Assignment 1

Gian Hug - András Izsó

February 2, 2022

1 Task

Task 1a): Logistic Regression

$$\frac{\partial C(w)}{\partial w_i} = -(y^n \frac{\partial}{\partial w_i} ln(\hat{y_n}) \frac{\partial}{\partial w_i} \hat{y}^n + (1-y^n) \frac{\partial}{\partial w_i} ln(1-\hat{y}^n)(-\frac{\partial \hat{}}{\partial w_i} y^n) =$$
$$= -(y^n \frac{1}{\hat{y}^n} x_i^n f(x^n)(1-f(x^n)) + (y^n-1) \frac{1}{1-\hat{y}^n} x_i^n f(x^n)(1-f(x^n)) =$$
$$= -(y^n \frac{1}{\hat{y}^n} + (y^n-1) \frac{1}{1-\hat{y}^n}) x_i^n f(x^n)(1-f(x^n)) =$$
$$= -(y^n(1-f(x^n)) + (y^n-1) f(x^n)) x_i^n =$$
$$= -(y^n - y^n f(x^n) + y^n f(x^n) - f(x^n)) x_i^n =$$
$$= -(y^n - \hat{y}^n) x_i^n$$

Task 1b): Softmax

$$\frac{\partial C^n(w)}{\partial w_{l,m}} = \frac{\partial}{\partial w_{l,m}} (-\sum_{k=1}^{K} y_k^n \cdot ln(\hat{y}_k^n)) =$$
$$= -\sum_{k=1}^{K} y_k^n \frac{\partial}{\partial w_{l,m}} ln\left(\frac{e^{z_k}}{\sum_{k'}^{K} e^{z_{k'}}}\right) =$$
$$= -\sum_{k=1}^{K} y_k^n \frac{\partial}{\partial w_{l,m}} \left(z_k - ln\left(\sum_{k'}^{K} e^{z_{k'}}\right)\right) =$$
$$= -\sum_{k=1}^{K} y_k^n \left(\frac{\partial}{\partial w_{l,m}} \left(\sum_{i=1}^{I} w_{k,i} x_i\right) - \frac{1}{\sum_{k'}^{K} e^{z_{k'}}} \frac{\partial}{\partial w_{l,m}} \left(\sum_{k'}^{K} e^{z_{k'}}\right)\right) =$$
$$= -\sum_{k=1}^{K} y_k^n \frac{\partial}{\partial w_{l,m}} \left(\sum_{i=1}^{I} w_{k,i} x_i\right) + \sum_{k=1}^{K} y_k^n \frac{1}{\sum_{k'}^{K} e^{z_{k'}}} \sum_{k'}^{K} e^{z_{k'}} \frac{\partial}{\partial w_{l,m}} \sum_{i=1}^{I} w_{k',i} x_i =$$

The two part with $w_{k,i}x_i$ is only non-zero if $k(ork') = l$ and $i = m$. In that case both equal to x_m.

$$= -y_k^n x_m + \sum_{k=1}^{K} y_k^n \frac{e^{z_l}}{\sum_{k'}^{K} e^{z_{k'}}} x_m =$$
$$= -y_k^n x_m + \sum_{k=1}^{K} y_k^n \hat{y}_l^n x_m =$$
$$= -y_k^n x_m + \hat{y}_l^n x_m \sum_{k=1}^{K} y_k^n =$$
$$= -x_m(y_k^n - \hat{y}_l^n)$$

2 Task

Task 2b)

