DNN normalization in DLKit study

Ryan Reece (UCSC)

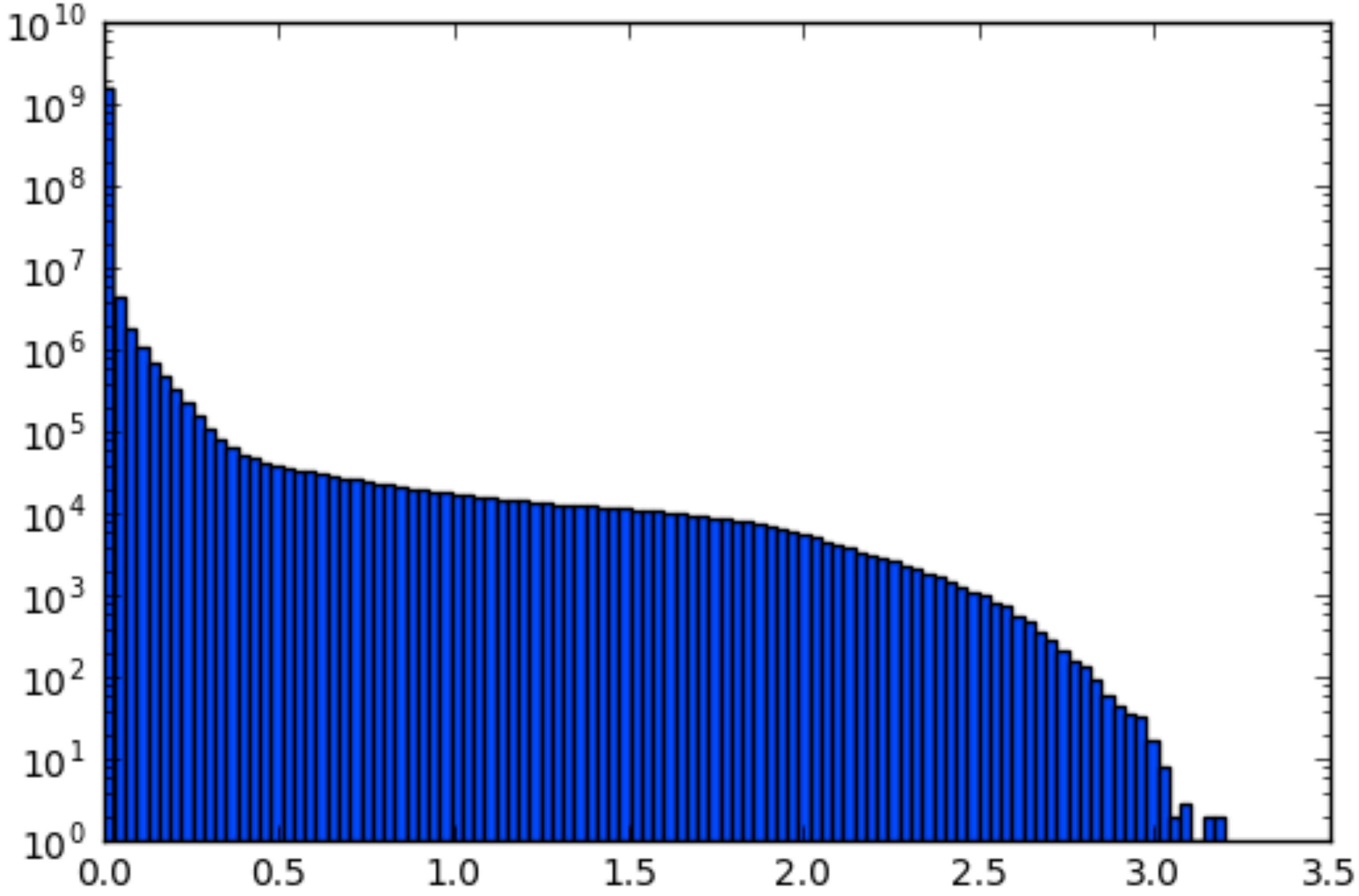
ryan.reece@cern.ch

with **Amir Farbin (UTA)**

CaloDNN Meeting

May 4, 2017

ECAL x-data



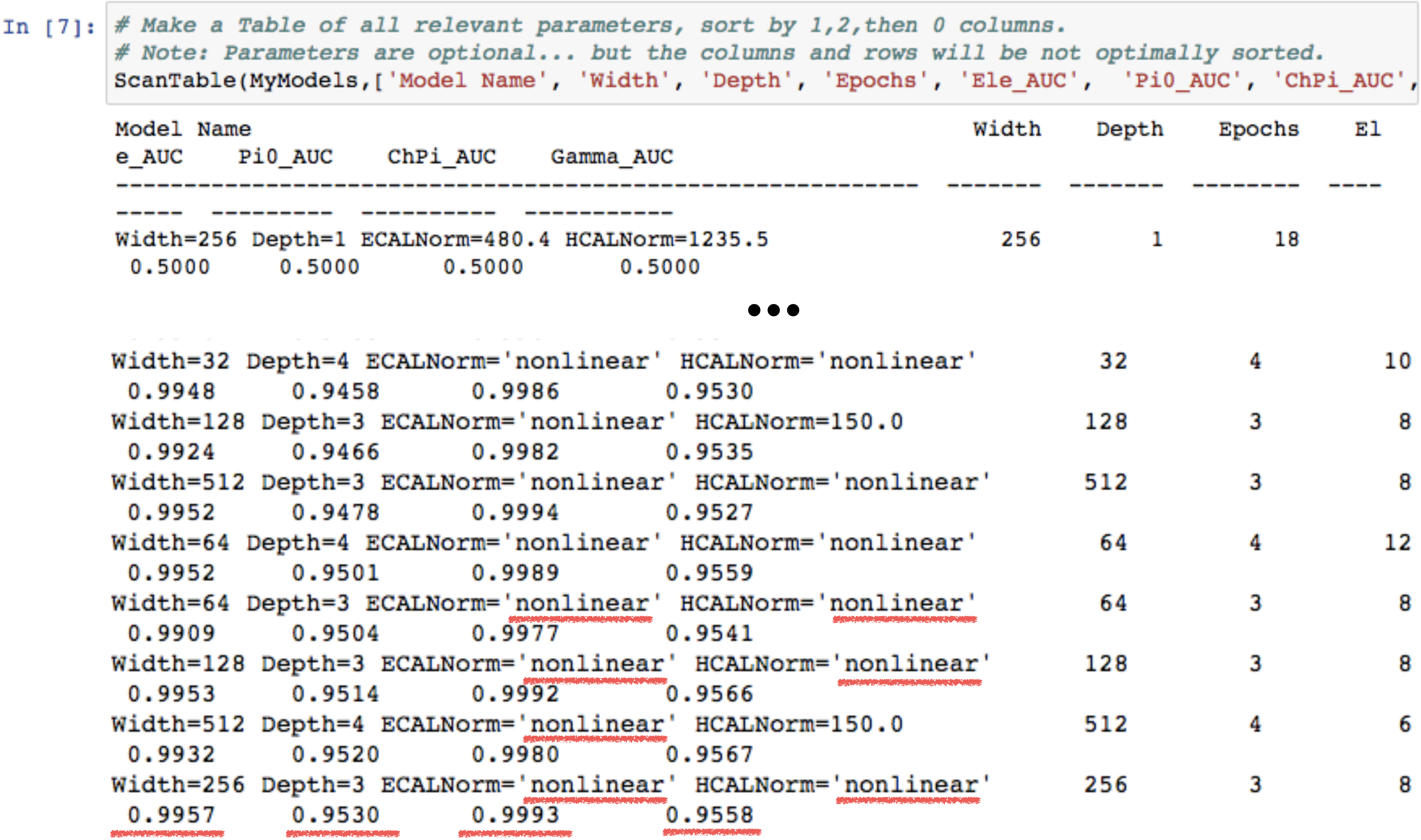
Maximum 3.2 with our old normalization factor of 1/150.

New nonlinear normalization

- I setup a hyper-parameter scan over width, depth, and the ECAL and HCAL normalization types, trying the
  - 1/150 past default
  - 1/480 (1/3.2 * past default)
  - “nonlinear” function suggested by Amir below:

$$X' = \tanh(\mathrm{sign}(X)\,\ln(\mathrm{abs}(X) + 1)/2)$$

Scan results



```
In [7]: # Make a Table of all relevant parameters, sort by 1,2,then 0 columns.
        # Note: Parameters are optional... but the columns and rows will be not optimally sorted.
        ScanTable(MyModels,['Model Name', 'Width', 'Depth', 'Epochs', 'Ele_AUC',  'Pi0_AUC', 'ChPi_AUC',
```

```
Model Name                                                    Width    Depth    Epochs    El
e_AUC     Pi0_AUC     ChPi_AUC     Gamma_AUC
-----------------------------------------------------------  -------  -------  --------  ----
-----  ---------  ----------  -----------
Width=256 Depth=1 ECALNorm=480.4 HCALNorm=1235.5                  256        1        18
 0.5000     0.5000      0.5000       0.5000
                                                    •••
Width=32 Depth=4 ECALNorm='nonlinear' HCALNorm='nonlinear'        32         4         10
 0.9948      0.9458      0.9986        0.9530
