TDT4265: Computer Vision

Assignment 1

Anders Kjelsrud

January 2023

Task 1

a) Have the binary cross entropy loss

$$C(w) = \frac{1}{N}\sum_{1}^{N} C^n \quad , \text{ where}$$

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

where $\hat{y}$ is the output and y is the ground truth.

Want $\frac{\partial C^n(w)}{\partial w_i}$, the gradient of the cost function

Using the chain rule,

$$\frac{\partial C^n(w)}{\partial w_i} = \underbrace{\frac{\partial C^n(w)}{\partial y_i}}_{1} \cdot \underbrace{\frac{\partial \hat{y}_i}{\partial z_i}}_{2} \cdot \underbrace{\frac{\partial z_i}{\partial w_i}}_{3}$$

or with vector notation:

$$\frac{\partial C}{\partial \underline{w}} = \frac{\partial C(\underline{w})}{\partial \underline{\hat{y}}} \frac{\partial \underline{\hat{y}}}{\partial \underline{z}} \frac{\partial \underline{z}}{\partial \underline{w}}$$

$$\frac{\partial C}{\partial \hat{y}} = -\left(\frac{y}{\hat{y}} - \frac{1-y}{1-\hat{y}}\right) = \underbrace{\frac{\hat{y}-y}{\hat{y}(1-\hat{y})}}_{1}$$

Using that $z = w^T x$

and the hint $\frac{\partial f^n}{\partial w_i} = x_i^n f(x^n)(1 - f(x^n))$

we can combine the last two terms from the chain rule, namely that

$$\frac{\partial f(x^n)}{\partial w_i} = ① \cdot ② = \frac{\partial \hat{y}_i}{\partial z_i} \frac{\partial z_i}{\partial w_i}$$

$$= x_i^n \hat{y} (1 - \hat{y})$$

Combining all 3 terms gives

$$\frac{\partial c}{\partial w} = \frac{\hat{y} - y}{\cancel{\hat{y}(1-\hat{y})}} \cdot x \cancel{\hat{y}(1-\hat{y})}$$

$$= x(\hat{y} - y) = -x(y - \hat{y}).$$

$$\Rightarrow \frac{\partial c^n(w)}{\partial w_i} = -(y^n - \hat{y}^n) x_i^n$$

$\square$

b)

Have the softmax activation function

$$\hat{y}_k = \frac{e^{z_k}}{\sum^{K} e^{z_k}} \quad , \quad z = w_k^T \cdot x$$

and the cross-entropy loss for multiple classes

$$C(w) = \frac{1}{N} \sum_1^N C^n(w) \quad , \quad C^n(w) = -\sum_1^K y_k^n \ln(\hat{y}_k^n)$$

Want $\dfrac{\partial C^n(w)}{\partial w_{kj}} = \dfrac{\partial C^n}{\partial z_k} \dfrac{\partial z_k}{\partial w_{kj}}$

Start with $\dfrac{\partial}{\partial z_j} \hat{y}_k = \dfrac{\partial}{\partial z_j} \dfrac{e^{z_k}}{\sum_1^K e^{z_k}}$

Use quotient rule; for $f(x) = \dfrac{g(x)}{h(x)}$

$$f'(x) = \frac{g'(x)\,h(x) - h'(x)\,g(x)}{h(x)^2} .$$

Choose $g(x) \triangleq e^{z_k}$ and $h(x) \triangleq \sum_1^K e^{z_k}$

Two cases:

if $k=j$: $\frac{\partial}{\partial z_j} \frac{e^{z_k}}{\sum_{k=1}^{K} e^{z_k}} = \frac{e^{z_k} \sum_{k=1}^{K} e^{z_k} - e^{z_j} e^{z_k}}{\left(\sum_{k=1}^{K} e^{z_k} \right)^2}$

$$= \frac{e^{z_k} \left(\sum_{k=1}^{K} e^{z_k} - e^{z_j} \right)}{\left(\sum_{k=1}^{K} e^{z_k} \right)^2} = \frac{e^{z_k}}{\sum_{k=1}^{K} e^{z_k}} \cdot \frac{\left(\sum_{k=1}^{K} e^{z_k} - e^{z_j} \right)}{\sum_{k=1}^{K} e^{z_k}}$$

