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.

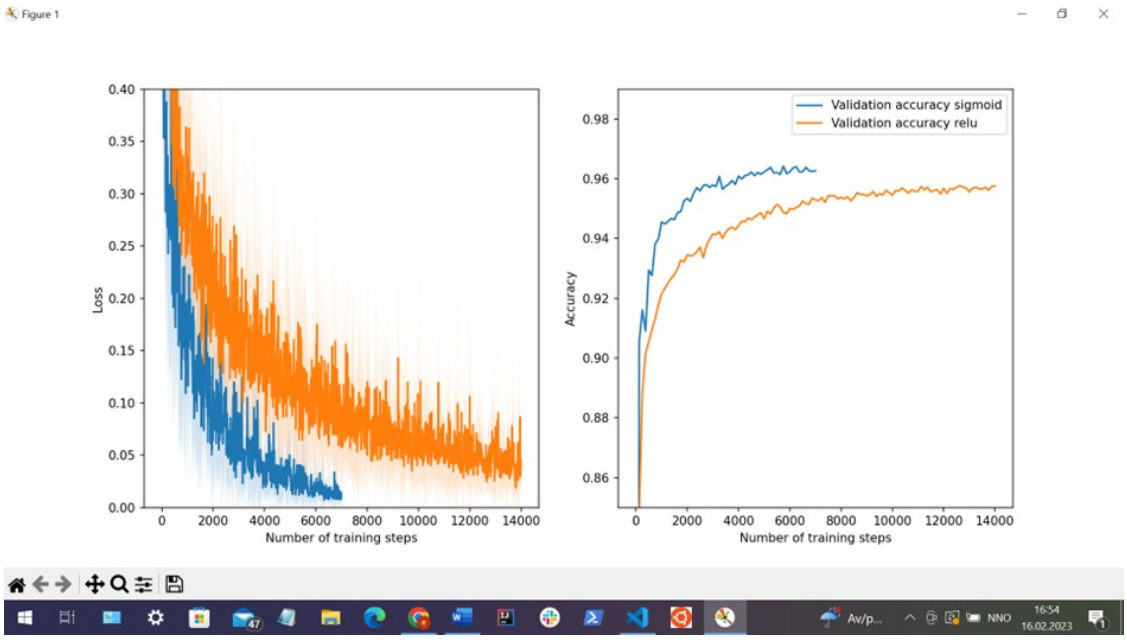
Before delivery, **remember to convert this file to PDF**. You can do it in two ways:

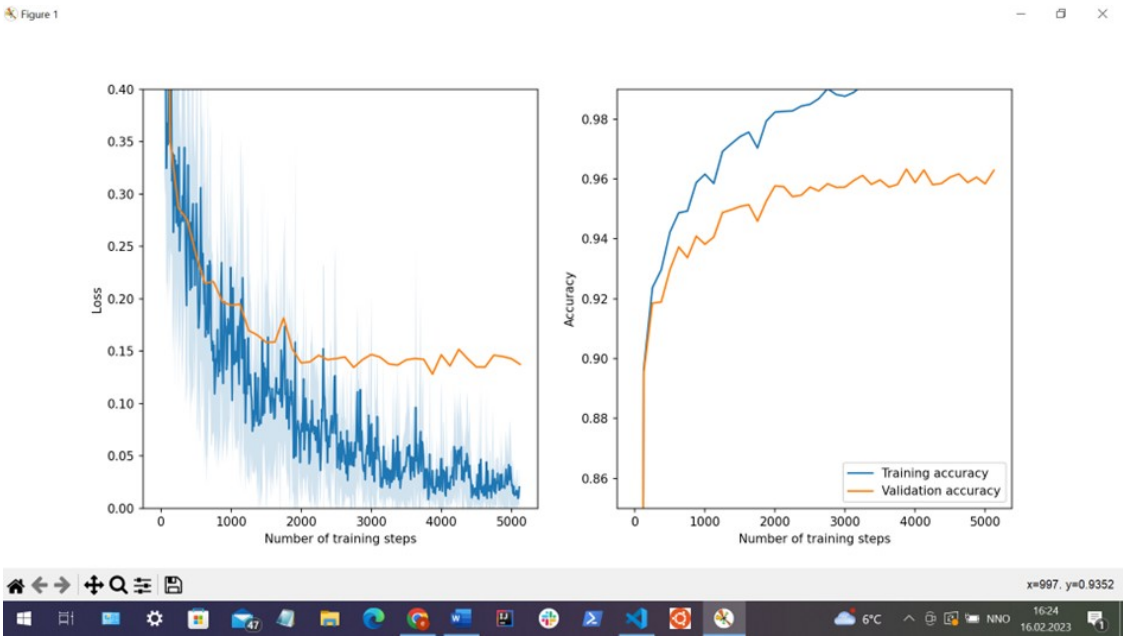
1. Print the webpage (ctrl+P or cmd+P)
2. Export with latex. This is somewhat more difficult, but you'll get somehwat of a "prettier" PDF. Go to File -> Download as -> PDF via LaTeX. You might have to install nbconvert and pandoc through conda; `conda install nbconvert pandoc`.

