TDT4195 Image processing assignment 1 report

Nils Reed

October 2021

Task 1.a

Sampling something (e.g. a signal) is measuring its value at a certain point in time.

Task 1.b

Quantization is the effect caused by only having a finite amount of discrete values when describing an analog signal, such that the analog values get rounded to the nearest representable value.

Task 1.c

A high contrast image will have clear peaks in the lower and higher end of the histogram, and small/zero values in between/in the middle.

Task 1 d)

Perform histogram equalization on this image:

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

(a) A 3×5 image.

Finding $p_r(r_k)$:

r_k	n_k	$p_r(r_k)$	$F_r(r_k)$
0	1	$1/15$	$1/15$
1	1	$1/15$	$2/15$
2	0	0	$2/15$
3	1	$1/15$	$3/15$
4	2	$2/15$	$5/15$
5	2	$2/15$	$7/15$
6	4	$4/15$	$11/15$
7	4	$4/15$	$15/15$

Using $p_r(r_k) = \frac{n_k}{M \cdot N}$, $F_r(r_k) = \sum_{j=0}^{k} p_r(r_j)$

Values from $0 - 7$

$\Rightarrow T(r) = 7 \cdot F_r(r)$

But, we should round down, thus the final transformation is

$$T(r) = \lfloor 7 \cdot F_r(r) \rfloor$$

Gives :

r	$T(r)$
0	$\lfloor 7 \cdot 1/15 \rfloor = \lfloor \frac{7}{15} \rfloor = 0$
1	$\lfloor 7 \cdot \frac{2}{15} \rfloor = \lfloor \frac{14}{15} \rfloor = 0$
2	$\lfloor 7 \cdot \frac{2}{15} \rfloor = \lfloor \frac{14}{15} \rfloor = 0$
3	$\lfloor 7 \cdot \frac{3}{15} \rfloor = \lfloor \frac{21}{15} \rfloor = 1$
4	$\lfloor 7 \cdot \frac{5}{15} \rfloor = \lfloor \frac{35}{15} \rfloor = 2$
5	$\lfloor 7 \cdot \frac{7}{15} \rfloor = \lfloor \frac{49}{15} \rfloor = 3$
6	$\lfloor 7 \cdot \frac{11}{15} \rfloor = \lfloor \frac{77}{15} \rfloor = 5$
7	$\lfloor 7 \cdot \frac{15}{15} \rfloor = \lfloor 7 \rfloor = 7$

Which gives the picture:

5	7	3	2	5
2	3	7	0	7
7	0	5	5	1

Task 1.e

The log transform squeezes the low end of the histogram and widens the high end. The result of this is that the dynamic range of an image is compressed. This is especially relevant if the image has a large variance in pixel intensities, because it means that we have pixels far from the mean of the intensities, and thus will see squeezing in the lower end and stretching in the higher end, leading to compression of the dynamic range.

Task 1f)

convolve this image:

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

(a) A 3 × 5 image.

With this kernel:

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

(b) A 3 × 3 Sobel kernel.

Trick:

easier to do correlation instead,
just rotate kernel 180°

Gives:

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

as the new kernel

handle boundary conditions by using
0-padding

Giving the image to be correlated:

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

This gives, for the result matrix:

$$A_{11} = (-1)\cdot 0 + 0\cdot 0 + 1\cdot 0 + (-2)\cdot 0 + 0\cdot 6 + 2\cdot 7 + (-1)\cdot 0 + 0\cdot 4 + 1\cdot 5$$

$$= 14 + 5 = 19$$

From here on, I will not write those terms that are multiplied with 0.

$$A_{21} = 1\cdot 7 + 2\cdot 5 + 1\cdot 1 = 7 + 5 + 2 = 14$$

$$A_{31} = 1\cdot 5 + 2\cdot 1 = 5 + 2 = 7$$

$$A_{12} = (-2)\cdot 6 + (-1)\cdot 4 + 2\cdot 5 + 1\cdot 7 = -12 - 4 + 10 + 7 = 1$$

$$A_{22} = (-1)\cdot 6 + (-2)\cdot 4 + (-1)\cdot 7 + 1\cdot 5 + 2\cdot 7 + 1\cdot 6$$

$$= -6 - 8 - 7 + 5 + 14 + 6 = 4$$

$$A_{32} = (-1)\cdot 4 + (-2)\cdot 7 + 1\cdot 7 + 2\cdot 6 = -4 - 14 + 7 + 12 = 1$$

