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)

Solution:

Step 1: Chain rule $\frac{dz}{dx} = \frac{dz}{dy} \cdot \frac{dy}{dz}$ $\qquad h'(x) = f'(g(x)) \cdot g(x)$

$$\Rightarrow \frac{\partial C^n(w)}{\partial w_i} = \frac{\partial C^n(w)}{\partial f(x^n)} \cdot \frac{\partial f(x^n)}{\partial w_i} = \boxed{\frac{\partial C^n(w)}{\partial \hat{y}^n}} \cdot \boxed{\frac{\partial \hat{y}^n}{\partial w_i}}$$

Step 2: Solve $\frac{\partial C^n(w)}{\partial f(x^n)} = \frac{\partial C^n(w)}{\partial \hat{y}^n} = \frac{\partial}{\partial \hat{y}^n}\left(-y^n \ln(\hat{y}^n) + (1-y^n)\ln(1-\hat{y}^n)\right)$

$$= \boxed{\frac{-y^n}{\hat{y}^n} + \frac{1-y^n}{1-\hat{y}^n}}$$

Step 3: Apply hint $\Rightarrow \frac{\partial \hat{y}^n}{\partial w_i} = \frac{\partial f(x^n)}{\partial w_i} = x_i^n f(x^n)(1-f(x^n))$

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

Step 4: Combine $\Rightarrow \left(\frac{-y^n}{\hat{y}^n} + \frac{1-y^n}{1-\hat{y}^n}\right) \cdot \left(x_i^n \hat{y}^n (1-\hat{y}^n)\right)$

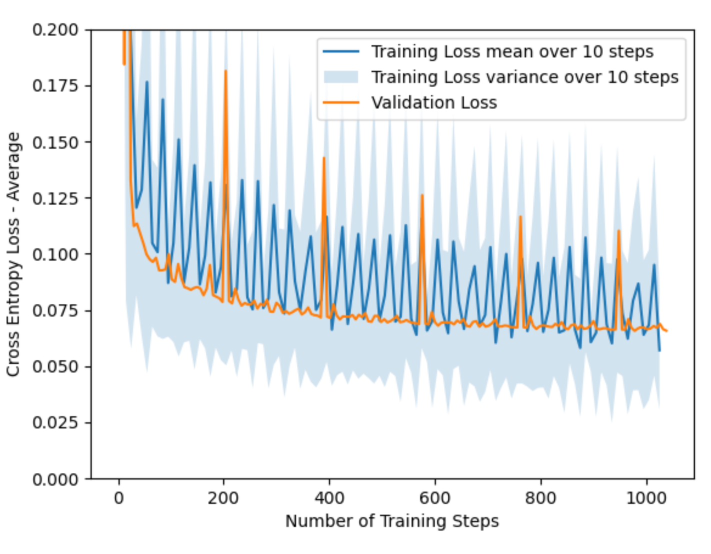
$$= x_i^n \left((1-y^n)\hat{y}^n - y^n(1-\hat{y}^n)\right)$$

$$= x_i^n \left(\hat{y}^n - y^n\hat{y}^n - y^n + y^n\hat{y}^n\right)$$

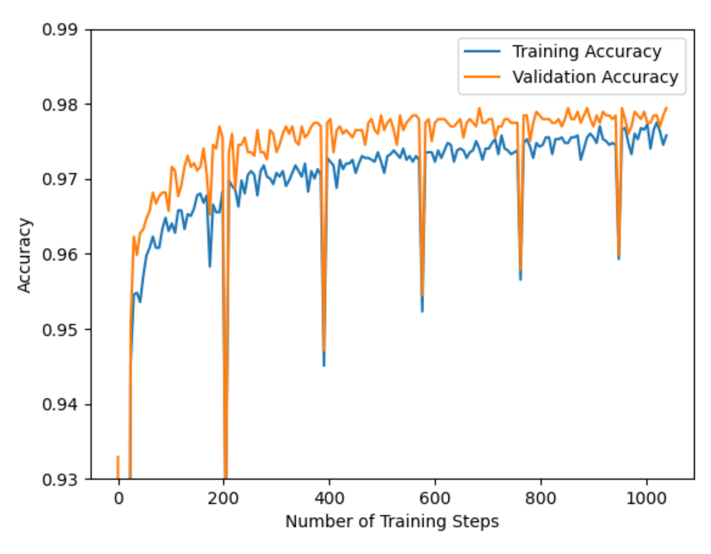
$$= \underline{\underline{x_i^n (\hat{y}^n - y^n)}}$$

