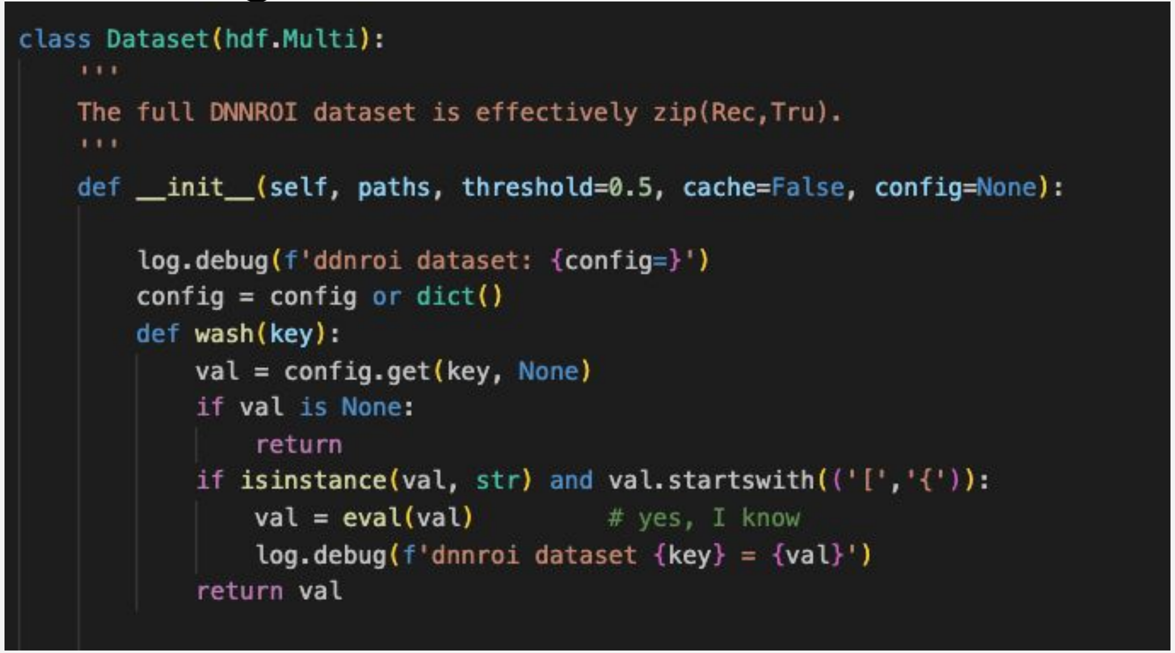
January 23

Training of the DNNROI

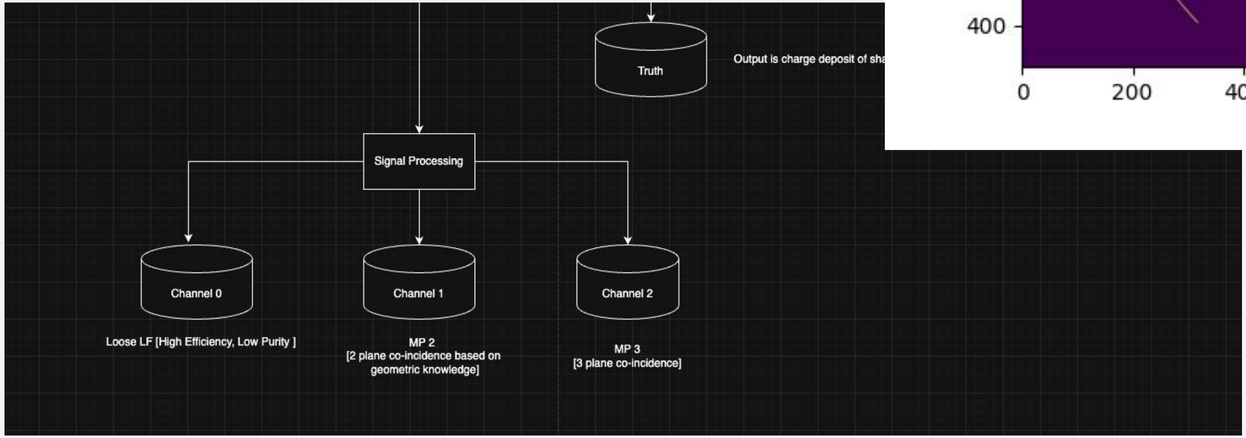


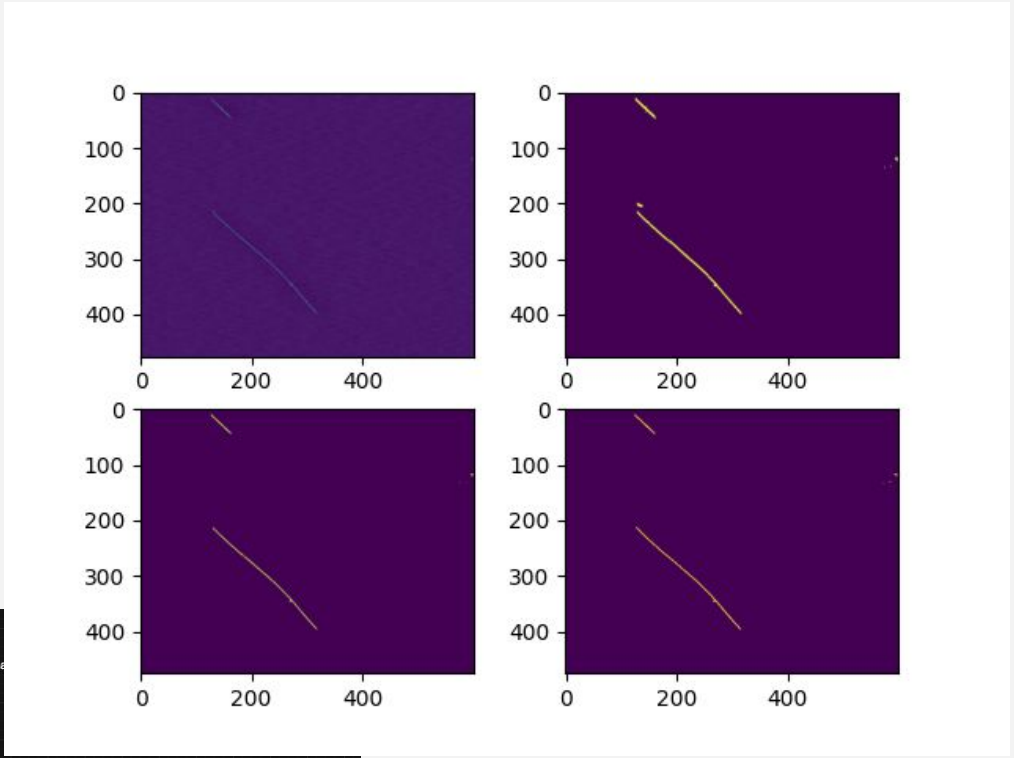
```
class Dataset(hdf.Multi):
    '''
    The full DNNROI dataset is effectively zip(Rec,Tru).
    '''
    def __init__(self, paths, threshold=0.5, cache=False, config=None):

        log.debug(f'ddnroi dataset: {config=}')
        config = config or dict()
        def wash(key):
            val = config.get(key, None)
            if val is None:
                return
            if isinstance(val, str) and val.startswith(('[','{')):
                val = eval(val)         # yes, I know
                log.debug(f'dnnroi dataset {key} = {val}')
            return val
```