$$A_{13} = (-2)\cdot 7 + (-1)\cdot 5 + 2\cdot 4 = -14 - 5 + 8 = -11$$

$$A_{23} = (-1)\cdot 7 + (-2)\cdot 5 + (-1)\cdot 1 + 1\cdot 5 + 1\cdot 6$$

$$= -7 - 10 - 1 + 5 + 6 = -7$$

$$A_{33} = (-1)\cdot 5 + (-2)\cdot 1 + 2\cdot 6 = -5 - 2 + 12 = 5$$

$A_{14} = (-2)\cdot 5 + (-1)\cdot 7 + 2\cdot 6 + 1\cdot 7$

$= -10 - 7 + 12 + 7 = 2$

$A_{24} = (-1)\cdot 5 + (-2)\cdot 7 + (-1)\cdot 6 + 1\cdot 6 + 2\cdot 7 + 1\cdot 3$

$= -5 - 14 - 6 + 6 + 14 + 3 = -2$

$A_{34} = (-1)\cdot 7 + (-2)\cdot 6 + 1\cdot 7 + 2\cdot 3$

$= -7 - 12 + 7 + 6 = -6$

$A_{15} = (-2)\cdot 4 = -8$

$A_{25} = (-1)\cdot 4 + (-1)\cdot 6 = -4 - 6 = -10$

$A_{35} = (-2)\cdot 6 = -12$

Thus, the result of the convolution is:

$$\begin{bmatrix} 19 & 1 & -11 & 2 & -8 \\ 14 & 4 & -7 & -2 & -10 \\ 7 & 1 & 5 & -6 & -12 \end{bmatrix}$$

