TDT4265 - Computer Vision and Deep Learning

Vegard Iversen, Sebastian Skogen Raa

Feb 2022

1

1.1 Task 1a

Answer
In logistic regression we have the $\hat{y}^n = f(x^n) = \frac{1}{1+e^{-w^T x^n}}$ which is the Sigmoid activation function. The cost function we want to take the gradient of is: $C(w) = \frac{1}{N}\sum_{n=1}^{N} C^n$, where $C^n(w) = -(y^n ln(\hat{y}^n) + (1-y^n)ln(1-\hat{y}^n))$ The expression can be divided into two parts and then sum them together after they have been differentiated. $g^n(w) = y^n ln(\hat{y}^n)$ and $h^n(w) = (1-y^n)ln(1-\hat{y}^n)$ By using the hint $\frac{\delta f(x^n)}{\delta w_i} = x_i^n f(x^n)(1-f(x^n))$ we get: $\frac{\delta g^n(w)}{\delta w_i} = \frac{\delta y^n ln f(x^n)}{\delta w_i} = y^n$

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

$$C(w) = \frac{1}{N}\sum_{n=1}^{N} C^n \tag{2}$$

$$C^n(w) = -(y^n ln(\hat{y}^n) + (1-y^n)ln(1-\hat{y}^n)) \tag{3}$$

We can split the function into two parts, since differentiation is a linear operator.

$$g^n(w) = y^n ln(\hat{y}^n) \tag{4}$$

$$h^n(w) = (1-y^n)ln(1-\hat{y}^n) \tag{5}$$

By using the hint

$$\frac{\delta f(x^n)}{\delta w_i} = x_i^n f(x^n)(1-f(x^n)) \tag{6}$$

We can write $g^n(w) = y^n ln(\frac{1}{1+e^{-w^T x^n}}) = y^n ln(f(x^n))$
We know that:

$$\frac{d}{dx} lnx = \frac{1}{x} \tag{7}$$

By using the chain rule and 6 we get

$$\frac{\delta g^n(w)}{\delta w_i} = \frac{\delta y^n ln(f(x^n))}{\delta w_i} = y^n(\frac{1}{f(x^n)} x_i^n f(x^n)(1-f(x^n)) \tag{8}$$

Simplifying gives:

$$\frac{\delta g^n(w)}{\delta w_i} = y^n x_i^n (1-f(x^n)) \tag{9}$$

Now we do the same for $h^n(w)$

$$\frac{\delta h^n(w)}{\delta w_i} = \frac{\delta(1-y^n)ln(1-f(x^n)))}{\delta w_i} = (1-y^n)(\frac{1}{1-f(x^n)}x_i^n f(x^n)(1-f(x^n))) \tag{10}$$

Simplifying gives:

$$\frac{\delta h^n(w)}{\delta w_i} = (1-y^n)x_i^n f(x^n) \tag{11}$$

Now we can sum $g^n(w)$ and $h^n(w)$ together

$$\frac{\delta C^n(w)}{\delta w_i} = -(\frac{\delta g^n(w)}{\delta w_i} + \frac{\delta h^n(w)}{\delta w_i}) = -(y^n x_i^n(1-f(x^n)) + (1-y^n)x_i^n f(x^n) = -x_i^n(y^n - f(x^n)) \tag{12}$$

Since $\hat{y}^n = f(x^n)$ we get:

$$\frac{\delta C^n(w)}{\delta w_i} = -x_i^n(y^n - \hat{y}^n) \tag{13}$$

q.e.d

1.2 Task 1b

Now we are deriving the gradient for Softmax Regression. With a given input x which can belong to K different classes, will the softmax regression output a vector $\hat{y}$ with length K and where each element $\hat{y_k}$ represent a probability that x is a member of class k The equation for softmax regression is given by this:

$$\hat{y_k} = \frac{e^{z_k}}{\sum_{k'}^K e^{z_{k'}}}, \quad \text{where } z_k = w_k^T \cdot x = \sum_i^I w_{k,i} \cdot x_i \tag{14}$$