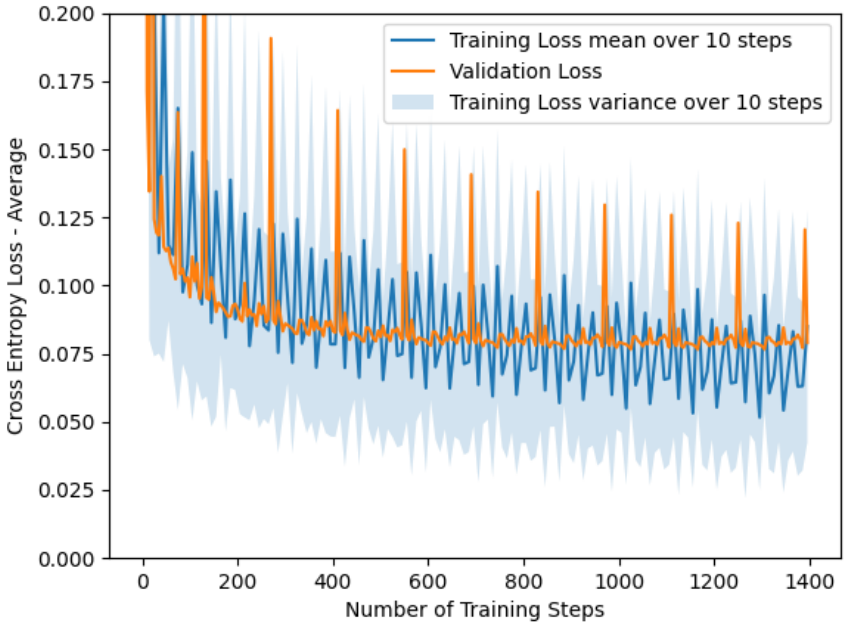

Figure 1: Evolution of cross entropy loss

Task 2c)

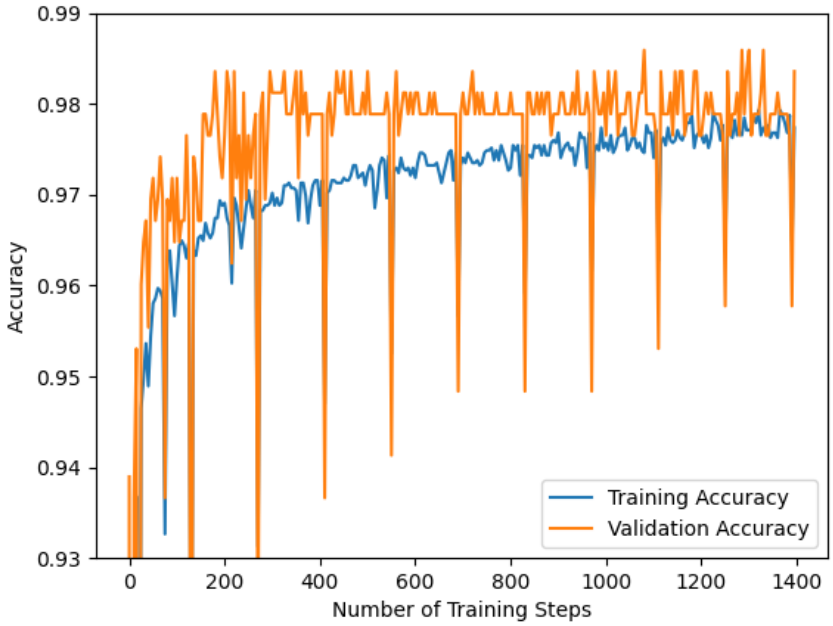

Figure 2: Evolution of accuracy

Task 2d)

The early stopping kicks in already after 10 epochs.

Task 2e)