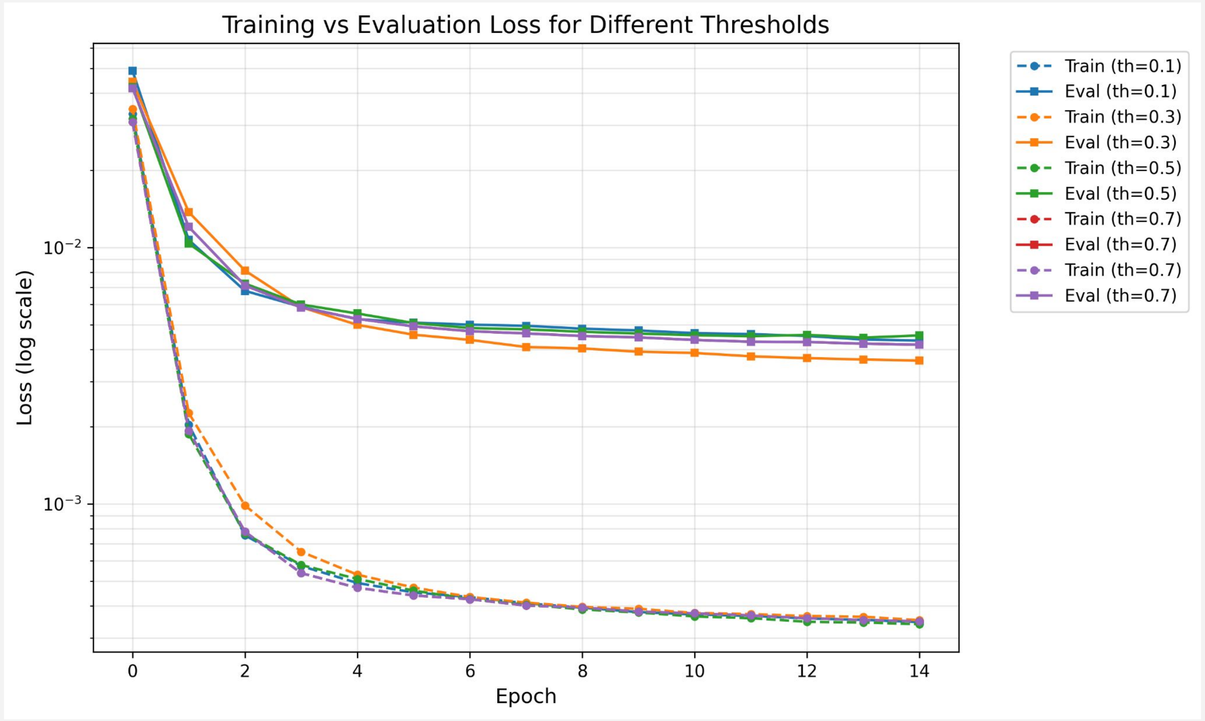
- Threshold to decide what is “noise” or “signal” in true data
- Currently set at 05

Training Samples


Truth
Output is charge deposit of sha
Signal Processing
Channel 0
Loose LF [High Efficiency, Low Purity]
Channel 1
MP 2
[2 plane co-incidence based on geometric knowledge]
Channel 2
MP 3
[3 plane co-incidence]



