1

a

Sampling is the process of digitizing the coordinate values of an image.

b

Quantization is the process of digitizing the amplitude values of a sample.

c

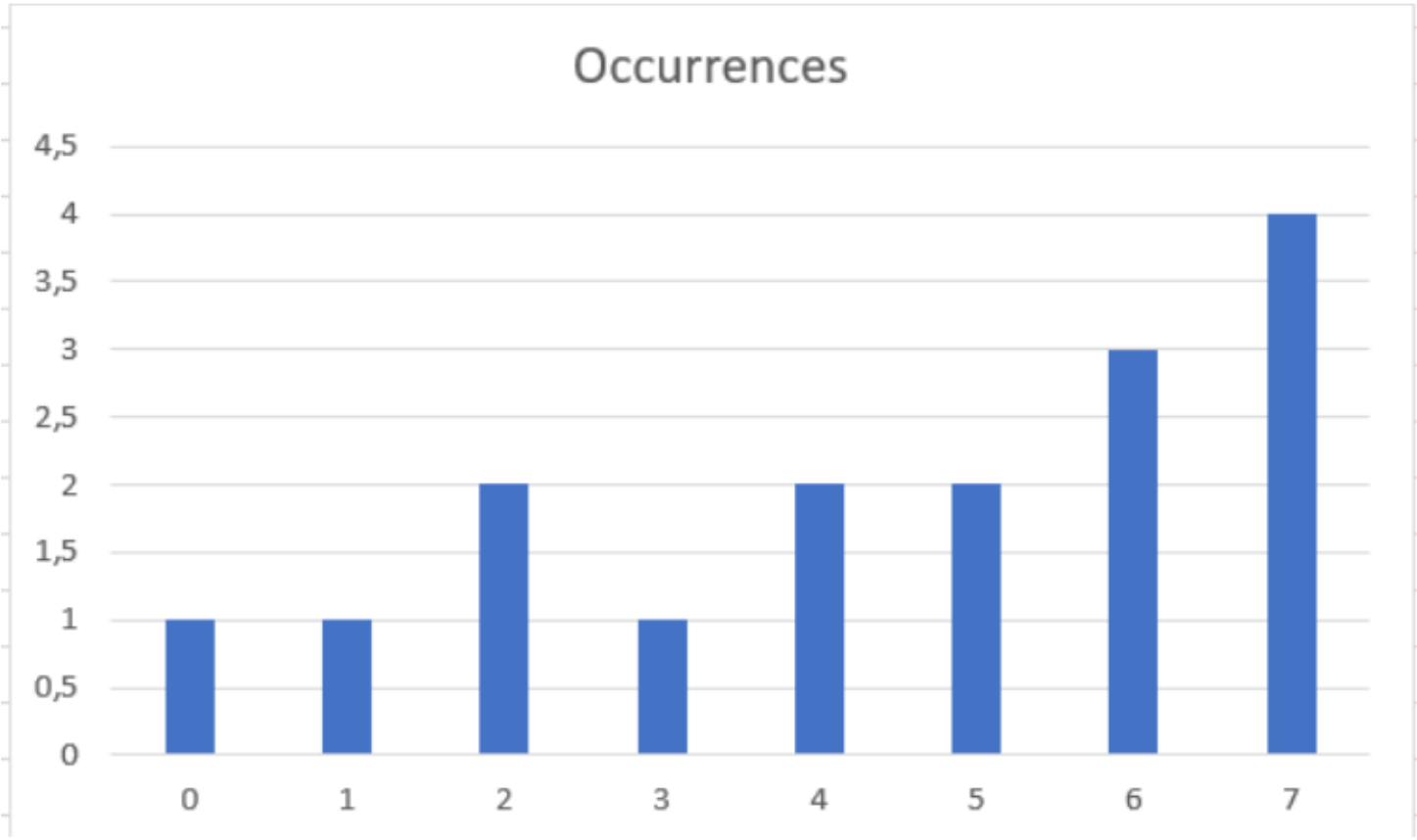
If the histogram is broad, then it has high contrast, while if it is narrow, then it has low contrast.

d

Finding original histogram by counting the number of occurrences of each intensity level:

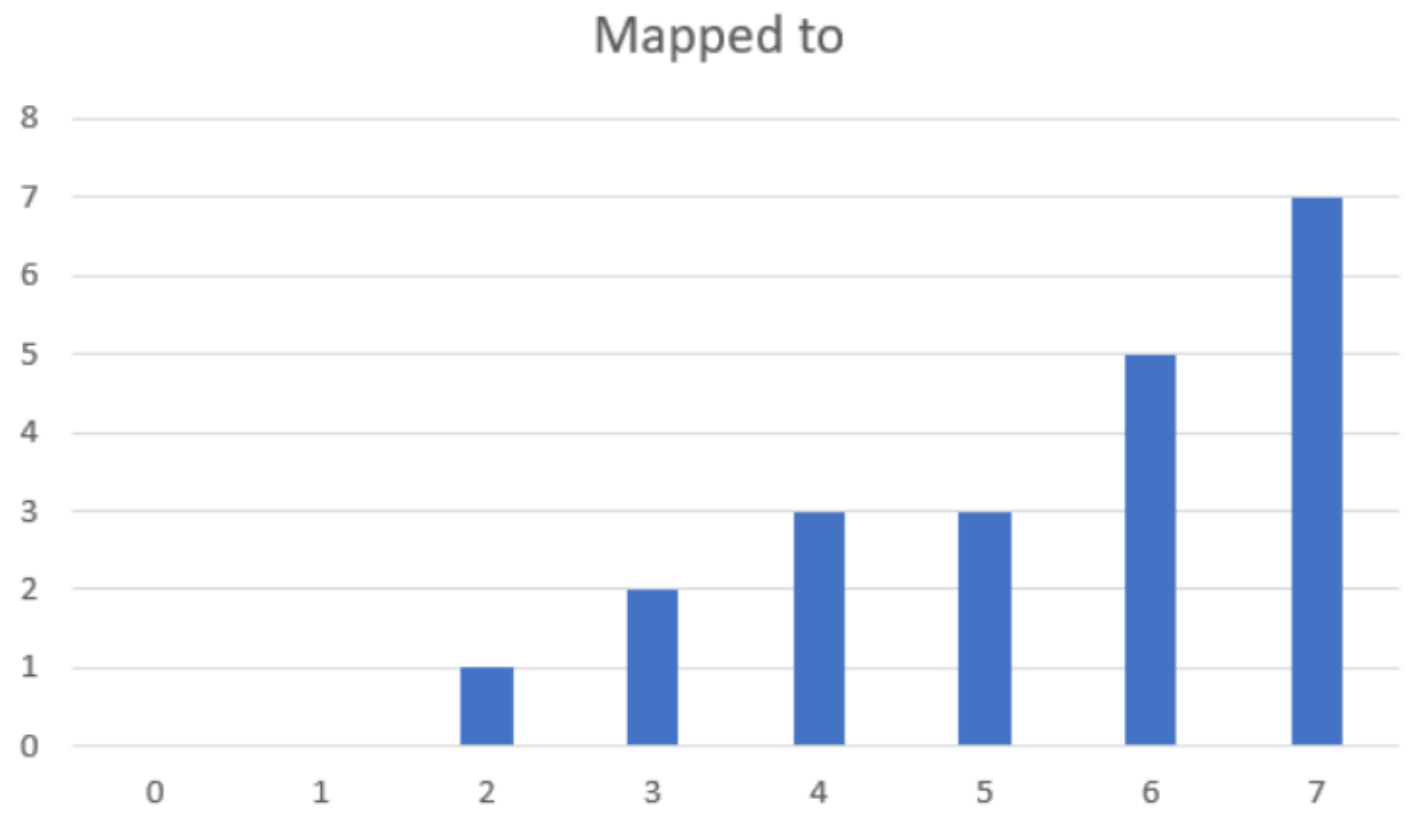
Intensity level	0	1	2	3	4	5	6	7
Occurrences	1	1	2	1	2	2	3	4