$$C(w) = \frac{1}{N}\sum_{n=1}^N C^n \tag{15}$$

$$C^n(w) = -\sum_{k=1}^K y_k^n ln(\hat{y}_k^n) \tag{16}$$

Goal is to show that

$$\frac{\delta C^n(w)}{\delta w_{kj}} = -x_j^n(y_k^n - \hat{y}_k^n) \tag{17}$$

We can again use

$$\hat{y_k^n} = \frac{e^{z_k^n}}{\sum_{k'}^K e^{z_{k'}^n}}, \quad \text{where } z_k^n = w_k^T \cdot x^n = \sum_i^I w_{k,i} \cdot x_i^n \tag{18}$$

$$C^n(w) = -\sum_{k=1}^K y_k^n ln(\frac{e^{z_k^n}}{\sum_{k'}^K e^{z_{k'}^n}}) \tag{19}$$

$$C^n(w) = -\sum_{k=1}^K (y_k^n(ln(e^{z_k^n}) - ln(\sum_{k'}^K e^{z_{k'}^n}))) \tag{20}$$

$$C^n(w) = -\sum_{k=1}^K (y_k^n z_k^n) + \sum_{k=1}^K (y_k^n(ln(\sum_{k'}^K e^{z_{k'}^n}))), \quad \text{where } \sum_{k=1}^K y_k^n = 1 \tag{21}$$

$$C^n(w) = -\sum_{k=1}^{K}(y_k^n z_k^n) + ln(\sum_{k'}^{K} e^{z_{k'}^n}) \tag{22}$$

We can now split the function $C^n(w)$ into two parts, $g^n(w) = \sum_{k=1}^{K}(y_k^n z_k^n)$ and $h^n(w) = ln(\sum_{k'}^{K} e^{z_{k'}^n})$. Then we can write the cost function as

$$C^n(w) = -g^n(w) + h^n(w) \tag{23}$$

When we are calculating the gradient of the cost function we can then do it in two steps. First we differentiate $g^n(w)$ and then $h^n(w)$

$$\frac{\delta g^n(w)}{\delta w_{kj}} = \frac{\delta}{\delta w_{kj}} \sum_{k=1}^{K}(y_k^n z_k^n) \tag{24}$$

$$\frac{\delta g^n(w)}{\delta w_{kj}} = \frac{\delta}{\delta w_{kj}} \sum_{k=1}^{K} y_k^n \sum_{k=1}^{K} z_k^n \tag{25}$$

By using the definition of $z_k^n = w_k^T \cdot x^n = \sum_i^I w_{k,i} \cdot x_i^n$

$$\frac{\delta g^n(w)}{\delta w_{kj}} = \frac{\delta}{\delta w_{kj}} \sum_{k=1}^{K} y_k^n \sum_{i=1}^{I} z_k^n \tag{26}$$

From this we can see that if we sum out the summation and differentiate with respect to w_{kj} will all the terms/parts where j != i (j not equal to i) be 0. Therefor we get that

$$\frac{\delta g^n(w)}{\delta w_{kj}} = y_k \cdot x_j^n \tag{27}$$

Now we will differentiate $h^n(w)$

$$\frac{\delta h^n(w)}{\delta w_{kj}} = \frac{\delta}{\delta w_{kj}} ln(\sum_{k'}^{K} e^{z_{k'}^n}) \tag{28}$$

by using the definition of $z_{k'}^n$ we get:

$$\frac{\delta h^n(w)}{w_{kj}} = \frac{\delta}{\delta w_{kj}} ln(\sum_{k'}^{K} e^{\sum_i^I w_{k',i} \cdot x_i^n}) \tag{29}$$

By using the chain rule we get

$$\frac{\delta h^n(w)}{w_{kj}} = \frac{1}{e^{\sum_i^I w_{k',i}}} \frac{\delta}{\delta w_{kj}} (\sum_{k'}^{K} e^{\sum_i^I w_{k',i} \cdot x_i^n}) \tag{30}$$

Again we can see that to get non-zero terms we have to have j = i and k = k' (by just writing out some of the summation we can prove this).
Therefor we get

$$\frac{\delta}{\delta w_{kj}} (\sum_{k'}^{K} e^{\sum_i^I w_{k',i} \cdot x_i^n}) = x_j e^{\sum_i^I w_{ki} \cdot x_i^n} \tag{31}$$

