PDHD APA1 “Fix” Network

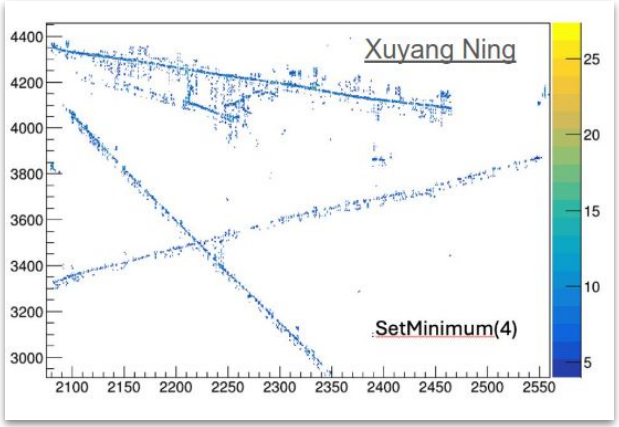
Jake Calcutt

Motivation

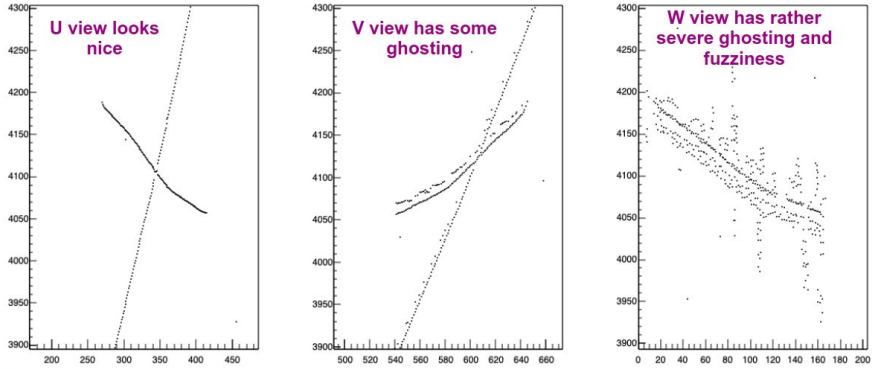
Xuyang has done a lot of work to determine a more accurate field response for PDHD

Even with this, downstream reco (i.e. Pandora) performance remains degraded

- Can we ‘fix’ this?
  - Note: this is not saying the Field Response is wrong – instead we want to see if we can recover what the image would have looked like with a properly-biased APA



- Example beam event from data - the collection plane looks fuzzy
  - Clearer if we zoom in and look at the points



Potential Solution

Can we simulate readout with signal (i.e. MIPs or MIP-like tracks) in the APA1 drift volume

1. With the ideal bias applied to the wires
    a. “Target”
2. With some varied/assumed bias configuration (i.e. $V_W = 0$)
    a. “Input”

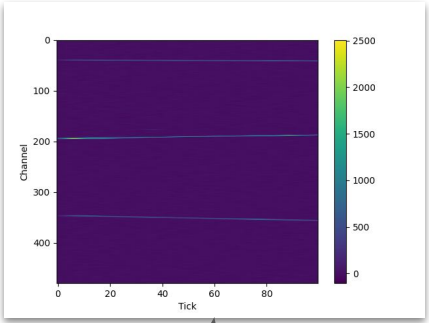
And train a UNet (like the DNNROI network) to transform the input ($V_W = 0$) to the ideal configuration

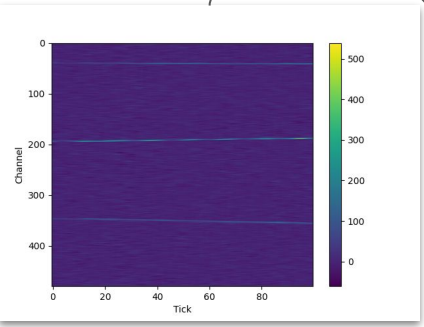
Network/Training Setup

Compare against all planes (shown here: drift-volume W)

