TDT4265 Assignment 1

Hallvard Stemshaug

January 2021

Task 1: Theory

a)

Throughout this task $f(x_i^n)$ will be written as $\hat{y}^n$

$$\begin{aligned}\frac{\partial C^n(w)}{\partial w_i} &= \frac{\partial C^n(w_i)}{\partial \hat{y}^n}\frac{\partial \hat{y}^n}{\partial w_i} \\ &= -\frac{\partial}{\partial \hat{y}^n}\big(y^n \ln(\hat{y}^n) + (1-y^n)\ln(1-\hat{y}^n)\big)\frac{\partial \hat{y}}{\partial w_i} \\ &= -\big(\frac{y^n}{\hat{y}^n} - \frac{(1-y^n)}{(1-\hat{y}^n)}\big)\frac{\partial \hat{y}}{\partial w_i}\end{aligned}$$

Using that $\frac{\partial \hat{y}^n}{\partial w_i} = x_i^n \hat{y}^n(1-\hat{y}^n)$ we get:

$$\begin{aligned}&= -\big(\frac{y^n}{\hat{y}^n} - \frac{(1-y^n)}{(1-\hat{y}^n)}\big)\big(x_i^n \hat{y}^n(1-\hat{y}^n)\big) \\ &= -x_i^n(y^n - \hat{y}^n y^n - \hat{y}^n + \hat{y}^n y^n) \\ &= \underline{\underline{-x_i^n(y^n - \hat{y}^n)}}\end{aligned}$$

b)

$$\begin{aligned}\frac{\partial C(w)}{\partial w_{kj}} &= -\frac{\partial}{\partial w_{kj}}\big(\sum_{k=1}^{K} y_k^n \ln(\hat{y}_k^n)\big) \\ &= -\frac{\partial}{\partial w_{kj}}\big(\sum_{k=1}^{K} y_k^n \ln(e^{z_k^n}) - \sum_{k=1}^{K} y_k^n \ln(\sum_{k'=k}^{K} e^{z_{k'}^n})\big) \\ &= -\frac{\partial}{\partial w_{kj}}\big(\sum_{k=1}^{K} y_k^n \ln(e^{z_k^n}) - \sum_{k=1}^{K} y_k^n \ln\big((\sum_{(k'\neq k^K} e^{z_{k'}^n}) + e^{z_k^n}\big) \\ &= -\big(x_{kj} y_k^n - x_{kj}\frac{e^{z_k^n}}{\sum_{k'=k}^{K} e^{z_{k'}^n}}\sum_{k=1}^{K} y_k^n\big) \\ &= \underline{\underline{-x_{kj}^n(y_k^n - \hat{y}_k^n)}}\end{aligned}$$

Task 2: Logistic Regression through Gradient Descent

b)

Final Train Cross Entropy Loss: 0.06509106
Final Validation Cross Entropy Loss: 0.06522831

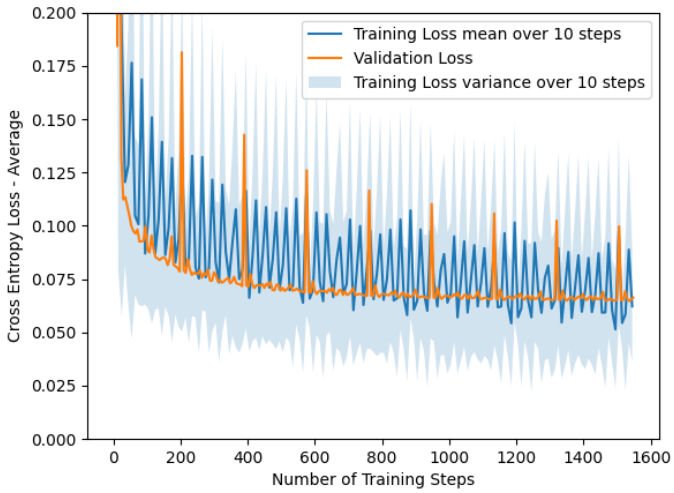

Figure 1: Binary training loss

c)

Train accuracy: 0.9792759051186017
Validation accuracy: 0.9789422135161606

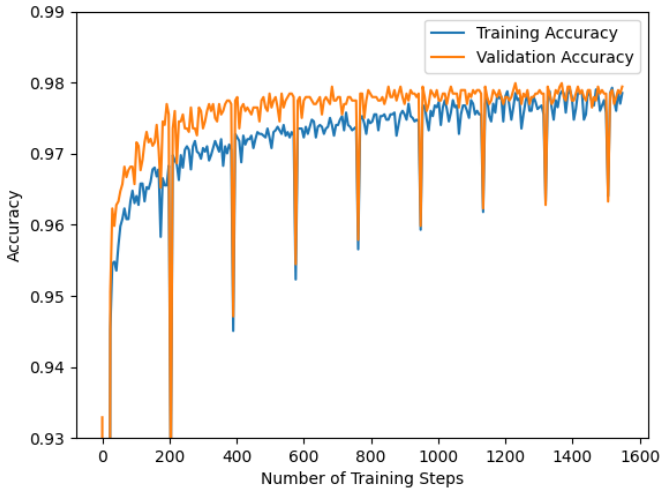

Figure 2: Binary training accuracy

d)

After implementing early stopping the training stopped at epoch 33, with the results:

Final Train Cross Entropy Loss: 0.07088706
Final Validation Cross Entropy Loss: 0.06577188
Train accuracy: 0.9757802746566792
Validation accuracy: 0.9794319294809011

e)

The number of epochs has been set back to 50 and early stopping has been disabled.

