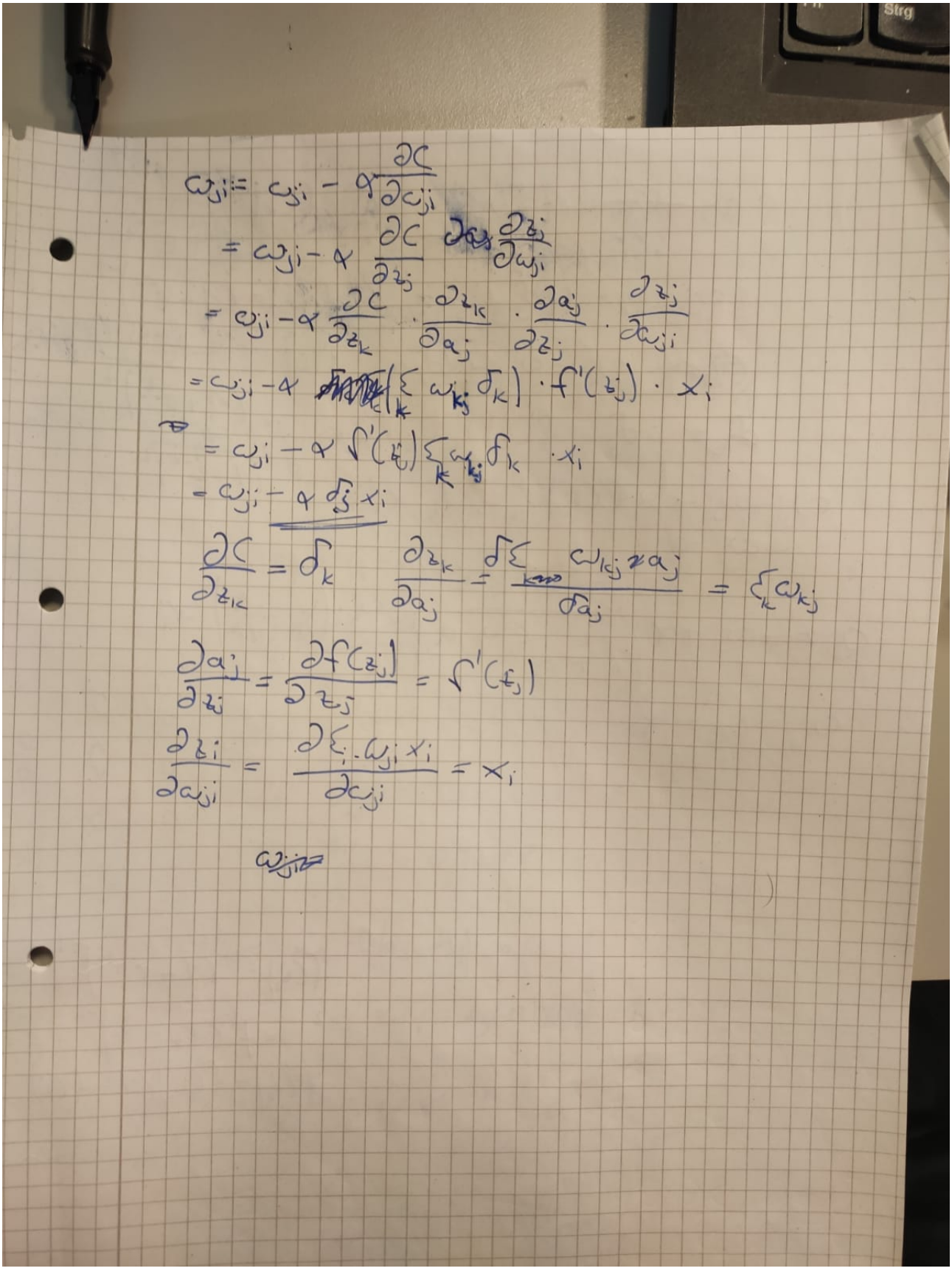
1a)