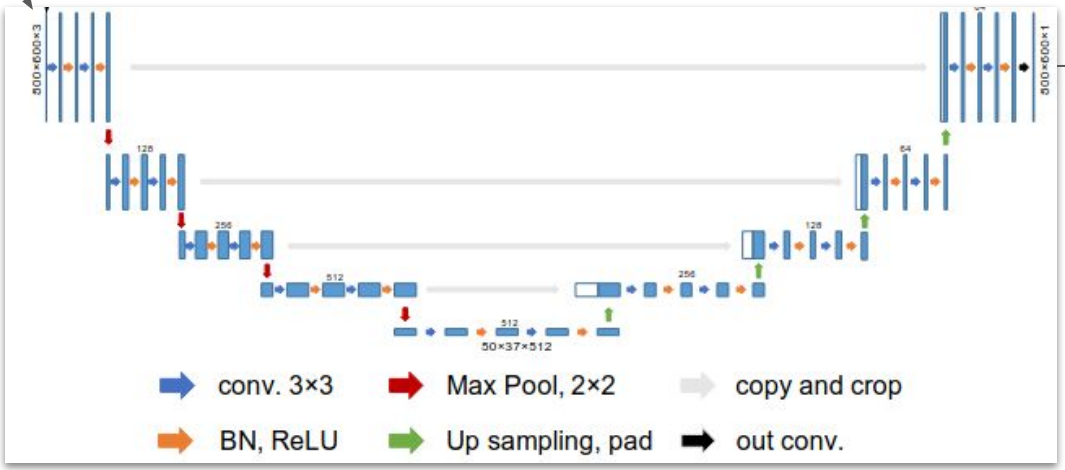
To learn: Shape & height of signals

Target – Ideal





Input – ‘Bad’ APA1



Compare against

Input Creation

Made some PDHD cosmics simulation with both APA configurations

- Same simulation (CORSIKA + G4) up to wirecell drift simulation then split into APA configs

Extract first 100 ticks (500ns each –> 50us total) from each

- Simulated field response known to 100us
  - I made a mistake when setting up the extraction – can rerun jobs to make sure I get the correct size)
  - “Fine” for proof of principle

Instead – can set up wirecell simulation workflow with random directions/positions/lengths of MIP tracks simulated in a single PDHD TPC with the 2 different APA setups

- Then: only need to simulate 200 ticks exactly

First Attempts

Copied DNNROI UResNet implementation (model, parts) and trained using only simulated readout from the drift-volume TPC

N Features: 16, 32, 64, 128

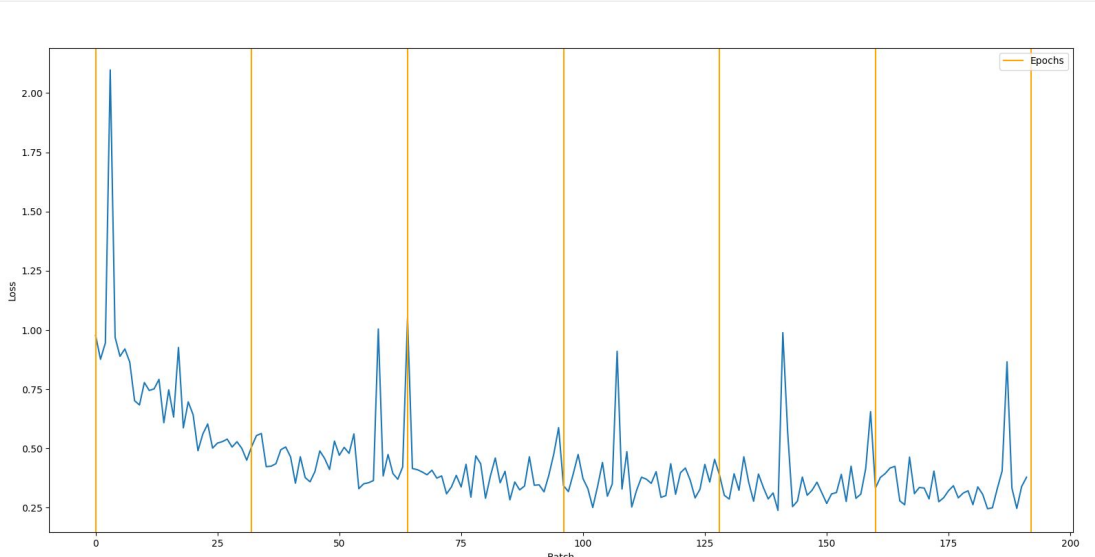
Scaling inputs & truth by 1./128.