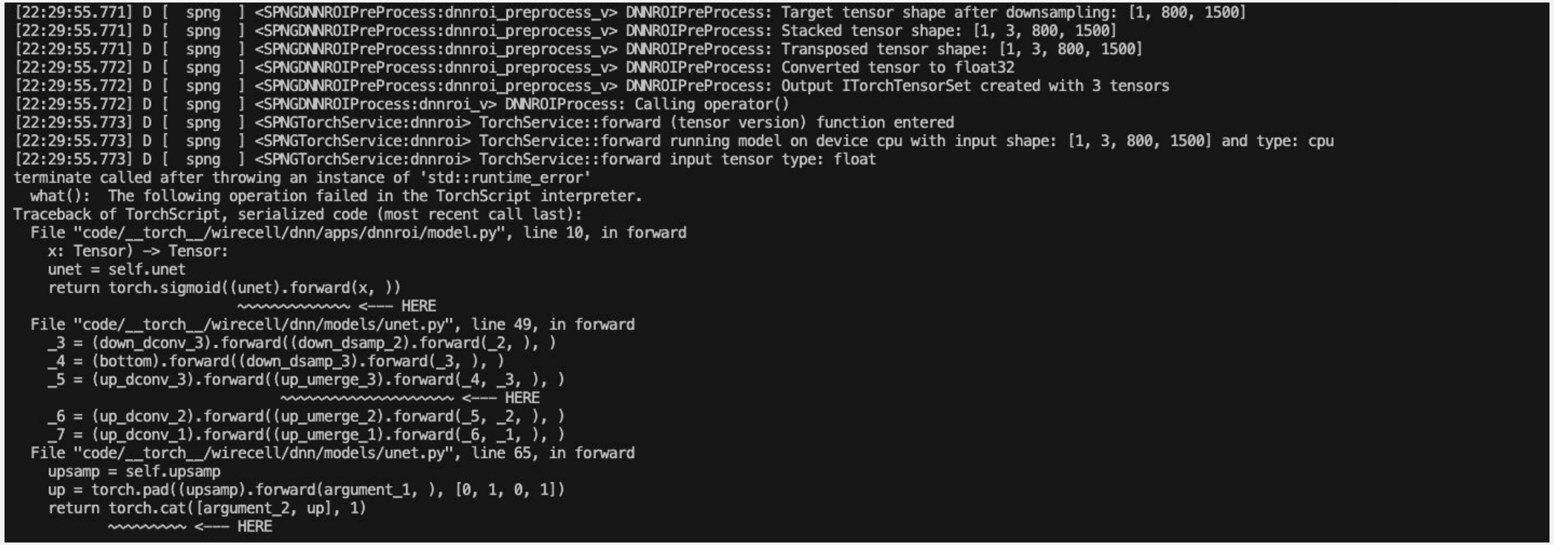
Training and evaluation loss with different threshold values



Training/evaluation loss are not that much affected by threshold value

Model evaluation

- Wanted to evaluate with the actual SPNG training….