$$w_{ji} := w_{ji} - \alpha \frac{\partial C}{\partial w_{ji}}$$

$$= w_{ji} - \alpha \frac{\partial C}{\partial z_j} \frac{\partial z_j}{\partial w_{ji}}$$

$$= w_{ji} - \alpha \frac{\partial C}{\partial z_k} \cdot \frac{\partial z_k}{\partial a_j} \cdot \frac{\partial a_j}{\partial z_j} \cdot \frac{\partial z_j}{\partial w_{ji}}$$

$$= w_{ji} - \alpha \left[\sum_k w_{kj} \delta_k \right] \cdot f'(z_j) \cdot x_i$$

$$= w_{ji} - \alpha f'(z_j) \sum_k w_{kj} \delta_k \cdot x_i$$

$$= w_{ji} - \alpha \delta_j x_i$$

$$\frac{\partial C}{\partial z_k} = \delta_k \qquad \frac{\partial z_k}{\partial a_j} = \frac{\partial \sum_k w_{kj} a_j}{\partial a_j} = \sum_k w_{kj}$$

$$\frac{\partial a_j}{\partial z_j} = \frac{\partial f(z_j)}{\partial z_j} = f'(z_j)$$

$$\frac{\partial z_j}{\partial w_{ji}} = \frac{\partial \sum_i w_{ji} x_i}{\partial w_{ji}} = x_i$$

1b)

1b, $X = B\_size \times 785$

$W_{ji} = 785 \times 64$

$$Z_j = \sum_{i=0}^{d} x_i w_{ji} \Rightarrow Z_j = B\_size \times 64$$

$$\Downarrow$$

$Z_j = X W_{ji} = B\_size \times 64$

$A_j = f(Z_j) = B\_size \times 64$

$\Delta_j = f'(Z_j) \sum_k W_{kj} \delta_k$

$W_{kj} = 64 \times 10$

$\Delta_k = -(y_k - \hat{y}_k) = -(y_k - f(Z_k))$

$Z_k = A_j W_{kj} = B\_size \times 10$

$y_k = B\_size \times 10$

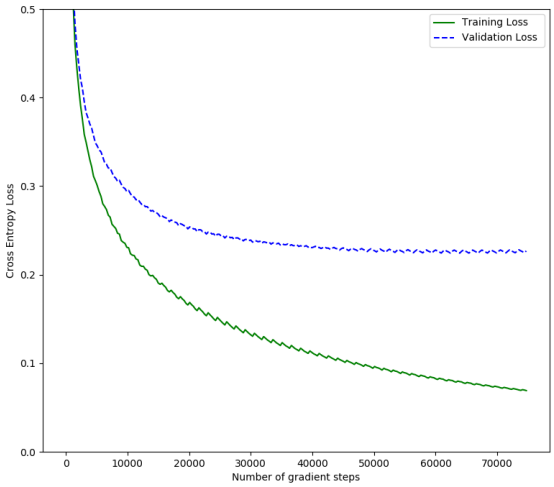
$\Delta_k = B\_size \times 10$

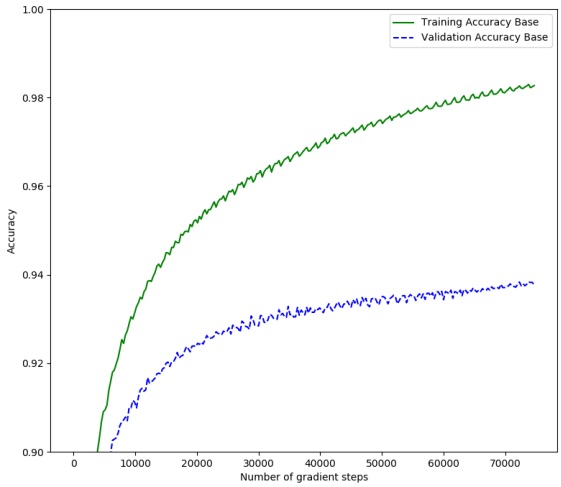
$\Delta_j = f'(Z_j) \odot \Delta_k W_{kj}^T = B\_size \times 64$