Width=128 Depth=3 ECALNorm='nonlinear' HCALNorm=150.0            128         3          8
 0.9924      0.9466      0.9982        0.9535
Width=512 Depth=3 ECALNorm='nonlinear' HCALNorm='nonlinear'      512         3          8
 0.9952      0.9478      0.9994        0.9527
Width=64 Depth=4 ECALNorm='nonlinear' HCALNorm='nonlinear'        64         4         12
 0.9952      0.9501      0.9989        0.9559
Width=64 Depth=3 ECALNorm='nonlinear' HCALNorm='nonlinear'        64         3          8
 0.9909      0.9504      0.9977        0.9541
Width=128 Depth=3 ECALNorm='nonlinear' HCALNorm='nonlinear'      128         3          8
 0.9953      0.9514      0.9992        0.9566
Width=512 Depth=4 ECALNorm='nonlinear' HCALNorm=150.0            512         4          6
 0.9932      0.9520      0.9980        0.9567
Width=256 Depth=3 ECALNorm='nonlinear' HCALNorm='nonlinear'      256         3          8
 0.9957      0.9530      0.9993        0.9558
```

≈.996 ≈.95 ≈.999 ≈.96

The best models tend to use the “nonlinear” normalization.

Thoughts

- Using a nonlinear normalization function should be equivalent to using a particular nonlinear activation function at the first layer.
- So maybe one would want to use the linear 1/max normalization and push the nonlinearities to the activation, but maybe this is a nice combination in the end.
- In CaloDNN/Models.py, the Fully3DImageClassification model we are using uses *relu* normalization for every layer except the last, where it uses *softmax*.