Task 2.a

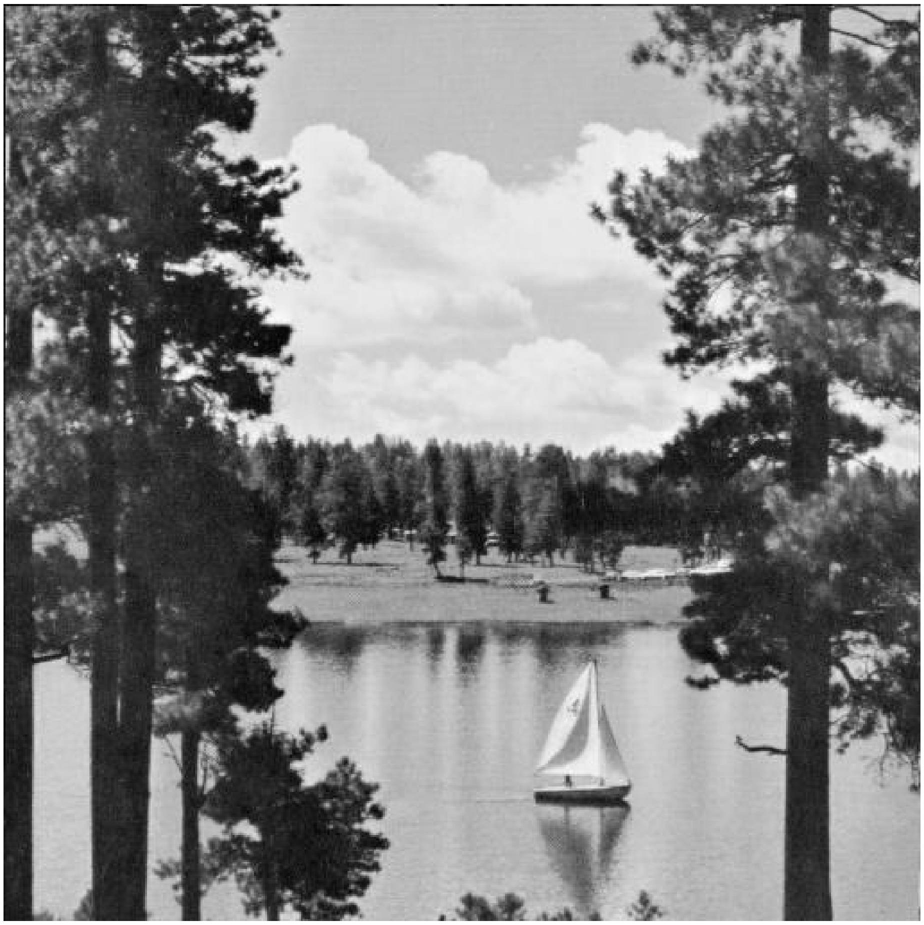


Figure 1: Greyscale version of lake.jpg

Task 2.b

Figure 2: Inverted version of greyscale version of lake.jpg

Task 2.c

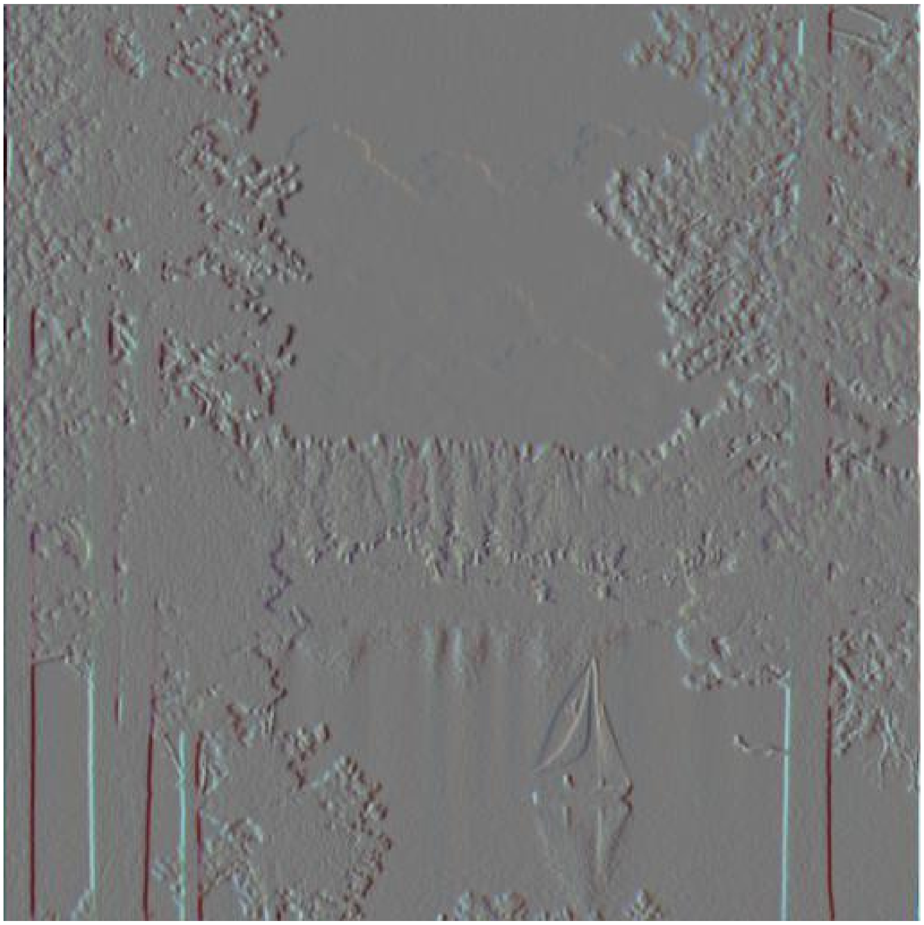


Figure 3: Image after convolution with Sobel kernel

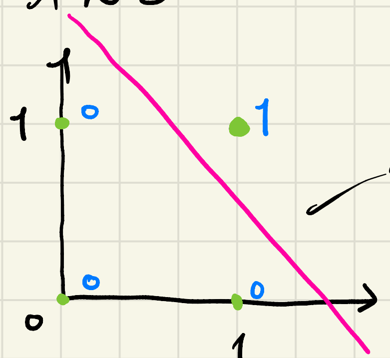
Figure 4: Image after convolution with smoothing kernel

Task 3a)

Single-layer NNs are, as mentioned in the task, linear functions.

Thus, they can only solve linearly separable problems, i.e. problems where the types of input points can be separated by a linear function

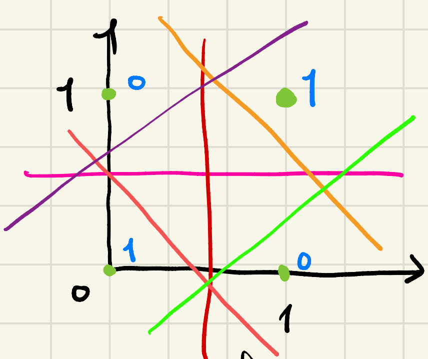
e.g. AND:



separated by straight line → linearly separable

Blue numbers are the output of the function.