$W_{ji} := W_{ji} - \alpha X^T \Delta_j = 785 \times 64$

$$\left[f(Z_j) = \frac{1}{1 + e^{-Z}} \right]$$

2c)

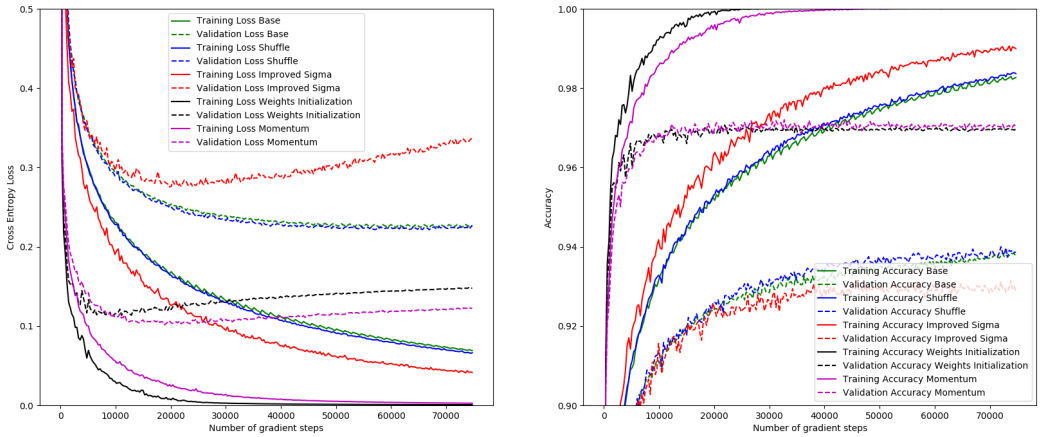

Training Loss
Validation Loss
Cross Entropy Loss
Number of gradient steps
0.0
0.1
0.2
0.3
0.4
0.5
0
10000
20000
30000
40000
50000
60000
70000


Training Accuracy Base
Validation Accuracy Base
Accuracy
Number of gradient steps
0.90
0.92
0.94
0.96
0.98
1.00
0
10000
20000
30000
40000
50000
60000
70000

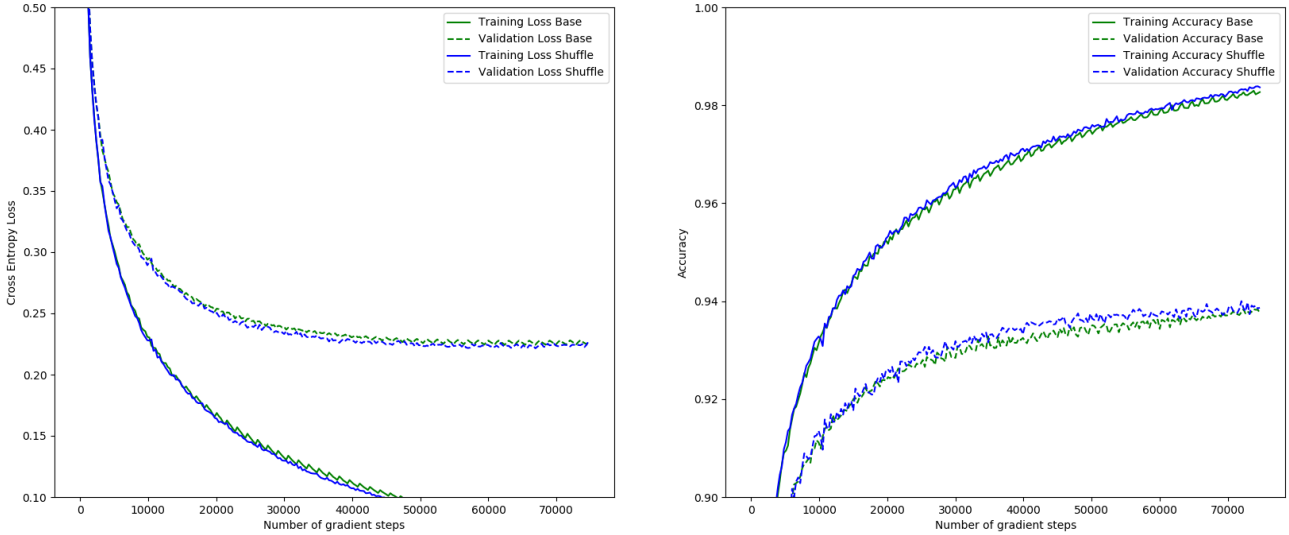
2d)