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

if $k \neq j$: Here we have $g(x) = 0$

$$\Rightarrow \frac{\partial}{\partial z_j} \frac{e^{z_k}}{\sum_{k=1}^{K} e^{z_k}} = \frac{0 - e^{z_j} e^{z_k}}{\left(\sum_{k=1}^{K} e^{z_k} \right)^2}$$

$$= \frac{-e^{z_j}}{\sum_{k=1}^{K} e^{z_k}} \cdot \frac{e^{z_k}}{\sum_{k=1}^{K} e^{z_k}} = \underline{-\hat{y}_j \hat{y}_k}$$

$$\Rightarrow \frac{\partial \hat{y}_k}{\partial z_j} = \begin{cases} \hat{y}_k(1-\hat{y}_j) & \text{for } k=j \\ -\hat{y}_j \hat{y}_k & \text{for } k \neq j \end{cases}$$

Now for the multiclass cross-entropy loss:

$$c^n(\omega) = -\sum_1^K y_k^n \ln(\hat{y}_k^n)$$

Look at one case of n:

$$\Rightarrow \frac{\partial c}{\partial z} = \frac{\partial c}{\partial \hat{y}_k} \frac{\partial \hat{y}_k}{\partial z_k}$$

$$= -\sum_{k=1}^K y_k \frac{\partial}{\partial z} \ln(\hat{y}_k) = -\sum_{k=1}^K y_k \frac{\partial}{\partial \hat{y}} \log(\hat{y}_k) \cdot \frac{\partial \hat{y}_k}{\partial z_j}$$

$$= -\sum_{k=1}^K y_k \frac{1}{\hat{y}_k} \cdot \frac{\partial \hat{y}_k}{\partial z_j}$$

Using the result of softmax derivative:

$$\frac{\partial c}{\partial z_k} = -y_k(1-\hat{y}_k) - \sum_{i \neq k}^K y_i \frac{1}{\hat{y}_i}(-\hat{y}_i \cdot \hat{y}_k)$$

$$= -y_k + y_k \hat{y}_k + \sum_{i \neq 1}^K y_i \hat{y}_k$$

$$= \hat{y}_k \left(y_k + \sum_{i \neq 1}^{k} y_i \right) - y_k$$

using that $\sum_{i}^{k} y_i = 1$ and $y_k + \sum_{i \neq 1}^{k} y_i = 1$

$$\Rightarrow \underline{\frac{\partial c}{\partial z_k} = \hat{y}_k - y_k}$$

To get $\frac{\partial c}{\partial w_{kj}}$ we use the chain rule

$$\frac{\partial c}{\partial w_{kj}} = \frac{\partial c}{\partial z_k} \frac{\partial z_k}{\partial w_j} = \left(\hat{y}_k - y_k\right) x_j$$

$z = w^T x$

$\frac{\partial z}{\partial w} = x$

$$= -x_j \left(y_k - \hat{y}_k \right)$$

or for each datasample n

$$\underline{\underline{\frac{\partial c^n(w)}{\partial w_{kj}} = -x_j^n \left(y_k^n - \hat{y}_k^n \right)}}$$

$\square$

1 Task 2

b) Training and validation loss over training is shown in Figure 1.

