中央研究院物理研究所
INSTITUTE OF PHYSICS, ACADEMIA SINICA

Status report

2025/05/15

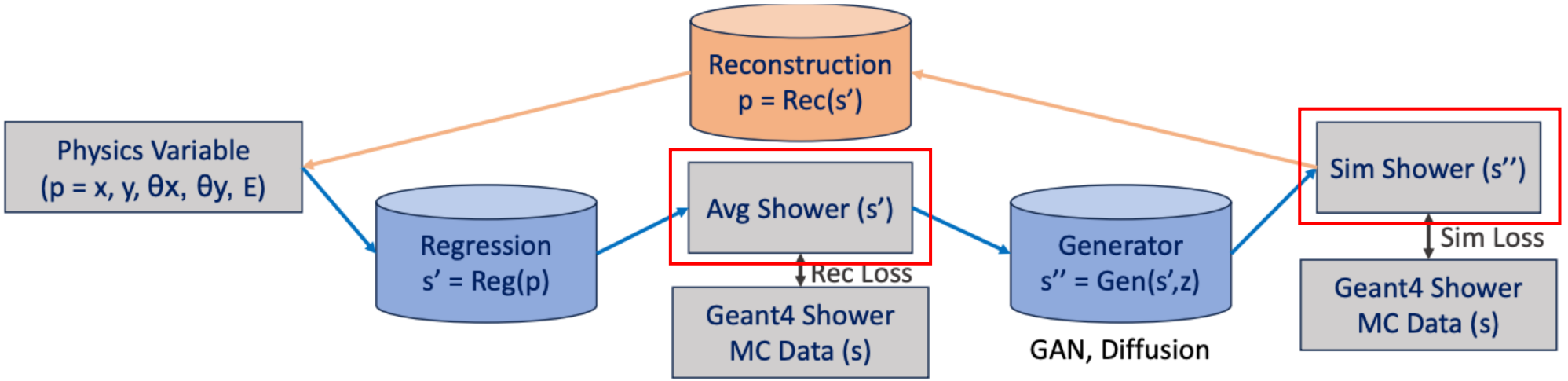
ZDC Internal

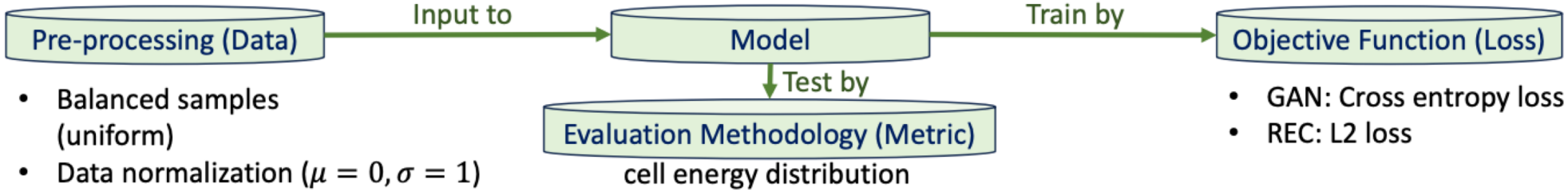
WAI YUEN CHAN

Overview

- Focus on improving the performance of GAN-CNN

Introduction

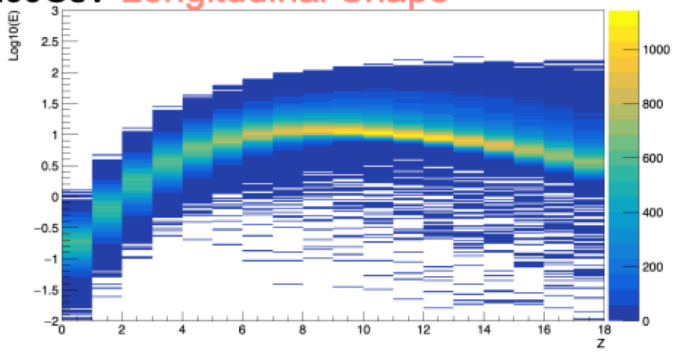




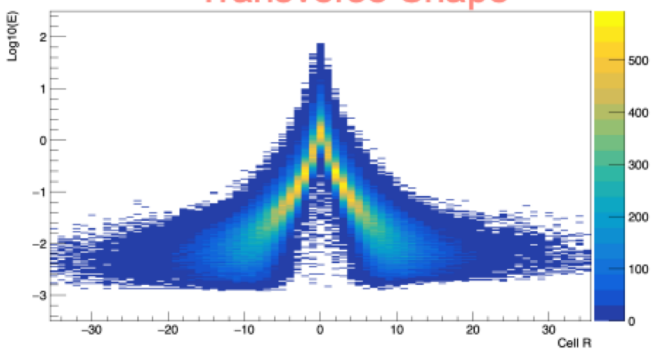
- Balanced samples (uniform)
- Data normalization ($\mu = 0, \sigma = 1$)

- GAN: Cross entropy loss
- REC: L2 loss

e-, 100GeV Longitudinal Shape



Transverse Shape



The longitudinal shower profile is described by the Gamma distribution:

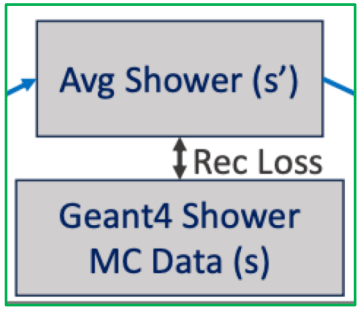
$$\frac{dE}{dt}(t) = E_0 \frac{(\beta t)^{\beta T_0} \beta e^{-\beta t}}{\Gamma(\beta T_0 + 1)}, \quad (1)$$

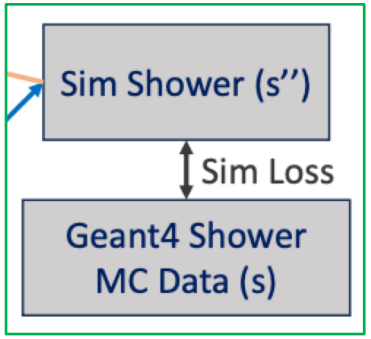
using parameters described above. The scale parameter β is found to be constant over the wide energy range, therefore it is fixed in the present analysis. Individual shower parameters E_0 and T_0 are obtained from a fit to observed energy depositions in the ECAL cells of that shower. Fig. 7 shows the high quality of the description of electron showers over a wide energy range by the model based on Eq. (1).