As we are using the bias trick, all the weights and biases are included in the weight matrices.
There are two weight matrices with the dimensions [785, 64] and [64, 10].
In total there are 785 * 64 + 64 * 10 = 50880 parameters.

This image contains all plots created in task 3 for a general overview. The changes on the accuracy and the loss is discussed in detail by looking at each specific plot in the following subsections.

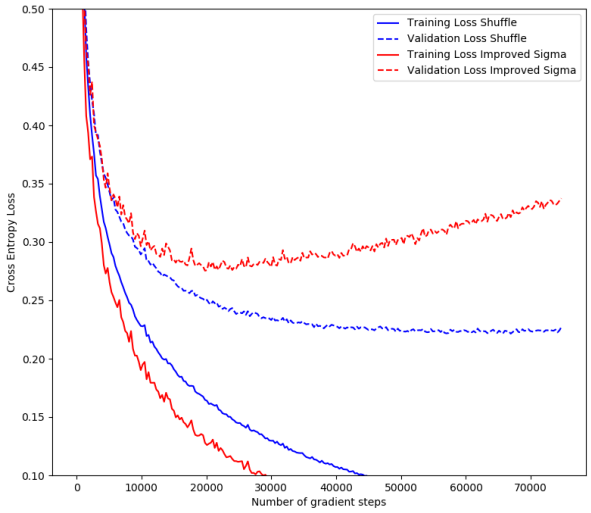


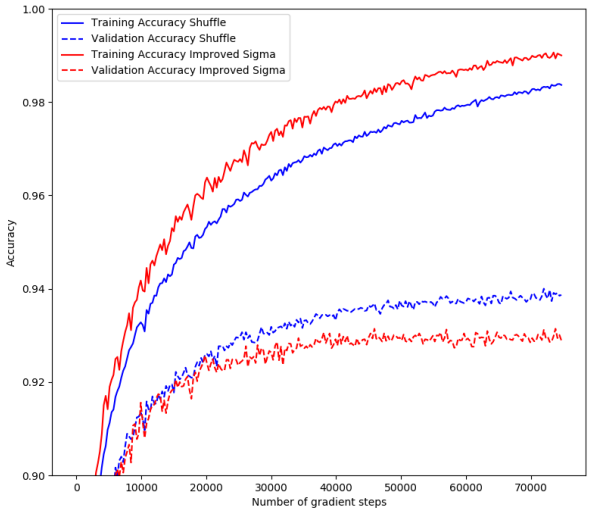
3a)
When shuffling the training data, the convergence happens a little faster and the accuracy is marginally higher. The network does not seem to get generalized/overfitted during training, but it flattens at 60000 gradient steps. The accuracy increases during the whole training, and early stopping does not necessarily need to be applied yet. There seems to be mixed opinions on the topic of early stopping[1]. Stopping at the lowest cross entropy loss point returns a network with the most sure opinion. If you continue training after this point, the network might guess correctly more often, but it will be not as sure.



