Multi. = 0.5:0.1875



Exploring BT results

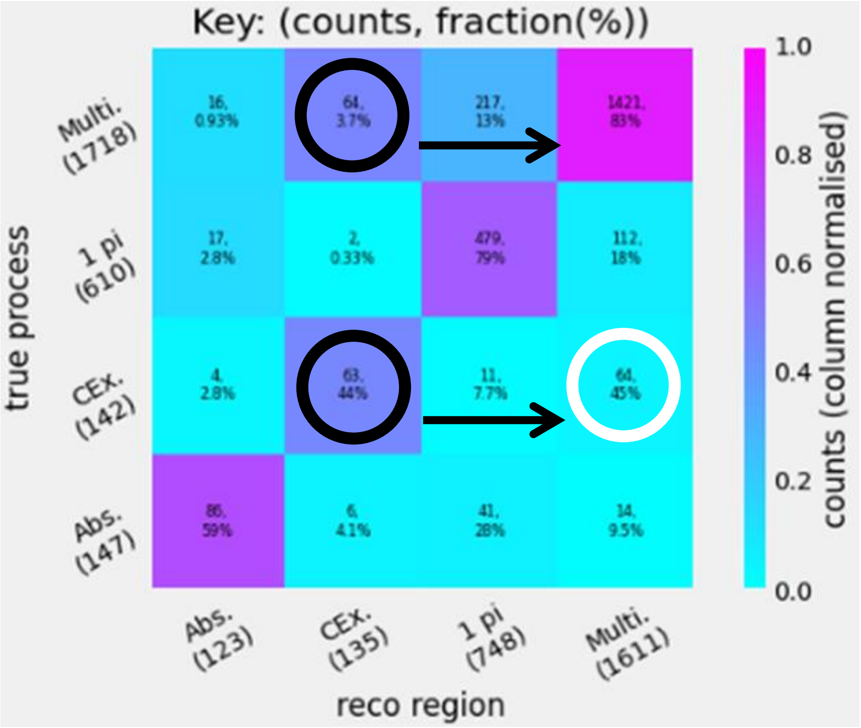
Left

CEx. : Multi. = 0.7:0.13



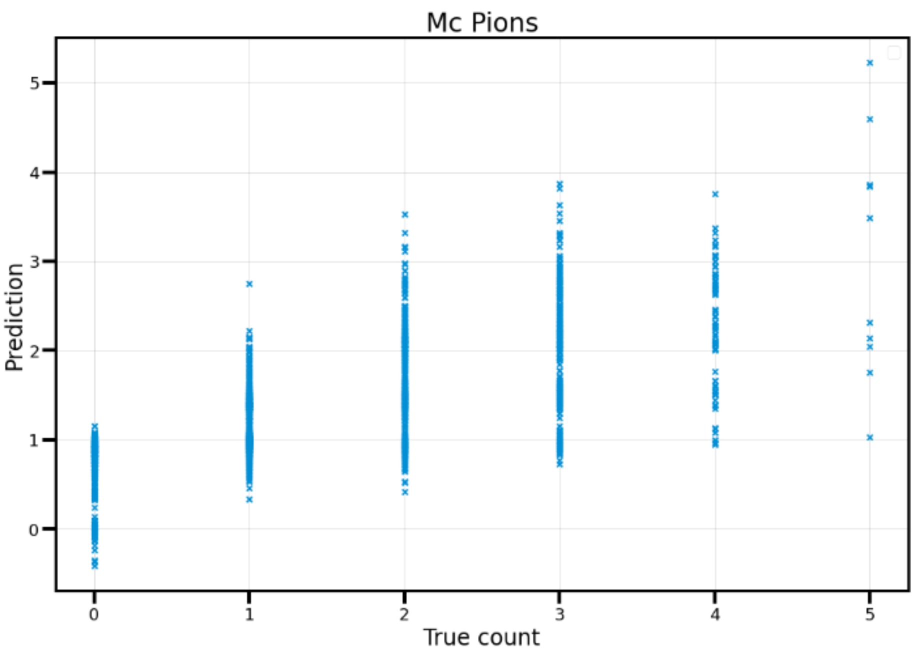
Right

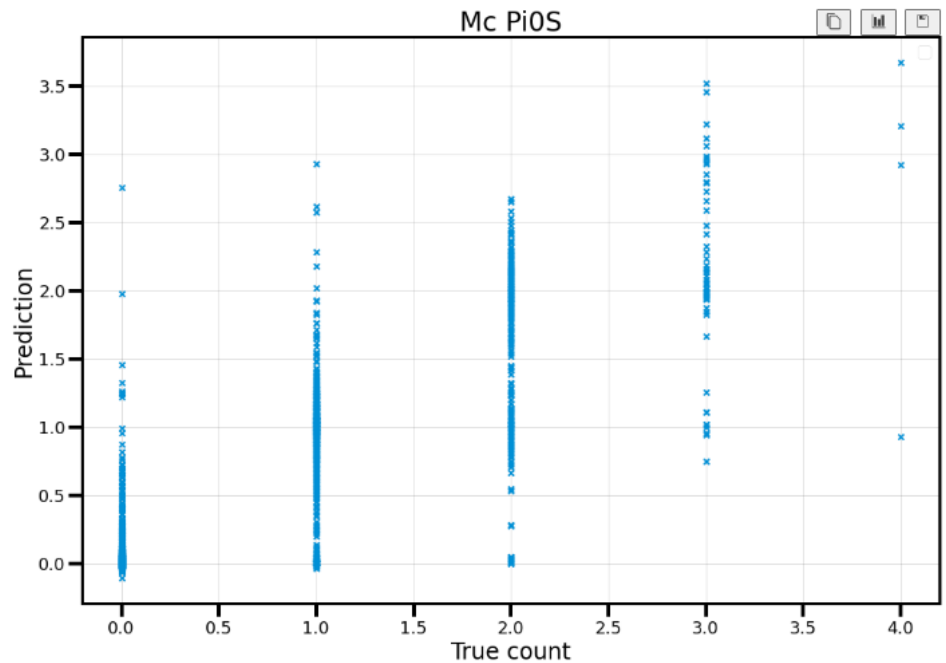
CEx. : Multi. = 0.5:0.1875



Understanding BT results

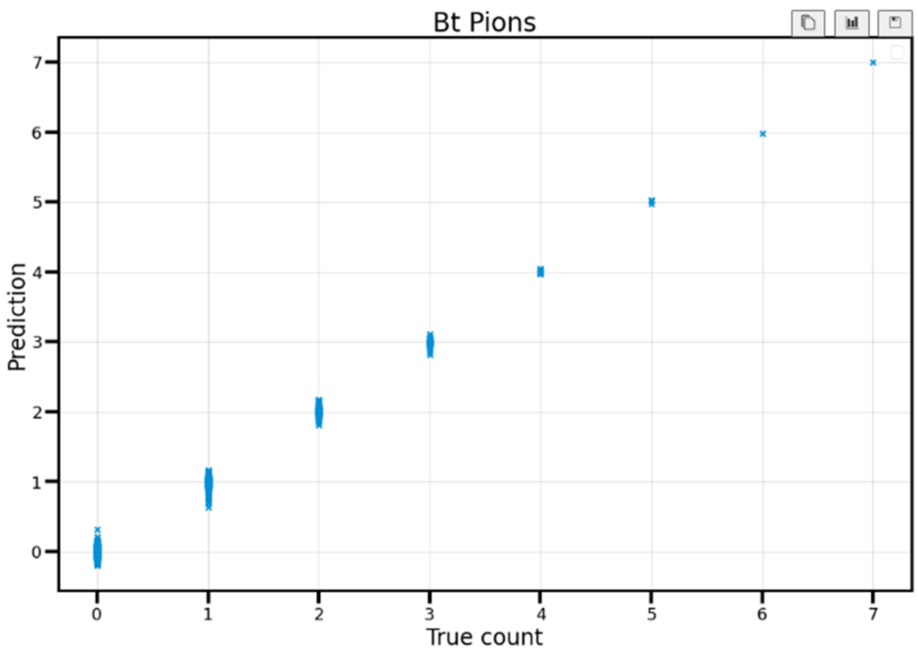
- Network isn't doing a very good job of predicting the counts…