Task 2

Task 2b)



Task 2c)



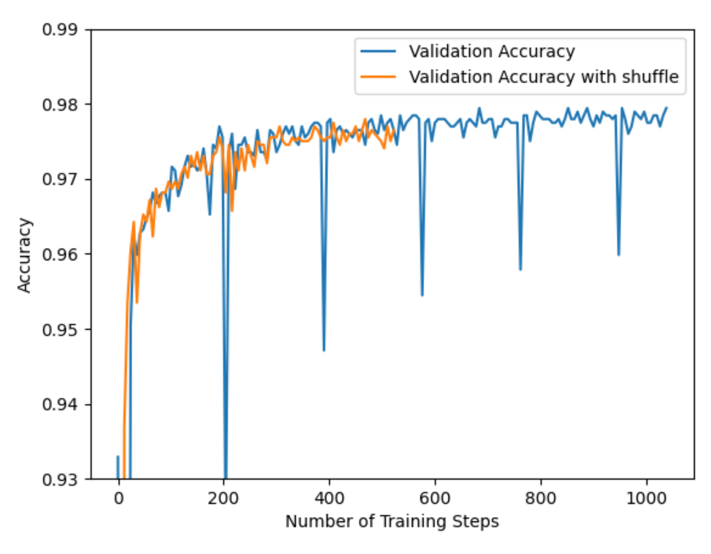
Task 2d)

Early stopping afther 34 epochs and 1039 global steps

```
❯ /Users/eirikvarnes/anaconda3/bin/python /Users/eirikvarnes/TDT
Train shape: X: (4005, 784), Y: (4005, 1)
Validation shape: X: (2042, 784), Y: (2042, 1)
Early stopping triggered after 34 epochs and 1039 global steps.
Final Train Cross Entropy Loss: 0.07088705555893647
Final Validation Cross Entropy Loss: 0.06577187753329566
Train accuracy: 0.9757802746566792
Validation accuracy: 0.9794319294809011
```

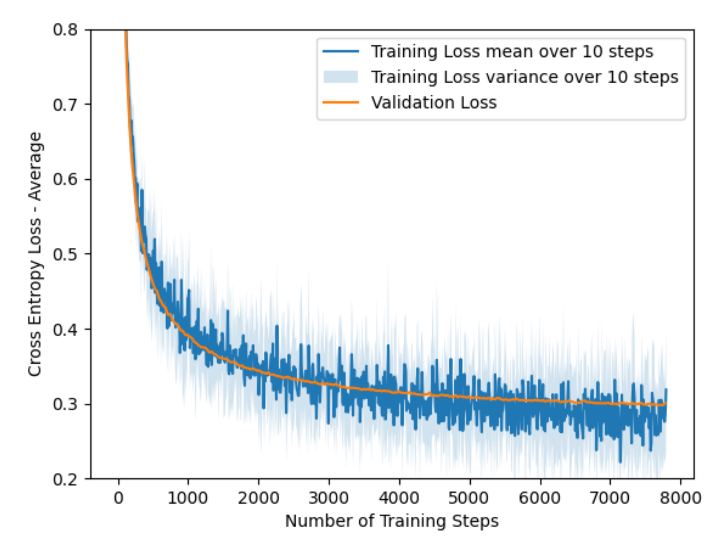
Task 2e)

The data's original structure, often featuring long sequences of identical numbers, can hinder model generalization and cause learning spikes with new sequences. Shuffling the data prior to training is recommended to ensure a more balanced distribution and improve generalization.