XOR on the other hand:



We see that no single one of the above lines is enough to separate the input types based on their output

→ Not linearly separable, and thus XOR cannot be represented by a single-layer N.N.!

Task 3.b

A hyperparameter is a parameter of a neural network that is set only once per training process of the network.

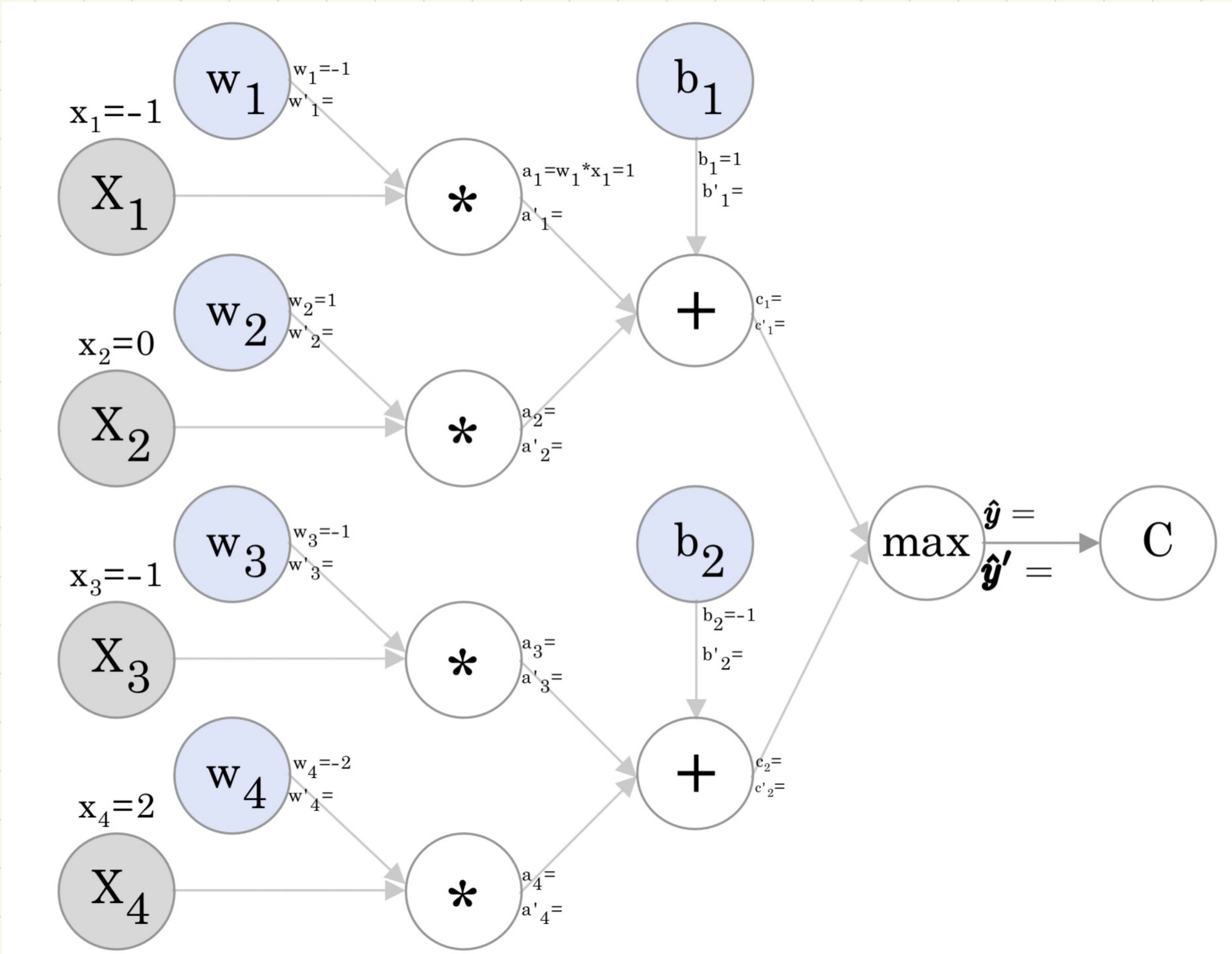
Examples of hyperparameters are the learning rate α and the size of a mini-batch if mini-batches are used for the gradient descent.

Task 3.c

The softmax function is used in the last layer of a neural network because the output gives confidence levels that sum to 1, similar to a probability distribution.