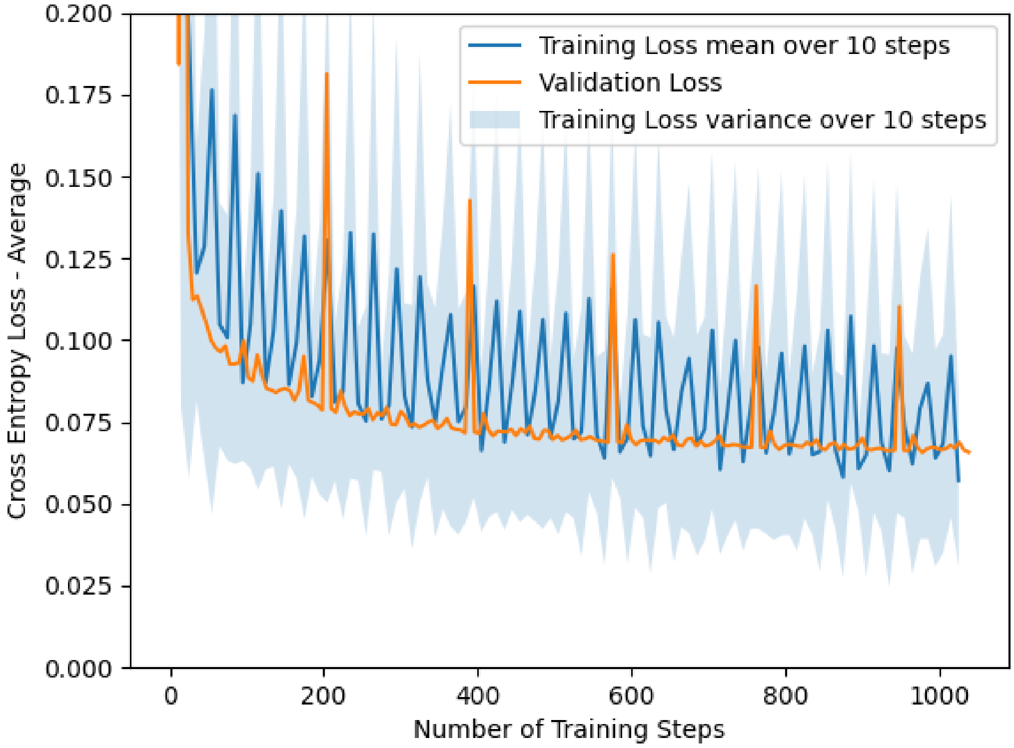

Figure 1: Loss

c) Accuracy is shown in Figure 2.

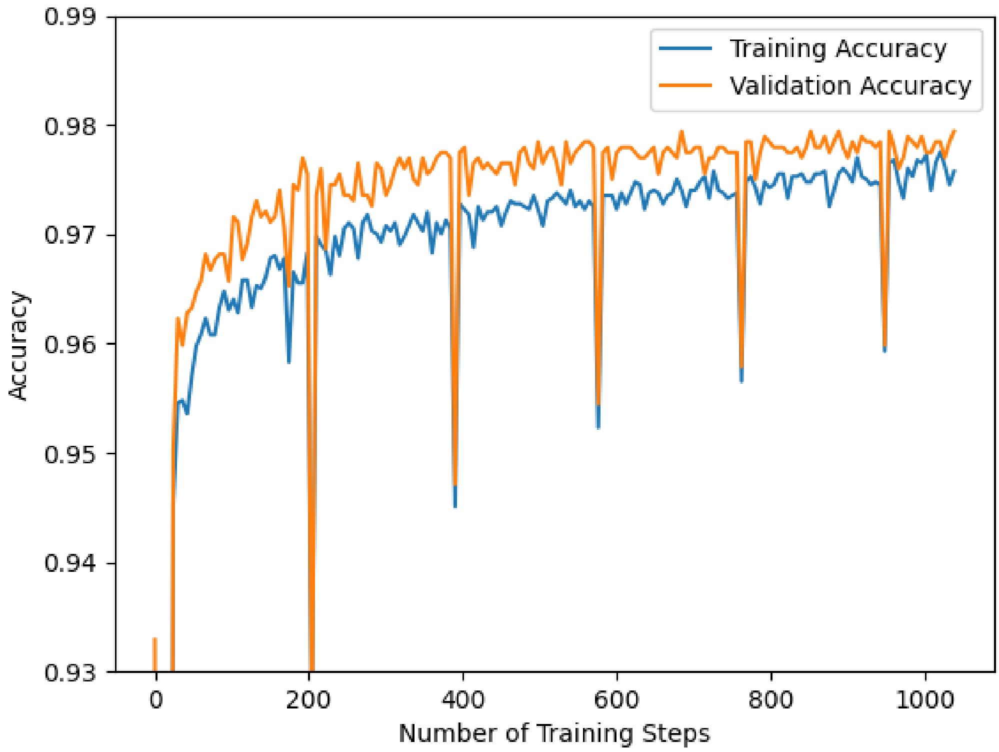

Figure 2: Accuracy

d) Early stopping kicks in after 33 epochs.

