Assignment 1 Report

This is an outline for your report to ease the amount of work required to create your report. Jupyter notebook supports markdown, and I recommend you to check out this cheat sheet. If you are not familiar with markdown.

Before delivery, **remember to convert this file to PDF**. You can do it in two ways:

1. Print the webpage (ctrl+P or cmd+P)
2. Export with latex. This is somewhat more difficult, but you'll get somehwat of a "prettier" PDF. Go to File -> Download as -> PDF via LaTeX. You might have to install nbconvert and pandoc through conda; `conda install nbconvert pandoc`.

Task 1

task 1a)

$$C^n = -(y^n * ln(\hat{y}^n) + (1-y^n) * ln(1-\hat{y}^n))$$

$$\frac{\partial C^n}{\partial w} = -(y^n * \frac{(\hat{y}^n)'}{\hat{y}^n} + (1-y^n) * \frac{(1-\hat{y}^n)'}{1-\hat{y}^n})$$

$$\hat{y}^n = f(x) = \frac{1}{1+e^{-w^Tx}}$$

$$(\hat{y}^n)' = f'(x) = -\frac{e^{-w^Tx} * x}{(1+e^{-w^Tx})^2}$$

$$f'(x) = \frac{1}{1+e^{-w^Tx}} * \frac{e^{-w^Tx}}{1+e^{-w^Tx}} * x$$

We recognize that the first fraction is the sigmoid function, and the second fraction can also be simplified to the sigmoid function by adding and subtracting by 1

$$f'(x) = f(x) * (1-f(x)) * x$$

We now use this in our original equation:

$$\frac{\partial C^n}{\partial w} = -(y^n * \frac{\hat{y}^n * (1-\hat{y}^n) * x}{\hat{y}^n} + (1-y^n) * \frac{-\hat{y}^n * (1-\hat{y}^n) * x}{1-\hat{y}^n})$$

$$\frac{\partial C^n}{\partial w} = -(y^n x - y^n\hat{y}^n x - \hat{y}^n x + y^n\hat{y}^n x)$$

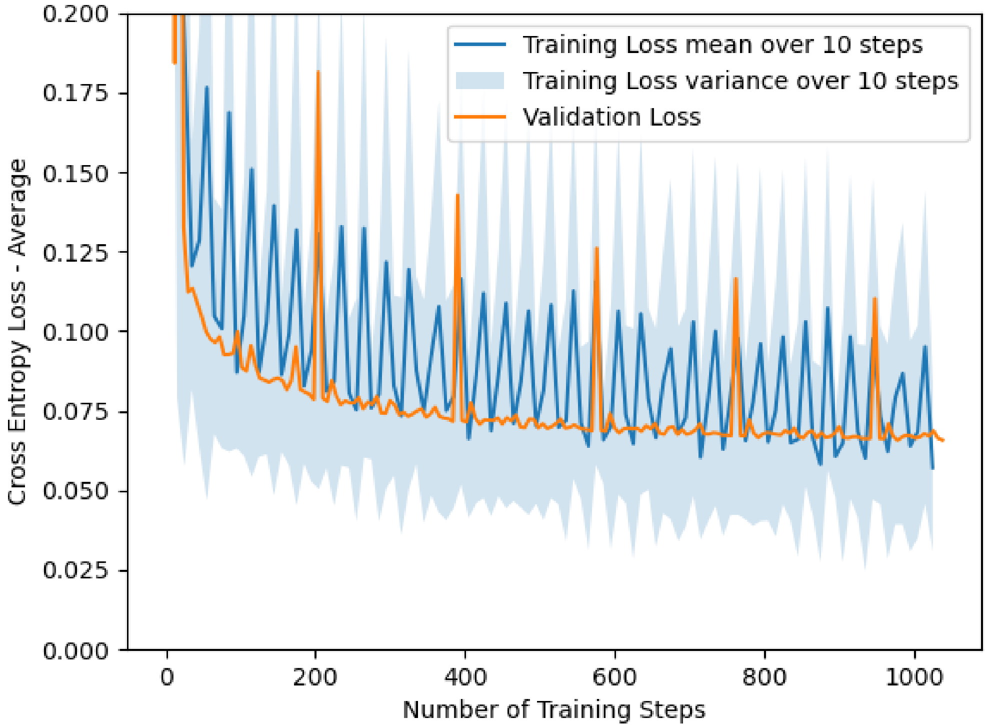
$$\frac{\partial C^n}{\partial w} = -(y^n - \hat{y}^n) * x$$

task 1a)