Task 1

task 1a)

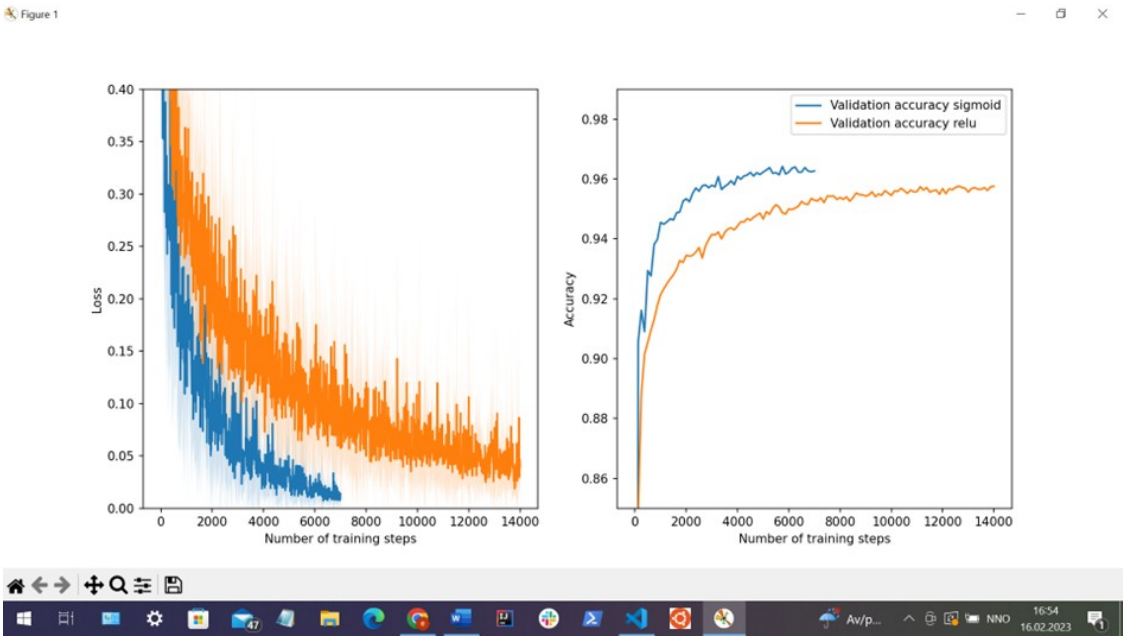
Fill in task 1a image of hand-written notes which are easy to read, or latex equations here





Task 2

Task 2a)



Task 2c)