e) It happens because shuffling makes it less likely that the validation set will contain only examples from the same class. Thus the model's performance on the validation set will be more representative of how it would perform on new data. The opposite would be overfitting to the validation set. The difference in validation accuracy is shown in Figure 3. It is observed that early stopping kicks in way earlier.

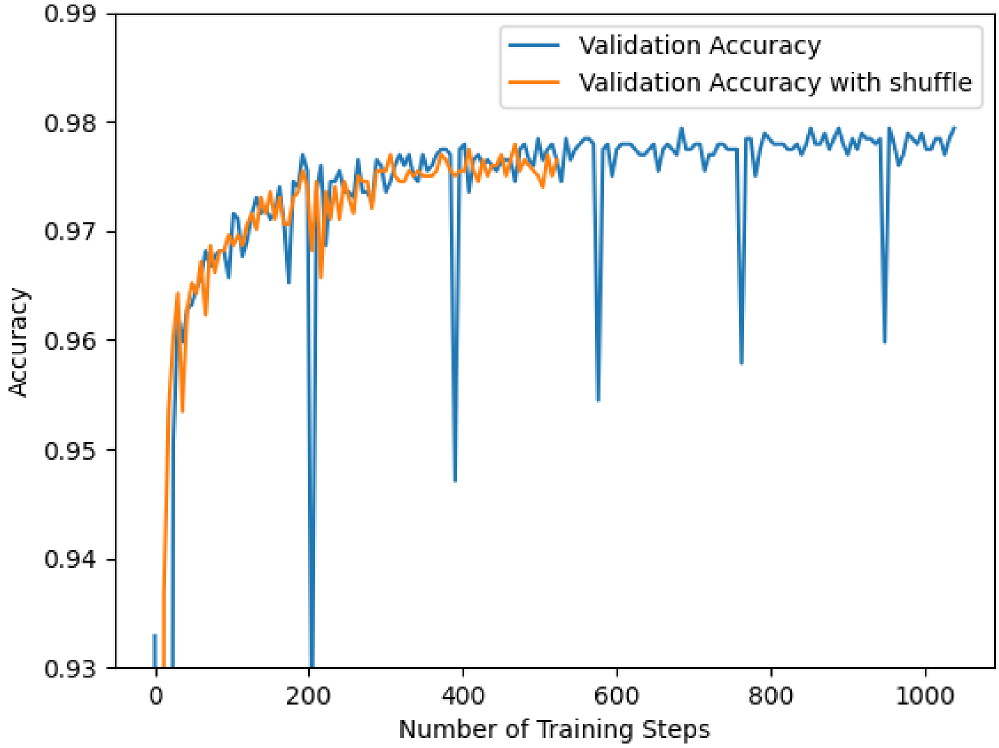

Figure 3: Shuffling

2 Task 3

b) Figure 4 shows the training and validation loss for Softmax.