- But this is the number of particles in MC, what about particles which have energy deposition?

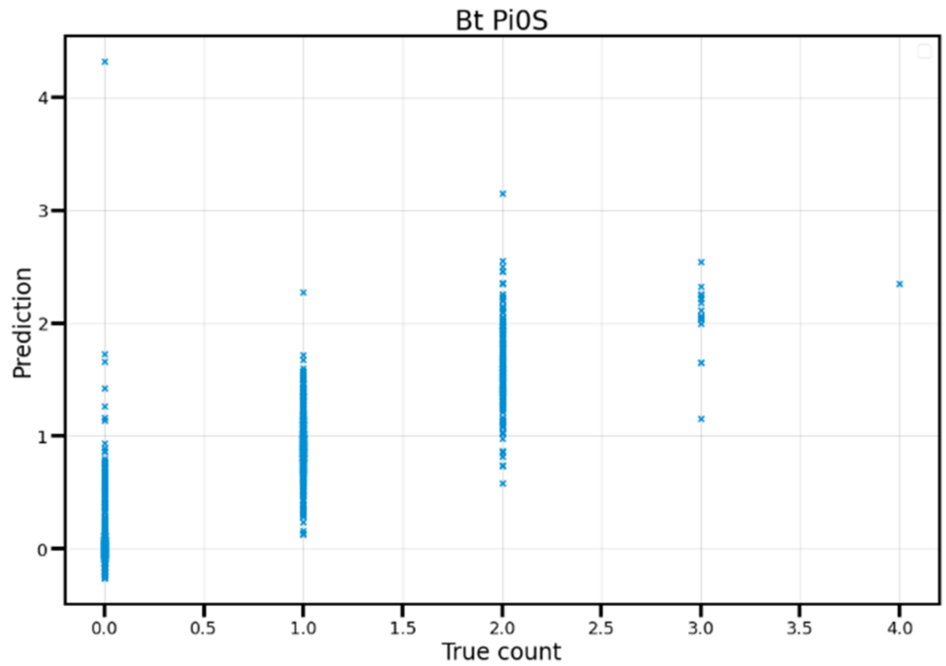




Understanding BT results

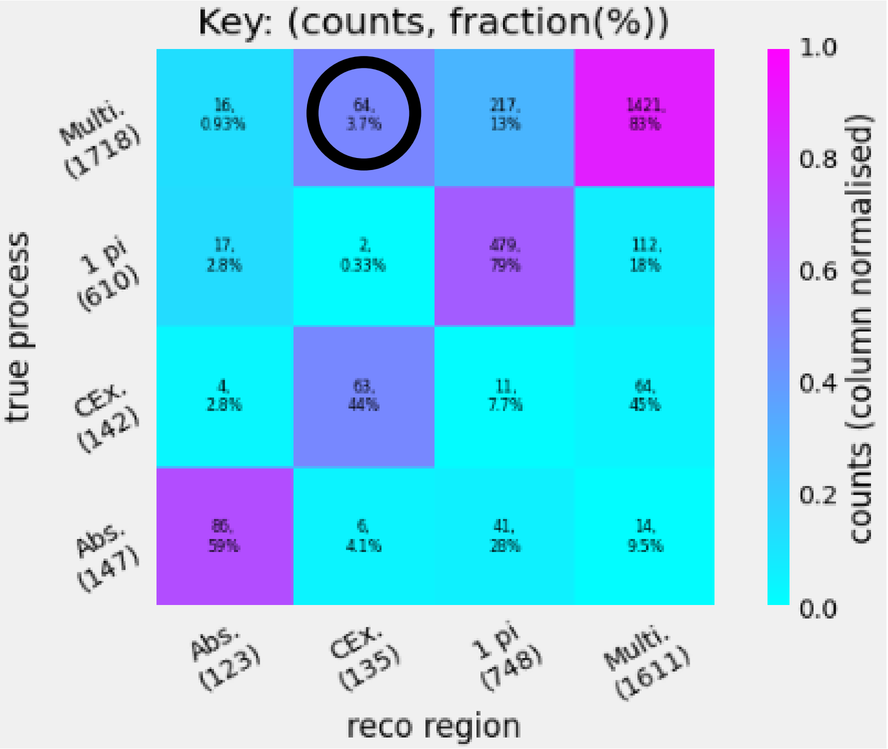
- When we compare with the number of particles reco particles which back-track to a pion or photon from pi0, it looks much better.
    - (Different model – retrained)