[1] https://datascience.stackexchange.com/questions/37186/early-stopping-on-validation-loss-or-on-accuracy

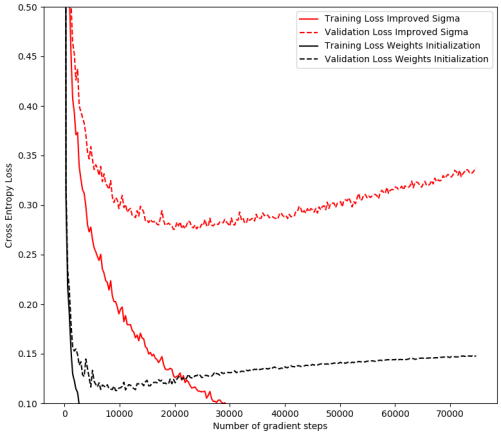
3b)
Using an improved sigmoid function increases the convergences speed significantly. This can be seen by the substantially lower training loss in comparison to the setup only using the shuffling. A major drawback is the worsened validation accuracy, this must be improved to achieve a better model. The network does get generalized/overfitted during training as the cross entropy loss plot increases after approximately 25000 gradient steps. The accuracy continues to increase until approximately 45000 gradient steps.

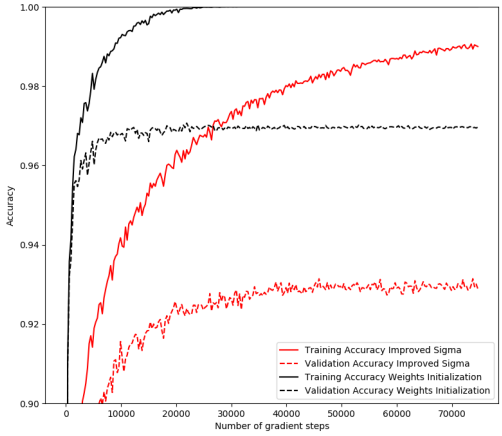




3c)
Fan-in:

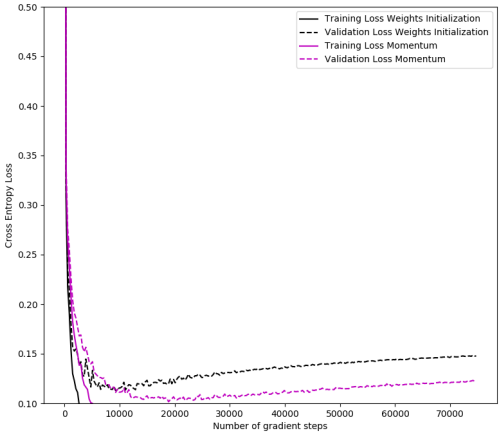
- First Layer = 785
- Second Layer = 64

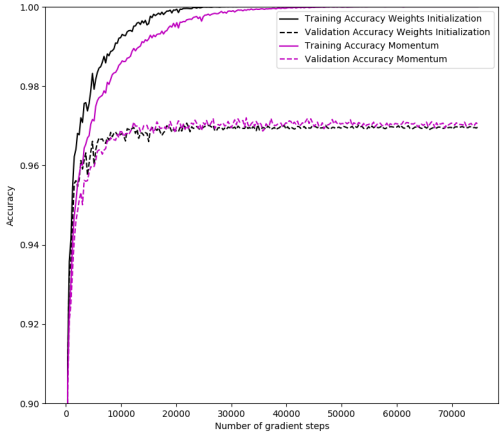




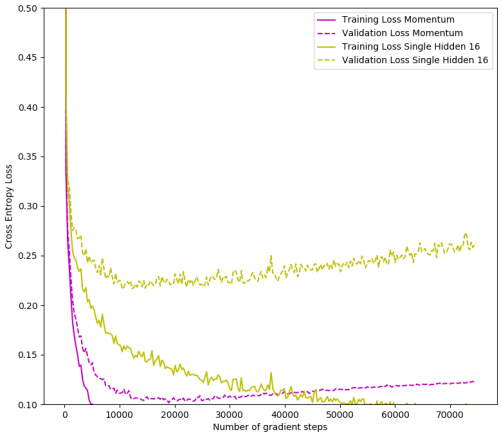
Using the improved weight initialization leads to a substantial increase in the convergence speed and the accuracy.
This can be seen in the significant and quick decrease of the cross entropy loss in the training set and the significant increase of the accuracy in the validation set. The network does get generalized/overfitted during training. It happens quite early at approximately 10000 gradient steps. The accuracy does not change much after 10000 gradient steps.