Histogram:


Occurrences
4,5
4
3,5
3
2,5
2
1,5
1
0,5
0
0
1
2
3
4
5
6
7

Intensity	Occurrences	PDF = Occurrences/sum	CDF = sk	sk*7	Histogram equalization level (floor)
0	1	1/16 = 0.0625	0.0625	0.4375	0
1	1	1/16 = 0.0625	0.125	0.875	0
2	2	2/16 = 0.125	0.25	1.75	1
3	1	1/16 = 0.0625	0.3125	2.1875	2
4	2	2/16 = 0.125	0.4375	3.0625	3
5	2	2/16 = 0.125	0.5625	3.9375	3
6	3	3/16 = 0.1875	0.75	5.25	5
7	4	4/16 = 0.25	1	7	7

Computed histogram:



New image (found by replacing the original intensity levels with the equalized levels):

7	5	3	5	3
3	3	7	7	0
0	7	5	2	5

e

The dynamic range will then be compressed, because each pixel value will be replaced by its logarithm.

f

Image:

7	6	5	6	4
5	4	7	7	0
1	7	6	3	6

Kernel:

1	0	-1
2	0	-2
1	0	-1

We are handling boundary conditions by returning zero in cases when the convolutional kernel goes outside the original image.

convolved (0,0) = 0*7* + *(-2*6) + (5*0)* + *(4*-1) = -16
convolved (1,0) = 0*6* + *(5*-2) + (7*-1) + (4*0)* + *(5*1) + (2*7) = 2

convolved (2,0) = 0*5* + *(6*-2) + (7*-1) + (7*0)* + *(4*1) + (2*6) = -3

convolved (3,0) = 0*6* + *(4*-2) + (0*-1) + (7*0)* + *(7*1) + (2*5) = 9

convolved (4,0) = 0*4* + *(0*0) + (1*7)* + *(6*2) = 19

convolved(0,1) = 0*5* + *(4*-2) + (7*-1) + (1*0)* + *(7*0) + (6*-1) = -21

convolved(1,1) = 7*1* + *(6*0) + (5*-1) + (5*2)* + *(4*0) + (7*-2) + (1*1)* + *(7*0) + (6*-1) = -7

convolved(2,1) = (6*1)* + *(5*0) + (6*-1) + (4*2)* + *(7*0) + (7*-2) + (7*1)* + *(6*0) + (3*-1) = -2

convolved(3,1) = (5*1)* + *(6*0) + (4*-1) + (7*2)* + *(7*0) + (0*-2) + (6*1)* + *(3*0) + (6*-1) = 15

convolved(4,1) = (0*0)* + *(6*0) + (3*1)* + *(7*2) + (6*1)* + *(4*0) = 23

convolved(0,2) = (1*0)* + *(7*-2) + (5*0)* + *(4*-1) = -18

convolved(1,2) = (5*1)* + *(4*0) + (7*-1) + (1*2)* + *(7*0) + (6*-1) = -6

convolved(2,2) = (4*1)* + *(7*0) + (7*-1) + (7*2)* + *(6*0) + (3*-1) = 8

convolved(3,2) = (7*1)* + (*7*0) + (0*-1) + (6*2)* + (*0*3) + (6*-1) = 13

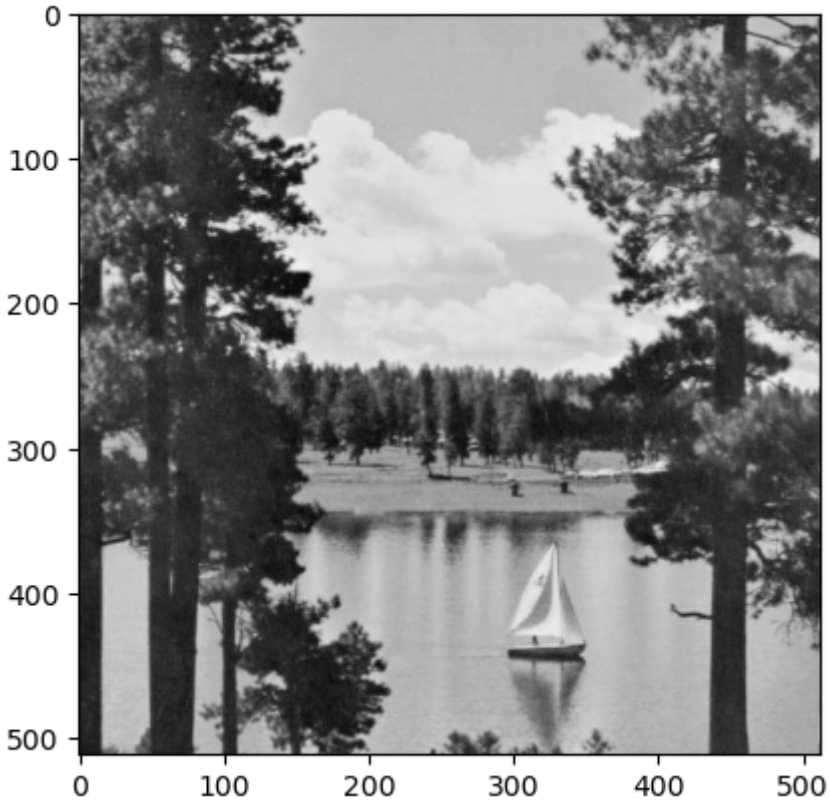
convolved(4,2) = (6*0)* + (*3*2) + (7*1)* + (*0*0) = 13