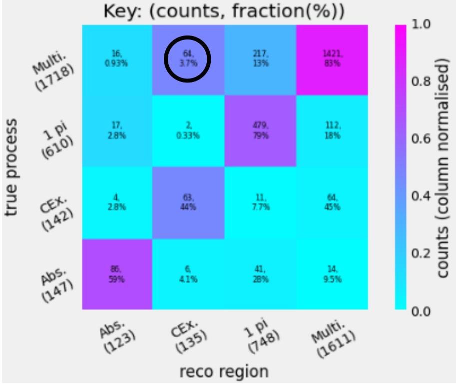


Understanding BT results

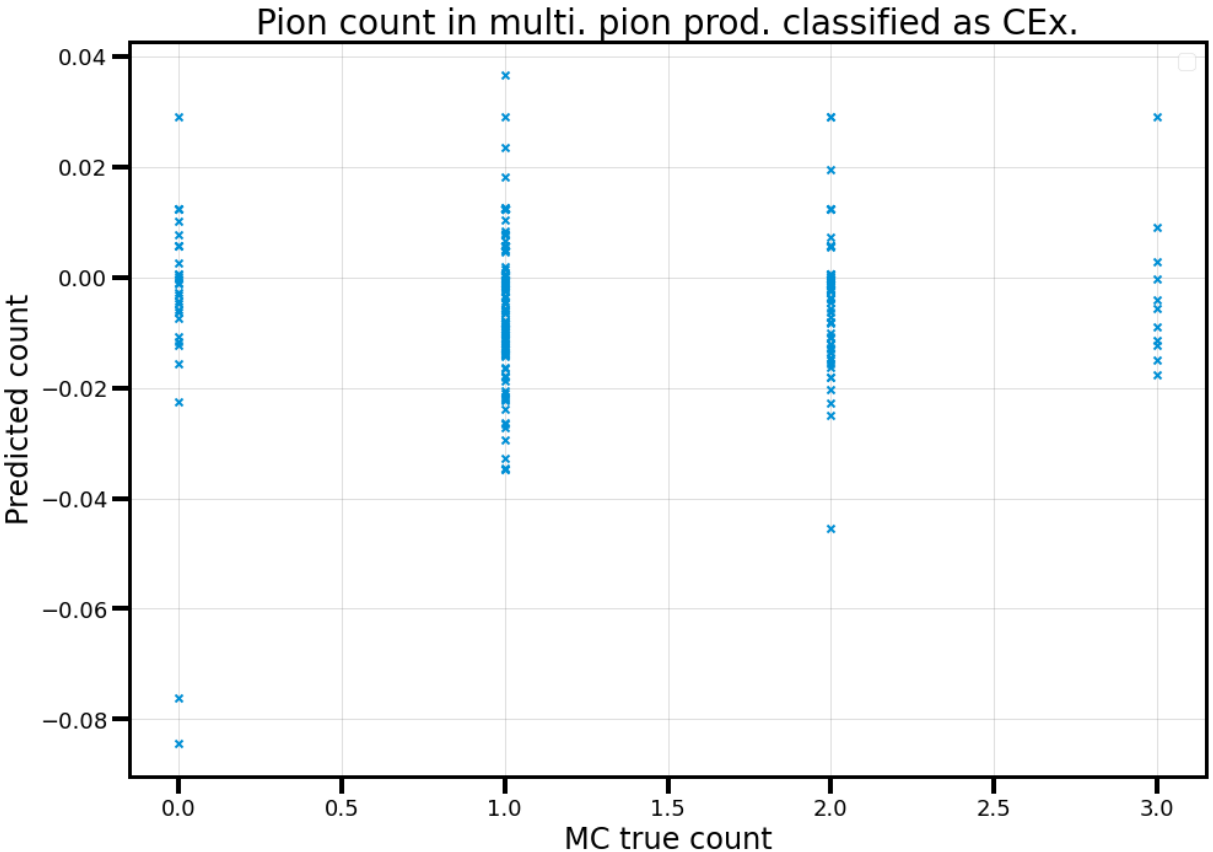
- Does this explain why the network does so poorly?
- Let's consider the multiple pion events misclassified as charge exchange:

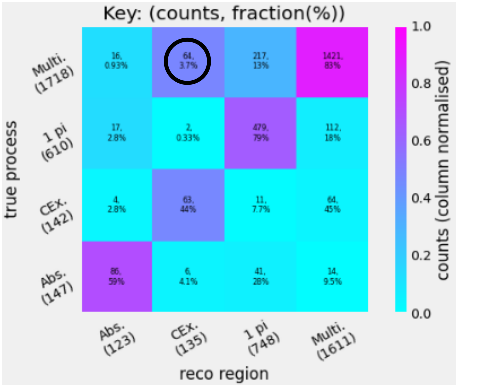


Understanding BT results



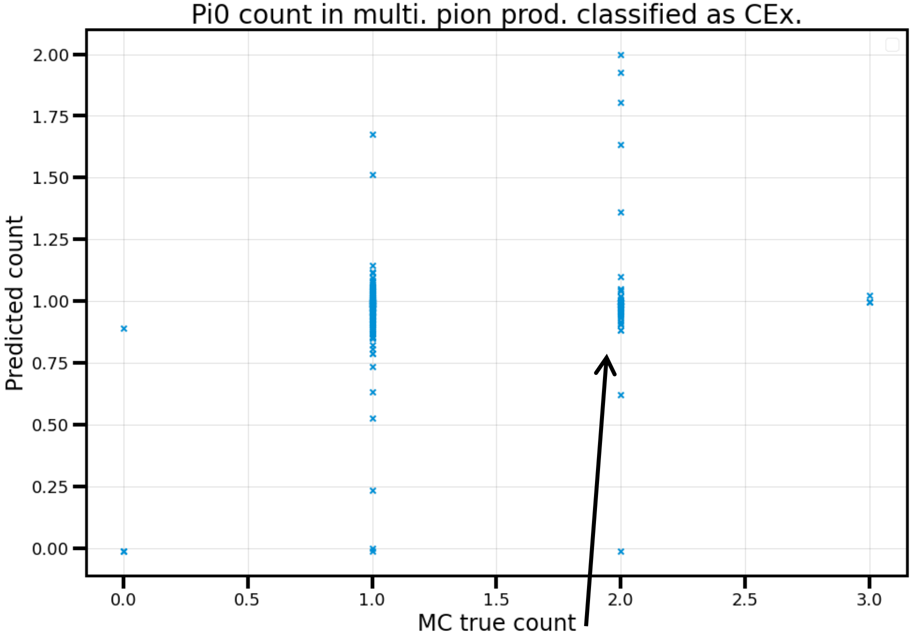
- Pions in that region (recall predicted pion count is very accurate)
- There aren't any reconstructed pions in this region!





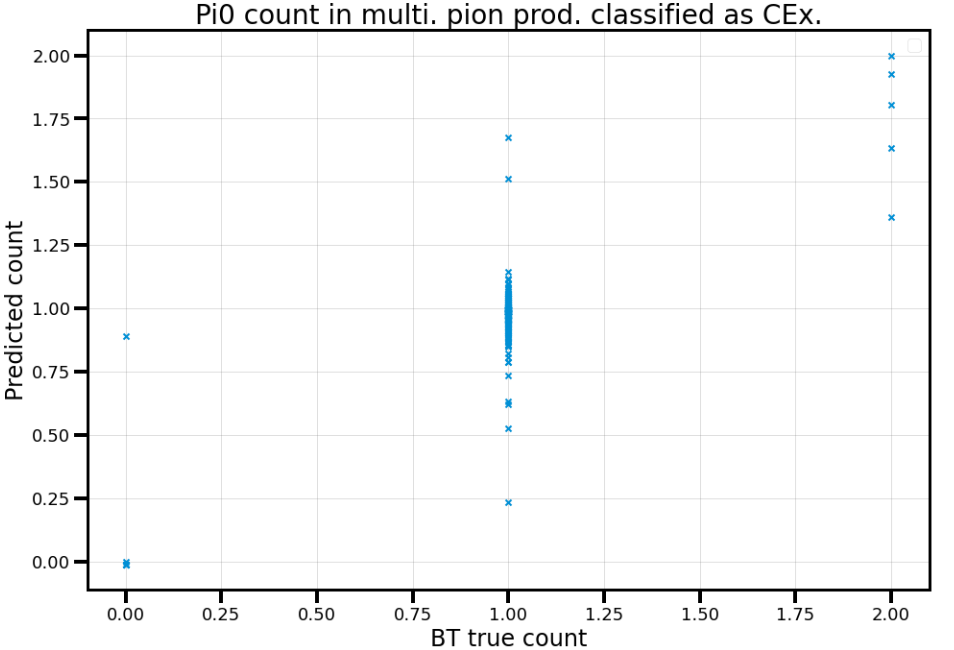
Understanding BT results

- Pi0s in the region



Many cases in MC where only one of the pions are producing backtrack-able photons

BT classification is good in this region



Understanding BT results

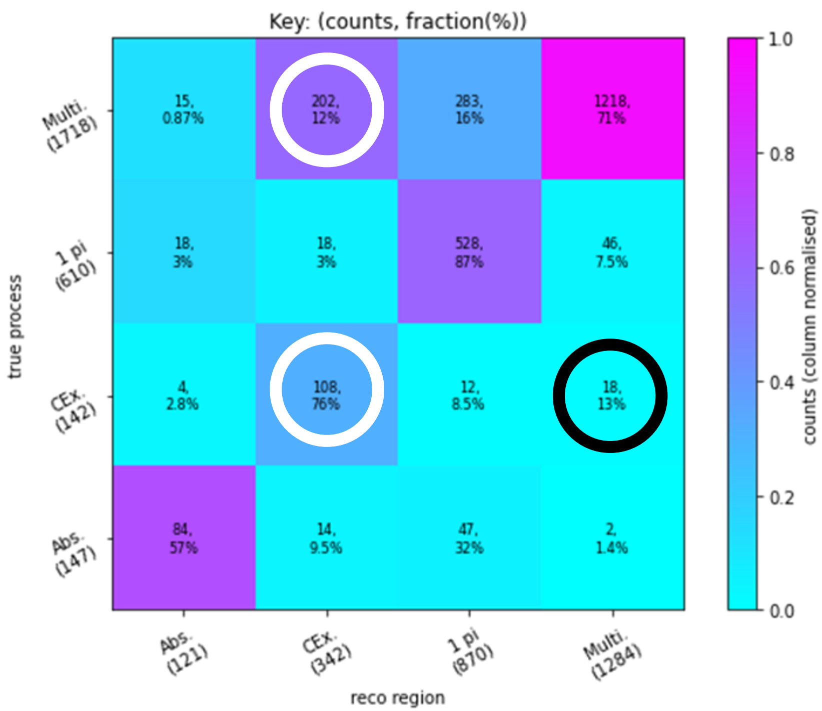
- Big disparity between reconstructed an MC particle counts.
- Create a new classification based on the reconstructed counts
  - Easier way to tell the disparity
- Retry the comparison of different class weightings, with the ability to compare the reconstructed classification matrix