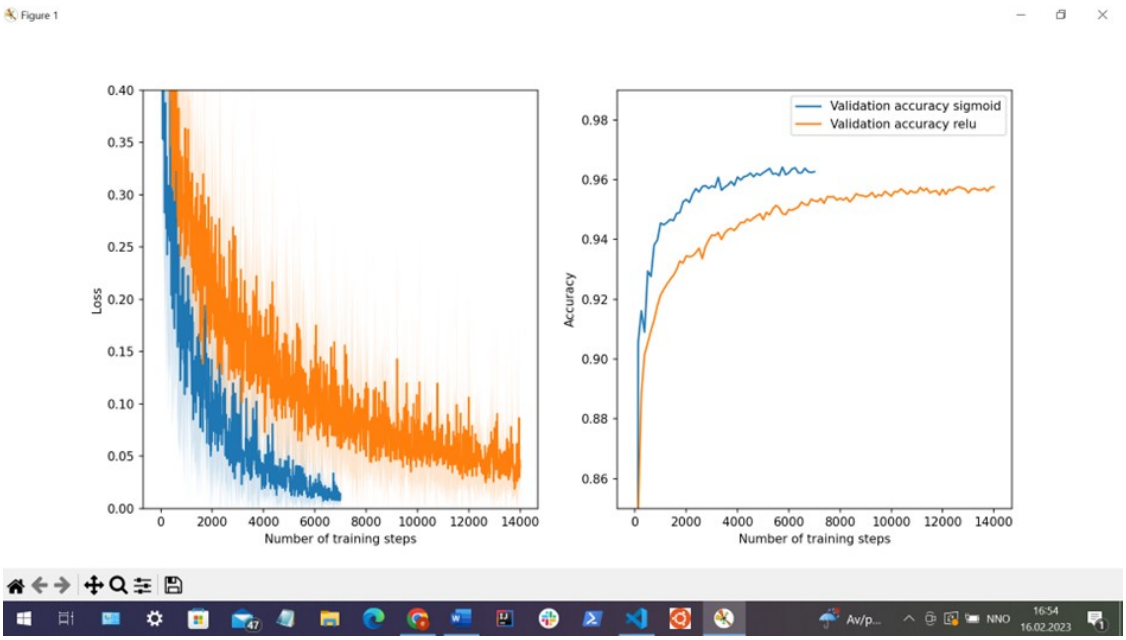
Task 2d)

How many parameters are there in the network defined in task 2c?

So for the number of parameters we add the total number of weights and biases together. The total number of biases are 64, one for every node in the hidden layer. Then we add the weights together which is 1 weight per connection, so for the input layer to the hidden layer there would be 784x64 weights and from the hidden layer to the output layer this would 64x10, so in total this adds up to 50880 total parameters.

Task 3

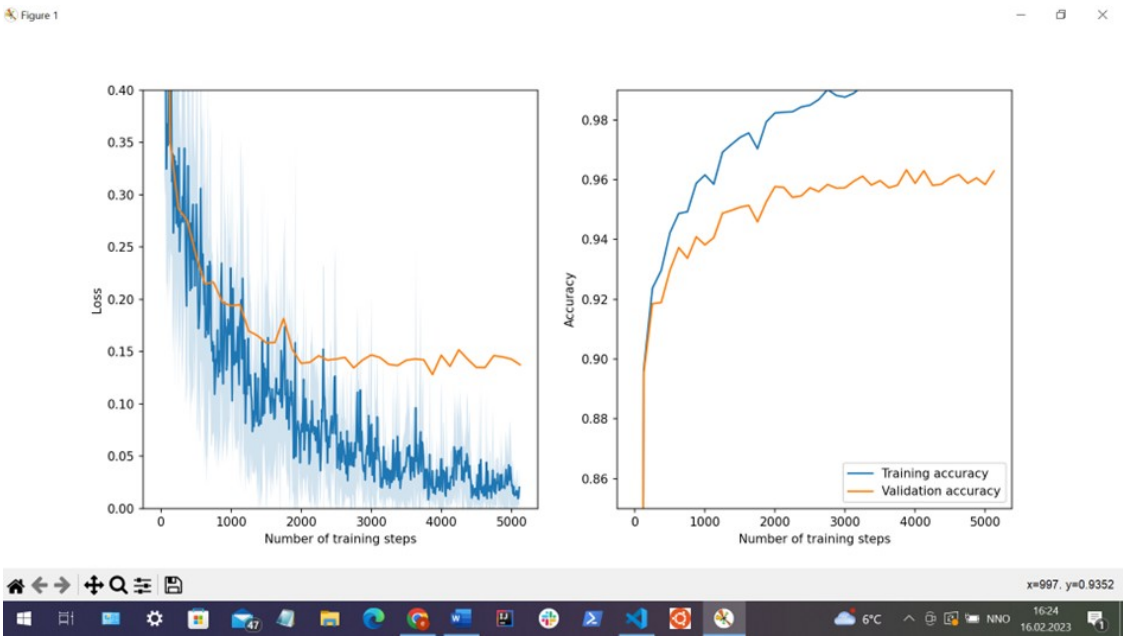
Task 3)