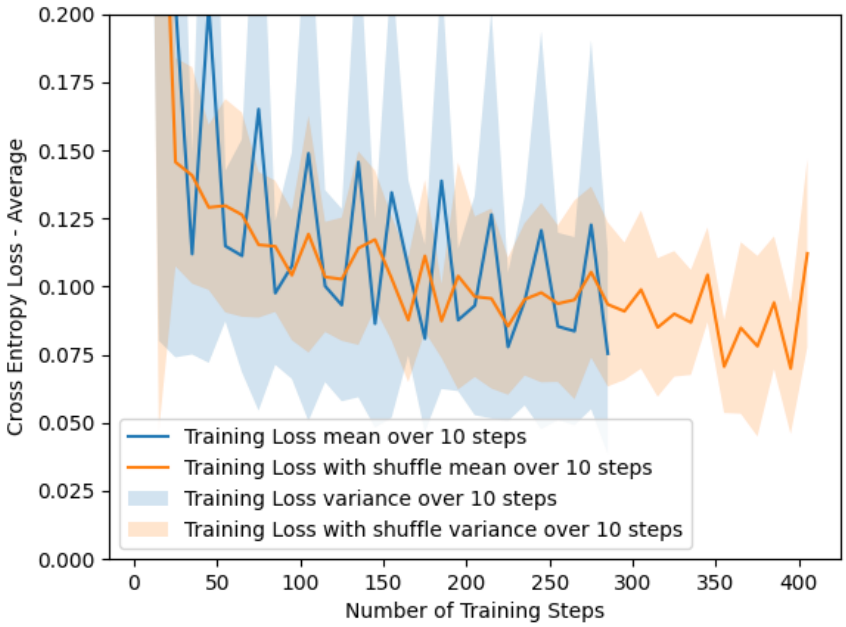

Figure 3: Evolution of loss for with and without reshuffling after each epoch

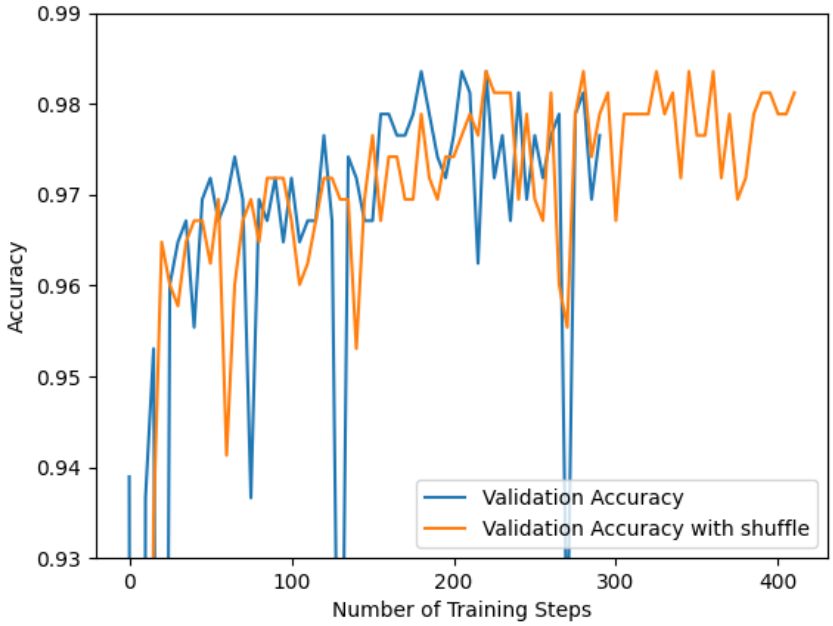

Figure 4: Evolution of accuracy for with and without reshuffling after each epoch

If we do not use reshuffle the model will also learn the pattern at which the data samples appear and overfit to that. This is the reason we get the spikey behaviour in loss and accuracy. This overfit to the order at which samples appear is highly undesirable in practice and thus makes reshuffling a crucial step.

Task 2e)

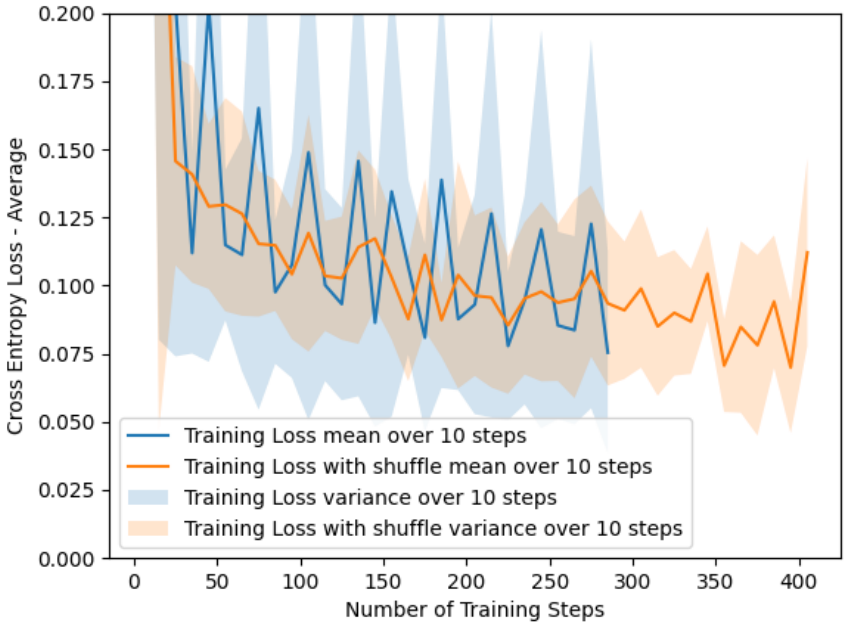

Figure 5: Evolution of loss for with and without reshuffling after each epoch

3 Task

Task 3b)