Understanding BT results

True classification

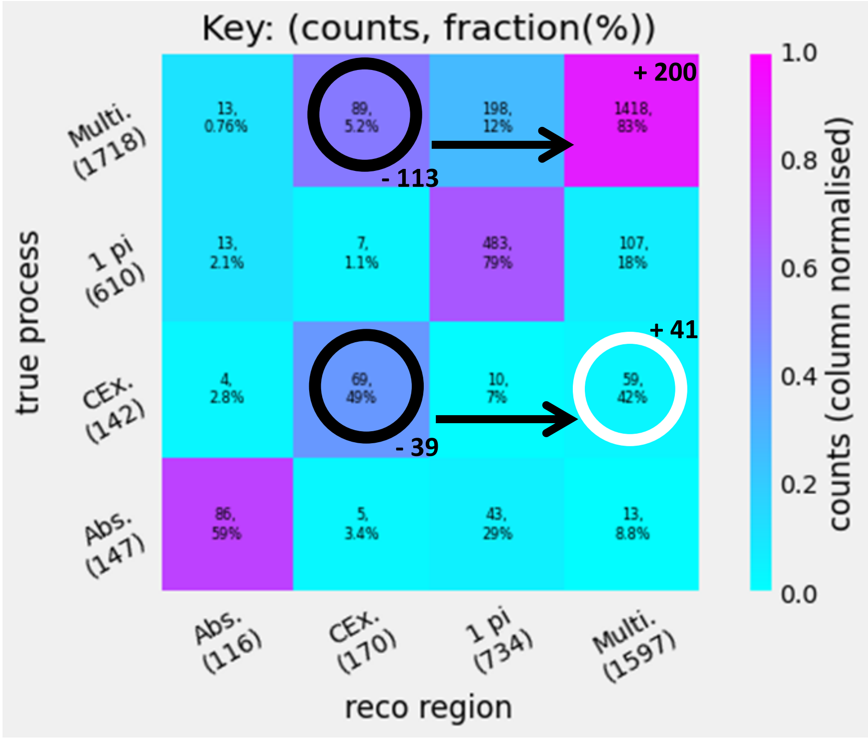
Left

CEx. : Multi. = 0.7:0.13



Right

CEx. : Multi. = 0.5:0.1875

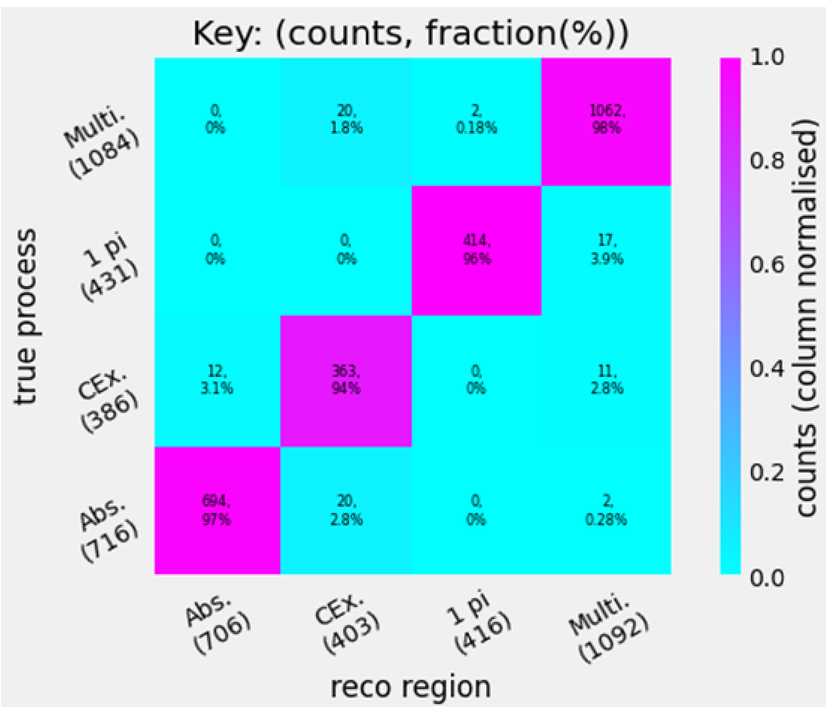


Understanding BT results

Reconstructed classification

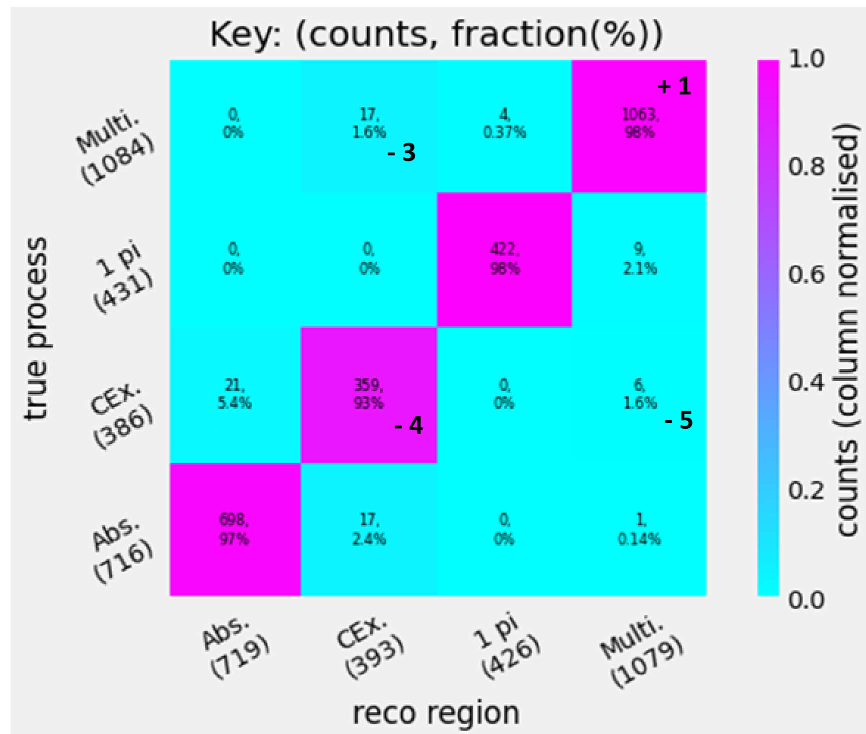
Left

CEx. : Multi. = 0.7:0.13



Right

CEx. : Multi. = 0.5:0.1875



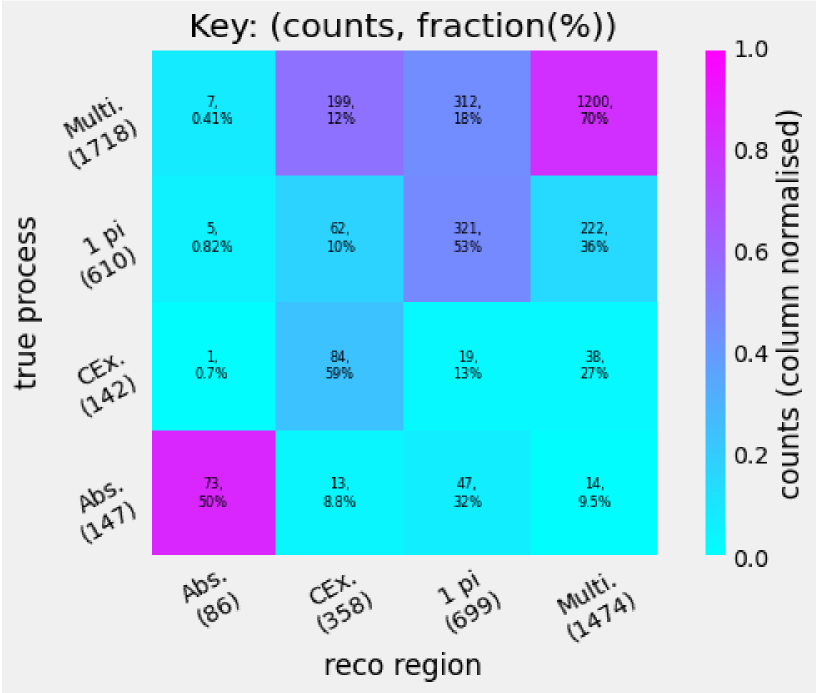
All values within 9 of each other

Next steps

- Remove BT data from edges (done)
  - Lose a lot of pion counting capacity – need to improve ability to deduce relationships
  - Pi0 identification includes many more extra pi0s predicted with 0 pi0s present – use momentum to reduce this?
- Remove BT data from particle identification
  - Extra loss to indication particle ID success