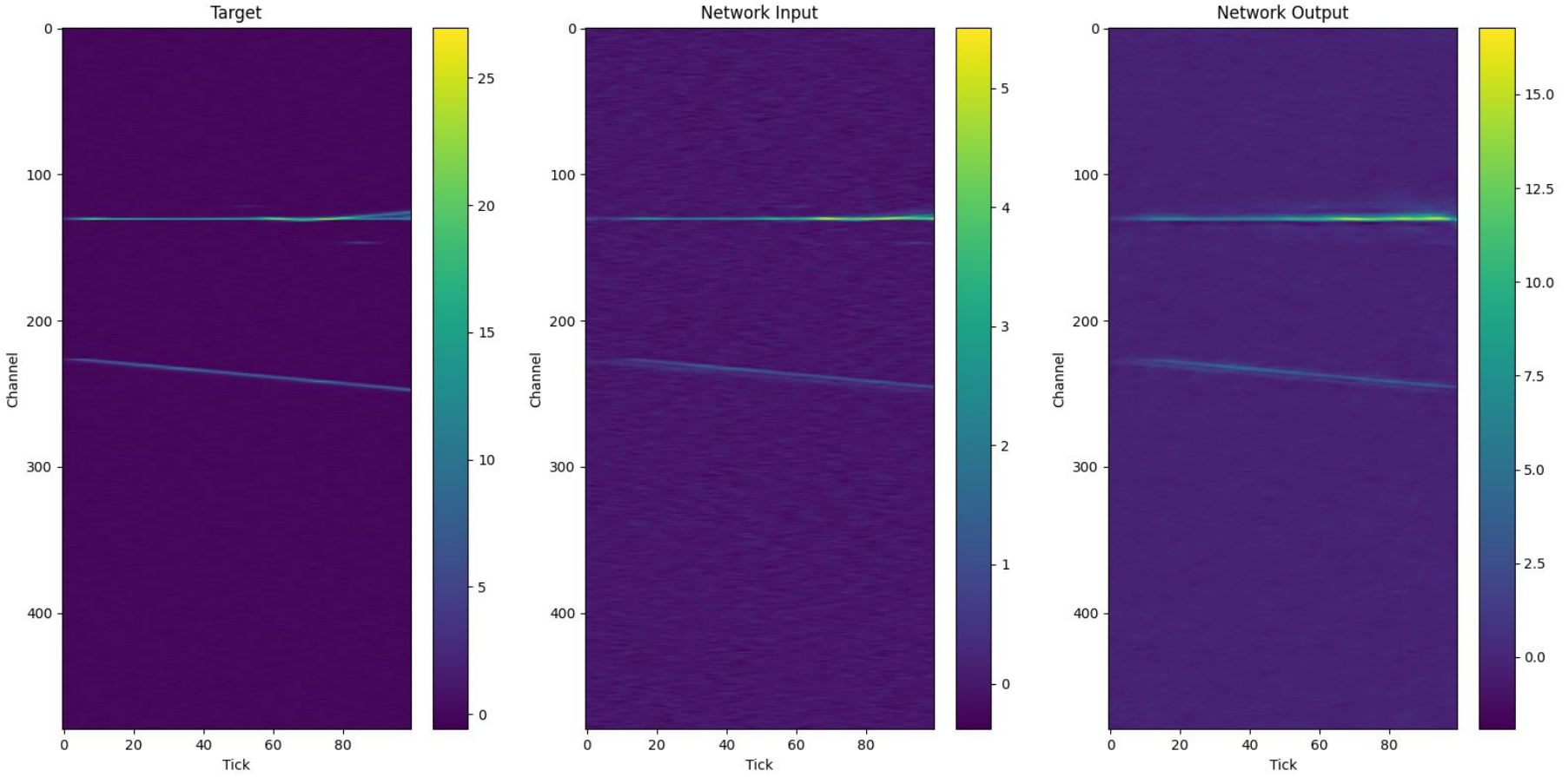
- Need to figure out better scale