First of all we can see that with any of the additions the accuracy increases quite a lot. We can also see that after each addition the convergence speed increases both in validation accuracy but also the training loss. We also notice that the number of training steps used decrease with each addition which indicates that the model learns faster. We can also see that for the model with weights and the standard model, it requires many more training steps than the two other models. This could lead to overfitting, however, we see that the number of training steps decrease with every addition which saves tons of computation, but also is less likely to overfit. We also see that the validation accuracy is way higher with the "tricks of the trade" than the standard model without and the training loss also drops significantly with each addition.

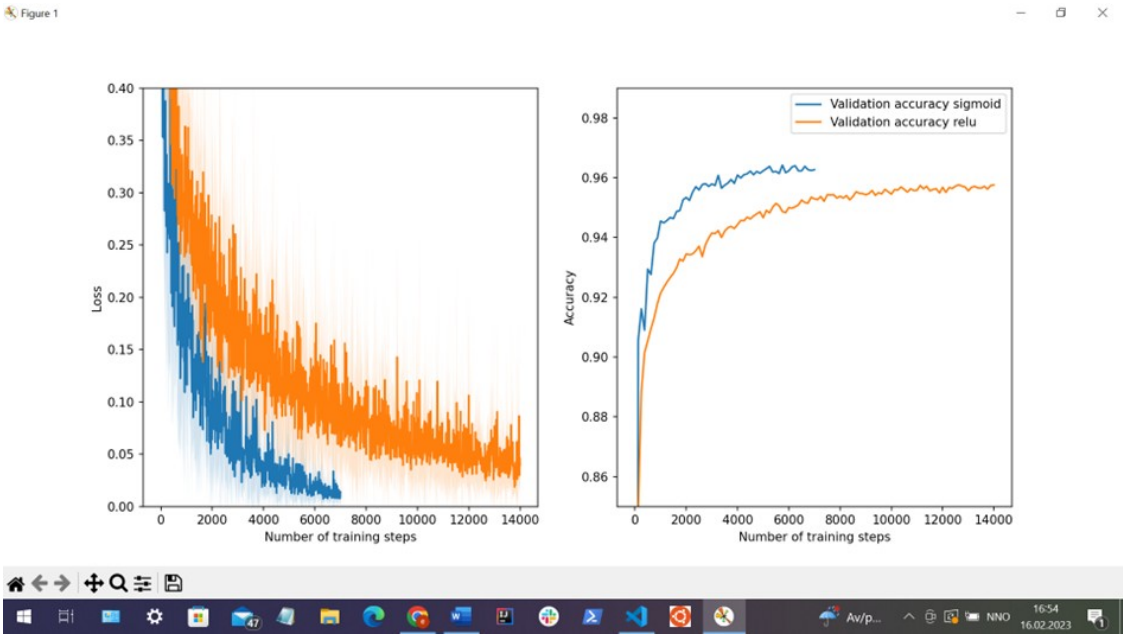
Task 4

Task 4a)



Here we see that the convergence speed is slightly slower with 32 units compared to 64, and it doesn't reach the same accuracy.