Now we can simplify $\frac{\delta h^n(w)}{w_{kj}}$

$$\frac{\delta h^n(w)}{w_{kj}} = \frac{1}{e^{\sum_i^I w_{k',i}}} e^{\sum_i^I w_{ki} \cdot x_i^n} \tag{32}$$

by using the definition of z_k^n and $\hat{y_k^n}$

$$\frac{\delta h^n(w)}{w_{kj}} = \frac{e^{z_k^n} \cdot x_j^n}{\sum_{k'}^K e^{z_{k'}^n}} \tag{33}$$

and $\hat{y_k^n}$

$$\frac{\delta h^n(w)}{w_{kj}} = x_j^n \hat{y_k^n} \tag{34}$$

We can now combine the two parts together

$$\frac{\delta C^n(w)}{\delta w_{kj}} = -\frac{\delta g^n(w)}{\delta w_{kj}} + \frac{\delta h^n(w)}{\delta w_{kj}} \tag{35}$$

$$\frac{\delta C^n(w)}{\delta w_{kj}} = -y_k \cdot x_j^n + x_j^n \hat{y_k^n} \tag{36}$$

$$\frac{\delta C^n(w)}{\delta w_{kj}} = -x_j^n (y_k^n - \hat{y}_k^n) \tag{37}$$

q.e.d

2b

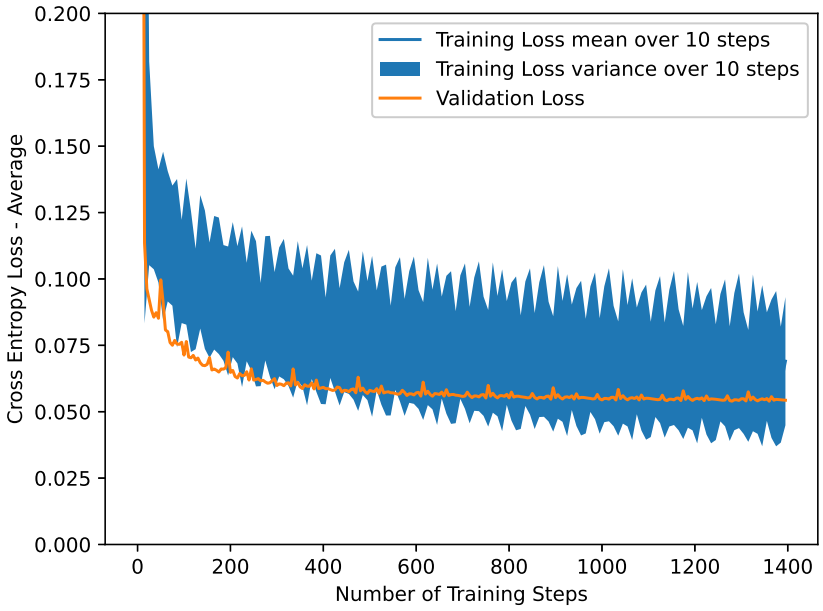

Figure 1: plot of the training and validation loss over training

2c

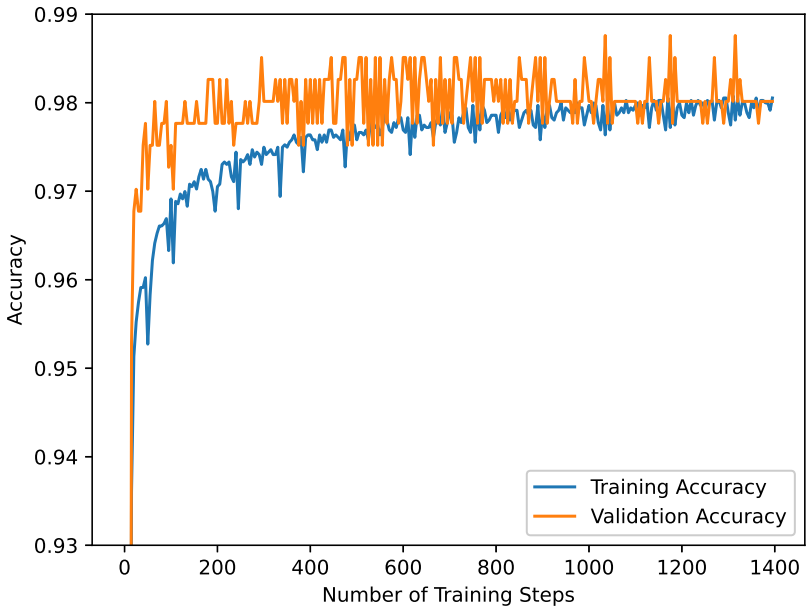

Figure 2: Plot of accuracy on the training set and validation set over training.