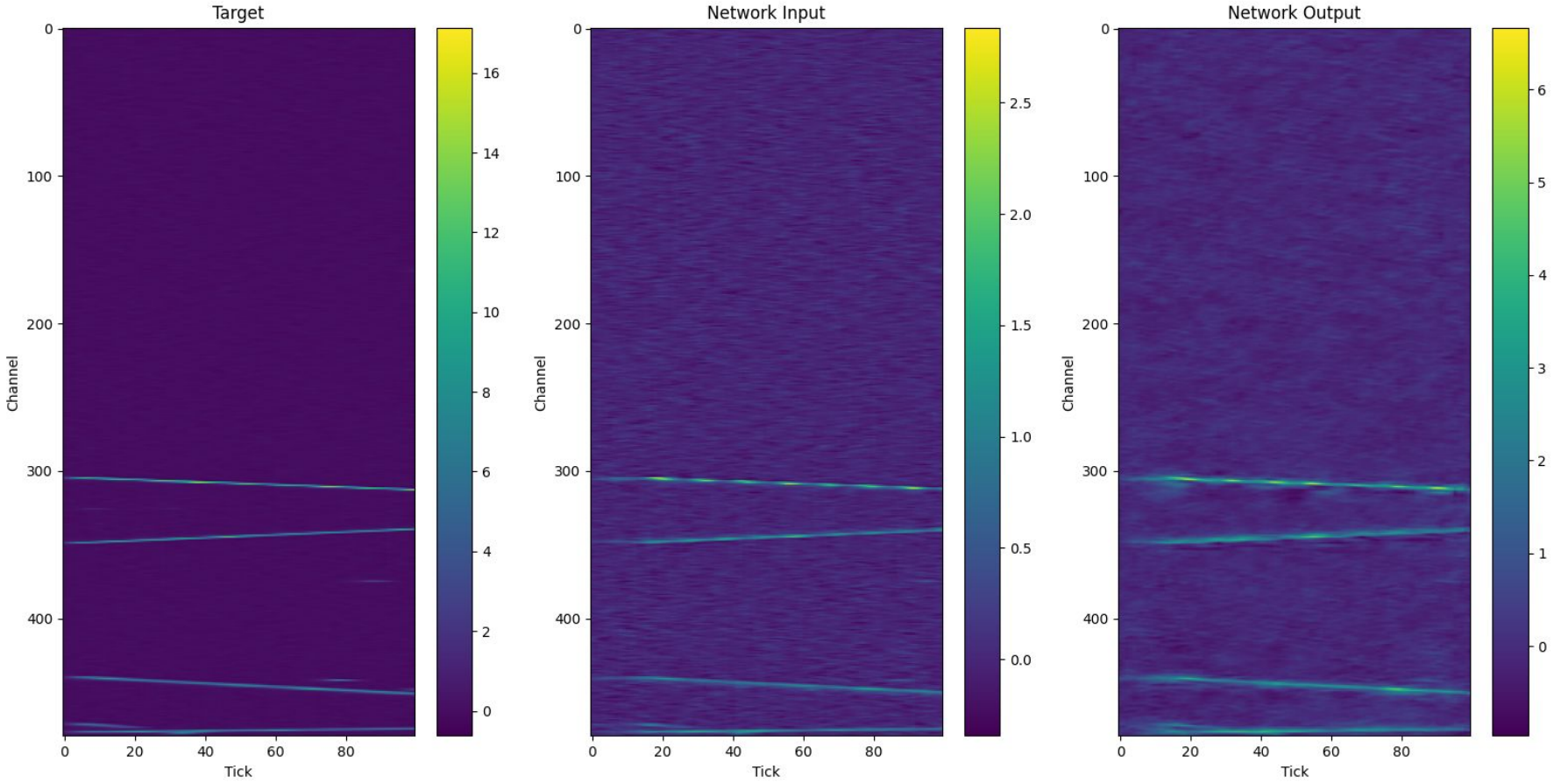
I believe the spikes are from higher-occupancy events