In addition to this, the outputs are all positive at the end, which is a nice property, because a negative confidence level is not very intuitive.

Task 3.d)



Target value: $y = 1$

Forward pass:

$$c_1 = a_1 + a_2 + b_1$$

$$= w_1 \cdot x_1 + w_2 x_2 + b_1$$

$$= (-1)\cdot(-1) + 0\cdot(1) + 1$$

$$= 1 + 1 = \underline{2}$$

$$c_2 = a_3 + a_4 + b_2$$

$$= w_3 x_3 + w_4 x_4 + b_2$$

$$= (-1)\cdot(-1) + (-2)\cdot 2 - 1$$

$$= 1 - 4 - 1 = \underline{-4}$$

$$\hat{y} = \max\{c_1, c_2\}$$

$$= \max\{2, -4\} = \underline{\underline{2}}$$

Cost function:

$$C(y, \hat{y}) = \frac{1}{2}(y - \hat{y})^2$$

$$C(1, 2) = \frac{1}{2}(1-2)^2$$

$$= \frac{1}{2}(-1)^2 = \underline{\underline{\frac{1}{2}}}$$

Backward pass:

$$\frac{\partial C}{\partial \hat{y}} = \frac{\partial}{\partial \hat{y}} \frac{1}{2}(y-\hat{y})^2 = (-1)\cdot 2 \cdot \frac{1}{2}(y-\hat{y})$$

$$= -(y-\hat{y}) = \hat{y} - y = 2 - 1 = \underline{1}$$

$$\frac{\partial C}{\partial c_1} = \frac{\partial C}{\partial \hat{y}} \cdot \frac{\partial \hat{y}}{\partial c_1} = \frac{\partial C}{\partial \hat{y}} \cdot \frac{\partial}{\partial c_1} c_1 \quad \text{— since } \hat{y} = \max\{c_1, c_2\} \overset{c_1 > c_2}{=} c_1$$

$$= \frac{\partial C}{\partial \hat{y}} \cdot 1 = \frac{\partial C}{\partial \hat{y}} = \underline{1}$$

$$\frac{\partial C}{\partial c_2} = \frac{\partial C}{\partial \hat{y}} \cdot \frac{\partial \hat{y}}{\partial c_2} = \frac{\partial C}{\partial \hat{y}} \cdot \frac{\partial}{\partial c_2} c_1 = \frac{\partial C}{\partial \hat{y}} \cdot 0 = \underline{0}$$

$$\frac{\partial C}{\partial b_1} = \frac{\partial C}{\partial c_1} \frac{\partial c_1}{\partial b_1} = \frac{\partial C}{\partial c_1} \cdot \frac{\partial}{\partial b_1}(w_1 x_1 + w_2 x_2 + b_1)$$

$$= \frac{\partial C}{\partial c_1} \cdot 1 = \underline{\underline{1}}$$

$$\frac{\partial C}{\partial w_1} = \frac{\partial C}{\partial c_1} \cdot \frac{\partial c_1}{\partial w_1} = \frac{\partial C}{\partial c_1} \frac{\partial}{\partial w_1}(w_1 x_1 + w_2 x_2 + b_1)$$

$$= \frac{\partial C}{\partial c_1} \cdot x_1 = 1 \cdot (-1) = \underline{\underline{-1}}$$

$$\frac{\partial C}{\partial w_2} = \frac{\partial C}{\partial c_1} \cdot \frac{\partial c_1}{\partial w_2} = \frac{\partial C}{\partial c_1} \cdot \frac{\partial}{\partial w_2}(w_1 x_1 + w_2 x_2 + b_1)$$

$$= \frac{\partial C}{\partial c_1} \cdot x_2 = 1 \cdot 0 = 0$$

$$\frac{\partial C}{\partial b_2} = \frac{\partial C}{\partial C_2} \cdot \frac{\partial C_2}{\partial b_2} = 0 \cdot \frac{\partial C_2}{\partial b_2} = \underline{\underline{0}}$$

$$\frac{\partial C}{\partial w_3} = \frac{\partial C}{\partial C_2} \cdot \frac{\partial C_2}{\partial w_3} = 0 \cdot \frac{\partial C_2}{\partial w_3} = \underline{\underline{0}}$$

$$\frac{\partial C}{\partial w_4} = \frac{\partial C}{\partial C_2} \cdot \frac{\partial C_2}{\partial w_4} = 0 \cdot \frac{\partial C_2}{\partial w_4} = \underline{\underline{0}}$$