2d

When only considering the latest 19 of 500 epochs, the early stop is triggered at global step 15.

2e

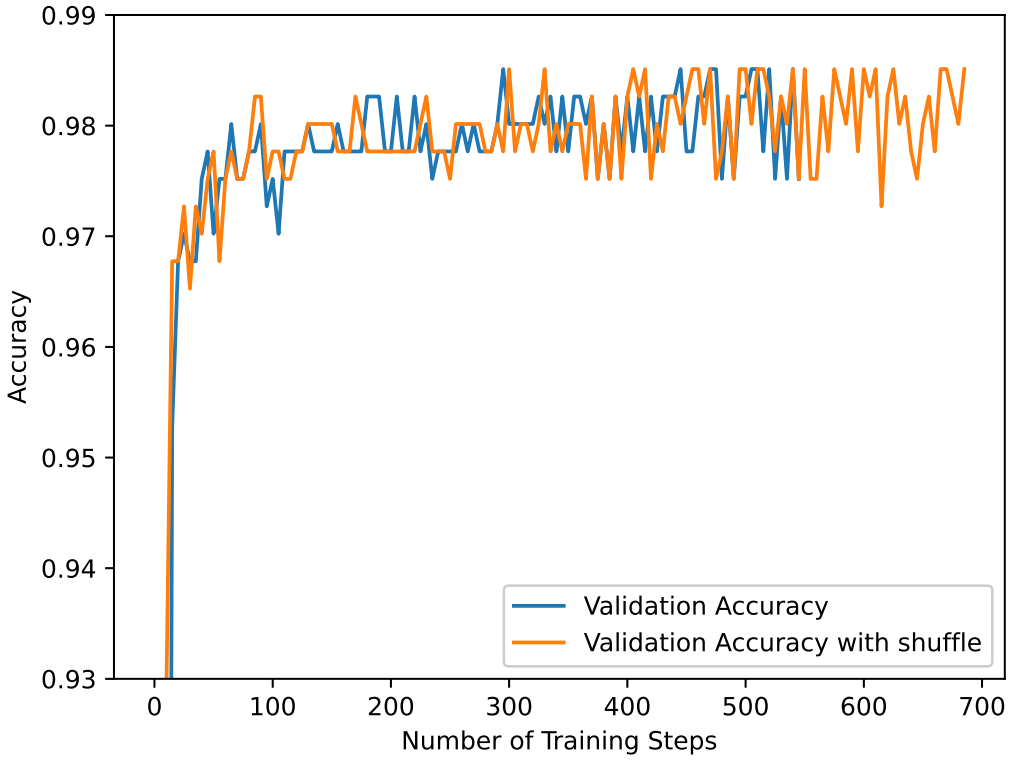

Figure 3: Plot of the validation accuracy with and without shuffle.

Shuffling can help with improving the accuracy of the gradient calculation when using mini-batch. Because of this the model "follows" the optimum path to local minima better.