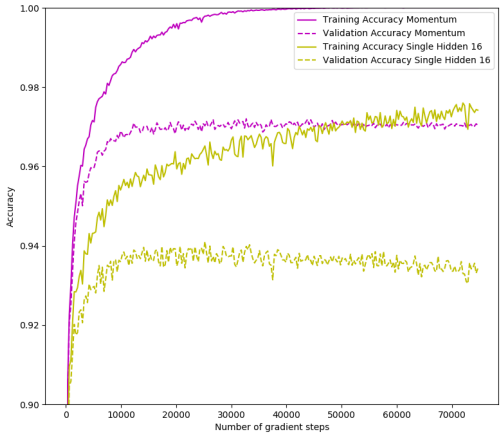
3d)
Implementing the momentum yields in a slightly higher validation accuracy with the cost of slightly decreased convergence speed. The network does get generalized/overfitted during training. It happens after the previous version of the network at approximately 20000 gradient steps. At this points the two versions perform more or less the same, but with a smaller loss on the network with momentum implemented.





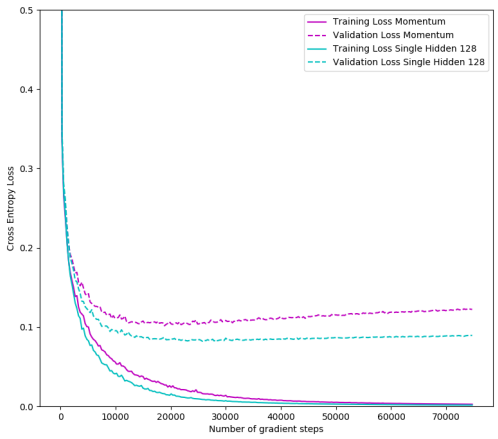
4a)

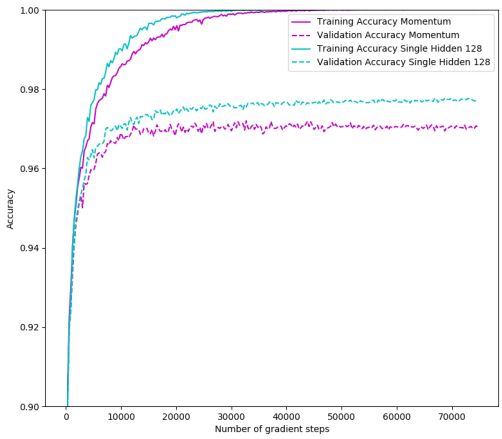




When a single hidden layer with 16 weights is used the accuracy of the neural network decreases substantially. The convergence speed is also slowed down.
Apart from this fact, the network is quicker to train when the hidden layer consists of only 16 weights.

4b)
Using a bigger hidden layer yields a significantly higher accuracy. The convergence (the point when the validation accuracy is approximately stable) takes more gradient steps. Additionally, it takes more computation to train the bigger network as it contains a lot more parameters.





4d)
The network from 3) had weight matrices with the shape [785,64] and [64,10]. This results in a total number of 785 * 64 + 64 * 10 = 50880 parameters.
Using a neural network with two hidden layers with 60 nodes result in a similar number of parameters. The weight matrices for such a network have the shape [785,60], [60, 60], [60, 10]. A total of 785 * 60 + 60 * 60 + 60 * 10 = 51360 parameters.

The validation accuracy of a network consisting of two hidden layers with 60 nodes each results in a similar validation accuracy. The convergence speed is also comparable to the network in 3)

In the provided code the gradient approximation test fails. After talking to the lab instructors this result is still considered correct, as the training of the network is working.