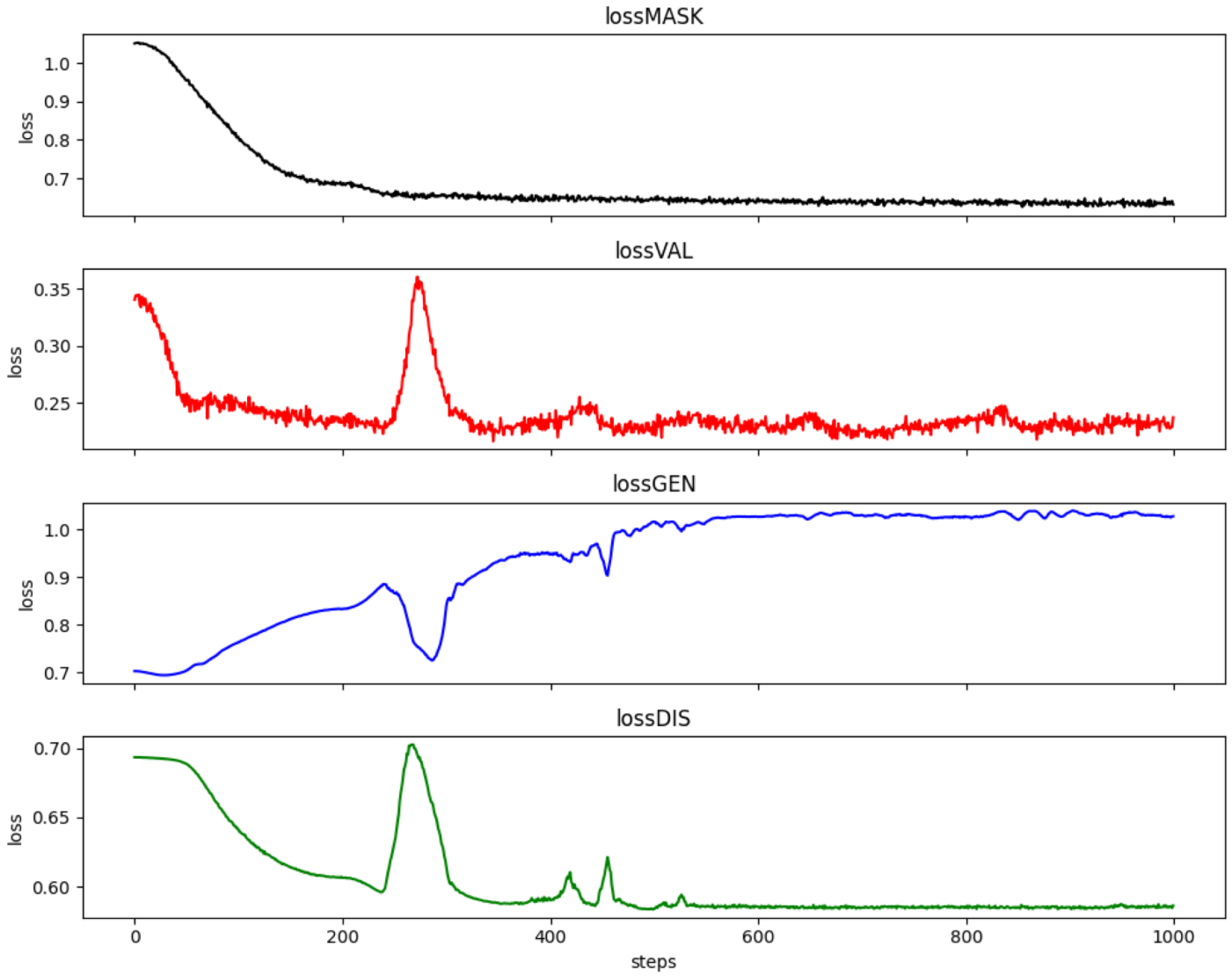
At a given shower depth, the transverse shower shape as a function of the distance from the shower axis r is described by the sum of a narrow core and a wide tail:

$$\frac{dE}{dr}(t, r) \propto Q_C \frac{2rR_C^2}{(r^2 + R_C^2)^2} + (1 - Q_C) \frac{2rR_T^2}{(r^2 + R_T^2)^2}, \quad (2)$$

Recap: Training result





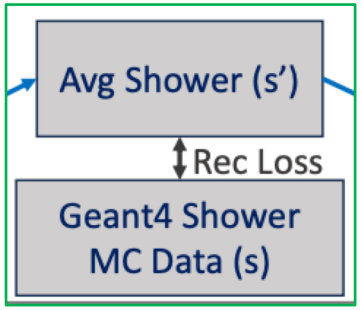


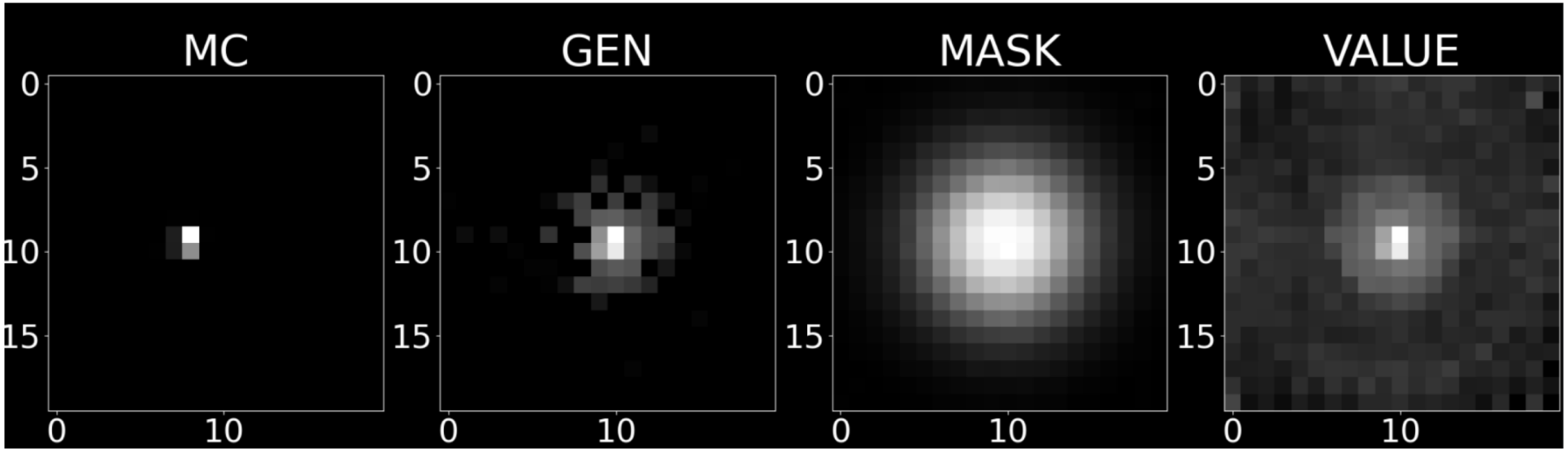
We have 4 losses and corresponding to different parameters.

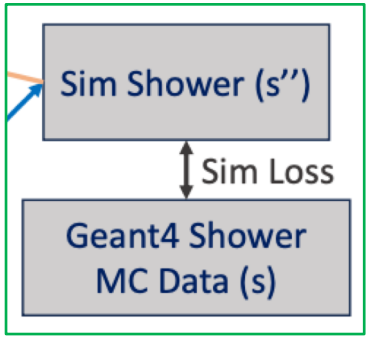
The training use 4000 events and the test use 1000 events in this test run.