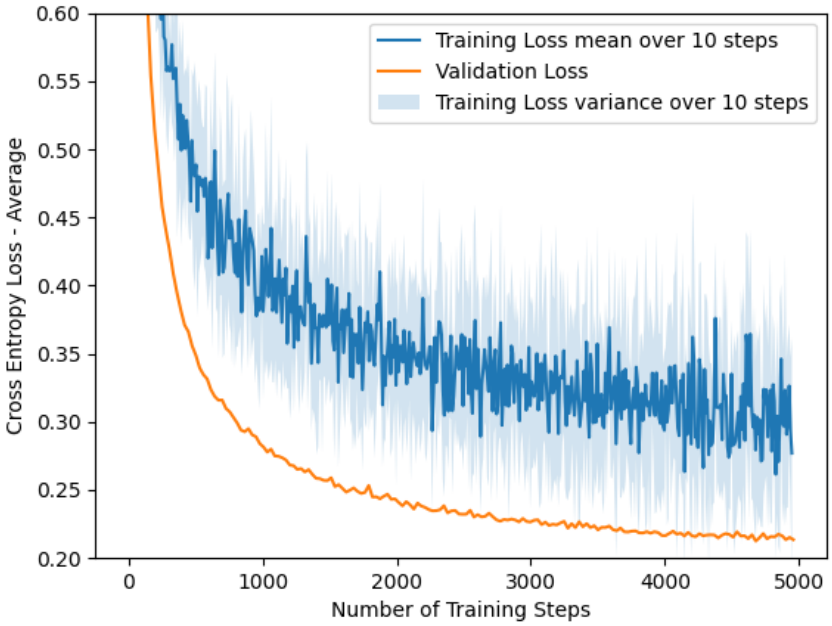

Figure 6: Evolution of training and validation loss with $n_{val} = 2000$

Task 3c)

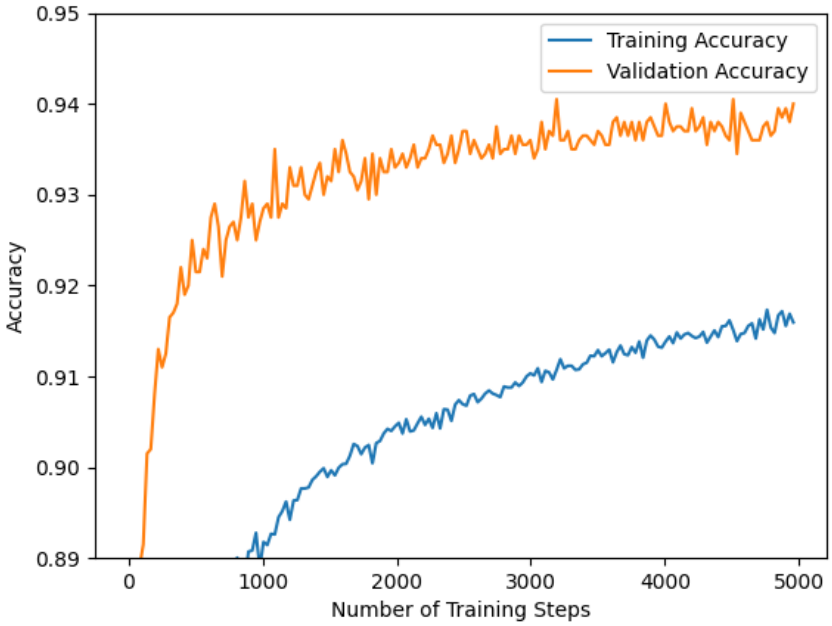

Figure 7: Evolution of training and validation accuracy with $n_{val} = 2000$

Task 3d)