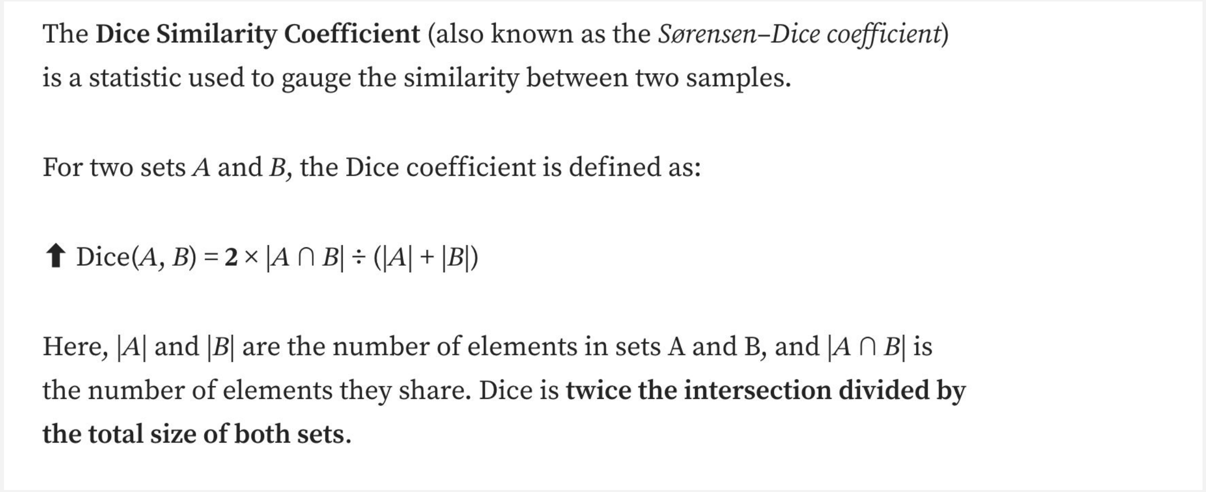


```
[22:29:55.771] D [  spng  ] <SPNGDNNROIPreProcess:dnnroi_preprocess_v> DNNROIPreProcess: Target tensor shape after downsampling: [1, 800, 1500]
[22:29:55.771] D [  spng  ] <SPNGDNNROIPreProcess:dnnroi_preprocess_v> DNNROIPreProcess: Stacked tensor shape: [1, 3, 800, 1500]
[22:29:55.771] D [  spng  ] <SPNGDNNROIPreProcess:dnnroi_preprocess_v> DNNROIPreProcess: Transposed tensor shape: [1, 3, 800, 1500]
[22:29:55.772] D [  spng  ] <SPNGDNNROIPreProcess:dnnroi_preprocess_v> DNNROIPreProcess: Converted tensor to float32
[22:29:55.772] D [  spng  ] <SPNGDNNROIPreProcess:dnnroi_preprocess_v> DNNROIPreProcess: Output ITorchTensorSet created with 3 tensors
[22:29:55.772] D [  spng  ] <SPNGDNNROIProcess:dnnroi_v> DNNROIProcess: Calling operator()
[22:29:55.773] D [  spng  ] <SPNGTorchService:dnnroi> TorchService::forward (tensor version) function entered
[22:29:55.773] D [  spng  ] <SPNGTorchService:dnnroi> TorchService::forward running model on device cpu with input shape: [1, 3, 800, 1500] and type: cpu
[22:29:55.773] D [  spng  ] <SPNGTorchService:dnnroi> TorchService::forward input tensor type: float
terminate called after throwing an instance of 'std::runtime_error'
  what():  The following operation failed in the TorchScript interpreter.
Traceback of TorchScript, serialized code (most recent call last):
  File "code/__torch__/wirecell/dnn/apps/dnnroi/model.py", line 10, in forward
    x: Tensor) -> Tensor:
    unet = self.unet
    return torch.sigmoid((unet).forward(x, ))
                         ~~~~~~~~~~~~~ <--- HERE
  File "code/__torch__/wirecell/dnn/models/unet.py", line 49, in forward
    _3 = (down_dconv_3).forward((down_dsamp_2).forward(_2, ), )
    _4 = (bottom).forward((down_dsamp_3).forward(_3, ), )
    _5 = (up_dconv_3).forward((up_umerge_3).forward(_4, _3, ), )
                              ~~~~~~~~~~~~~~~~~~~~ <--- HERE
    _6 = (up_dconv_2).forward((up_umerge_2).forward(_5, _2, ), )
    _7 = (up_dconv_1).forward((up_umerge_1).forward(_6, _1, ), )
  File "code/__torch__/wirecell/dnn/models/unet.py", line 65, in forward
    upsamp = self.upsamp
    up = torch.pad((upsamp).forward(argument_1, ), [0, 1, 0, 1])
    return torch.cat([argument_2, up], 1)
           ~~~~~~~~~ <--- HERE
```

- Need to change the default padding in umerge

Dice Similarity Coefficients to look at the performance

- Similarity between two samples
  - Apparently used in image segmentation evaluation…
- Trained on training samples
- Calculate Similarity Coefficient using evaluation sample



The **Dice Similarity Coefficient** (also known as the *Sørensen–Dice coefficient*) is a statistic used to gauge the similarity between two samples.

For two sets A and B, the Dice coefficient is defined as:

⬆ $\text{Dice}(A, B) = \mathbf{2} \times |A \cap B| \div (|A| + |B|)$

Here, $|A|$ and $|B|$ are the number of elements in sets A and B, and $|A \cap B|$ is the number of elements they share. Dice is **twice the intersection divided by the total size of both sets.**

- Tested with thresholds 0.1,0.3,0.5,0.7,0.9
- Dice Similarity Coefficients ~ 0.9