Task 3.e)

Updating the weights:

General formula

$$\theta^{n+1} = \Theta^{n} - \alpha \frac{\partial C}{\partial \Theta^{n}}$$

(Where the superscript does not signify an exponent, but rather an iteration number)

Thus:

$$w_1^2 = w_1^1 - \alpha \frac{\partial C}{\partial w_1^1}$$

$$= -1 - 0.1 \cdot (-1)$$

$$= -1 + 0.1 = \underline{\underline{-0.9}}$$

$$w_3^2 = w_3^1 - \alpha \frac{\partial C}{\partial w_3^1}$$

$$= -1 - 0.1 \cdot 0 = \underline{\underline{-1}}$$

$$b_1^2 = b_1^1 - \alpha \frac{\partial C}{\partial b_1^1}$$

$$= 1 - 0.1 \cdot 1 = 1 - 0.1 = \underline{\underline{0.9}}$$

Task 4.a

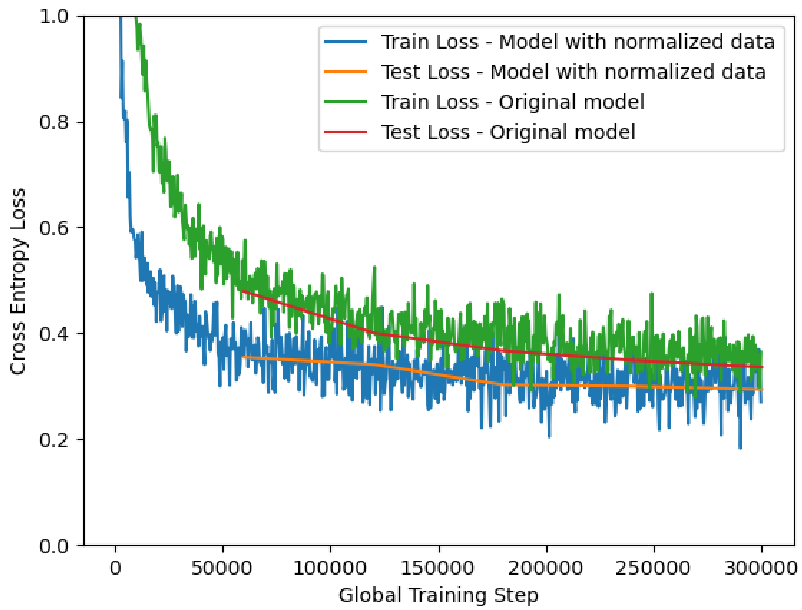


Figure 5: Training and validation loss for training of both networks in task 4a

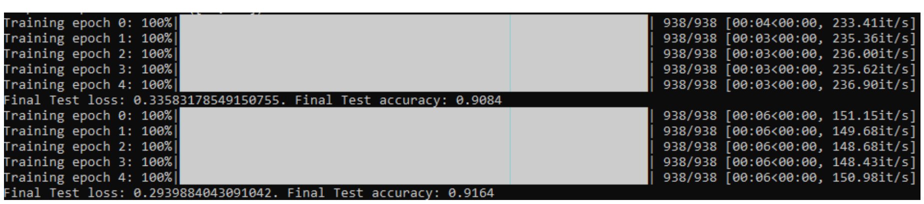


Figure 6: Terminal output from running script in 4a

As shown in figure 5, the neural network working on normalized data performs better than the one operating on non-normalized data. It also seems to converge faster, as its progression slows down before the original. Had one had more epochs of training, the original network could maybe end up at the same performance as the one with normalized data.

Normalizing the data helps with biases caused by some data types/points typically having a large value while others typically have a low value, and makes the network evaluate all inputs more equally.

In addition to the normalized model performing better than the original model, it also seems to take longer to train it, as in that the training actually takes more time.
This can be confirmed by looking at the terminal output when running the program, shown in figure 6. The training of the second network has significantly fewer iterations per second than the training of the first.