No there are no signs of over fitting. Over fitting would imply that the validation loss and accuracy would be significantly worse (higher and lower respectively) than their training counterpart since the model is over fitting to the training date. Here the exact opposite is occurring, namely the validation set is performing significantly better than the training set. This can have different reasons. Here the difference in size of the training and validation set seems to be the case. The validation set size for 7 and 6 is 2000 which are relatively few sample compared to the 18000 of the training set. Additionally the validation samples are sampled in a deterministic manner from the MNIST set which distorts the validation-training accuracy relation. In 7 and 8 the validation set was increased to 9000. We can see in 8 that the results of validation and training are more like expected. In 9 we can see first signs of over fitting, namely a flattening of the validation accuracy while the training accuracy is still increasing. But it is to early to talk about over fitting yet.

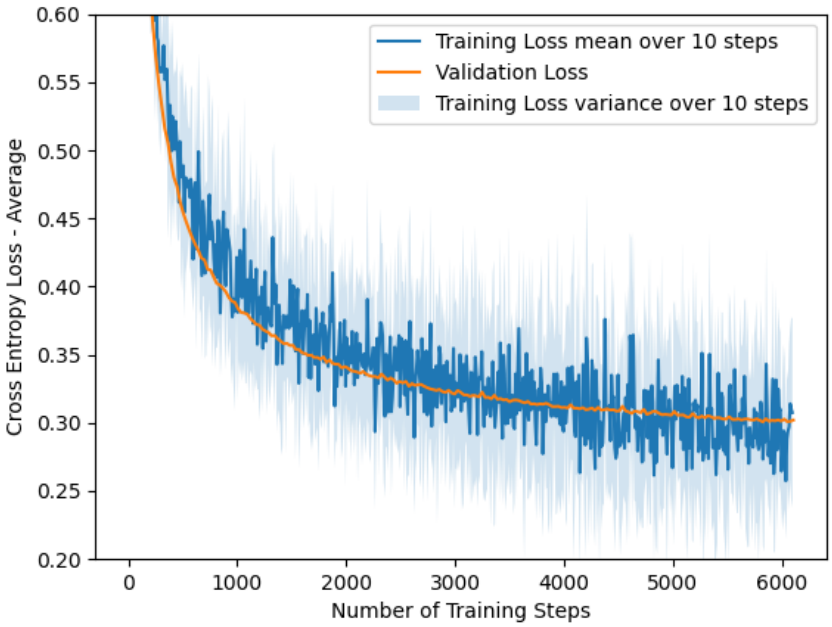