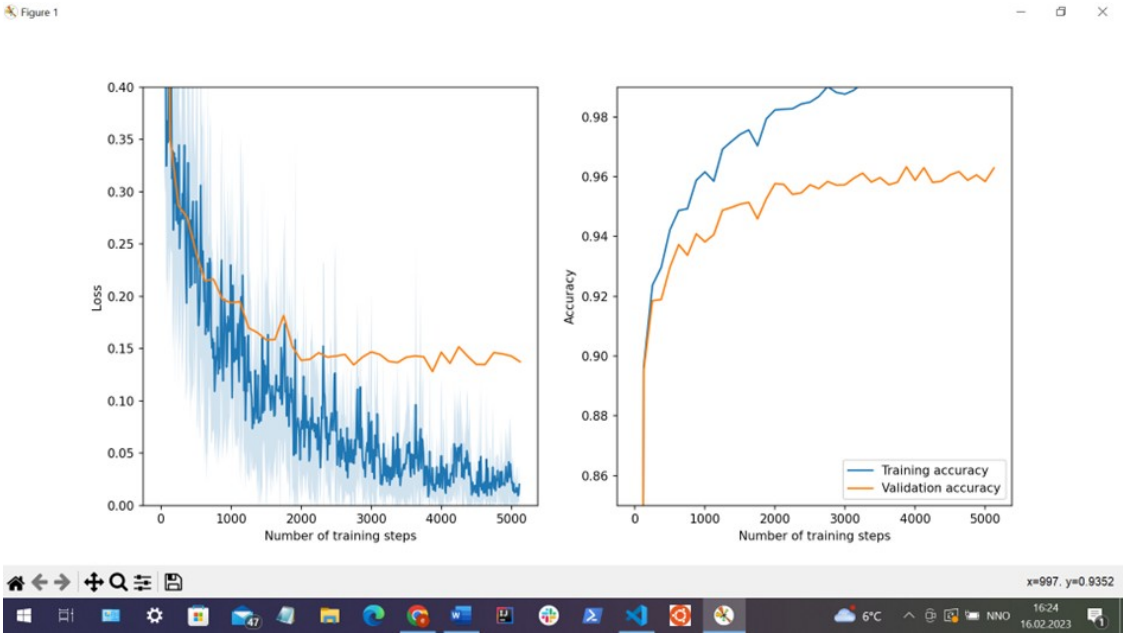
Task 4b)



We see with this one that it converges about the same speed, just slightly faster. And that the accuracy doesnt increase significantly.

Task 4d)

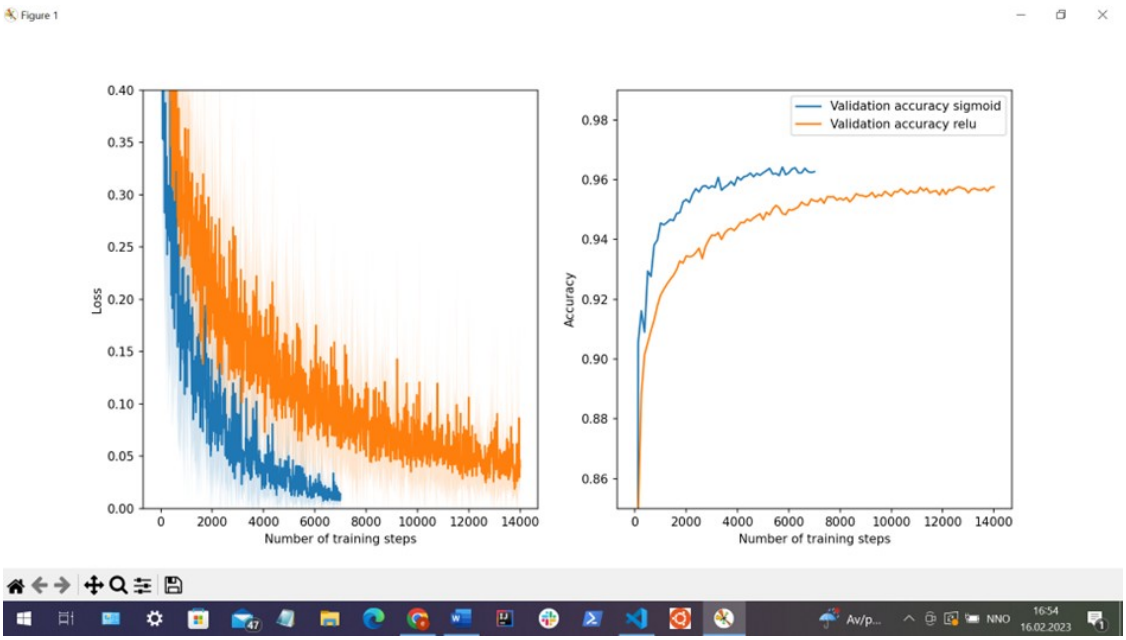
Plot of a model with 2 hidden layers with 60 hidden units each