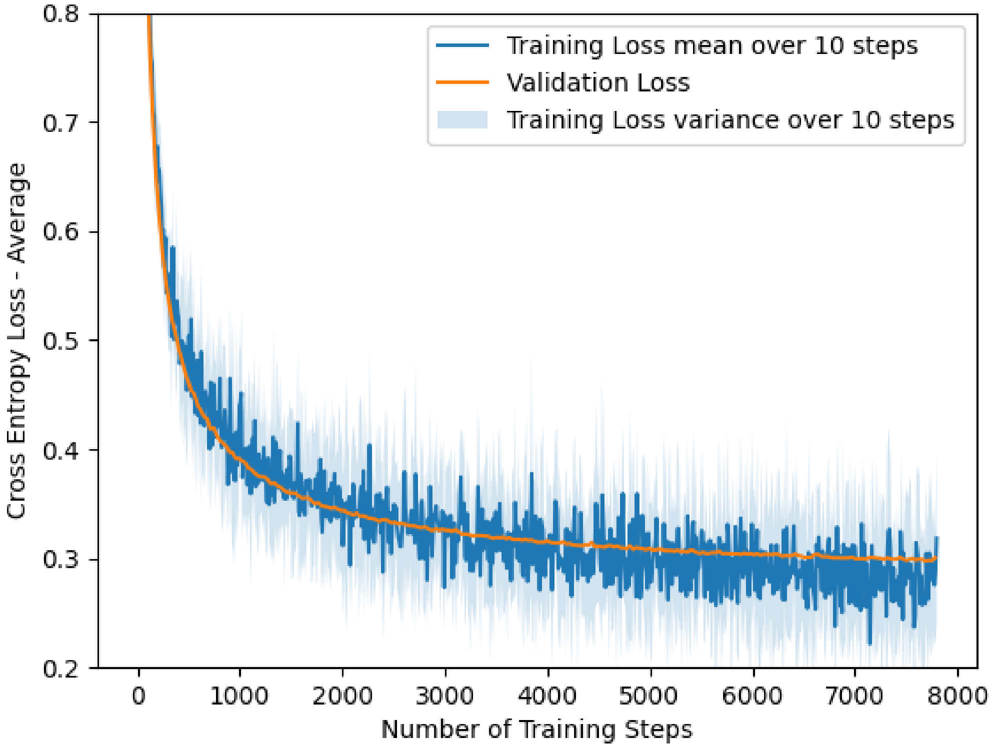

Figure 4: Softmax loss

c) Figure 5 shows the multi-class classification accuracy.

