TDT4265: Computer Vision

Assignment 2

Anders Kjelsrud

February 2023

Have $\quad w_{ji} \leftarrow w_{ji} - \alpha \frac{\partial c}{\partial w_{ji}}$

want $\quad w_{ji} \leftarrow w_{ji} - \alpha \delta_j x_i \qquad\qquad z_k = w_{kj} x_k$

chain rule: $\frac{\partial c}{\partial w_{ji}} = \underbrace{\frac{\partial c}{\partial z_k} \frac{\partial z_k}{\partial a_j} \frac{\partial a_j}{\partial z_j}}_{\frac{\partial c}{\partial z_j}} \frac{\partial z_j}{\partial w_{ji}}$

Using that

$\frac{\partial c}{\partial z_k} = \delta_k$, $\frac{\partial z_k}{\partial a_j} = w_{kj}$,

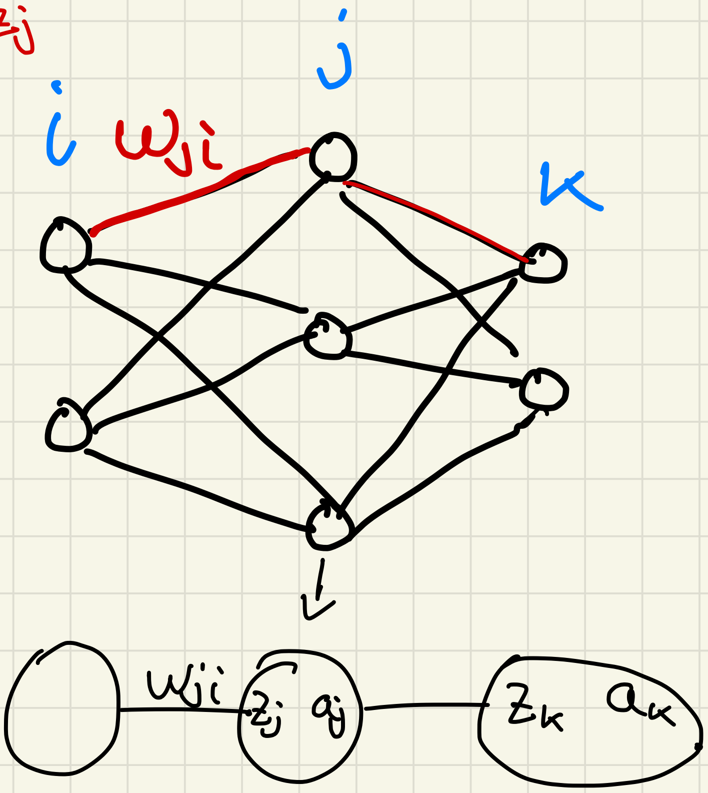
$\frac{\partial a_j}{\partial z_j} = f'(z_j)$, $\frac{\partial z_j}{\partial w_{ji}} = x_i$

$\Rightarrow \frac{\partial c}{\partial w_{ji}} = \underbrace{\delta_k w_{kj} f'(z_j)}_{\frac{\partial c}{\partial z_j} = \delta_j} x_i$

$\Rightarrow \frac{\partial c}{\partial w_{ji}} = \delta_j x_i$

or $\alpha \frac{\partial c}{\partial w_{ji}} = \alpha \delta_j x_i.$

$\square$



1 Task 2

a) Mean $\mu = 33.55$ and standard deviation $\sigma = 78.88$.

c) Accuracy is shown in Figure 1.

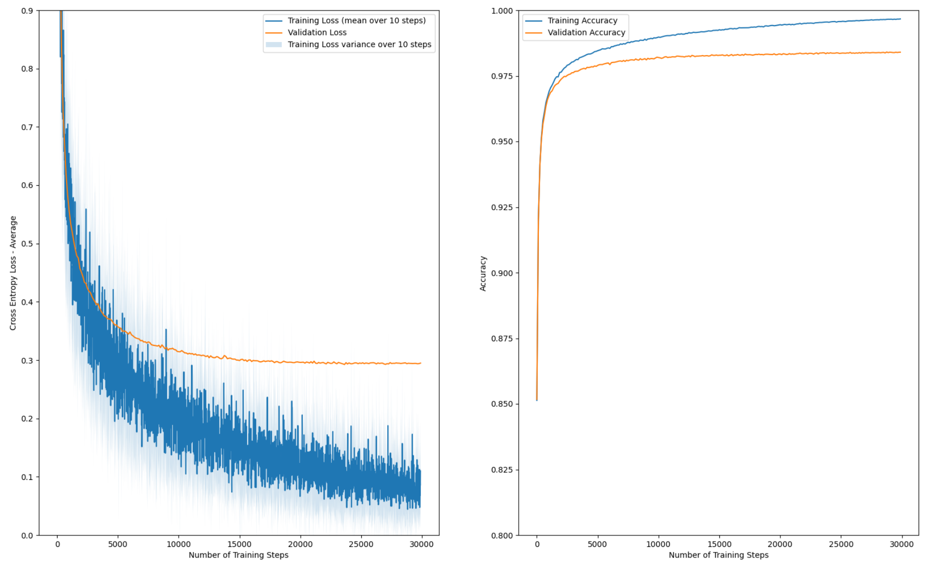


Figure 1: Loss

d) Number of parameters is $785 \cdot 64 + 64 \cdot 10 = 50880$. Here the biases for the last layer is neglected because there are none.