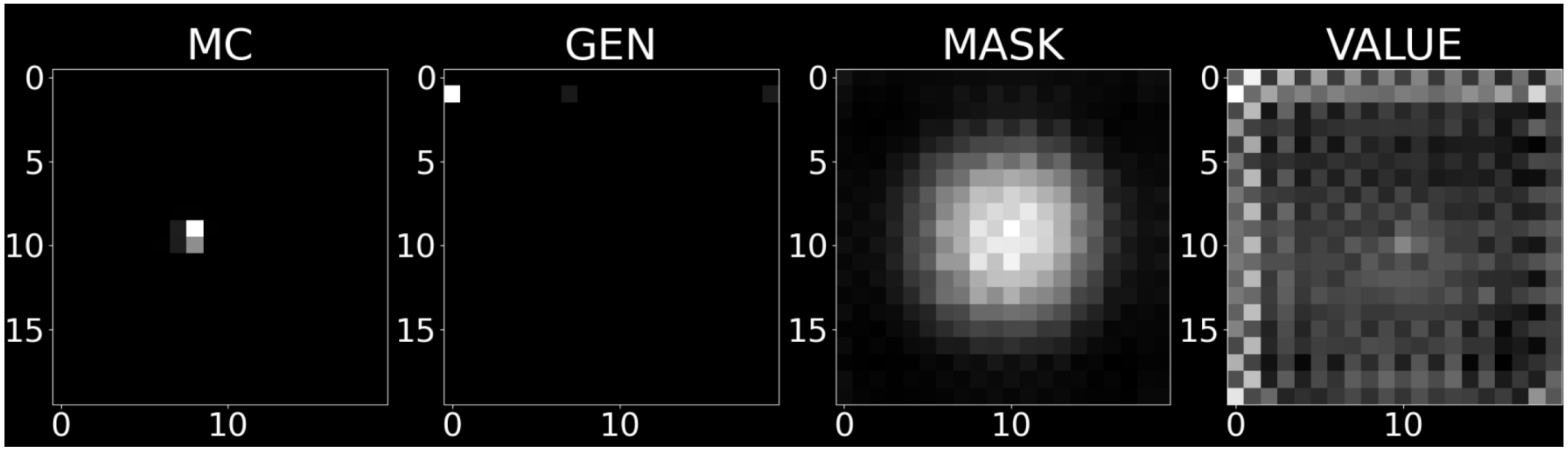
1000 steps = 500 epochs have been used.

Recap: Training result



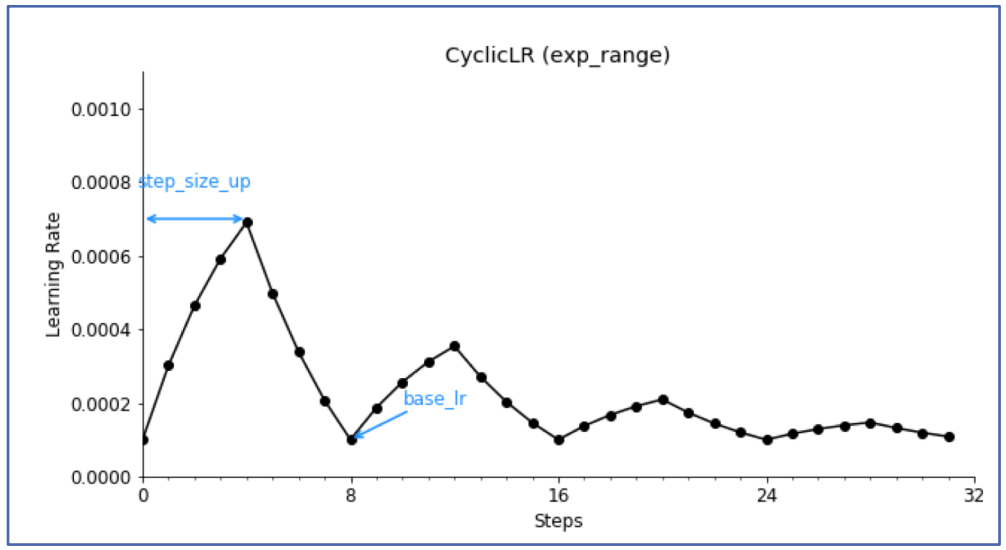




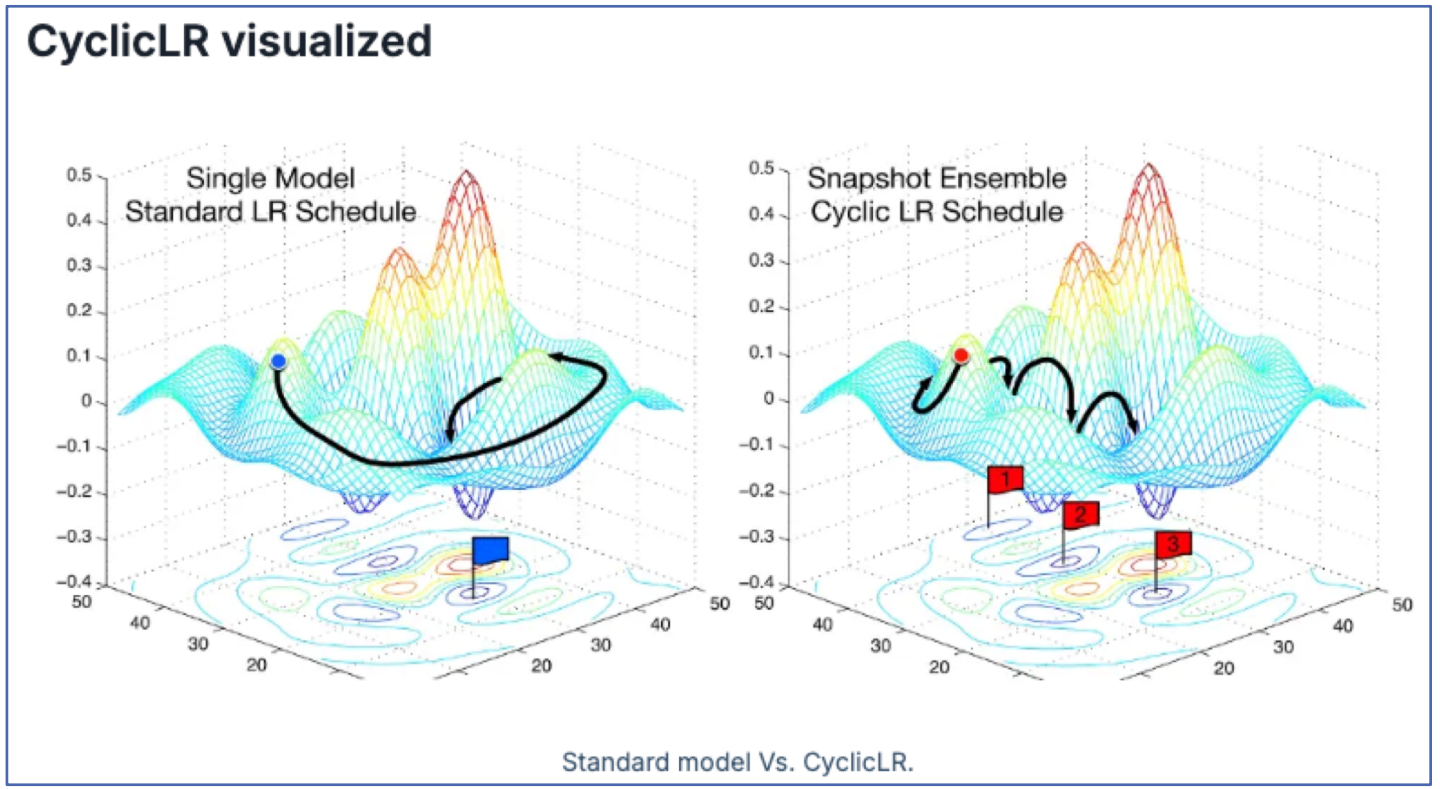


Hyperparameters in the ML training

- Learning rate scheduler: a function which dynamically adjusts the learning rate over the training. The learning rate cycles through these bounds in a triangular or exponential pattern, which helps the model escape local minima and converge faster.
- Here we use ”CyclicLR”




```
cyclic_base_lr = 1e-6
cyclic_max_lr = 1e-4
cyclic_step_size_up = 40
cyclic_step_size_down = 160
cyclic_mode = 'exp_range'
cyclic_gamma = 0.999925
```



Hyperparameters in the ML training

- We have tried different combination of hypermeters, in order to stabilise the loss function.