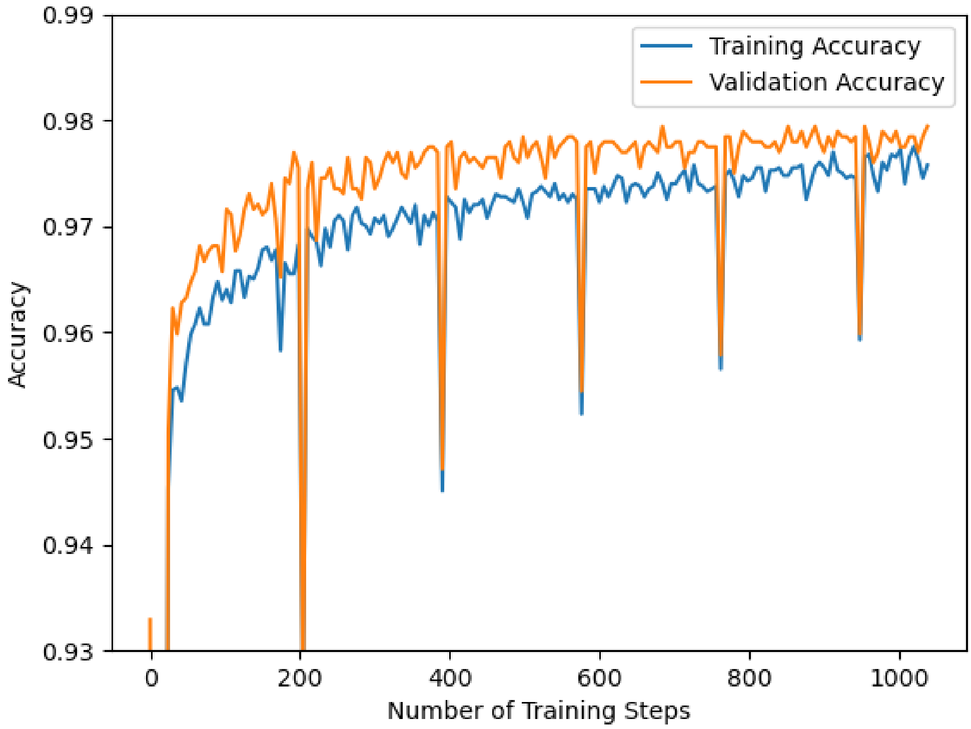
Fill in task 1a image of hand-written notes which are easy to read, or latex equations here

Task 2

Task 2b)



Task 2c)