Figure 8: Evolution of training and validation loss with $n_{val} = 4000$

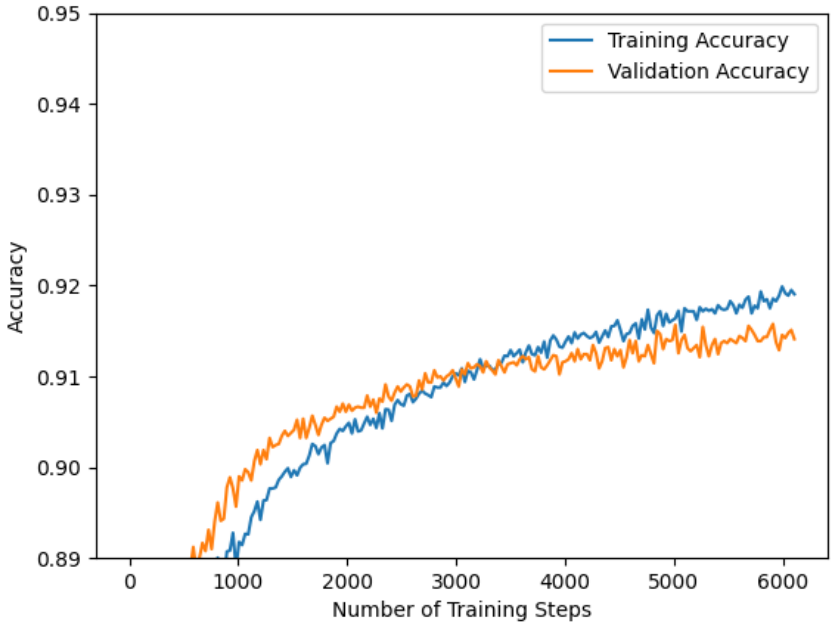

Figure 9: Evolution of training and validation accuracy with $n_{val} = 4000$

4 Task

Task 4a

$$\frac{\partial J(w)}{\partial w_{l,m}} = \frac{\partial}{\partial w_{l,m}}(C(w) + \lambda R(w)) = \frac{\partial C(w)}{\partial w_{l,m}} + \lambda \frac{\partial R(w)}{\partial w_{l,m}} = -x_m(y_l^n - \hat{y_l}^n) + \lambda \frac{\partial}{\partial w_{l,m}} \frac{1}{2} \sum_{i,j} w_{i,j}^2 =$$

$$= -x_m(y_l^n - \hat{y_l^n}) + \frac{\lambda}{2} 2 w_{l,m} = -x_m(y_l^n - \hat{y_l^n}) + \lambda w_{l,m}$$

Task 4b

I think the visualizations with higher lambda are less noisy, because the use of regulation makes the model to only increase the necessary weights, and keep the others down. This results in a more "focused" vision of the digits itself. It's also noticable that the weights are truly lower, because the bottom are fading out.

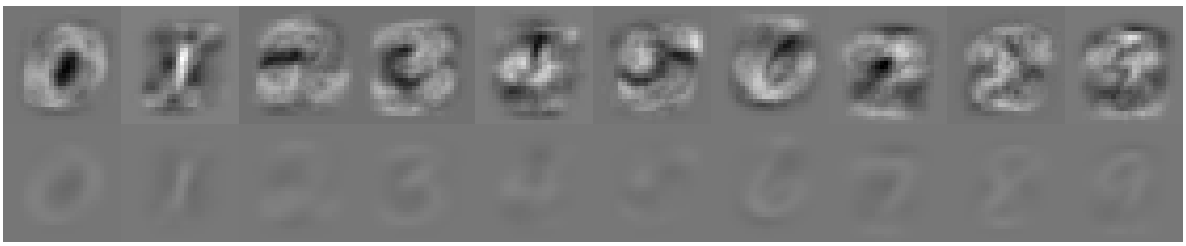

Figure 10: The visualizations of the weight matrices. Top row: $\lambda = 0$, bottom: $\lambda = 2.0$. Digits are increasing from left to right