  - Try with true edges first, to see the effect of bad at this stage.
  - Pure reconstructed data
- Other ideas:
  - Add an additional loss where the network tries to predict events where reco. and MC don’t align.
  - What data indicates this? Use the momenta for this task only? MC independence (randomly add/remove PFOs)?

Removing edge data

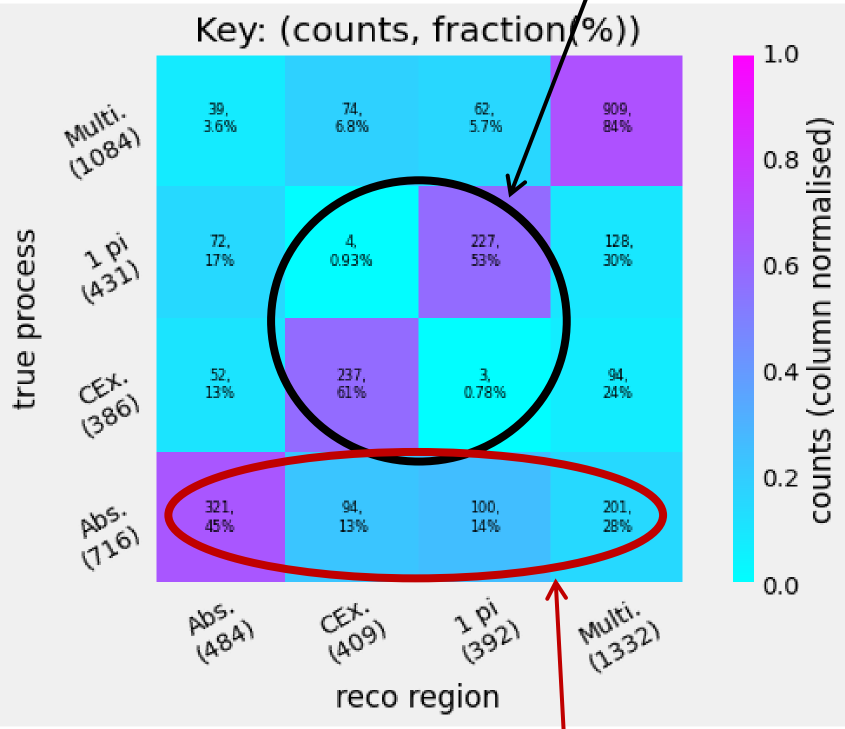
- Uses only reconstructed geometric properties on edges: impact parameter, separation, closest approach (non-beam edges only).
  - Uses CEx. : multi. = 0.7:0.13 weighting
  - For Abs.: **49.66%** efficiency, **84.88%** purity (prod.: 42.15%).
  - For CEx.: **59.15%** efficiency, **23.46%** purity (prod.: 13.88%).
  - For 1 pi: **52.62%** efficiency, **45.92%** purity (prod.: 24.17%).
  - For Multi.: **69.85%** efficiency, **81.41%** purity (prod.: 56.86%).