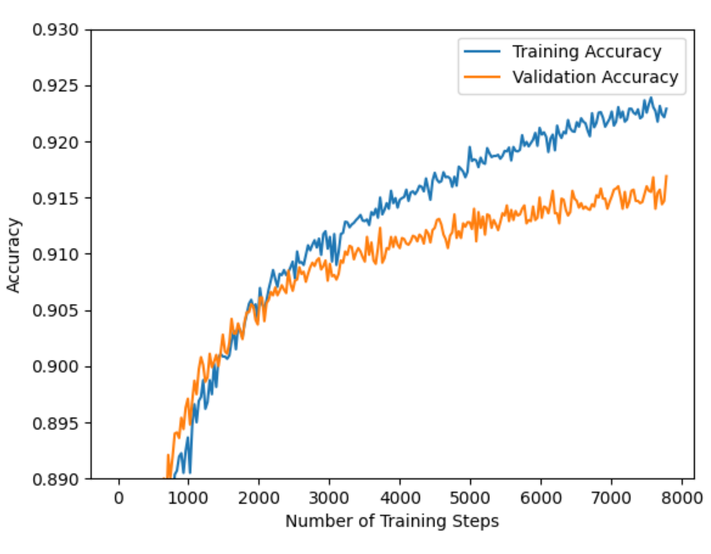


Task 3

Task 3b)



Task 3c)



Task 3d)

At 3000 steps, a divergence occurs between training and validation accuracy: training accuracy keeps improving, while validation accuracy appears to converge. This likely indicates overfitting, as the model starts learning dataset noise rather than underlying patterns.

Task 4

Task 4a)

Task 4

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

L_2 regularization term $R(w) = \|w\|^2 = \sum_{i,j} w_{i,j}^2 = \sum_{i=1}^{D}\sum_{j=1}^{K} w_{i,j}^2$

D = number of features , K = number of classes , regularization strength λ

Total regularization cost function : $J(w) = C(w) + \lambda R(w)$

Gradient of $J(w)$: $\frac{\partial J(w)}{\partial w} = \frac{\partial C(w)}{\partial w} + \lambda \frac{\partial R(w)}{\partial w}$

$\frac{\partial C(w)}{\partial w}$ will be the same as in Task 1.1.

$$\frac{\partial C(w)}{\partial w} = \frac{1}{N}\sum_{n=1}^{N}\left(-x_i^n (y^n - \hat{y}^n)\right)$$

Intuitivly this makes sense. $\frac{\partial C^n}{\partial w}$ tells you how the cost changes as you change each weight. If you increase a weight a tiny bit and the cost increases, gradiant is positive, and visa versa.

The cost $-\sum_{k=1}^{K} y_k^n \ln(\hat{y}_k^n)$ $\quad 0 \leq \hat{y}_k^n \leq 1$ $\quad \ln(x) \begin{cases} 0 & x = 1 \\ 0 < & x < 1 \end{cases}$

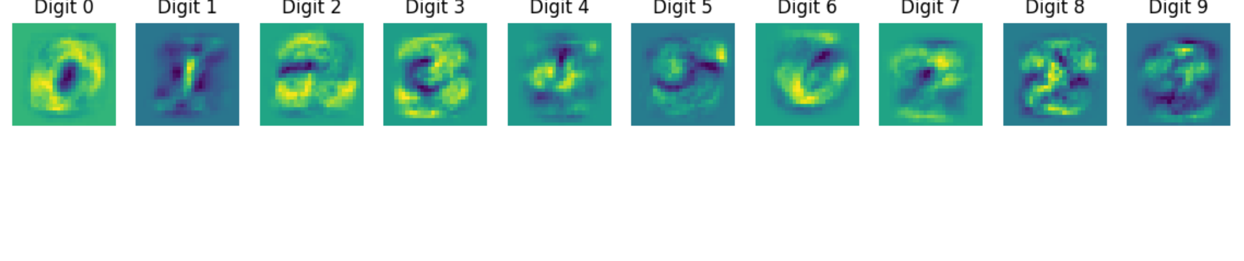
Every example n only have 1 correct K where $y_k^n = 1$. It is then clear that the larger $\hat{y}_k^n$ is when $y_k^n = 1$, the smaller is the cost. We choose the largest $\hat{y}_k^n$ for our guess So the rest dont matter. So the way w_{kj} affect the cost $C(w)$ is based on the output and target error $(y_k^n - \hat{y}_k^n)$ and the corrosponding node x_j^n connecting the output and weight.