Task 4c

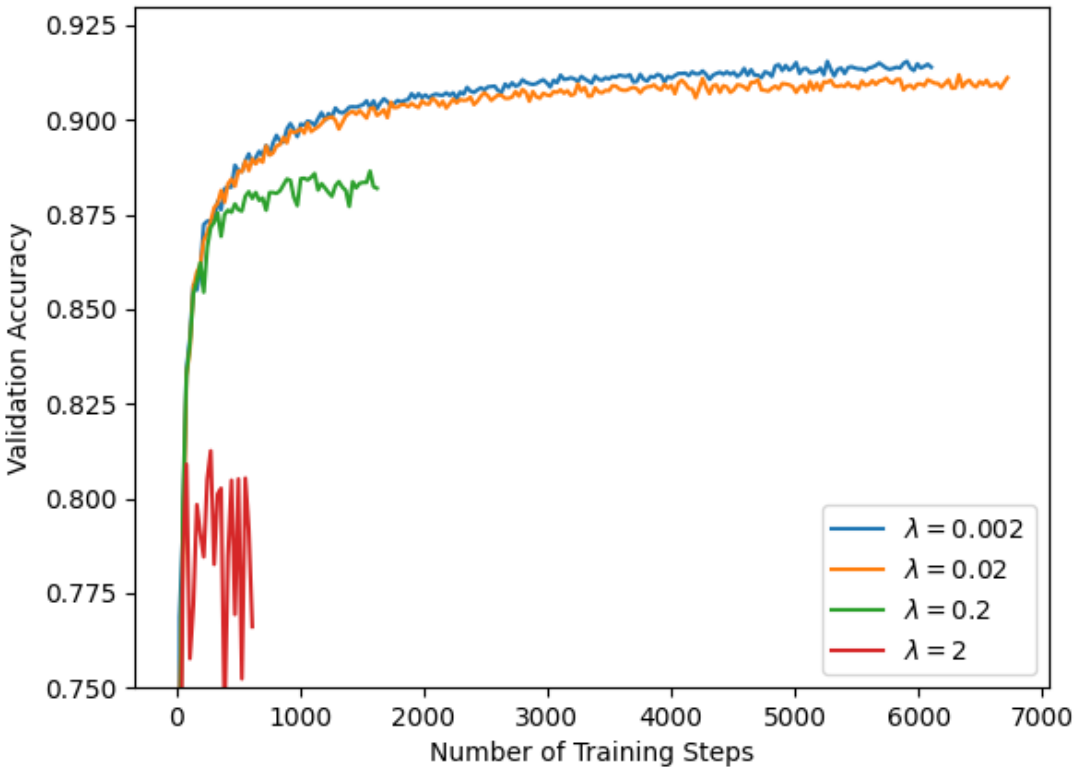

Figure 11: The validation accuracy of the model with different regularization coefficients over the training steps.With increased regularization the training stabilizes sooner, but the accuracy remains lower.

Task 4d

The use of regularisation shifts the bias-variance trade off towards higher bias and lower variance with the goal of achieving a better generalisation of the model and avoid over fitting to the current data set. Thus the model should perform better with samples different from the training set. If the validation set is very similar to the training data set the regularisation decreases the validation accuracy since the regularisation leads to a model which is not solely optimised for the loss/accuracy of the training set (which may be very similar to the validation set).

Task 4e

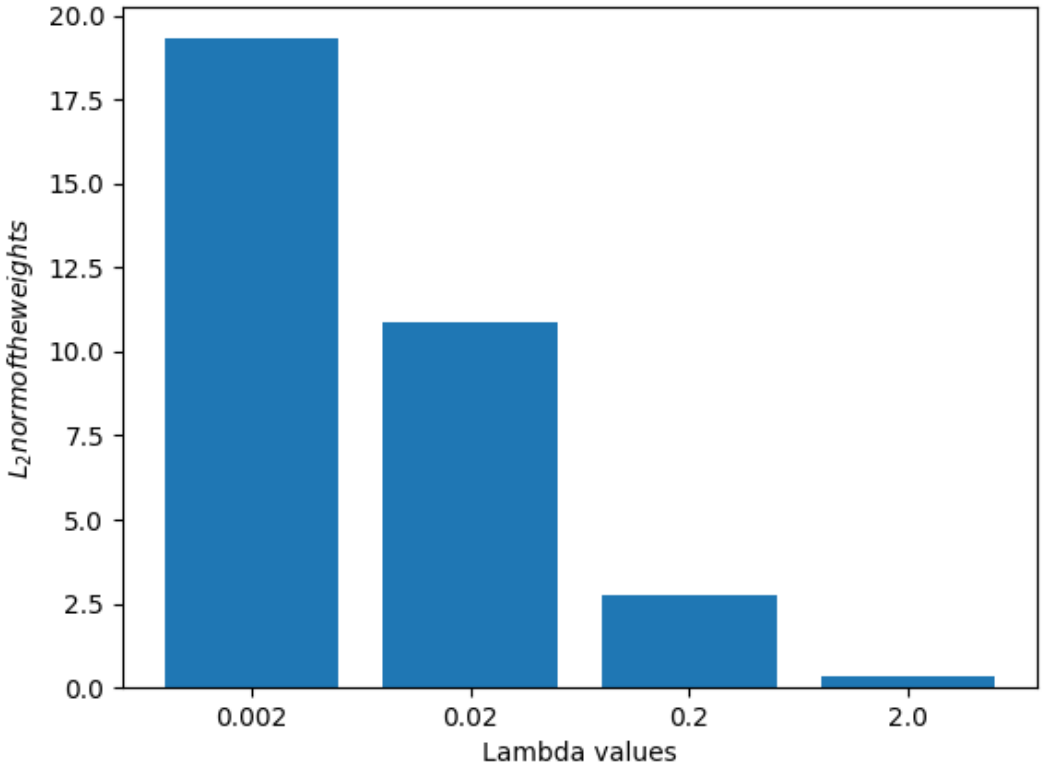

Figure 12: It is clearly visible, that the resulting weights are much lower with increased regularization