Task 4.b

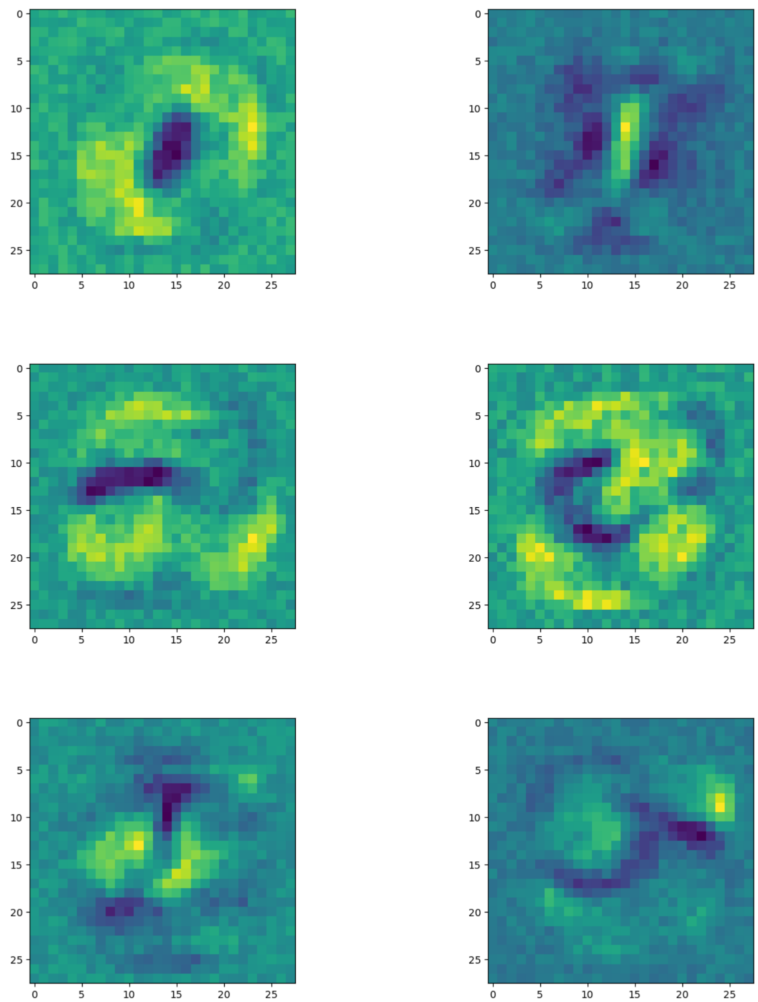


Figure 7: Weights from trained network, 0-5 from left to right, top to bottom

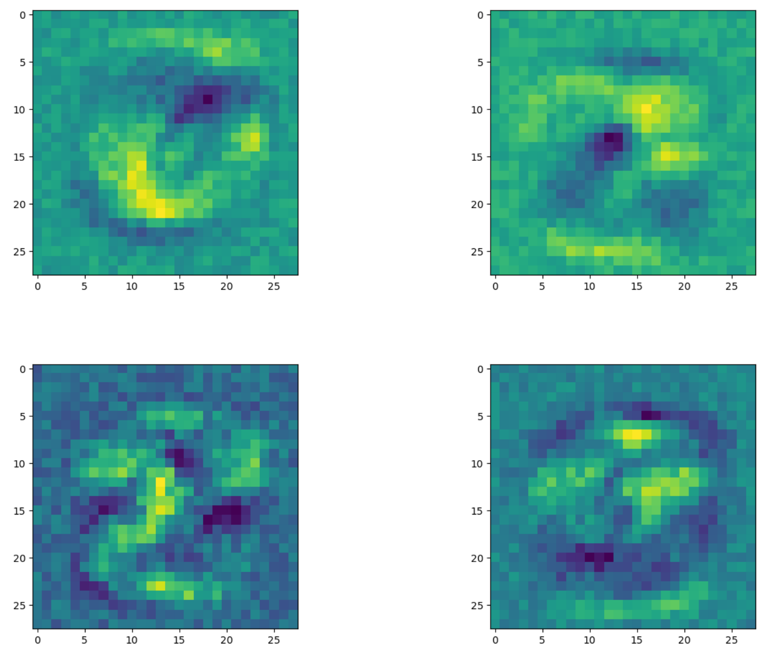


Figure 8: Weights from trained network, 6-9 from left to right, top to bottom

Comments:

0. The general shape of a zero can be seen, but the upper right part and lower left part are emphasized. The inside of the 0 is weighted with very small weights.