Convolved image:

-16	2	-3	9	19
-21	-7	-2	15	23
-18	-6	8	13	13

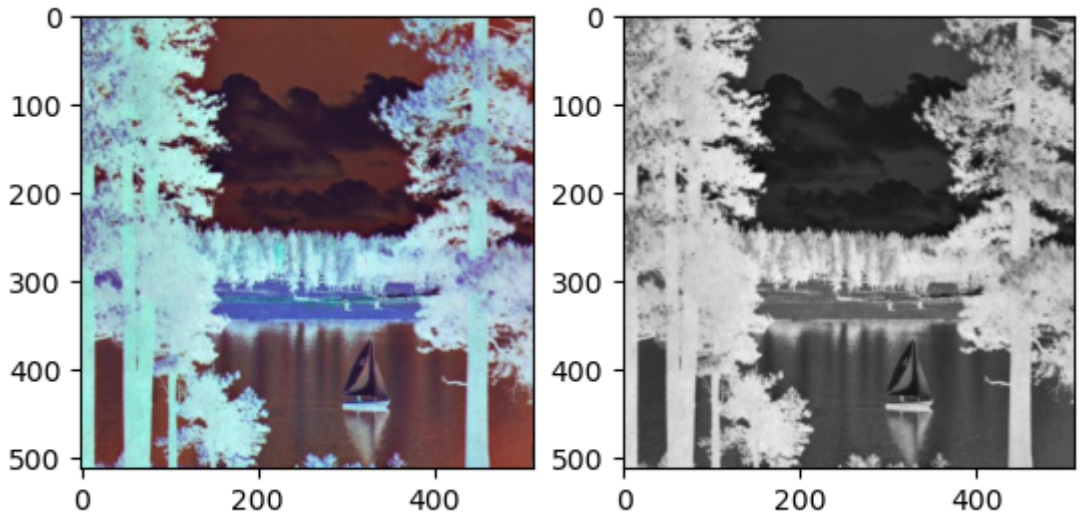
2

a



b

Normal image inverted and grey image inverted:

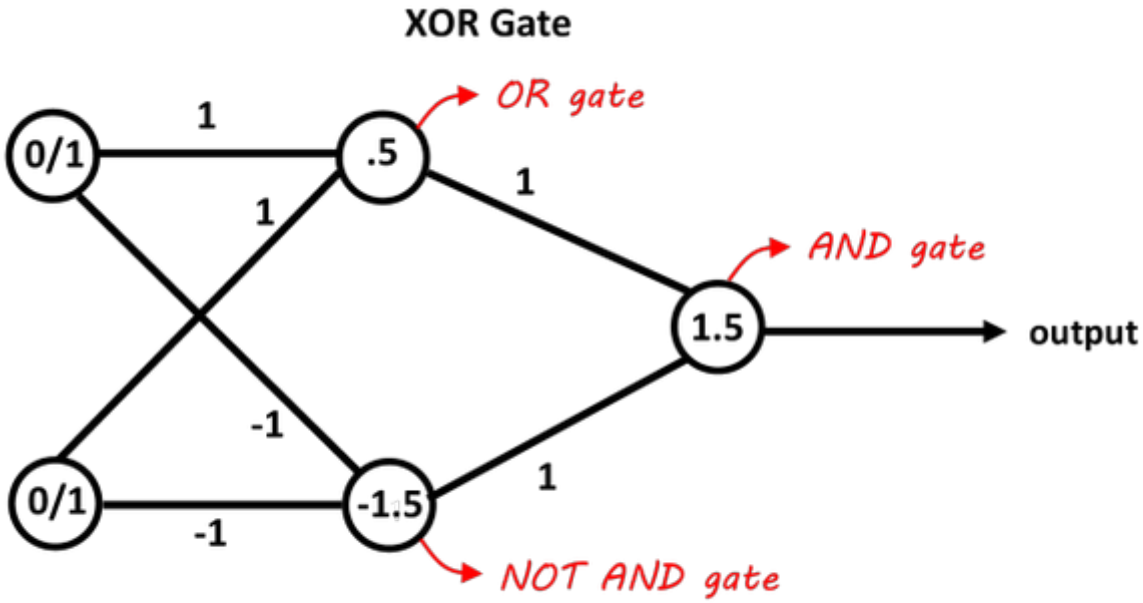


3

a

NOR, NAND, XOR

The image shows an example of an AND and XOR gate that require more than one layer:



b

Hyperparameters for a neural network are constants that can be applied which determines the structure of the network (e.g. dimentionality and size) as well as how it is trained (e.g. speed and evaluation methods).

- Learning rate is the degree of which the algorithm will correct itself according to the error. If the learning rate is too high, it may not find the optimal weights and biases by bouncing back and forth too much. If it is too low, it may take a long time to get there
- Number of epochs is the number of iterations where the entire training data is shown and trained on. In general, the higher this number is, the better the model gets - until usually a platou occurs. This can be measured by checking the accuracy of the model in between each epoch. Triaining can be terminated either by looking at when this graph starts to flatten out, or set it to a specific epoch number.

c

Softmax is one of many activation functions that wrap around the weights and biases calculated in the network. This is used so that the final output of each neuron is set to a probability between 1 and 0. For example, if the task of the network is to determine which digit is shown, a single neuron (representing a digit) from the final layer with the highest probability would be picked.

d

Forward pass:
$a1\_1 = x1*w1 = 1$

$a1\_2 = x2*w2 = 0$

$a1\_3 = x3*w3 = 1$

$a1\_4 = x4*w4 = -4$

$a2\_1 = a1\_1 + a1\_2 + b1 = 1+0+1 = 2$

$a2\_2 = a1\_3 + a1\_4 + b2 = 1-4-1 = -4$

$\hat{y} = max(a2\_1, a2\_2) = 2$

$\hat{y}' = 1$

Error:
$E_{tot} = C(\hat{y}', \hat{y}) = 0.5(\hat{y}'-\hat{y})^2 = 0.5(1-2)^2 = 0.5$

Derivatives for each weight and bias using chain rule:

$dC/dw1 = E_{tot} * a1\_1 * w1 = 0.5 * 1 * -1 = -0.5$

$dC/dw2 = E_{tot} * a1\_2 * w2 = 0.5 * 0 * 1 = 0$

$dC/dw3 = E_{tot} * a1\_3 * w3 = 0.5 * -1 * 1 = -0.5$

$dC/dw4 = E_{tot} * a1\_4 * w4 = 0.5 * -4 * -2 = 4$

$dC/db1 = E_{tot} * b1 = 0.5 * 1 = 0.5$

$dC/db2 = E_{tot} * b2 = 0.5 * -1 = -0.5$

e

New weights and biases:

$w1\_2 = w1 - dC/dw1 * a = -1 - -0.5 * 0.1 = -0.95$

$w2\_2 = w2 - dC/dw2 * a = 1 - 0 * 0.1 = 1$

$w3\_2 = w3 - dC/dw3 * a = -1 - -0.5 * 0.1 = -0.95$

$w4\_2 = w4 - dC/dw4 * a = -2 - 4 * 0.1 = -2.4$

$b1\_2 = b1 - dC/db1 = 1 - 0.5 = 0.5$

$b2\_2 = b2 - dC/db2 = -1 - -0.5 = -1.5$

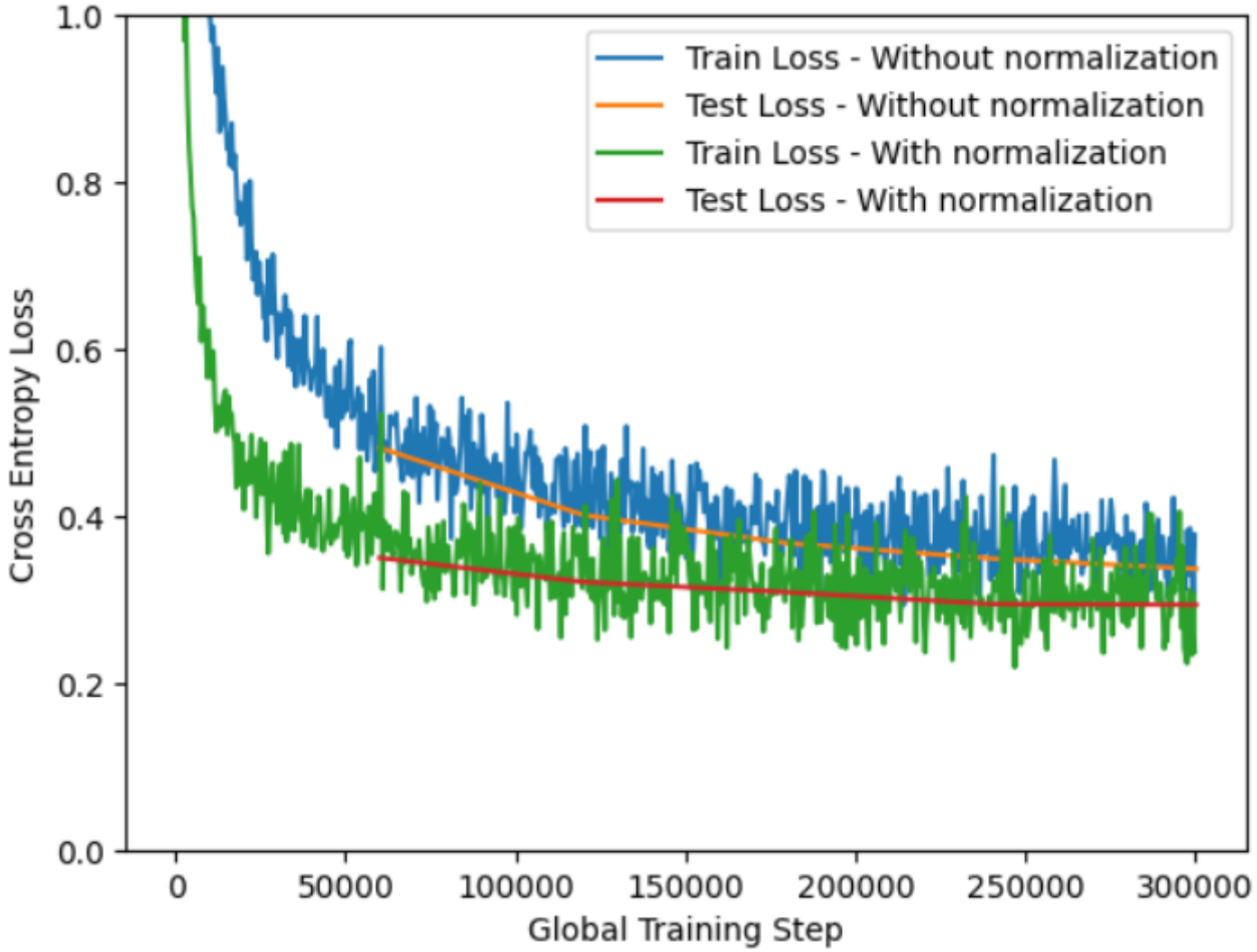
4

a

Accuracy: 0.919

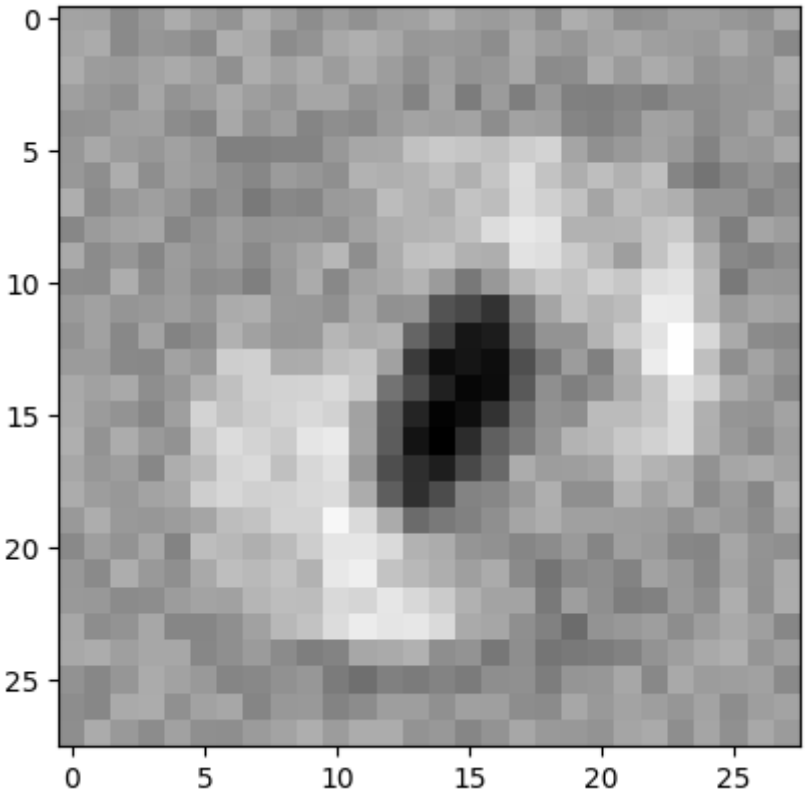
Average cross entropy loss: 0.288

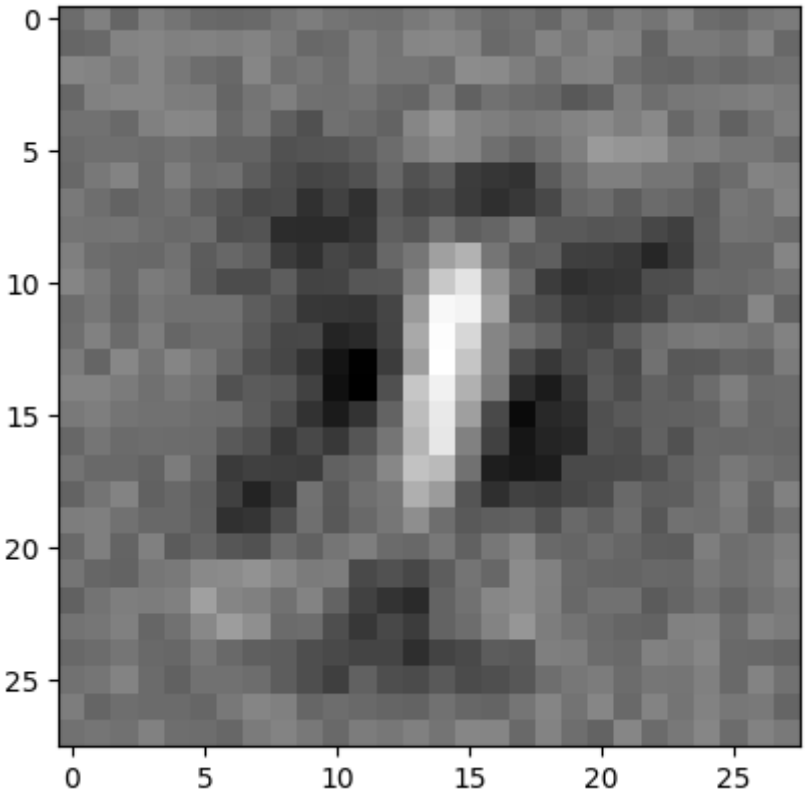
As the graph shows, the loss is greater without normalization than with normalization:

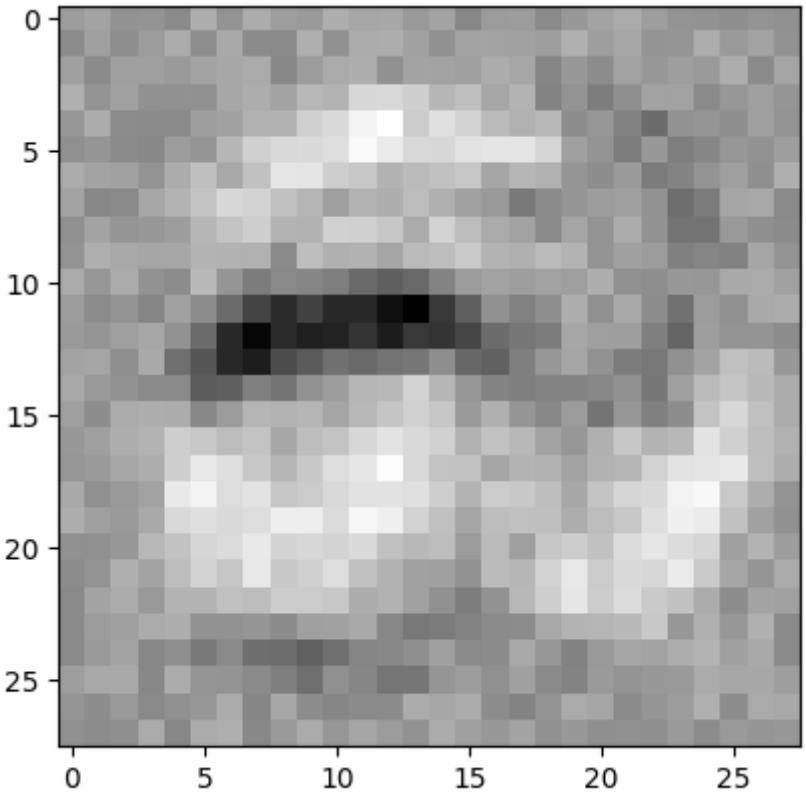


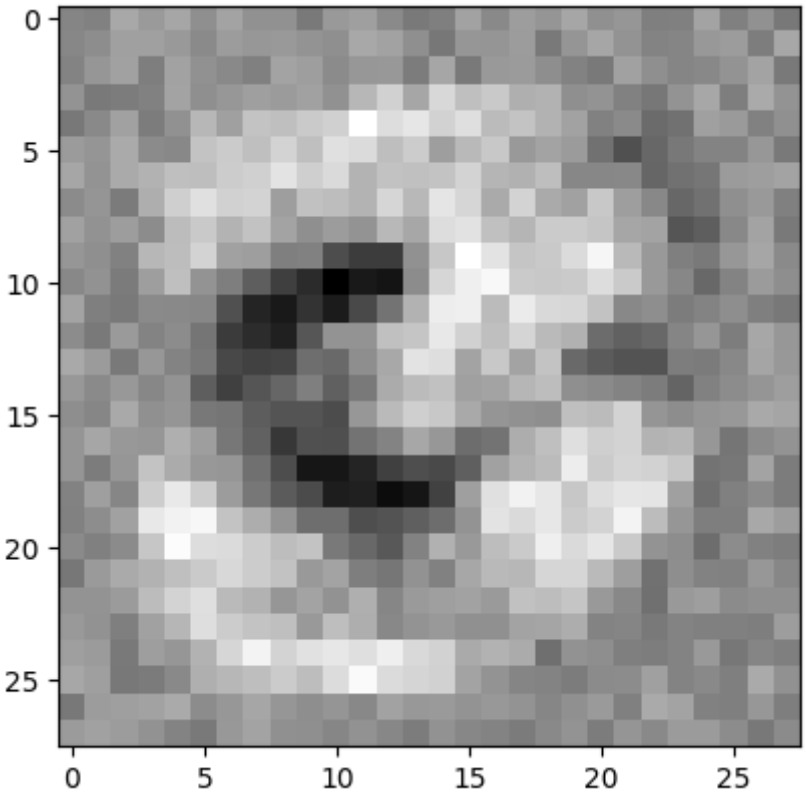
b

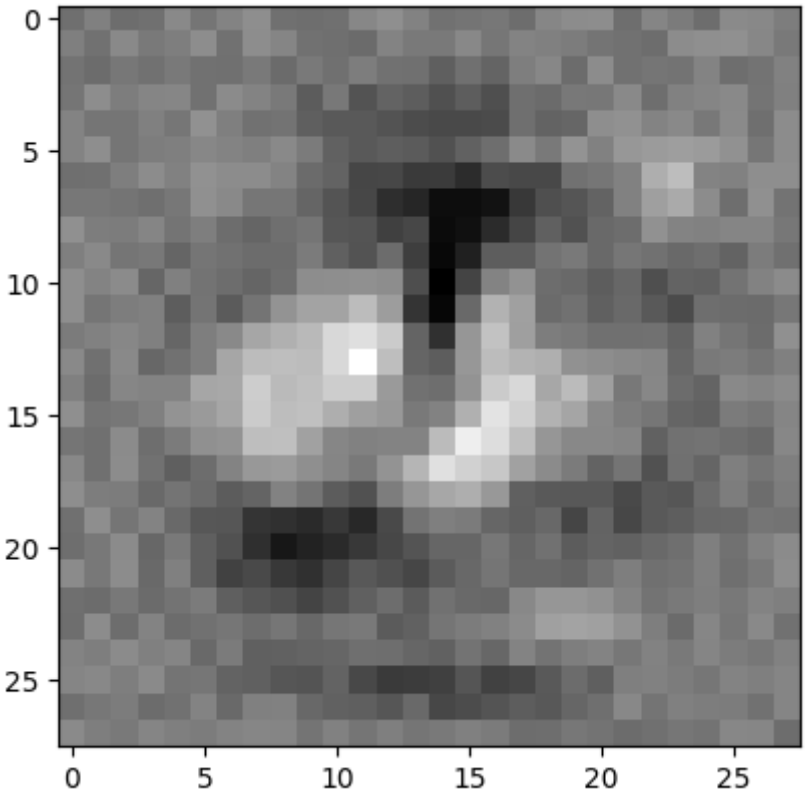
We observe that when we plot the learned weights for each class as a greyscale image, the image resebles the image of each class. This is because the model will begin by making random guesses, and as it learns, the weights will be updated to represent each number.