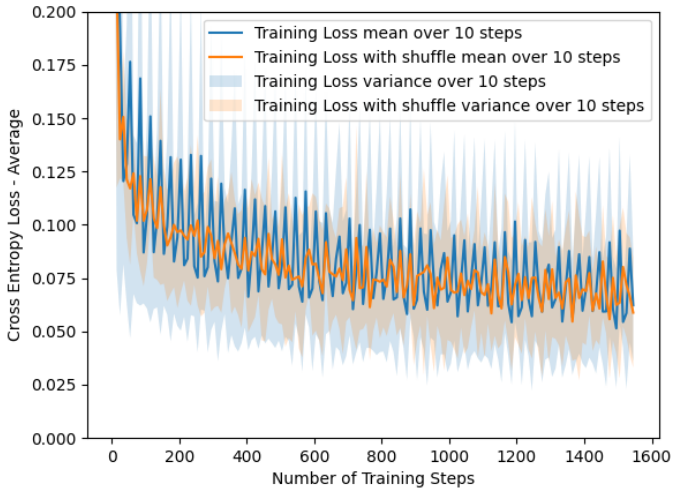

Figure 3: Binary training loss with shuffle

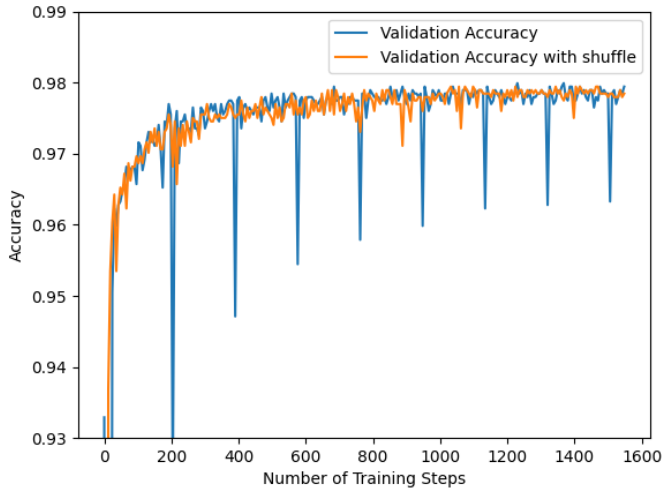

Figure 4: Binary training accuracy with shuffle

Shuffling the validation set leads a lower possibility that batches with hard to classify data gets repeated in every epoch. In the not shuffled data you can see a negative spike where the same set of data return in about every 200 steps.

Task 3: Softmax Regression through Gradient Descent

b)

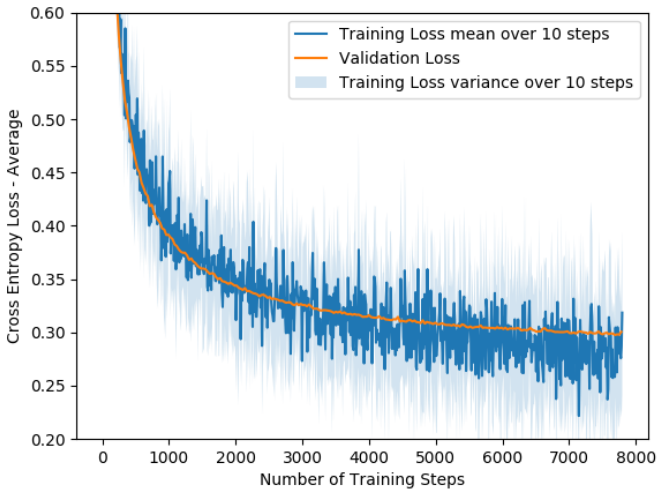

Figure 5: Softmax training loss

c)