CyclicLR: default

```
cyclic_base_lr = 1e-6
cyclic_max_lr = 1e-4
cyclic_step_size_up = 40
cyclic_step_size_down = 160
cyclic_mode = 'exp_range'
cyclic_gamma = 0.999925
```

CyclicLR: reduced

```
cyclic_base_lr = 1e-6
cyclic_max_lr = 5e-5
cyclic_step_size_up = 80
cyclic_step_size_down = 80
cyclic_mode = 'exp_range'
cyclic_gamma = 0.999
```

CyclicLR: asymmetry

```
cyclic_base_lr = 1e-6
cyclic_max_lr_GEN = 1e-4 #GEN
cyclic_max_lr_DIS = 7e-5 #DIS
cyclic_step_size_up = 100
cyclic_step_size_down = 100
cyclic_mode = 'exp_range'
cyclic_gamma = 0.995
```

Name	CNN	Label ratio	Training ratio	CyclicLR setting	epochs
Default	default	0.73:0.27	1:1	default	100
Test 1	simplified	0.73:0.27	1:1	default	100
Test 2	simplified	0.8:0.2	1:1	default	150
Test 3	simplified	0.8:0.2	1:1	deduced	150
Test 4	simplified	0.7:0.3	1:1	asymmetry	150
Test 5	simplified	0.65:0.35	1:3	asymmetry	150
Test 6	simplified	0.6:0.4	1:3	asymmetry	150

Hyperparameters in the ML training

- We have tried different combination of hypermeters, in order to stabilise the loss function.

CNN: default (example)

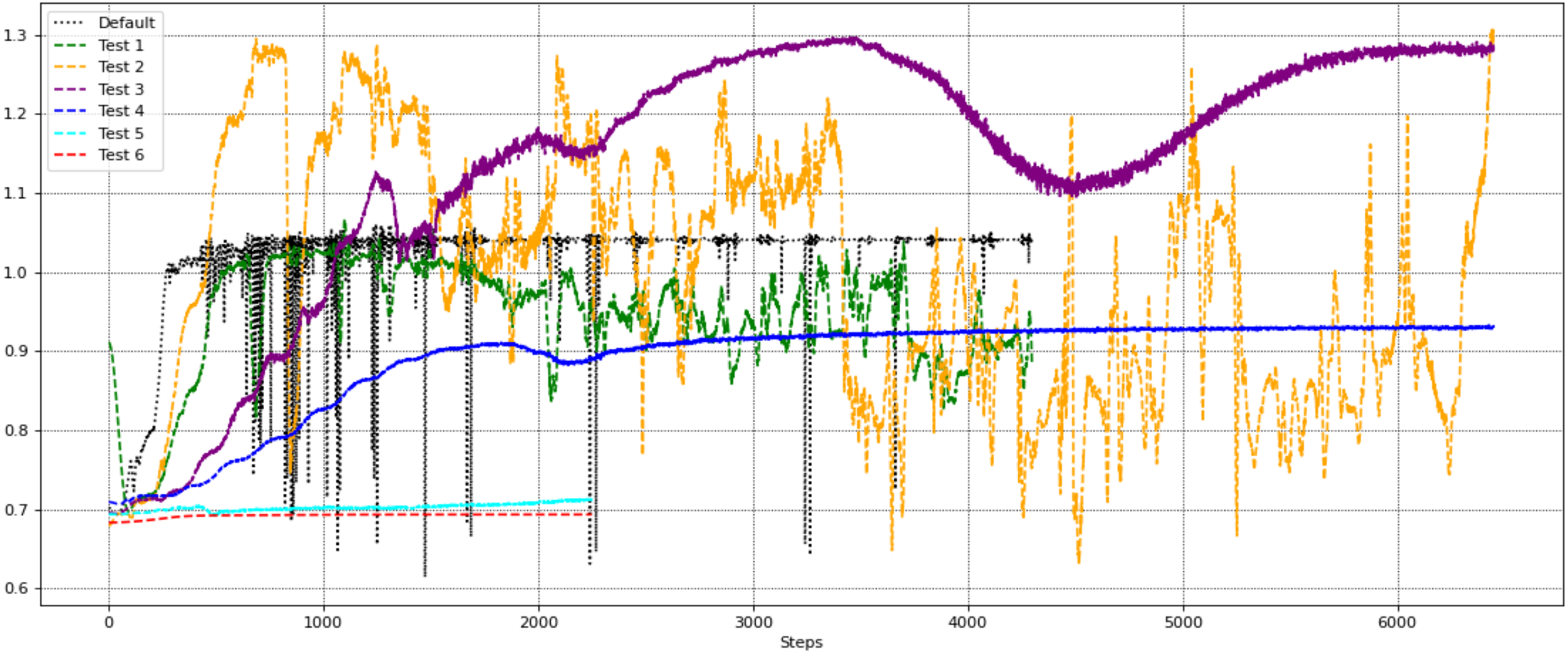
```
self.proj_head = nn.Sequential(
        Conv2dBlockH5W5(hidden_dim, hidden_dim),
        Conv2dBlockH5W5(hidden_dim, hidden_dim),
        PixelUnshuffle2D(2, 2), # (10, 10)
        Conv2dBlockH5W5(hidden_dim*4, hidden_dim*2),
        Conv2dBlockH5W5(hidden_dim*2, hidden_dim),
        PixelUnshuffle2D(2, 2), # (5, 5)
        Conv2dBlockH3W3(hidden_dim*4, hidden_dim*2),
        Conv2dBlockH3W3(hidden_dim*2, hidden_dim),
        nn.Conv2d(hidden_dim, hidden_dim,
                  kernel_size = (3, 3), stride = (1, 1),
                  padding = (0, 0)), # (3, 3)
        nn.Flatten(start_dim = 1, end_dim = 3), # 9 = 3 x 3
        LinearBlock(hidden_dim*9, hidden_dim*4, 4),
        LinearBlock(hidden_dim*4, hidden_dim*4, 4),
        nn.Linear(hidden_dim*4, 1),
        nn.Sigmoid()
        )
```