1. Only the middle part of the line of the 1 is emphasized. Most of the rest of the input is weighted with small weights

2. The general shape of a 2 can be seen, but the upper left part of the arc and the lower ”base line” are emphasized. The area below the upper arc is weighted with very small weights

3. The general shape of a 3 can be seen, and almost all of is emphasized. The ”inside” of the 3 is weighted with small weights

4. Only the middle of the 4 is emphasized. The inside of the upper part of the 4 is weighted with small values

5. The general shape of a 5 can be seen, very vaguely. The right part of the upper line is especially emphasized, and the area below it has small weights

6. The lower arc of the 6 is especially emphasized, and the part above this has especially small weights

7. The general shape of a 7 can be seen faintly but the upper half is most strongly emphasized. The inside of the 7 has small weights

8. The general shape of an 8 can be seen, but the left half is most strongly emphasized. The rest of the input has small weights, but the inside of the upper loop and the part inside the right "nook" of the 8 have especially low weights.

9. The upper loop and lower arc of the 9 is emphasized. The 0utline of the upper loop has especially low weights.

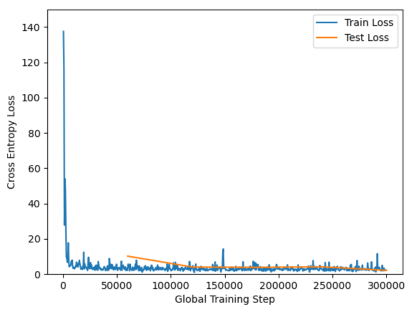


Figure 9: Training and validation loss eith learning rate = 1

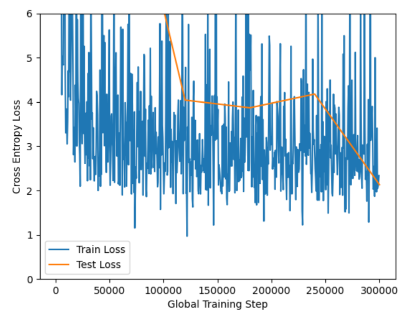


Figure 10: Zoomed in plot of loss

The final test accuracy is 0.8954, and the average training loss is 3.9627632697149866. This network acheives slightly worse performance than the one in a), and clearly has a lot larger average loss than the one in a).

An explanation of this is that the learning rate probably is to high, such that the steps taken in the gradient descent method overshoot minima of the cost function when training the network. This A large learning rate increases learning speed, but decreases granularity/accuracy of the learning.

Task 4.d

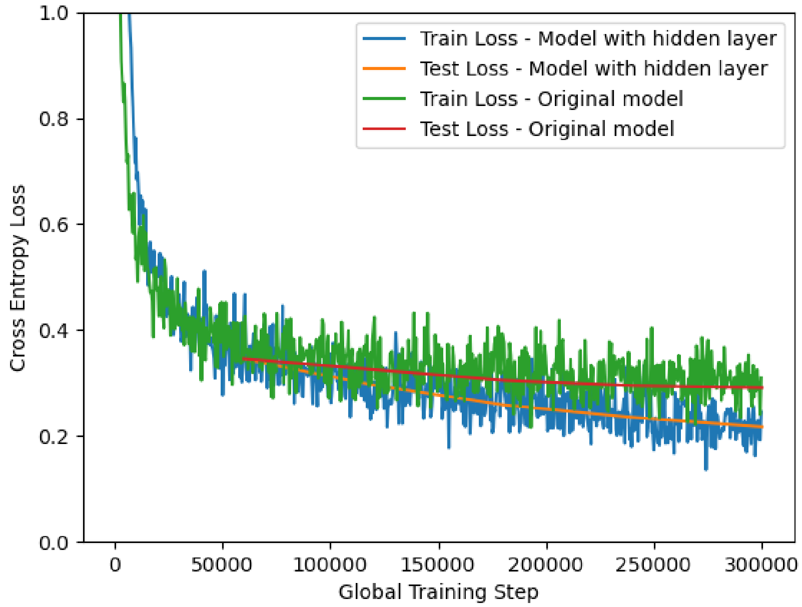


Figure 11: Neural network with one hidden layer vs neural network with zero hidden layers

Final test accuracy of the network with a hidden layer is 0.9368, which is slightly better than the one from a) (with normalized data), which had 0.9166

In conclusion, the network with one hidden layer performs slightly better than the one without. This is of course expected, as multi-layer neural network can "decode" more complex patterns than a single-layer one can.

It also seems like the network with a hidden layer has more potential than what is utilized here, as the lsoss graph is not completely flattened out. The single-layer network seems to have converged to a limit.