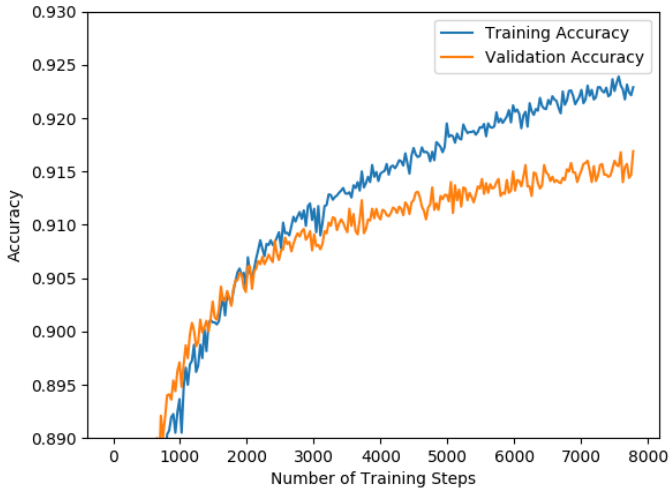

Figure 6: Softmax training accuracy

d)

The training accuracy is growing at an faster pace than the validation accuracy towards the end, this is a sign of overfitting, where the model gets fine-tuned to a small set that it's training on, but performs worse on the validation set that it doesn't train on.

Task 4: Regularization

a)

$$\frac{\partial J(w)}{\partial w_{kj}} = \frac{\partial C(w)}{\partial w_{kj}} + \frac{\partial R(w)}{\partial w_{kj}} \tag{1}$$

$$= \frac{\partial C(w)}{\partial w_{kj}} + \frac{\partial}{\partial w_{kj}} \lambda \sum_{k}^{K} w_{kj}^2 \tag{2}$$

$$= \frac{\partial C(w)}{\partial w_{kj}} + 2\lambda w_{kj} \tag{3}$$

This gives:

$$\frac{\partial J(w)}{\partial w_{i,j}} = \underline{\underline{-(y_{i,j} - \hat{y}_{i,j}) x_{kj} + 2\lambda w_{kj}}} \tag{4}$$

$$\tag{5}$$

b)

Figure 7: Comparison of the weights for a model trained without L2 Regularization (top) and with L2 Regularizarion with $\lambda = 1.0$ (bottom)

The regularization decreases the weights of the model, and removes some noise or variance. In the picture you can see that areas where it's not so common to write the different number less present. In this case this leads to the model not learning as well as it could, because there is a big variance in how numbers are represented. In more complex problems this would keep the model from putting too much weight on noise or irrelevant features.

c)

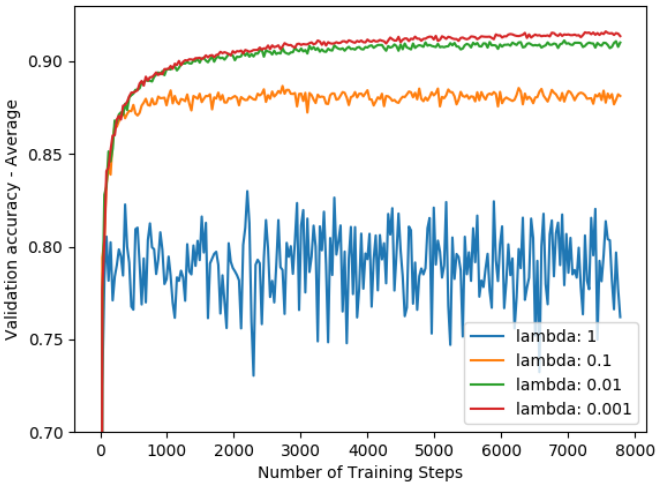

Figure 8: L2 regularization with different values of λ

d)

L2 regularization is used to prevent overfitting in complex problems with small data sets, where there are put to much focus on irrelevant details or noise. By decreasing the weights we are trying to stop the model from learning too much from outliers . This problem on the other hand is simple and we have a big dataset with not that much variety, the problem with overfitting is therefore not that present here, and instead the procedure stops the model from learning.

e)

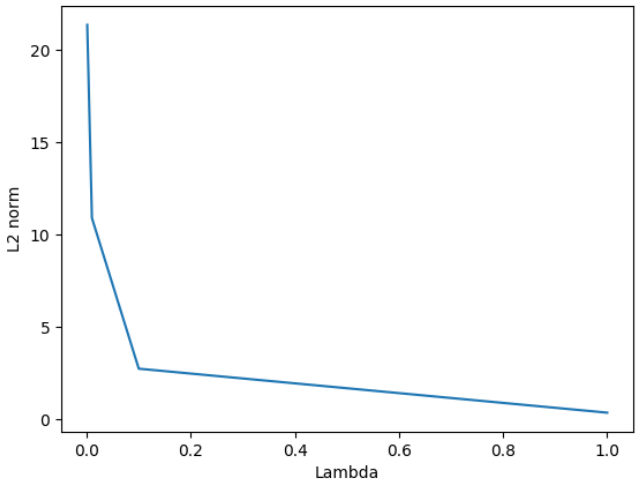

Figure 9: The norms of he weights after L2 regularization with different values of λ

The weight matrices where λ is low and the regularization has little impact on the weight matrix has larger norms, while the matrices with high λ has a lower norm, meaning that the values are smaller. This is to be expected with L2 regularization, which makes the larger weights have a smaller impact on the network.