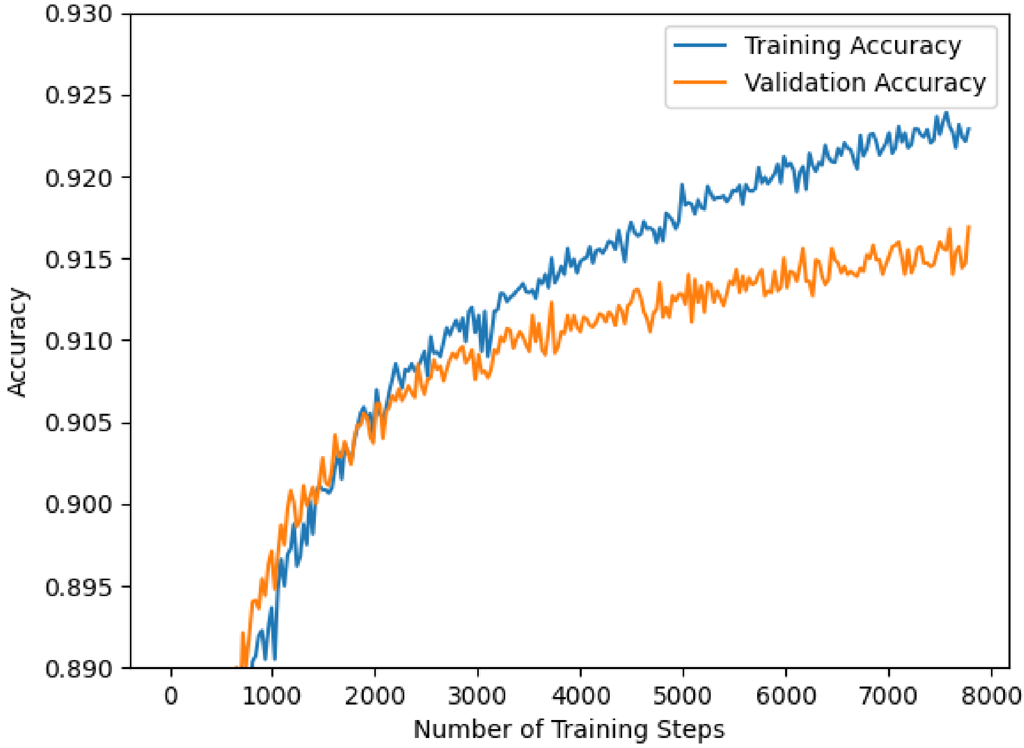

Figure 5: Softmax accuracy

d) There are signs of overfitting, namely that the accuracy is much higher on the training set than it is on the validation set. This is a sign that the model is unable to generalize well to new data, which is not very surprising considering the simple model we are using.

3 Task 4

a)

Task 4

Have that $\dfrac{\partial C^n(w)}{\partial w_{kj}} = -x_j^n (y_k^n - \hat{y}_k^n)$

Want $\dfrac{\partial J(w)}{\partial w}$ where $J(w) = C(w) + \lambda R(w)$

with L2-reg : $R = \frac{1}{2} \sum_{i,j} w_{ij}^2$

$\Rightarrow J(w) = C(w) + \lambda \frac{1}{2} \sum_{i,j} w_{ij}^2$

$\Rightarrow \boxed{\dfrac{\partial J}{\partial w} = -x_j (y_k - \hat{y}_k) + \lambda w}$

b) The model with regularization has less noisy weights because the regularization term penalizes large weights and therefore the model is encouraged to find weights that are smaller in magnitude which could mean that the weights are less noisy. This is because smaller weights tend to produce less change in the model's output for a given input, which can make the model's predictions more stable and less sensitive to small variations in the input. Figure 6 shows the weights for two models with different λ-values. The first has $\lambda = 0$ (no penalty), the second has $\lambda = 1$.

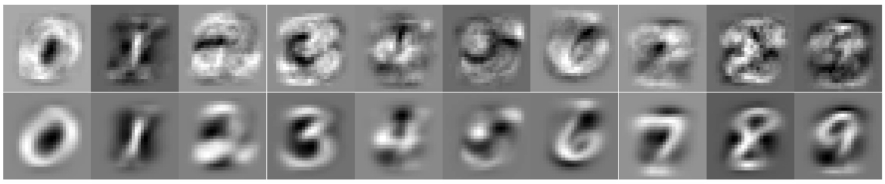

Figure 6: Model weights

c) Validation accuracy for different λ is shown in Figure 7.