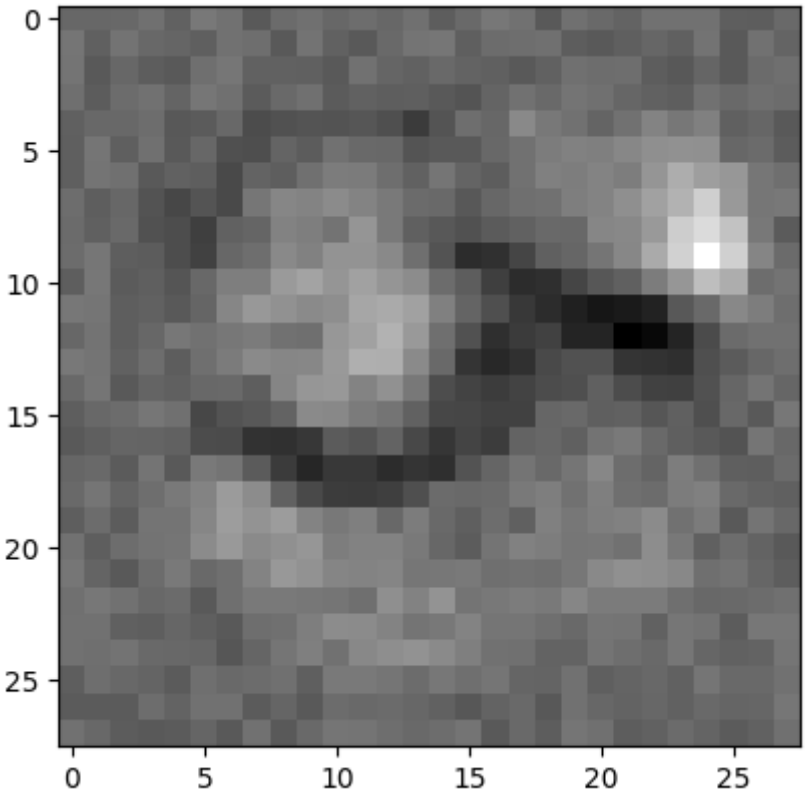

0
5
10
15
20
25
0
5
10
15
20
25

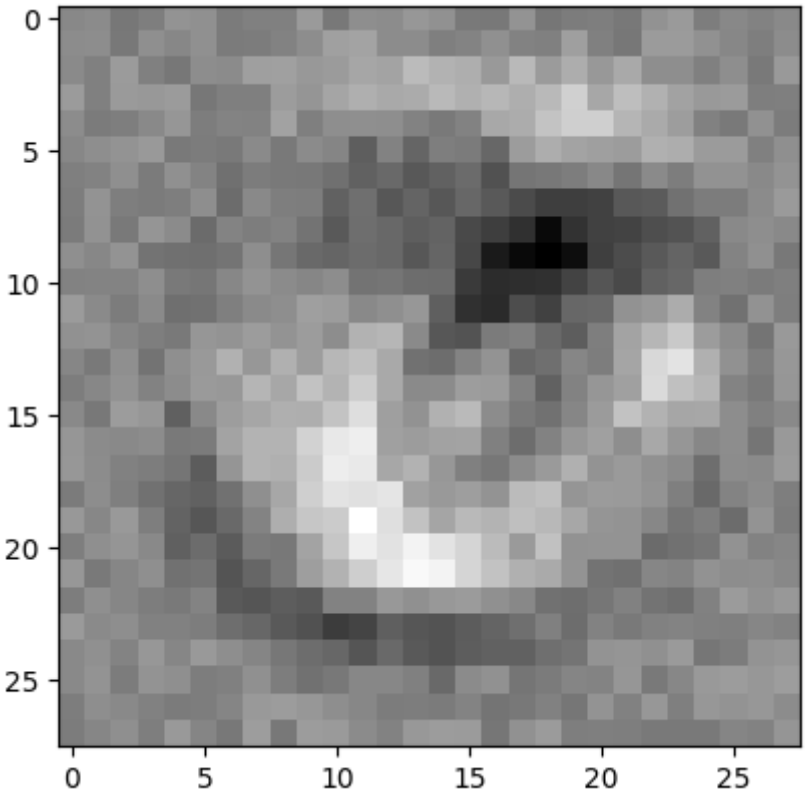

0
5
10
15
20
25
0
5
10
15
20
25

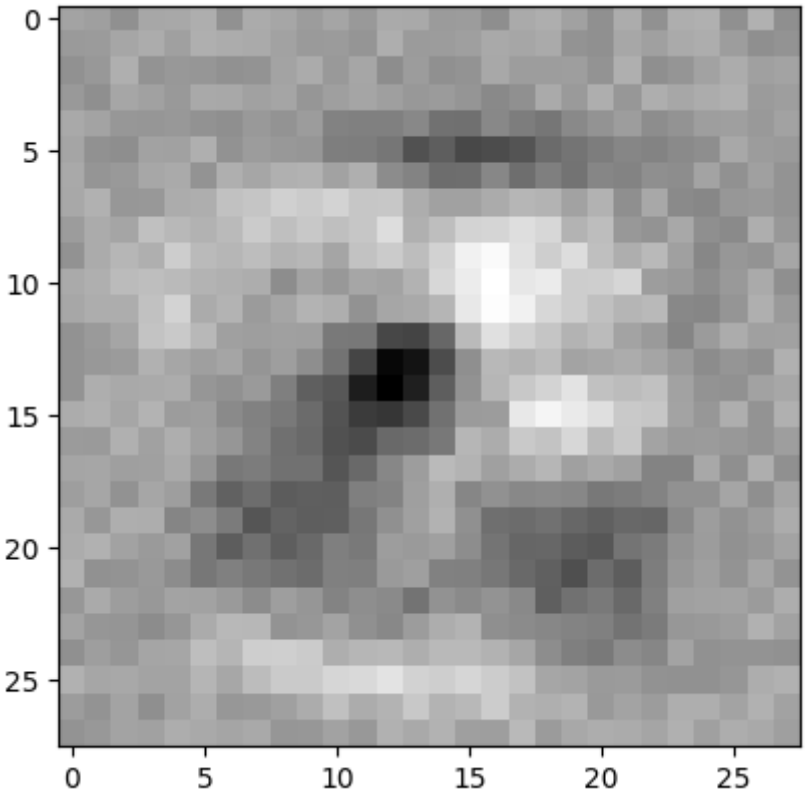

0
5
10
15
20
25
0
5
10
15
20
25

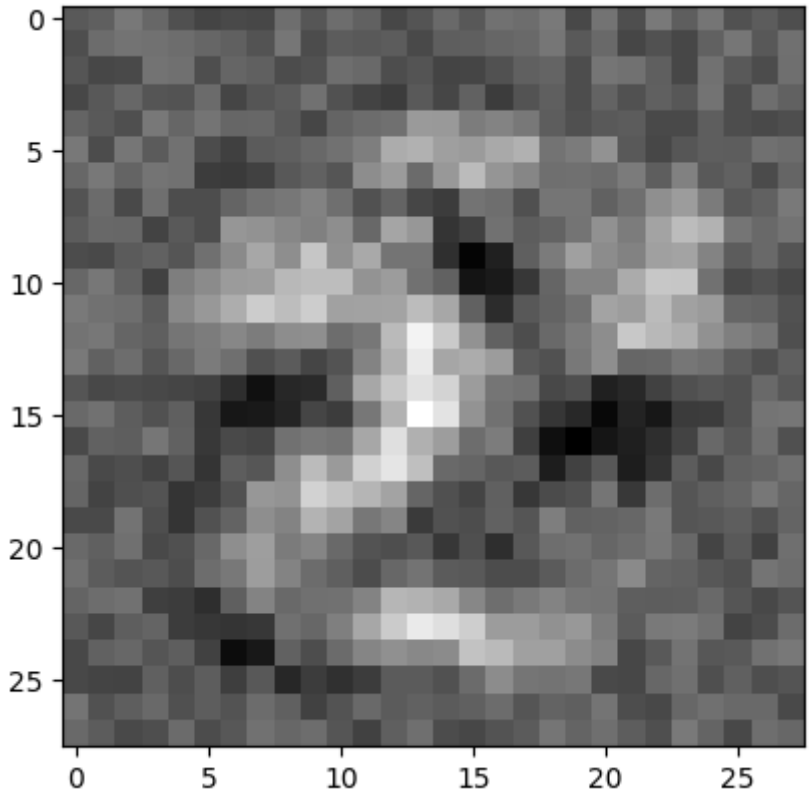

0
5
10
15
20
25
0
5
10
15
20
25

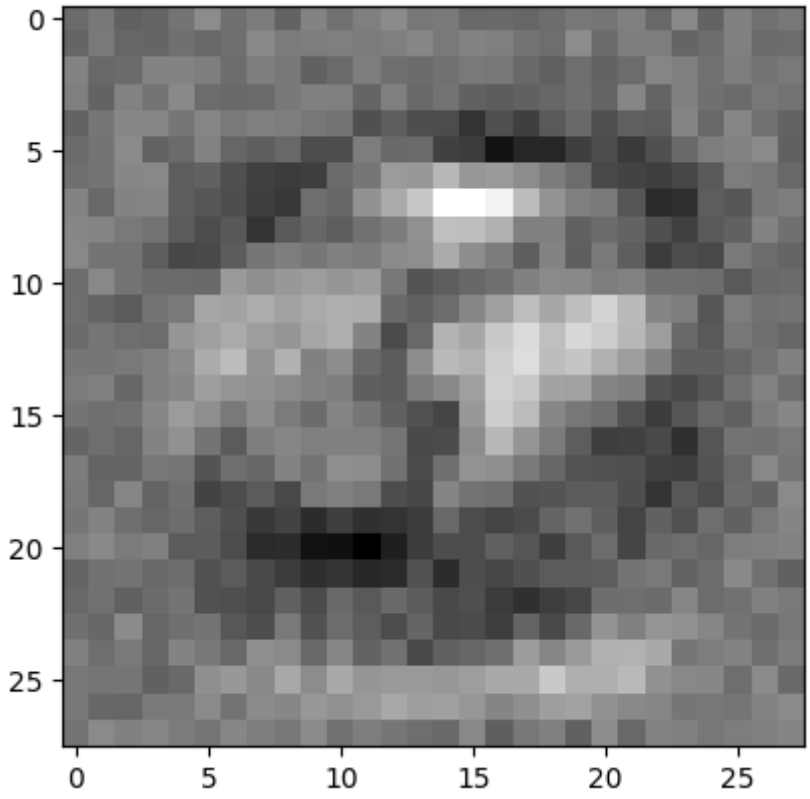

0
5
10
15
20
25
0
5
10
15
20
25


0