$\frac{\partial R(w)}{\partial w_{jk}} = 2w_{jk}$, Standard derivation rules

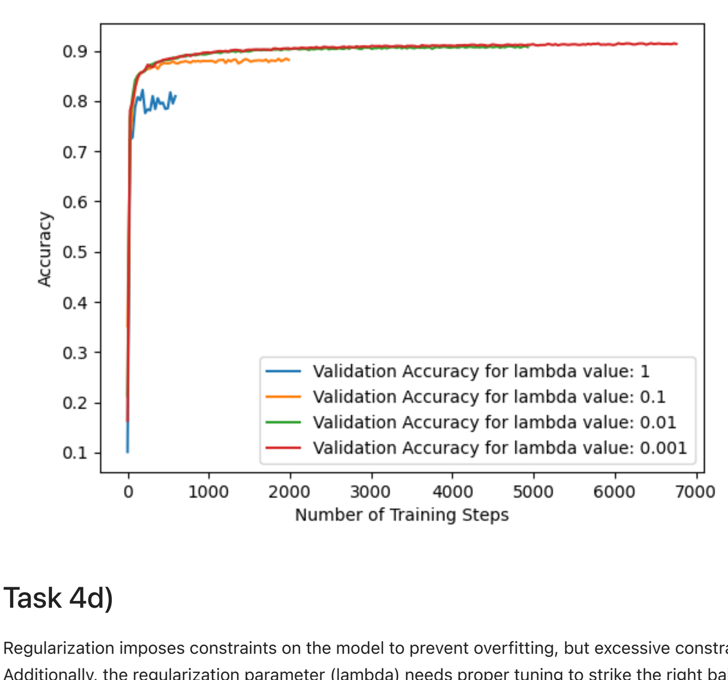
$$\Rightarrow \underline{\underline{\frac{\partial J(w)}{\partial w} = -\frac{1}{N}\sum_{n=1}^{N}\left(x_i^n (y^n + \hat{y}^n)\right) + 2\lambda w_{jk}}}$$

Task 4b)

Training two models with different L2 regularization values (`λ = 0.0` and `λ = 1.0`) and visualizing their weights reveals that the model with `λ = 1.0` has less noisy weights. This is because L2 regularization penalizes large weights, encouraging the model to maintain smaller, simpler weights, which reduces overfitting and results in a smoother, more generalized representation of the data.



Task 4c)



Task 4d)

Regularization imposes constraints on the model to prevent overfitting, but excessive constraints can lead to underfitting, resulting in poorer performance on both the training and validation datasets. Additionally, the regularization parameter (lambda) needs proper tuning to strike the right balance between model complexity and generalization performance. Failure to tune this parameter effectively can lead to suboptimal validation accuracy. Therefore, careful management of regularization strength and hyperparameter tuning is crucial for achieving optimal model performance.

Task 4e)

The plot illustrates an inverse relationship between the regularization parameter λ and the L2 norm of the weights. As λ increases, the L2 norm decreases, indicating that higher regularization leads to smaller weight vectors. This is a characteristic effect of L2 regularization, which aims to prevent overfitting by penalizing large weights. The most substantial decrease in the L2 norm is observed with increasing λ, especially transitioning from λ = 0.001 to λ = 1.0.

L2 Norm of Weights for different λ values