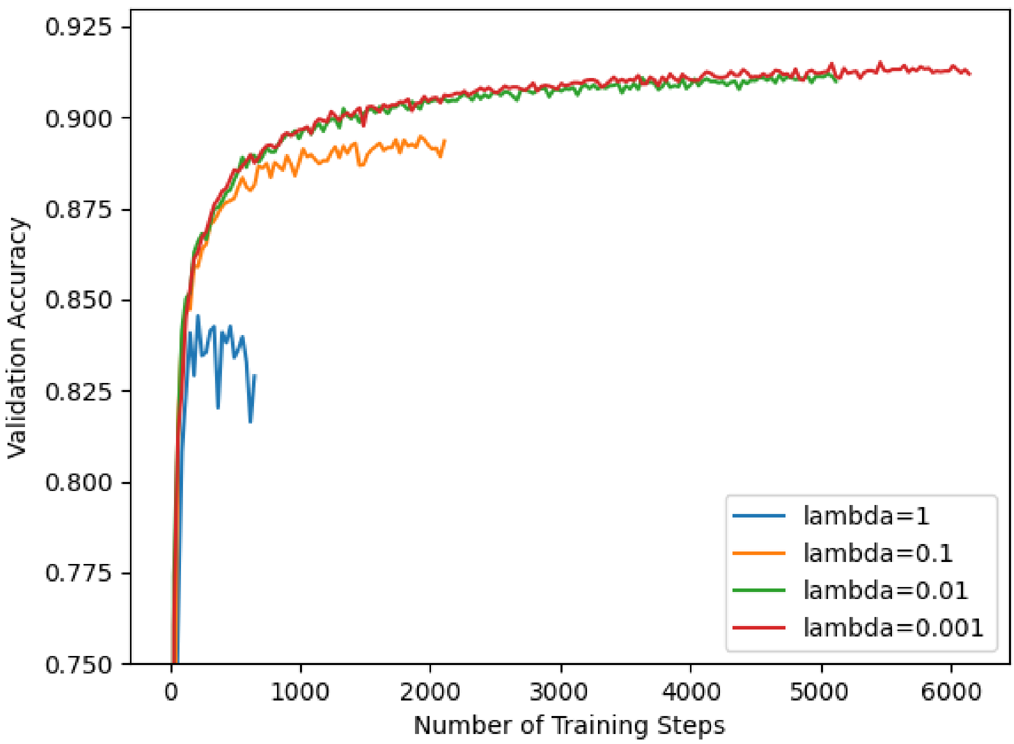

Figure 7: Accuracy for different λ

d) Because a single layer NN is already quite simple and probably not able to generalize very well, adding a further restriction will make it even harder. A model this simple needs all the wiggle room it can get in terms of finding good weights to be able to learn a good representation of the data, therefore adding large regularization will in this case be too restrictive.

e) We can see that the magnitude of the weights decrease as the regularization term increases.

This makes sense, because regularization is supposed to reduce model complexity by penalizing large weights. The magnitude of the weights are shown in Figure 8.

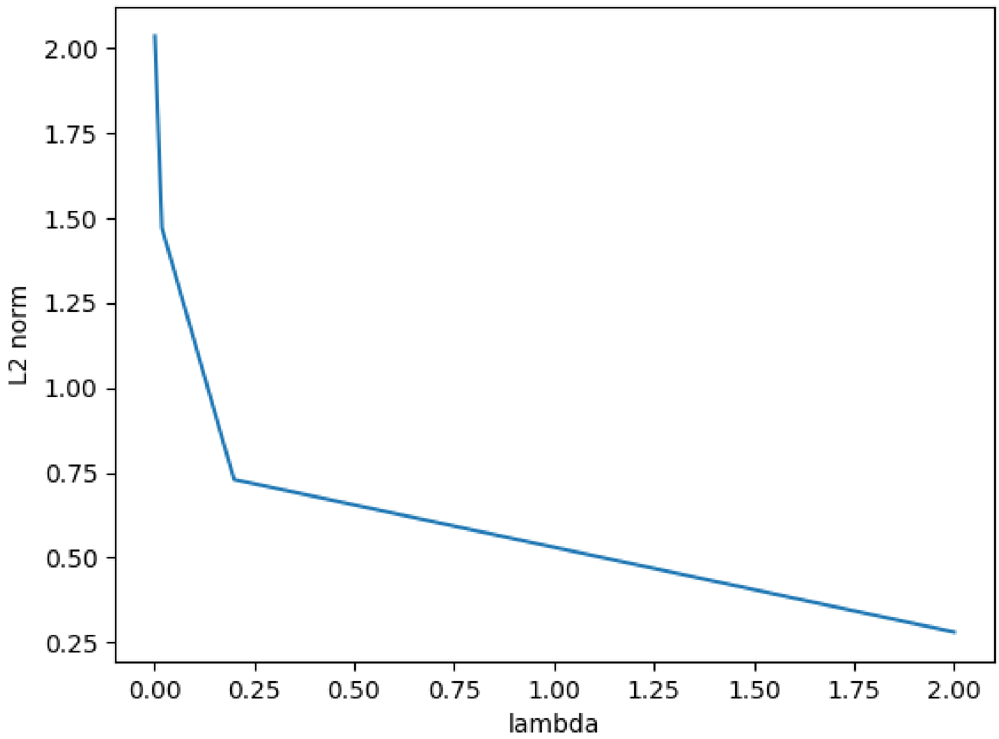

Figure 8: $\|w\|^2$