1.3 3b

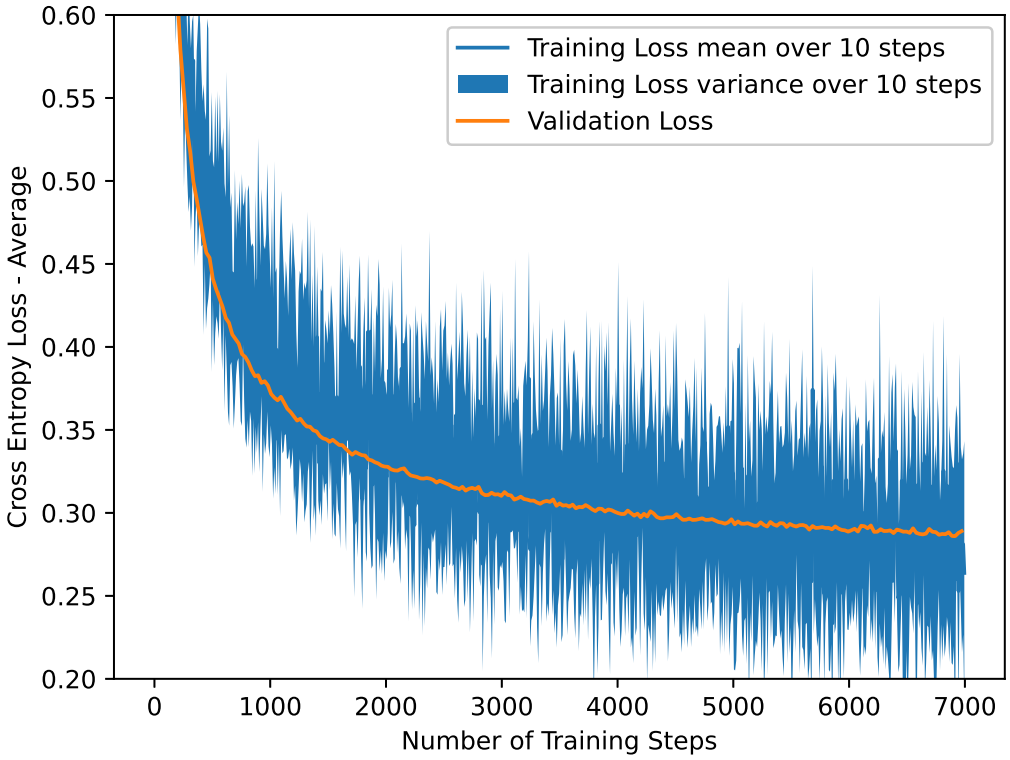

Figure 4: Plot of the training and validation loss over training.

1.4 3c

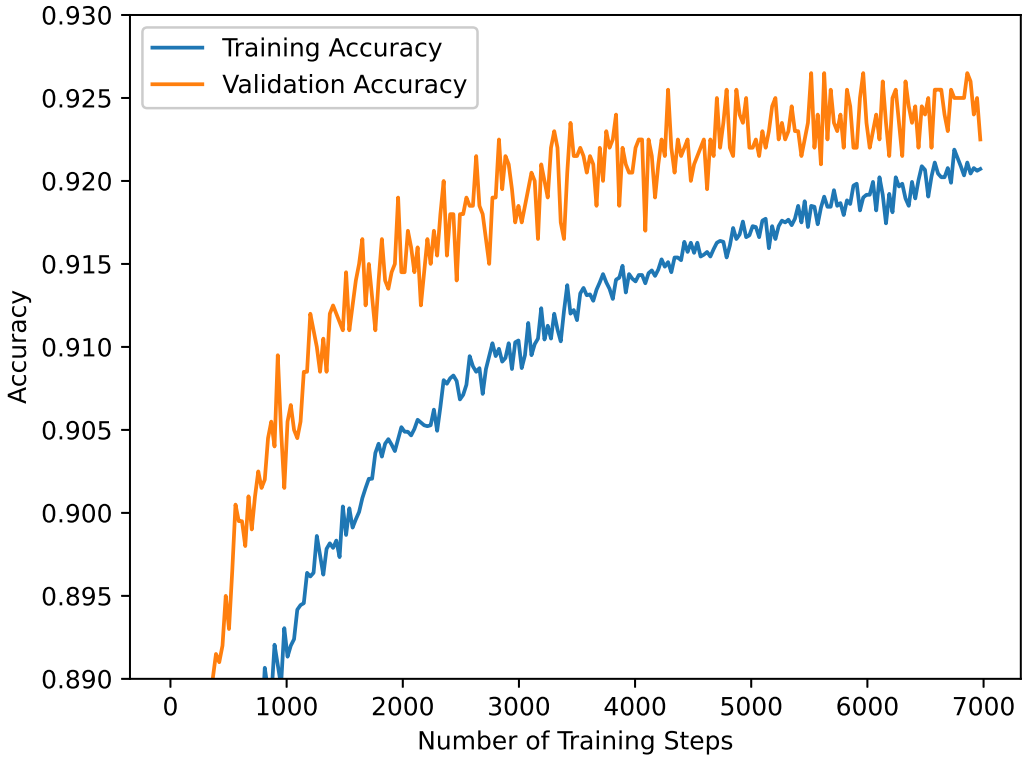

Figure 5: Plot of the training and validation accuracy of multi-class classification.

1.5 3d

As the training accuracy is not higher than the accuracy compared to the validation dataset, there does not appear to be any obvious overfitting seen from this plot.