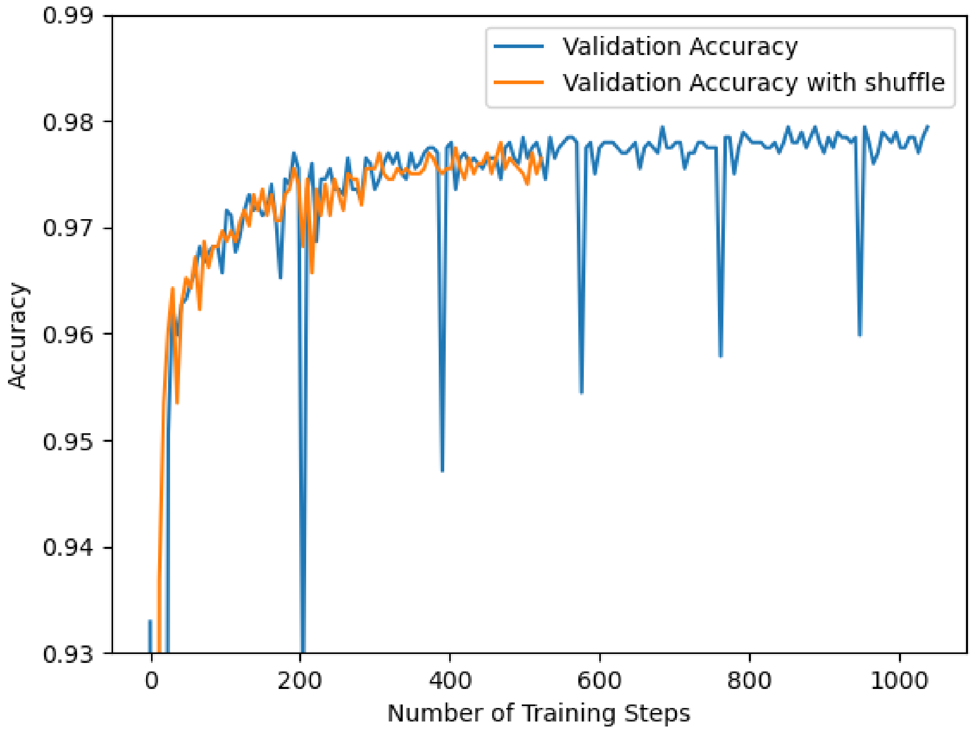


Task 2d)

Early stopping kicks in after only 33 epochs.

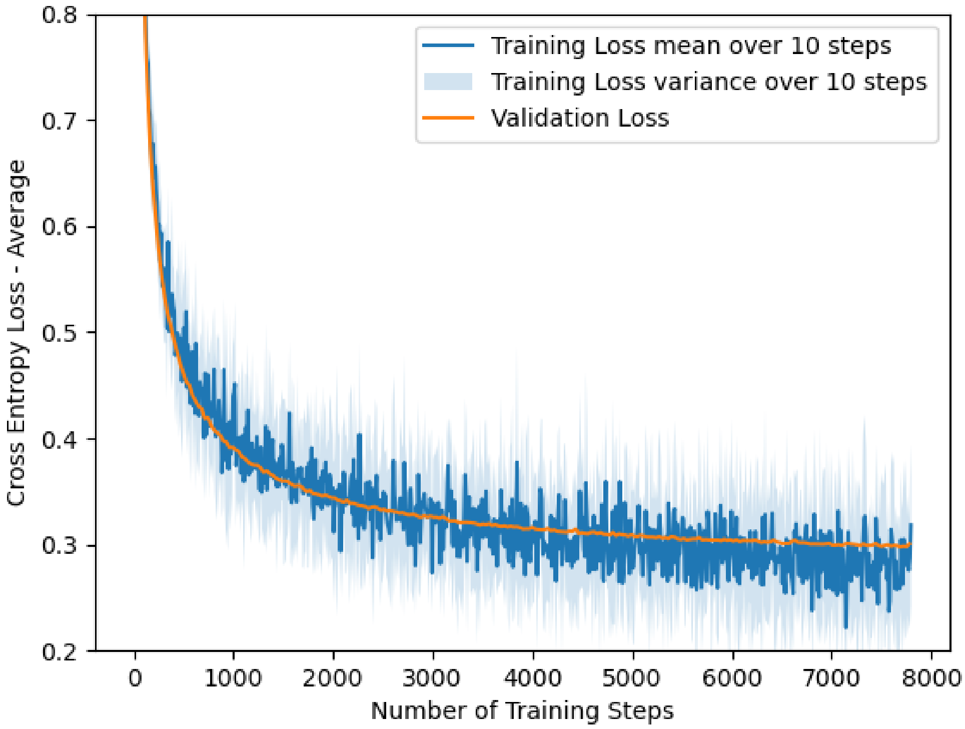
Task 2e)

The spikes in the accuracy when data is not shuffled is possibly due to the fact that a certain segment of the training data is not representative of the validation dataset. Shuffling the order of training data increases generality, and therefore the oscillations in accuracy are reduced.