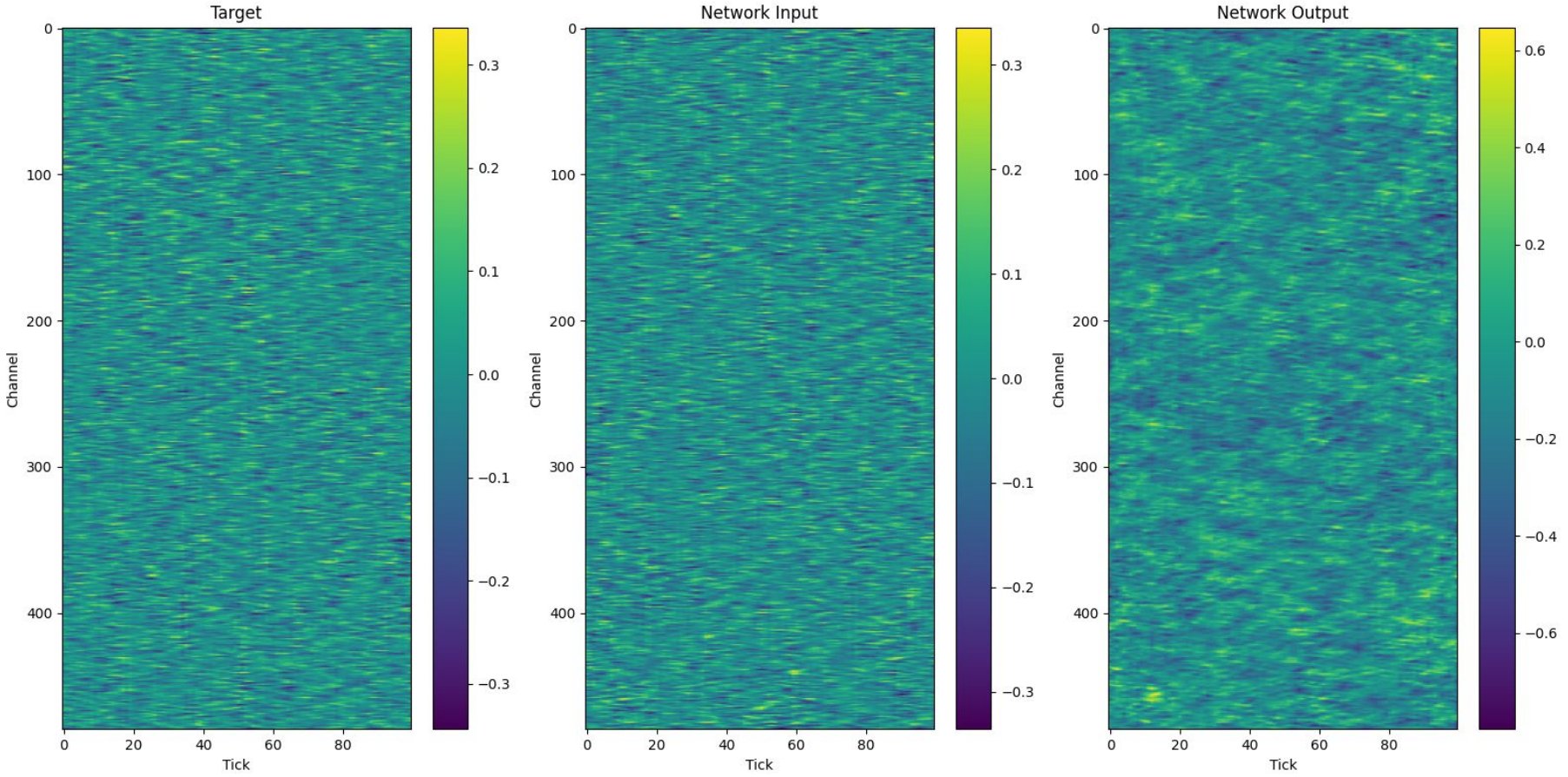
- Need to check this

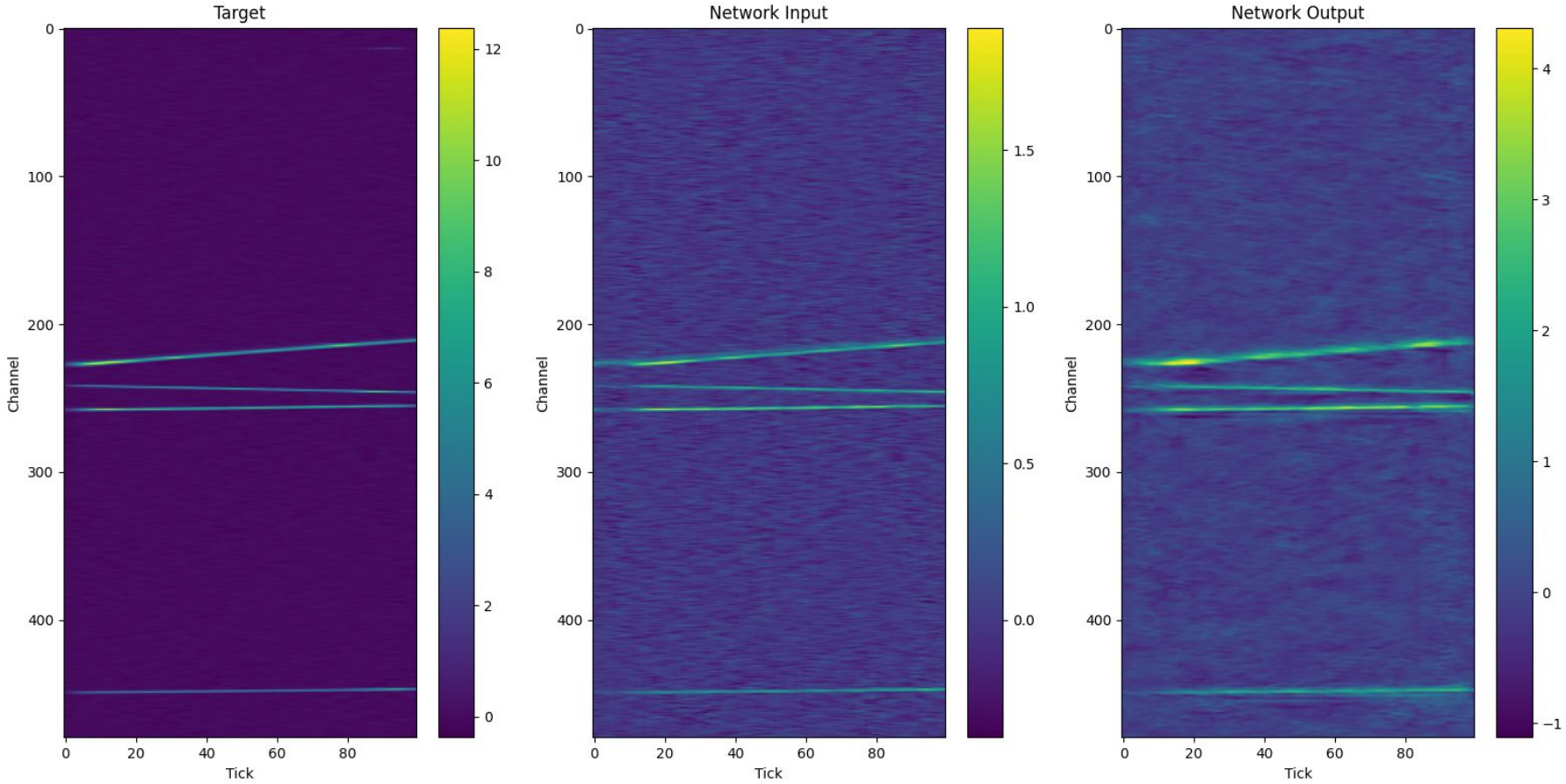
Loss flattens out after ~3 epochs

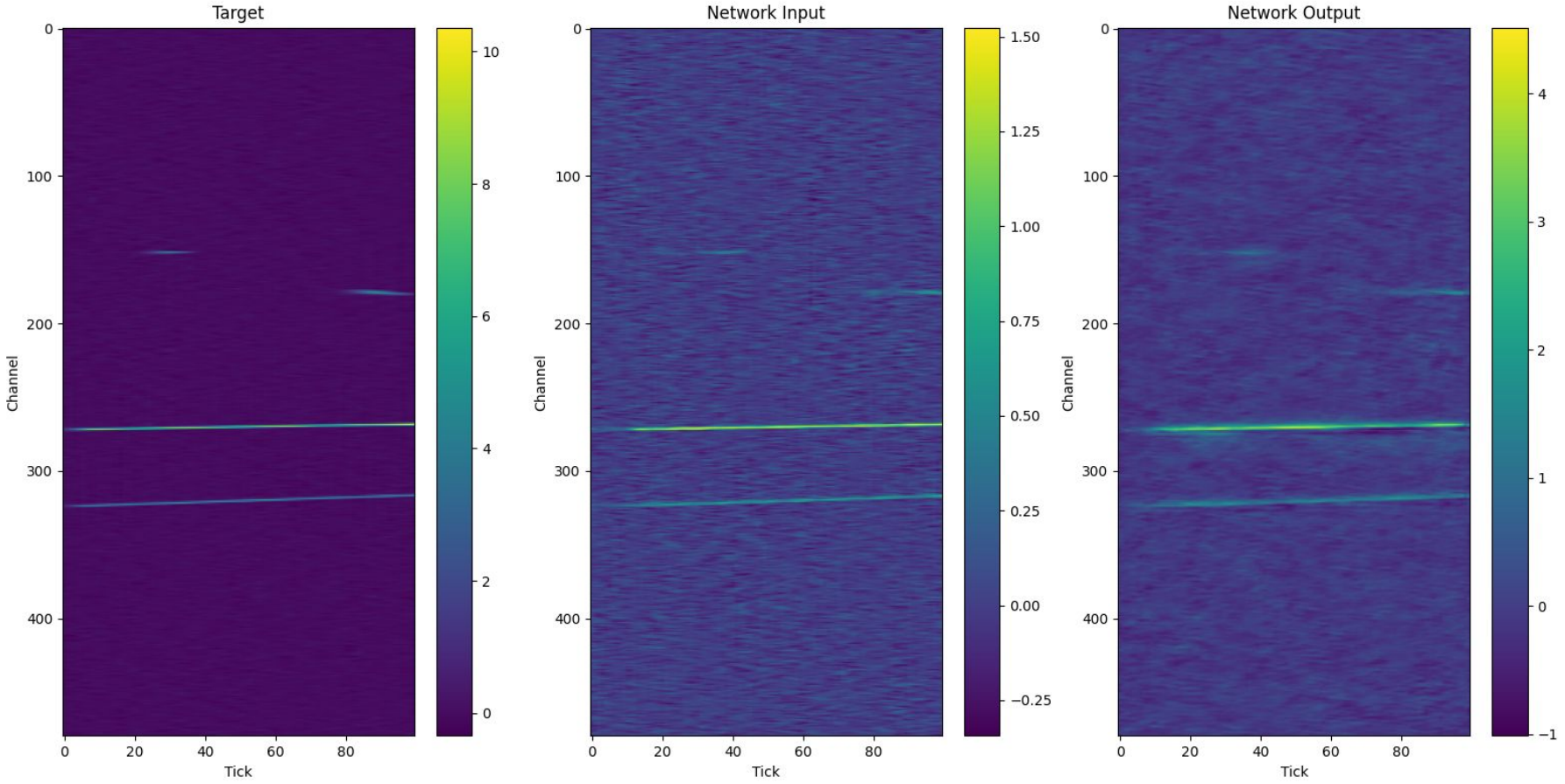


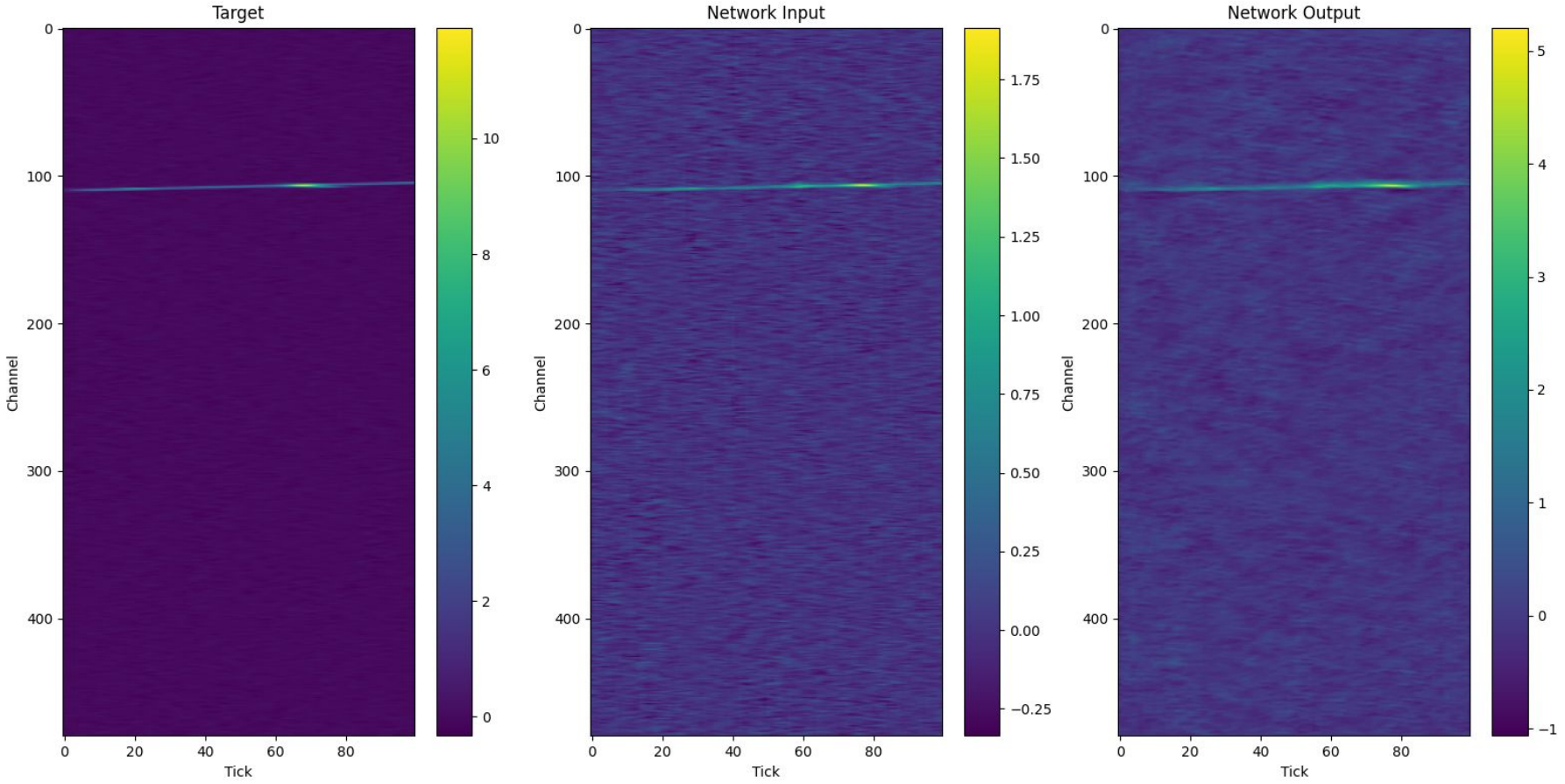

Target
Network Input
Network Output
Channel
Tick
0
100
200
300
400
0
20
40
60
80
25
20
15
10
5
0
4
3
2
1
15.0
12.5
10.0
7.5
5.0
2.5
0.0


Target
Network Input
Network Output
Channel
Tick
0
100
200
300
400
0
20
40
60
80
16
14
12
10
8
6
4
2
0
2.5
2.0
1.5
1.0
0.5
0.0
6
5
4
3
2
1
0


Target
Network Input
Network Output
Channel
Tick
0
100
200
300
400
0
20
40
60
80
0.3
0.2
0.1
0.0
−0.1
−0.2
−0.3
0.6
0.4
−0.4
−0.6


Target
Network Input
Network Output
Channel
Tick
0
100
200
300
400
0
20
40
60
80
12
10
8
6
4
2
0
1.5
1.0
0.5
0.0
4
3
2
1
0
−1


Target
Network Input
Network Output
Channel
Tick
0
100
200
300
400
0
20
40
60
80
10
8
6
4
2
0
1.50
1.25
1.00
0.75
0.50
0.25
0.00
−0.25
4
3
2
1
0
−1