4a

$$J(w) = C(w) + \lambda R(w \tag{38}$$

$$C(w) = \frac{1}{N}\sum_{n=1}^{N} C^n(w) \tag{39}$$

$$C^n(w) = -\sum_{k=1}^{K} y_k^n \ln(\hat{y}_k^n) \tag{40}$$

$$R(w) = ||w||^2 = \frac{1}{2}\sum_{i,j} w_{i,j}^2 \tag{41}$$

task is to find $\frac{\delta J}{\delta w}$.

$$\frac{\partial J}{\delta w} = \frac{\partial C(w)}{\partial w} + \lambda \frac{\partial R(w)}{\partial w} \tag{42}$$

from task 1 we have that

$$\frac{\partial C(w)}{\partial w} = \frac{1}{N}\sum_{n=1}^{N} \frac{\partial C^n(w)}{\partial w} \tag{43}$$

Differentiating $R(w)$

$$\lambda \frac{\partial R(w)}{\partial w} = \lambda \frac{\partial ||w||^2}{\partial w} = \lambda \frac{\partial \frac{1}{2}\sum_{i,j} w_{i,j}^2}{\partial w} = \lambda \frac{1}{2} \frac{\partial \sum_{i,j} w_{i,j}^2}{\partial w} \tag{44}$$

If we see the $\frac{\partial \sum_{i,j} w_{i,j}^2}{\partial w}$ as $\frac{\partial \sum_{i,j} w_{i,j}^2}{\partial w_{i',j'}}$. With the same logic as in task 1, we only get a non-zero answer when $i' = i$ and $j' = j$, therefor we get

$$\lambda \frac{1}{2} \frac{\partial \sum_{i,j} w_{i,j}^2}{\partial w} = \lambda \frac{1}{2} 2 w_{i',j'} = \lambda w. \tag{45}$$

4b

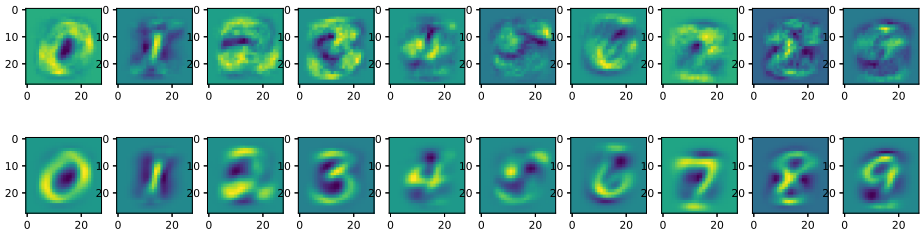

Figure 6: Final weights with models using $\lambda = 0$ on top and $\lambda = 2$ on bottom row respectively.

When comparing top and bottom row in Figure 7 we see that it becomes less noisy when applying regularization. This comes as the weight matrices become limited in how complex they can be by punishing this when training.