CNN: simplified (example)

```
self.proj_head = nn.Sequential(
        Conv2dBlockH5W5(hidden_dim, hidden_dim),
        #Conv2dBlockH5W5(hidden_dim, hidden_dim),
        PixelUnshuffle2D(2, 2), # (10, 10)
        Conv2dBlockH5W5(hidden_dim*4, hidden_dim),
        #Conv2dBlockH5W5(hidden_dim*2, hidden_dim),
        PixelUnshuffle2D(2, 2), # (5, 5)
        Conv2dBlockH3W3(hidden_dim*4, hidden_dim),
        #Conv2dBlockH3W3(hidden_dim*2, hidden_dim),
        nn.Conv2d(hidden_dim, hidden_dim,
                  kernel_size = (3, 3), stride = (1, 1),
                  padding = (0, 0)), # (3, 3)
        nn.Flatten(start_dim = 1, end_dim = 3), # 9 = 3 x 3
        LinearBlock(hidden_dim*9, hidden_dim*4, 4),
        #LinearBlock(hidden_dim*4, hidden_dim*4, 4),
        nn.Linear(hidden_dim*4, 1),
        nn.Sigmoid()
        )
```

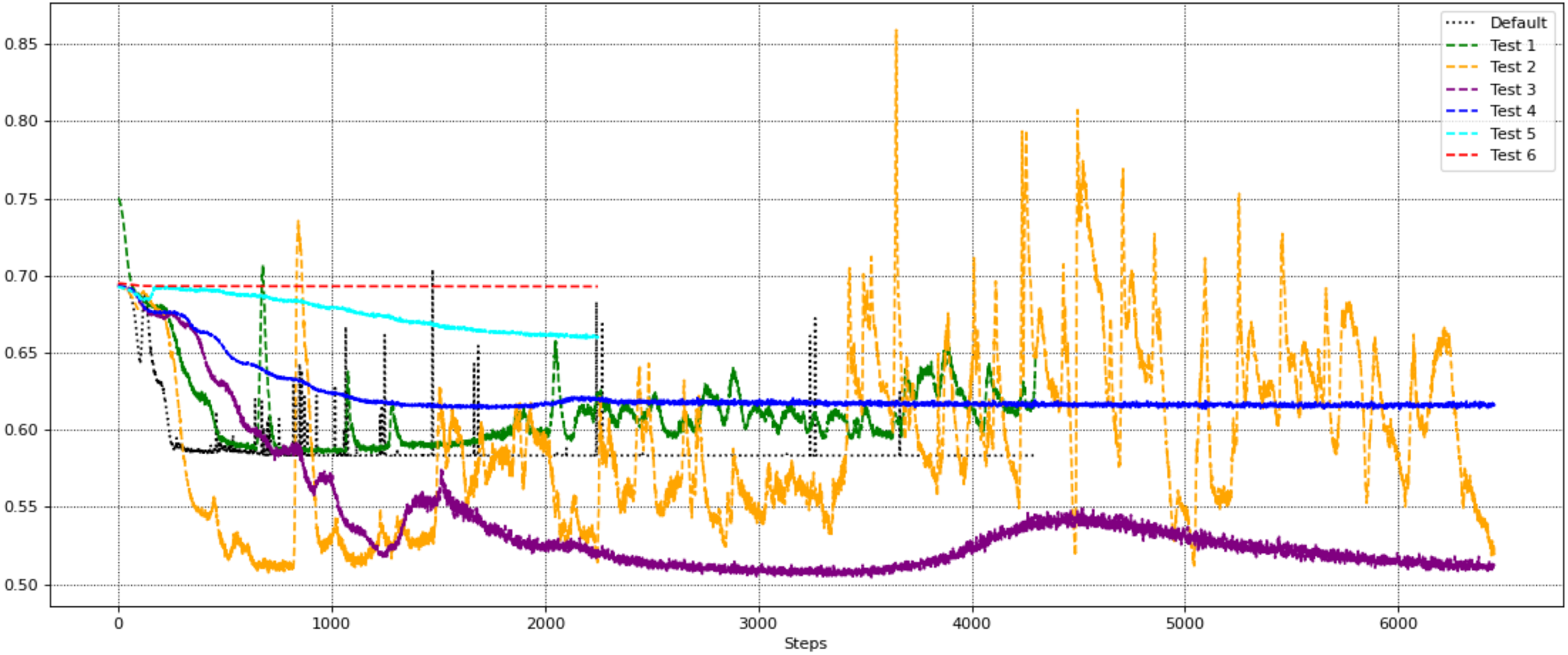
Hyperparameters in the ML training

lossGEN



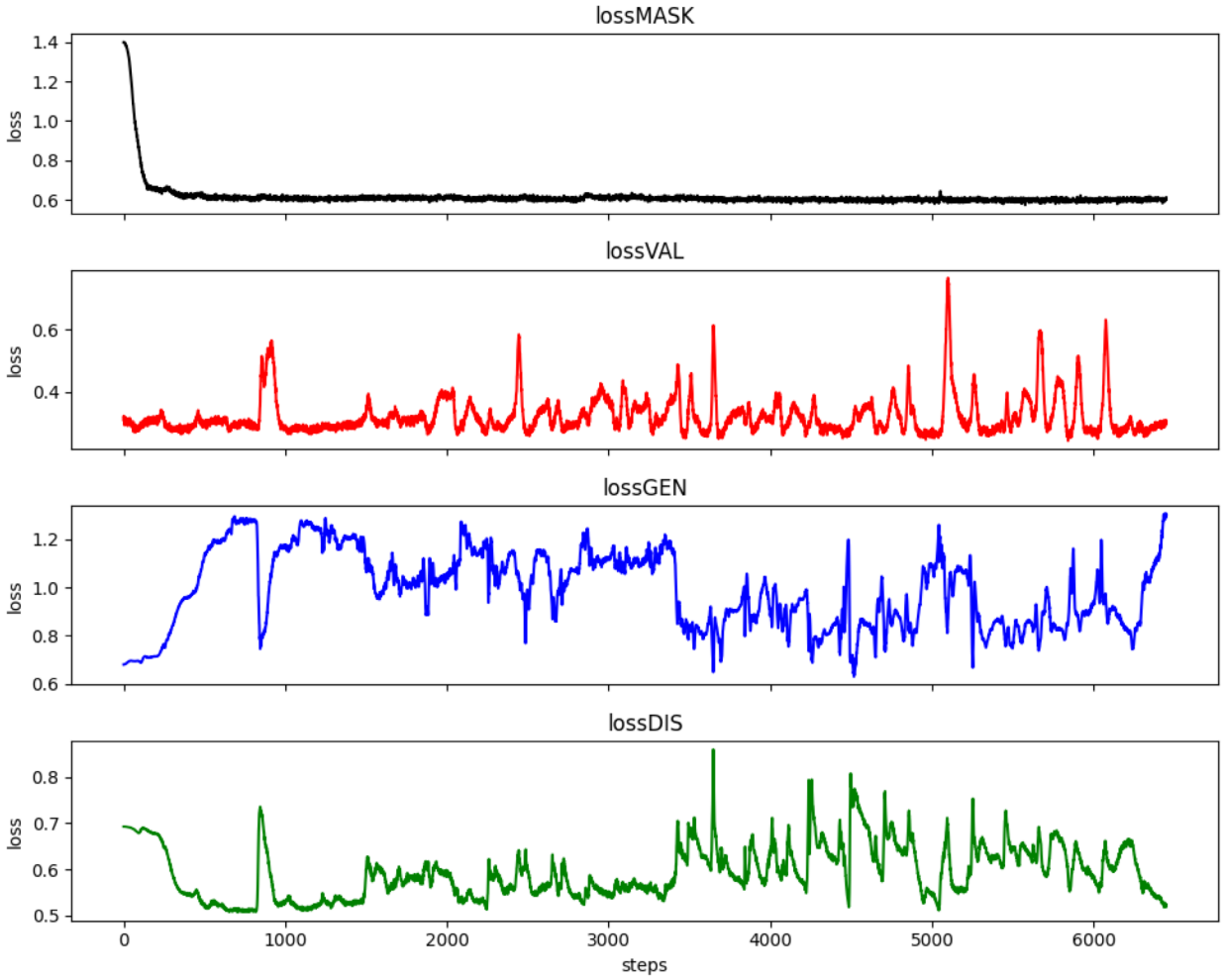
Hyperparameters in the ML training

lossDIS

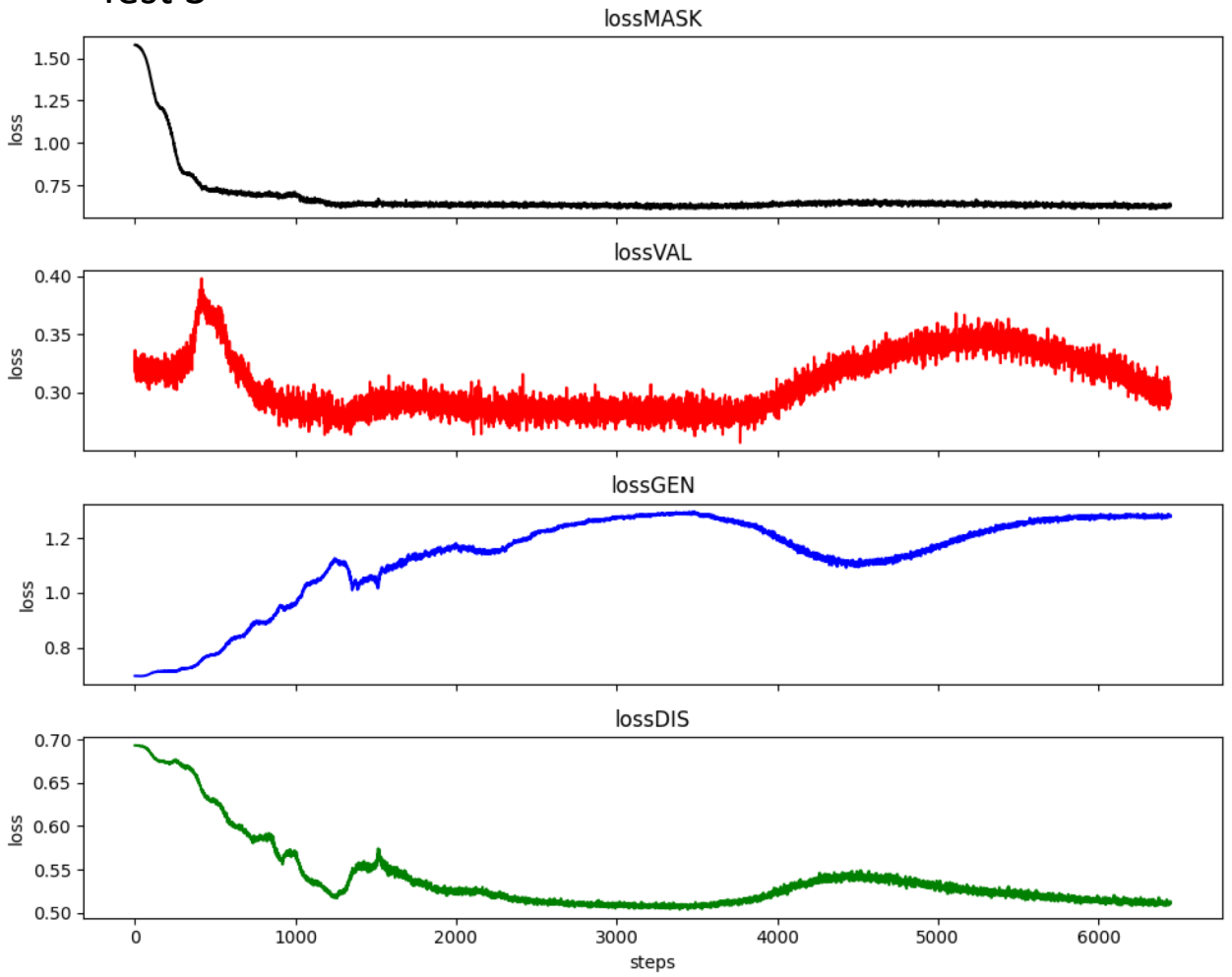


Hyperparameters in the ML training

Test 2

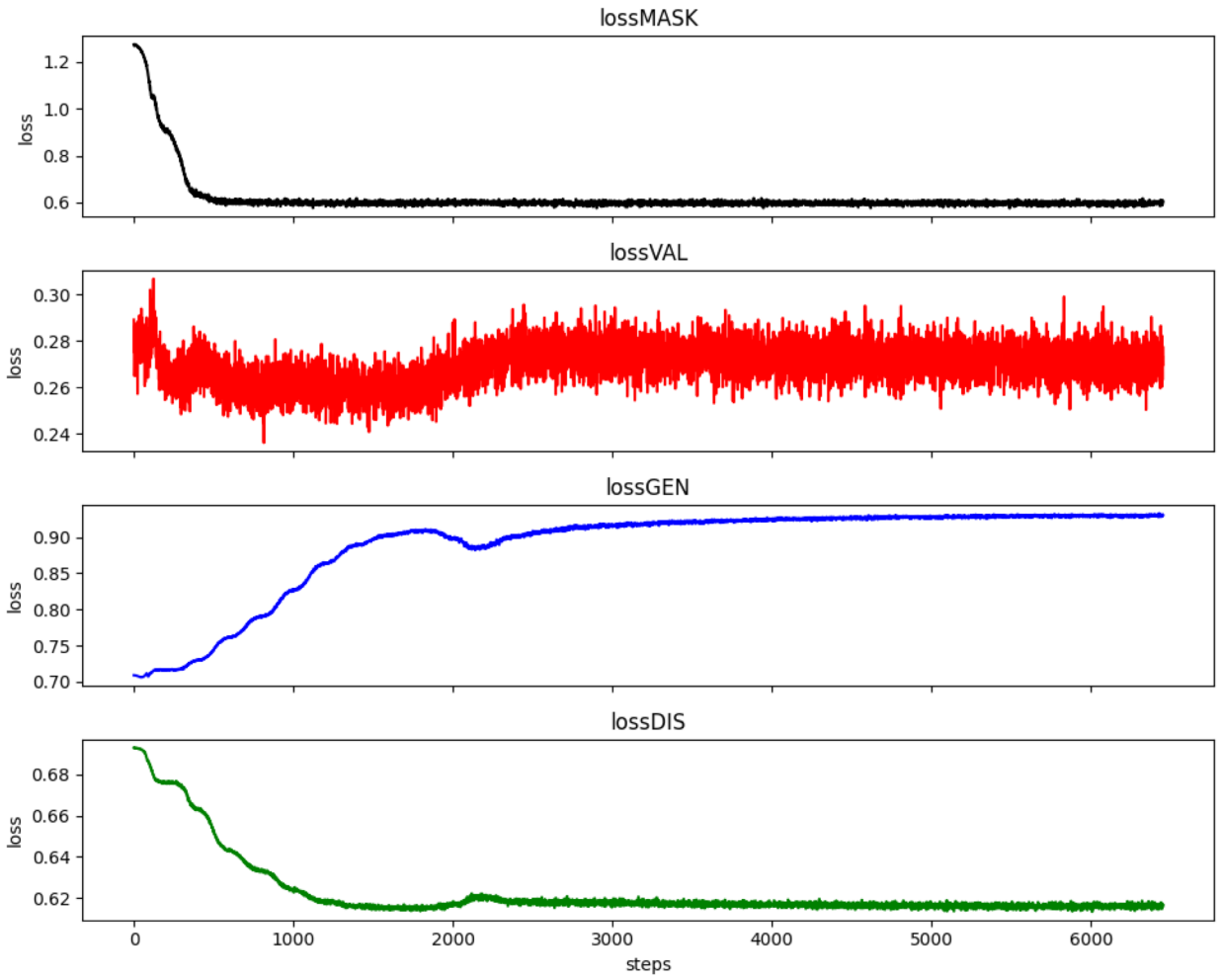


Test 3

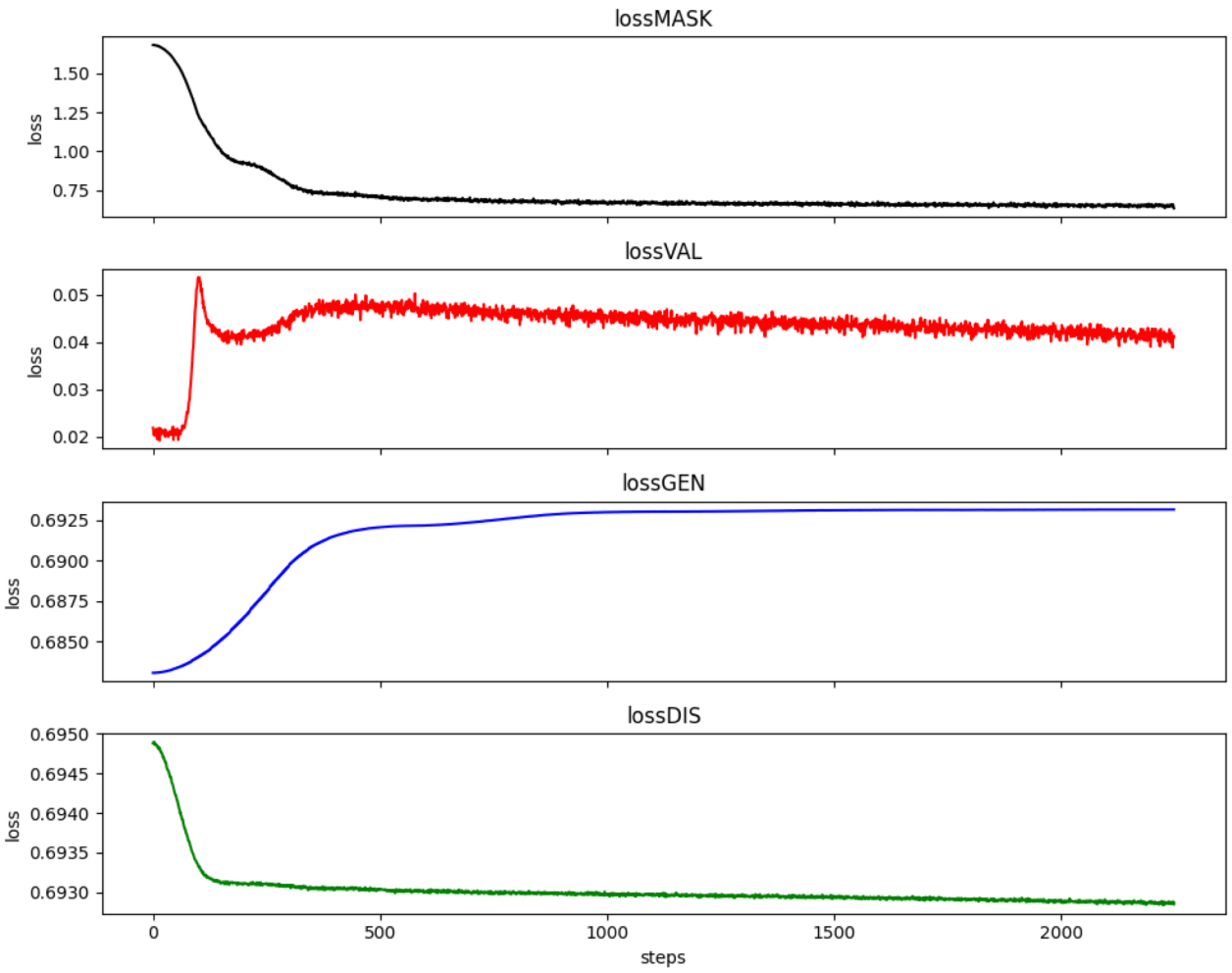


Hyperparameters in the ML training

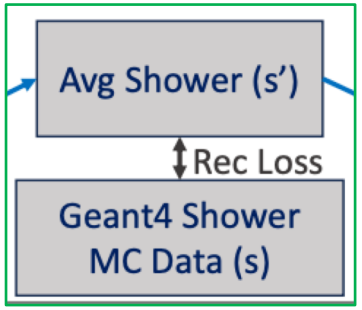
Test 4

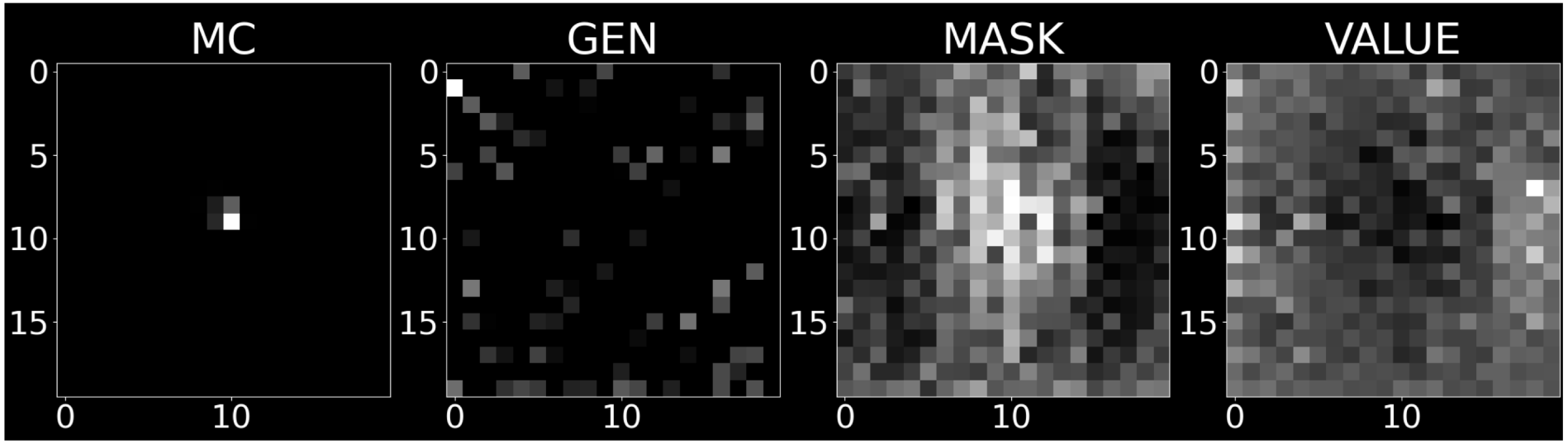


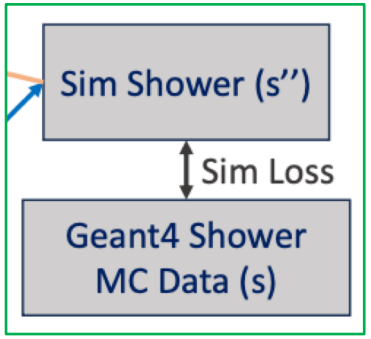
Test 6

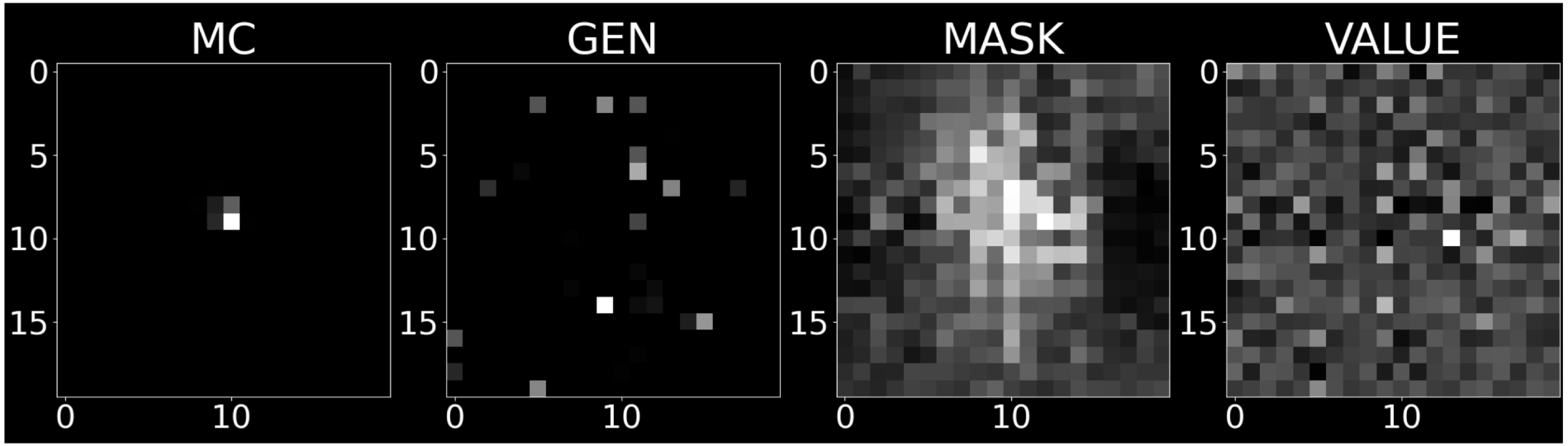


Test result from Test 9 model





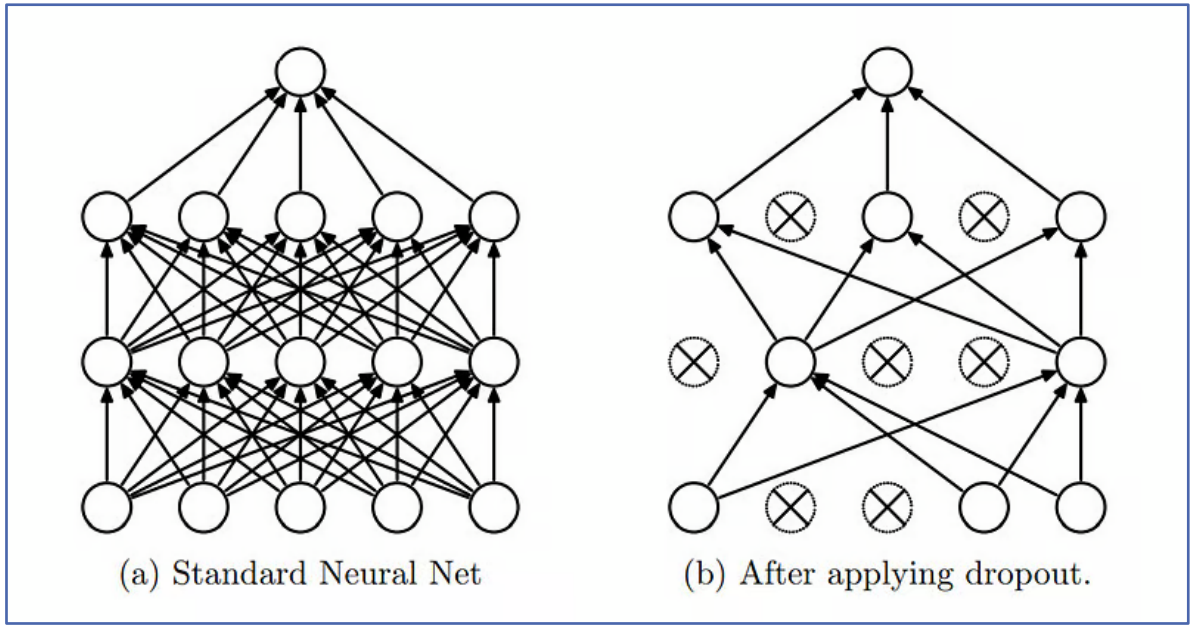