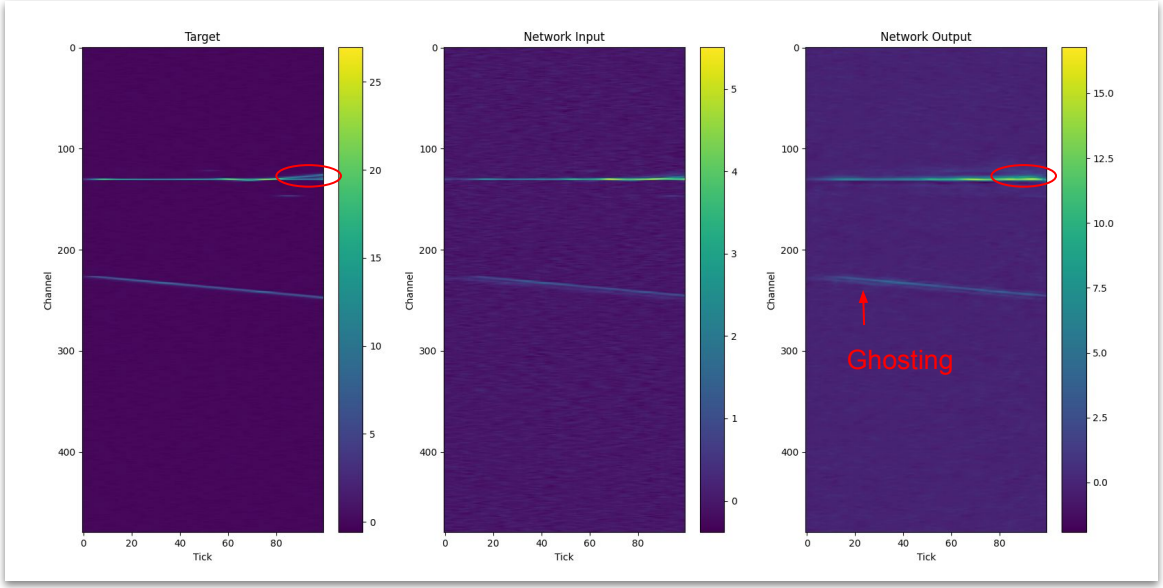

Target
Network Input
Network Output
Channel
Tick
0
100
200
300
400
0
20
40
60
80
10
8
6
4
2
0
1.75
1.50
1.25
1.00
0.75
0.50
0.25
0.00
−0.25
5
4
3
2
1
0
−1

Discussion

Overall scale of signal is closer to target

Some signal remains ‘washed out’

Ghosting remains



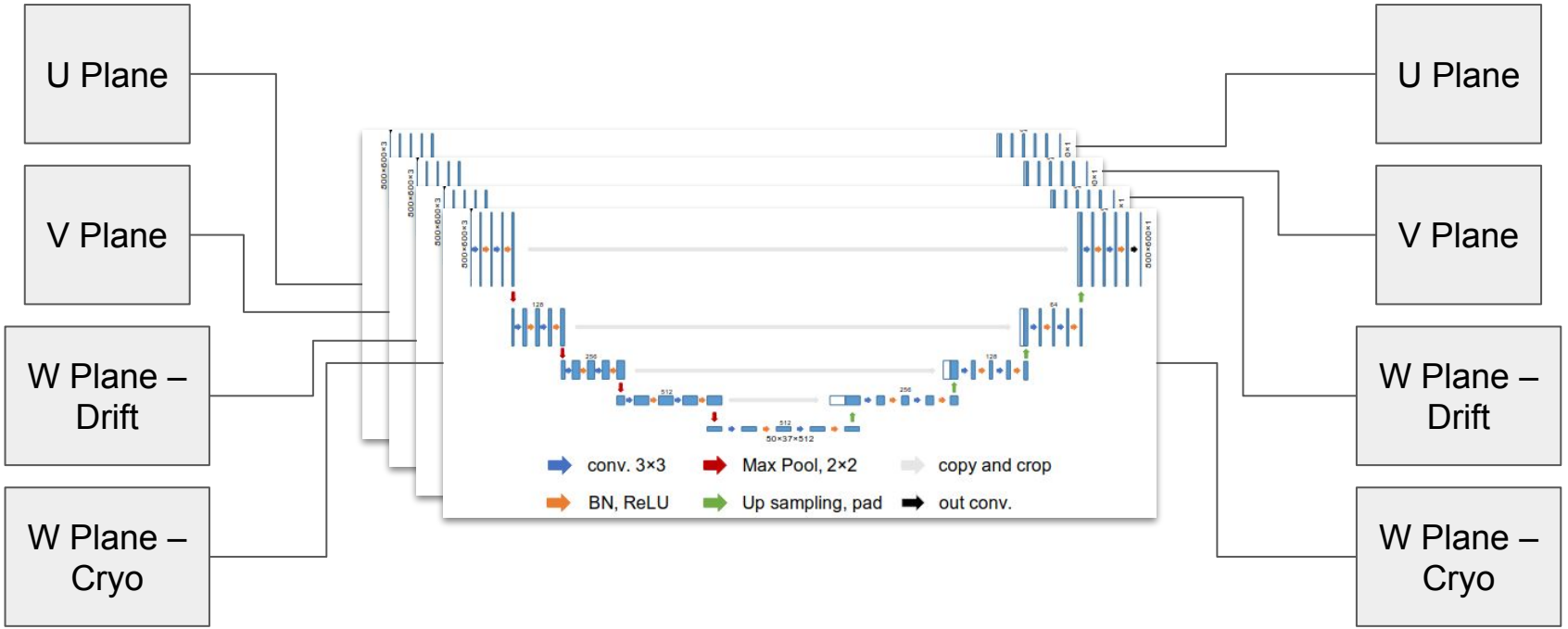
Future Improvements

More features in UNet layers

Including readout from all planes should ‘recover’ charge that is not collected on W planes

- How best to combine the info?
  - My naive intuition: separated UNets for each plane trained at once wouldn’t be so powerful

Simple extension for all planes



Extension for all planes