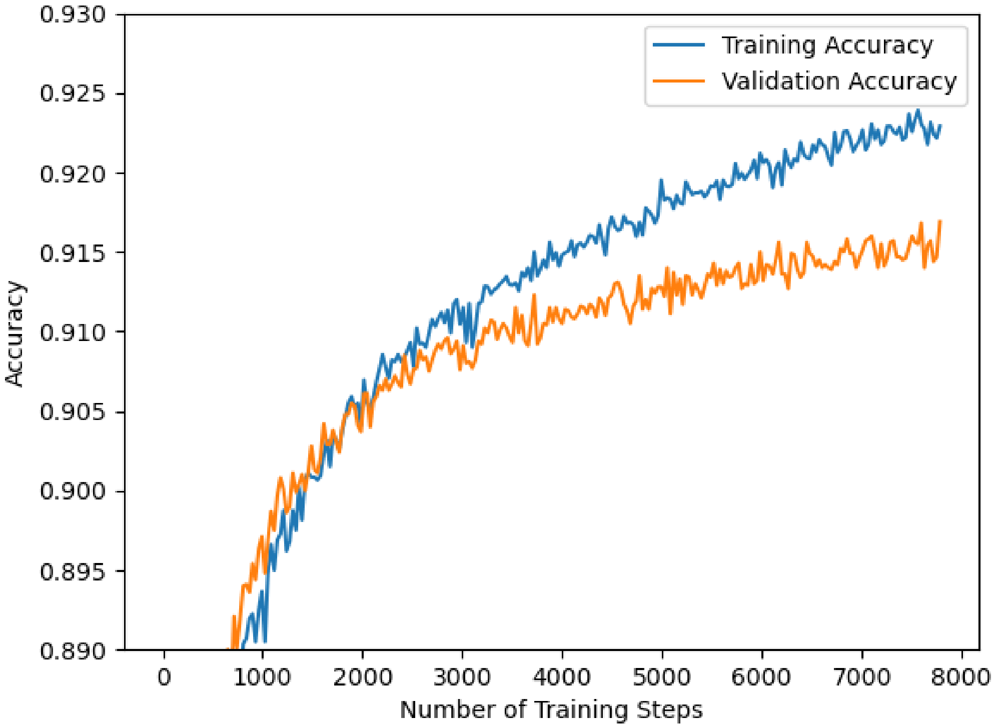


Task 3

Task 3b)



Task 3c)



Task 3d)

Yes, we can notice signs of overfitting based on the differing training accuracies. The gap between the training and validation is increasing towards the end of training, which is a sign of overfitting. However, the validation accuracy is still slightly increasing, so it is not clear-cut.

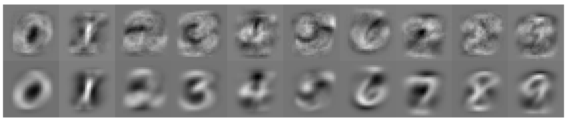
Task 4

Task 4a)

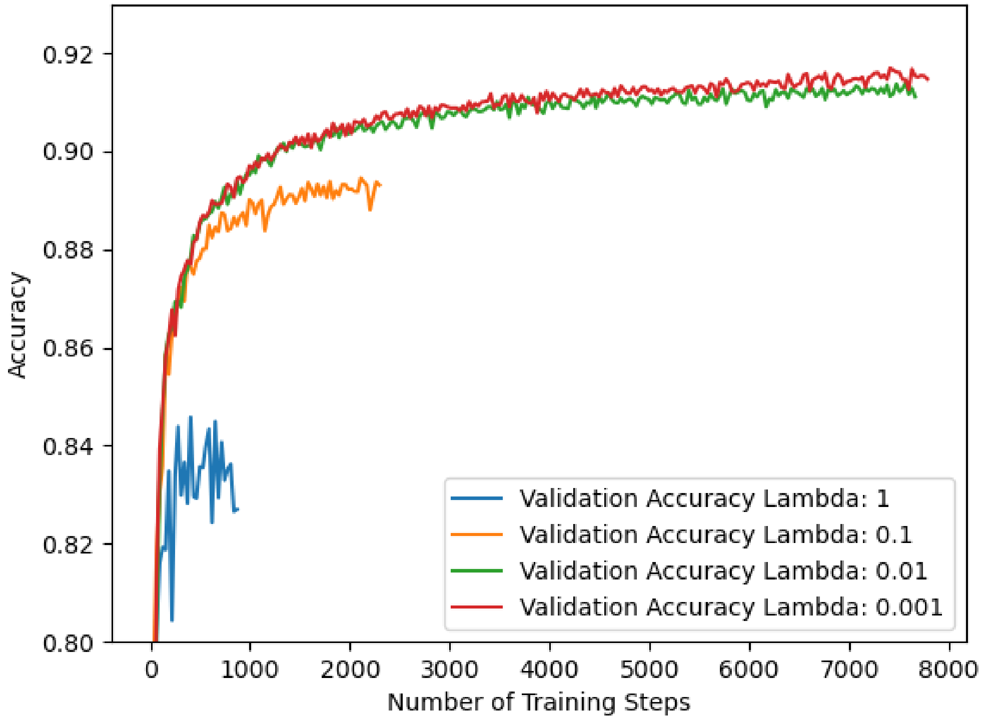
We have already found the update term for $\frac{\partial C(w)}{\partial w}$, so we only need to find out what the update term is for regularization. This will simply be λW.