4c

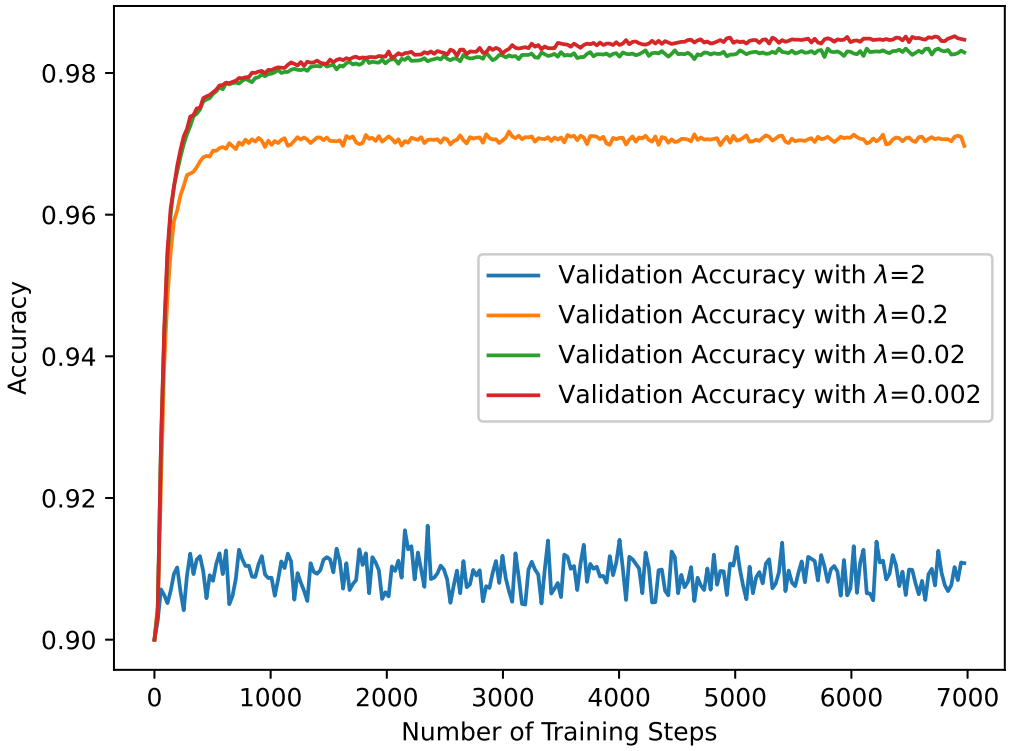

Figure 7: Validation accuracy for different λ values

4d

The reason for lower validation accuracy with regularization can be that it is not able/allowed to fit to the data as well as it could without. In practice with a larger more varied validation set, the accuracy would probably be more favorable with regularization.

This could also come by that the model overfit more when we have regularization. As seen in the plotted weights, the one that have gone through a regularization is less noisy. This could indicate that it will just remember the training set and not generalize. The reason for this could be that our model is really simple.

4e

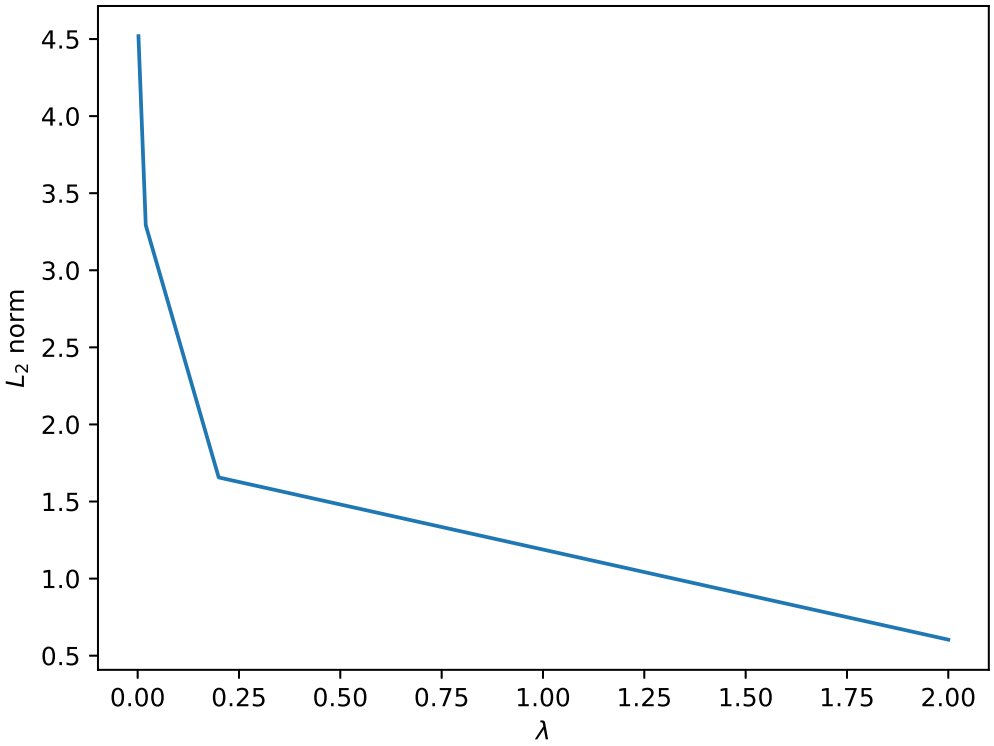

Figure 8: Task 4e: λ against the L_2 norm.

From Figure 8 we see that the L_2 norm decreases with higher λ values. This makes sense as it means that the weights become more limited in size and complexity, which is what we wanted by applying regularization.