Removing edge data

- Reconstructed classification:
  - For Abs.: 44.83% efficiency, 66.32% purity (product: 29.73%).
  - For CEx.: 61.40% efficiency, 57.95% purity (product: 35.58%).
  - For 1 pi: 52.67% efficiency, 57.91% purity (product: 30.50%).
  - For Multi.: 83.86% efficiency, 68.24% purity (product: 57.23%).

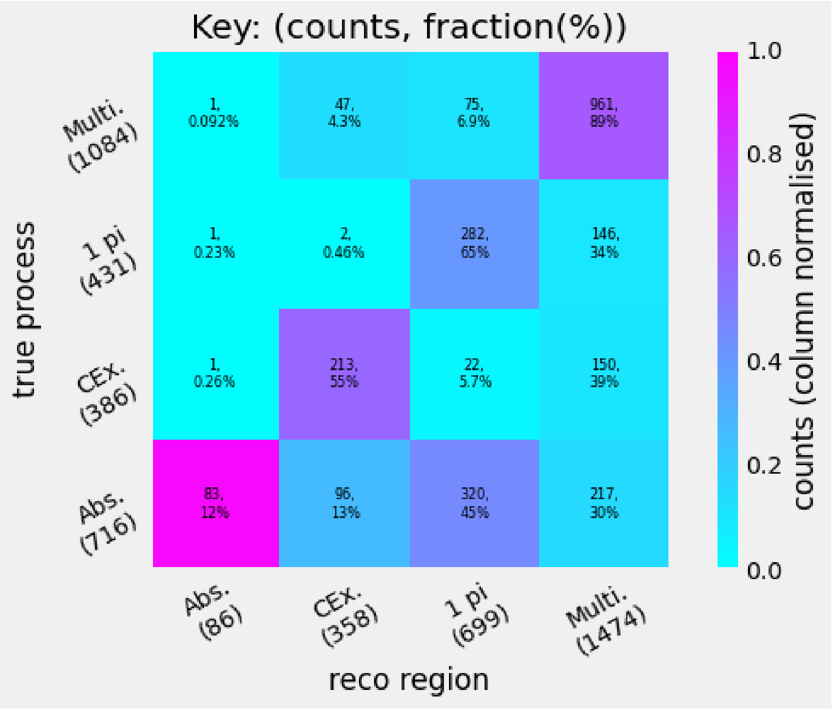
Don’t mix up CEx. and 1 pion type events



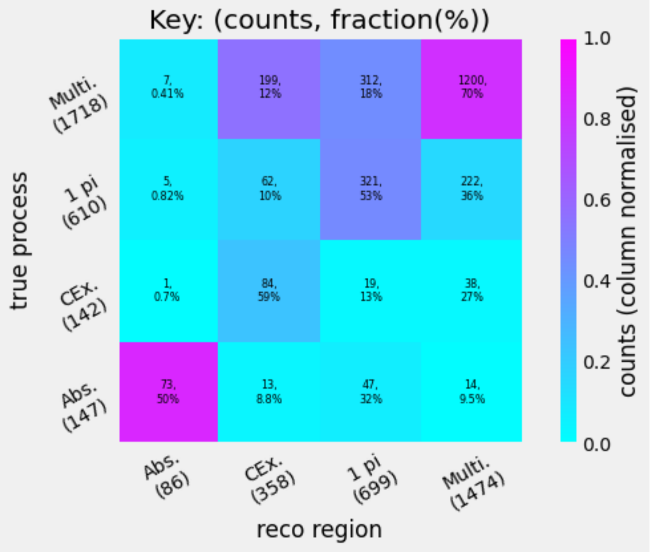
Lots of absorption put into other bins, likely something weird with this method…

Removing edge data

- Main classifier vs. reconstructed classification:

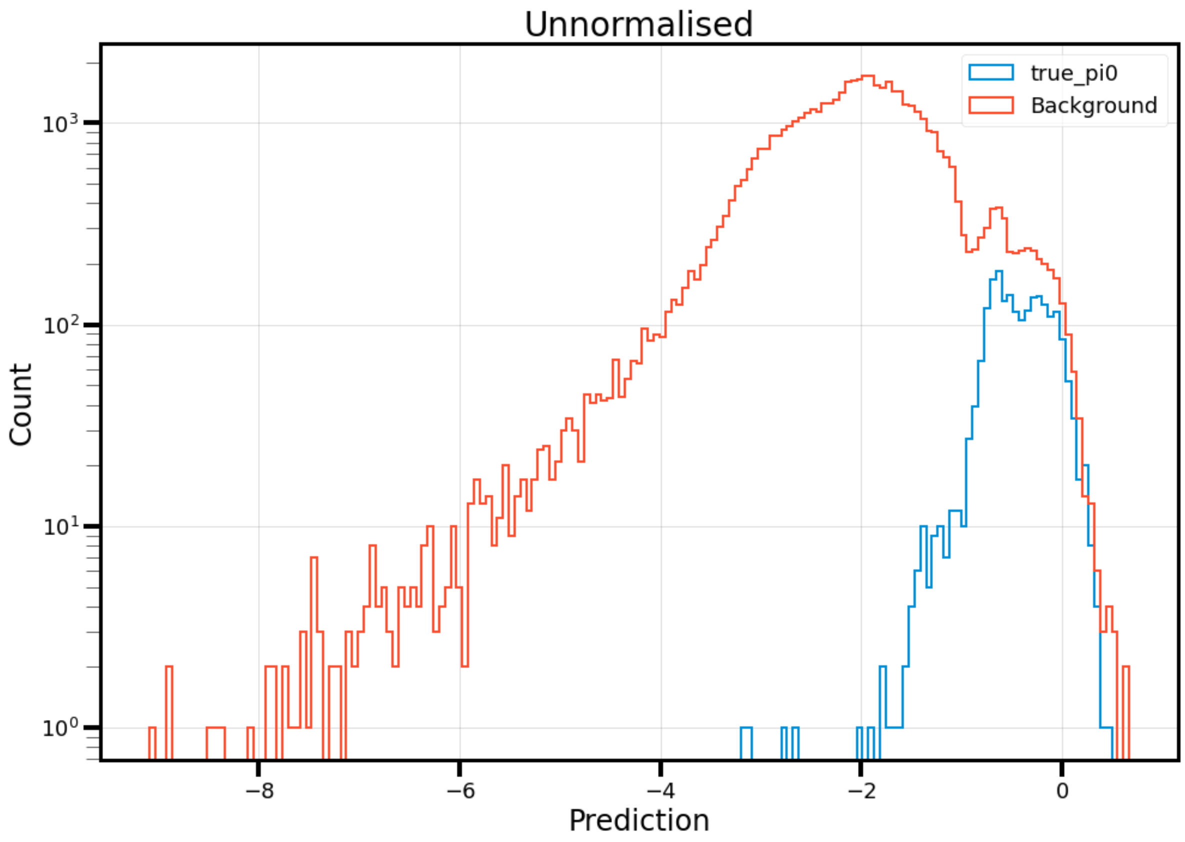


Main classifier vs. MC classification for comparis`on



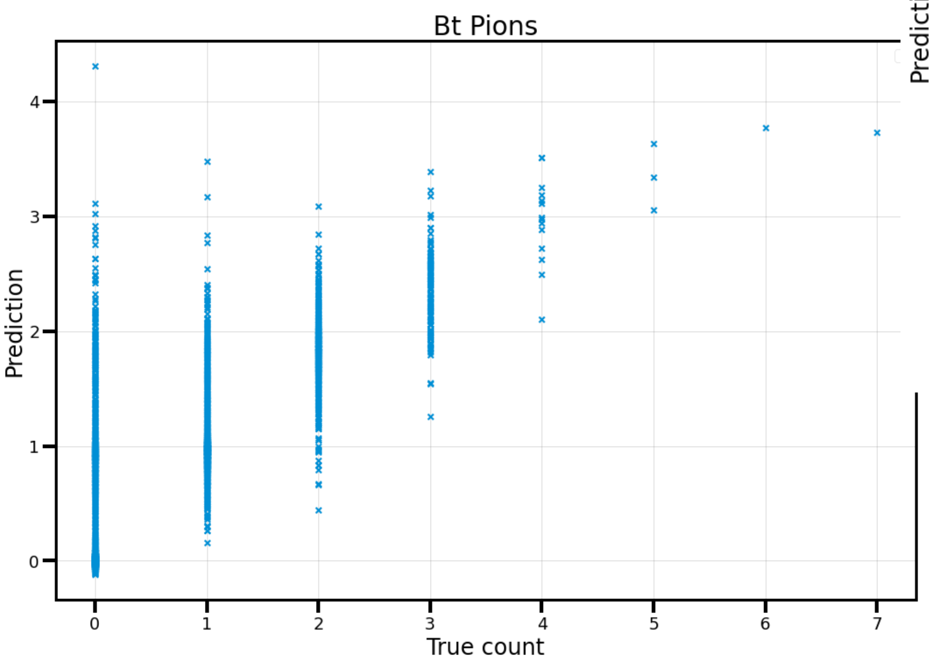
Removing edge data

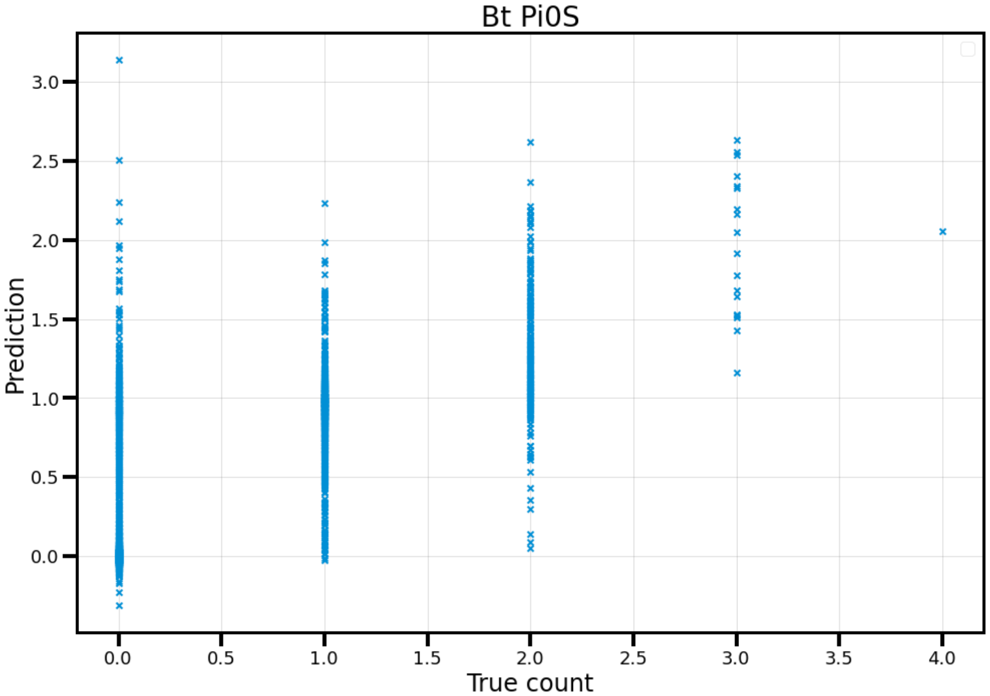
- Pi0 ID distribution



Removing edge data

- Particle counts





Removing edge data

- Particle counts