Task 4b)

The bottom row is less noisy due to the difference in weight values. The regularized weights have smaller differences in values, and therefore appear to have more smooth transitions visually. Regularization also forces the model to not overfit on noise, keeping the model more generalizable.



Task 4c)

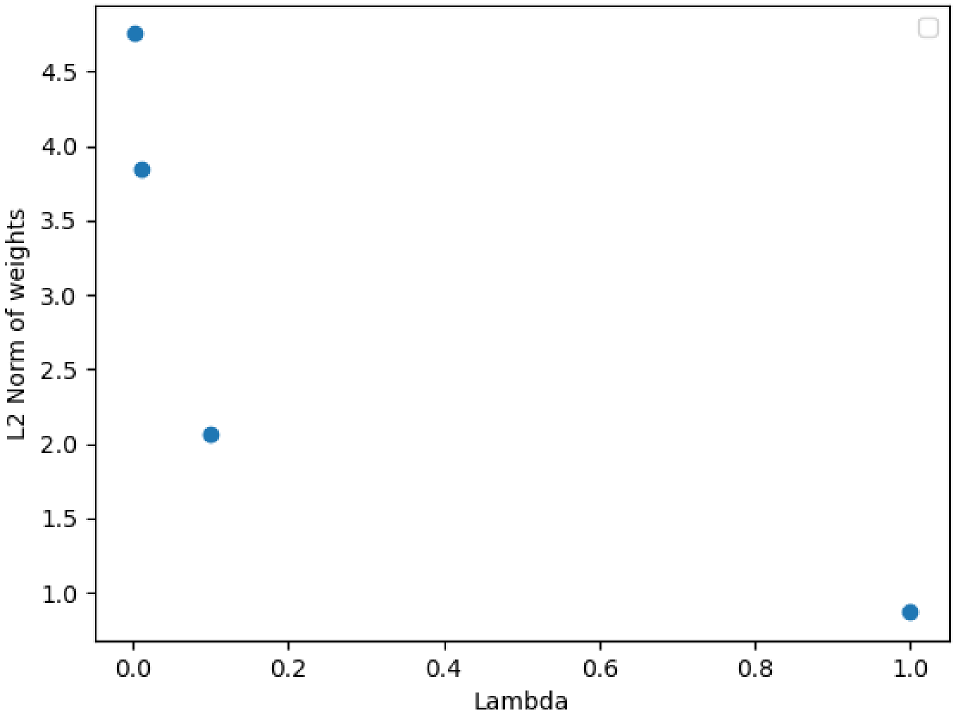


Task 4d)

It seems that the model is underfitting when using lambdas between 1 and 0.1, but I don't fully understand why the other models also perform worse. Perhaps the distribution between training and validation set is good enough that regularization has no meaningful impact, and simply worsens the result?

Task 4e)

We can observe that the models with lower lambdas for the l2 regularization clearly have a higher l2 norm, which makes sense given that regularization is meant to punish large weight values.


L2 Norm of weights
4.5
4.0
3.5
3.0
2.5
2.0
1.5
1.0
0.0
0.2
0.4
0.6
0.8
1.0
Lambda