On-going items

- We can see that the model learn more from the background noise rather than the MC samples. We have to improve the masking by changing the weighting and bias.
- Also, as the training is optimizing the sum of lossMASK, lossVAL and lossGEN, we have to add weighting into lossMASK and lossVAL to affect the combination (in order to change the learning dynamics).
- Balancing the GAN (Generator vs Discriminator):
  - Training ratio back to 1:1
  - Add dropout layers in the CNN
  - Label ratio back to 0.7:0.3

```
###Add dropout
self.encoder = nn.Sequential(
        nn.Linear(1, hidden_dim*4),
        LinearBlock(hidden_dim*4, hidden_dim*9, 4),
        nn.Dropout(0.1),  # Add dropout
        nn.Unflatten(1, (hidden_dim, 3, 3)),
        nn.ConvTranspose2d(hidden_dim, hidden_dim,
            kernel_size=(3, 3), stride=(1, 1), padding=(0, 0)),
        Conv2dBlockH3W3(hidden_dim, hidden_dim*4),
        nn.Dropout(0.1),
        PixelShuffle2D(2, 2),
        Conv2dBlockH5W5(hidden_dim, hidden_dim*4),
        nn.Dropout(0.1),
        PixelShuffle2D(2, 2),
        Conv2dBlockH5W5(hidden_dim, hidden_dim)
        )
```



(a) Standard Neural Net (b) After applying dropout.

Summary

- We improved the GAN dynamics, but the model didn’t generate a MC-like images.
- We have to improve the structure of CNN to affect the learning dynamics, in order to train a model which can learn the MC-like events rather than background noise.