5
10
15
20
25
0
5
10
15
20
25


0
5
10
15
20
25
0
5
10
15
20
25


0
5
10
15
20
25
0
5
10
15
20
25

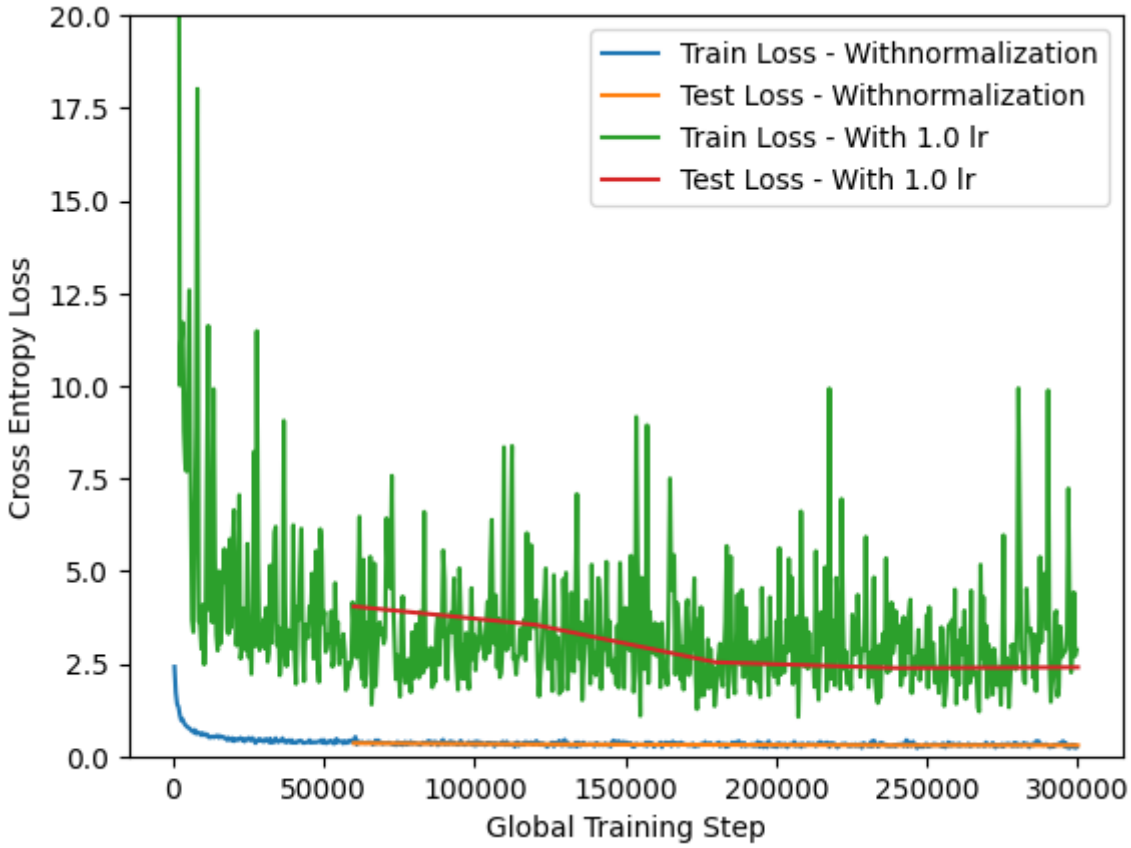




c

Accuracy: 0.902

Average cross entropy loss: 2.21



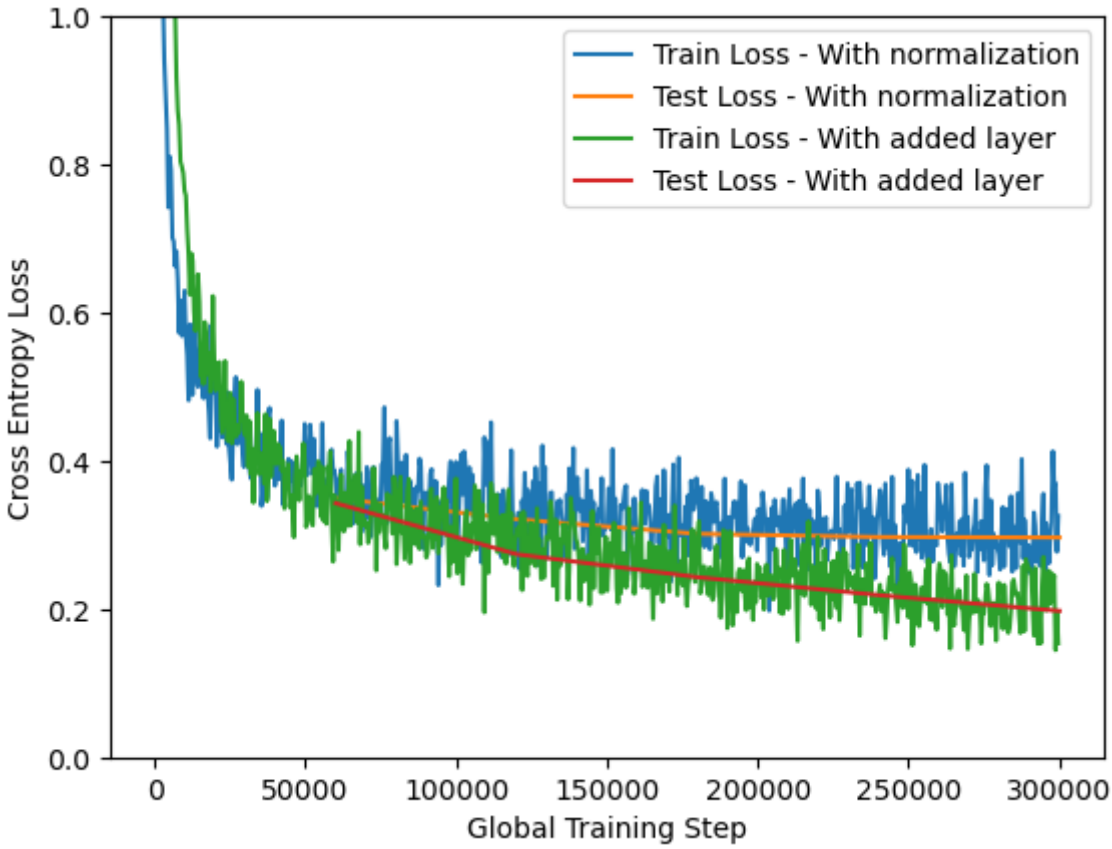
The accuracy decreases since the learning rate is too high. This causes the model to update each weight too much, causing the weights to swing back and forth without converging.

d

Accuracy: 0.9387

Average cross entropy loss: 0.204

We observe an improvement in both accuracy and loss:


Train Loss - With normalization
Test Loss - With normalization
Train Loss - With added layer
Test Loss - With added layer
Cross Entropy Loss
Global Training Step
1.0
0.8
0.6
0.4
0.2
0.0
0
50000
100000
150000
200000
250000
300000