In task 3 the number of parameters we had was 50880. Now, using the same approach we have 2 layers of 60 hidden units each which means we have 784x60 + 60x60 + 60x10 + 60 = 51300. And the number of hidden layer units is 60*2 = 120.

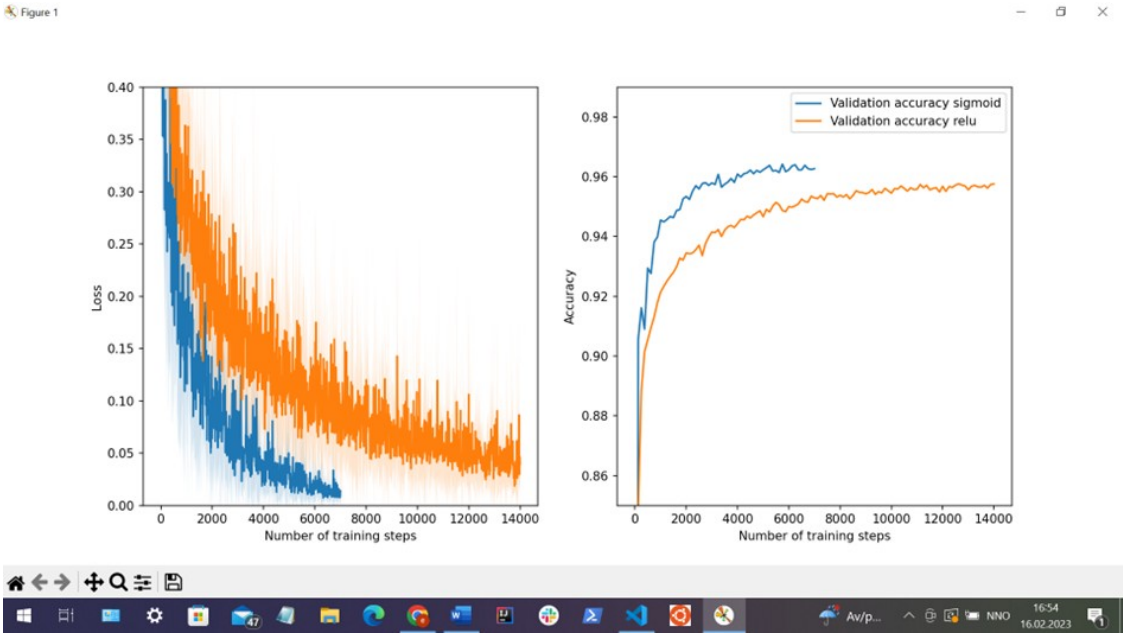
It seems like this network performs about the same when it comes to accuracy, or even slightly worse. And the loss seems to be about the same as well.

Task 4e)



We see here that the model converges about the same speed but doesn't get as low loss as the standard model and the accuracy is a bit lower as well. This could come from the model being too complex and finding connections and/or patterns of insignificance. This just proves that aimlessly adding layers to a neural network does not necessarily increase its performance.

Task 4f)



Here we use the standard model of 1 hidden layer with 64 hidden units where one model uses the improved sigmoid and the other uses relu. We can see here that the model performance does not increase according to this plot, in fact the accuracy of the relu model is lower and the loss is not as low as sigmoid. The relu model also does not converge as fast.