2 Task 3

a) Using Xavier initialization for the weights increase the training accuracy from `0.996685` to `0.999845`, while the validation accuracy increases from `0.98401` to `0.99168`. Training time went down from `69` seconds to `54`. Early stopping kicks in earlier with `44` epochs compared to `47`.

b) Using the improved zero-centered sigmoid gives a training accuracy of `1.0` and validation accuracy of `0.99237`. Early stopping kicks in after `28` epochs. This change increased convergence dramatically.

c) Momentum decreases the number of epochs needed before early stopping further, now training stops after `19` epochs. Improvements are shown in Figure 2.

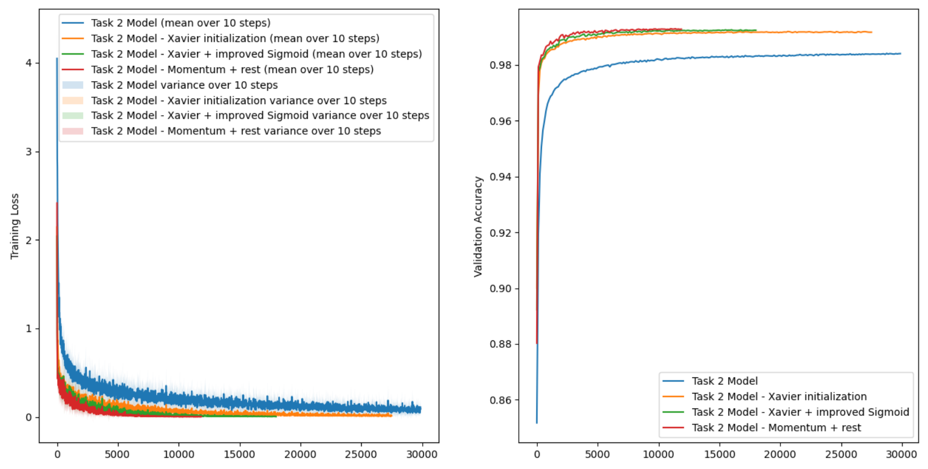


Figure 2: Tricks of the trade

3 Task 4

a) Figure 3 shows the training loss and validation accuracy of the different topologies in task a) and b). It looks like 32 neurons are too few to properly learn the different classes.

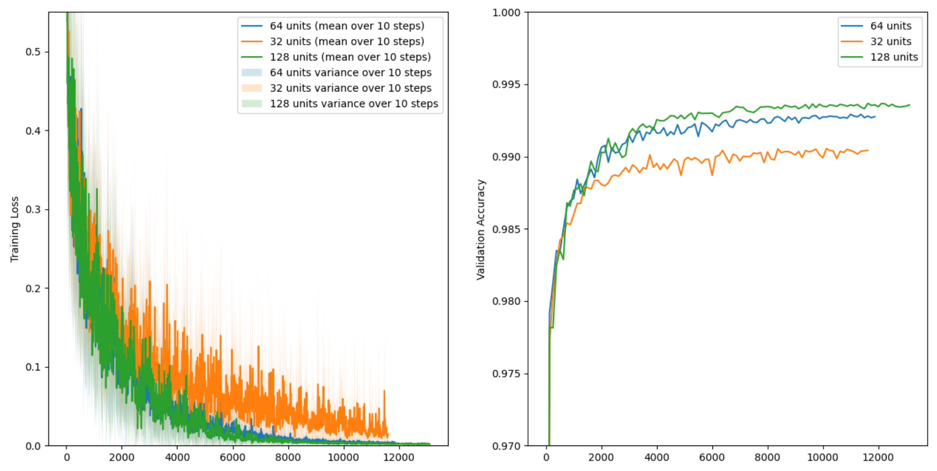


Figure 3: Different number of neurons

b) 128 neurons learn at about the same rate as 64, but there are signs of overfitting, namely that validation accuracy stops increasing but the training is not stopped.

d) Using 60 neurons on the hidden layers gives almost the same number of parameters. The network from (2) has $785 \cdot 64 + 64 \cdot 10 = 50880$ parameters. Here we have $785 \cdot 60 + 60 \cdot 60 + 60 \cdot 10 = 51300$. In other words the 2-layer NN has 420 more parameters. Figure 4 shows the training and validation loss as well as the validation accuracy for the two models. They perform very similarly, which is to be expected because one hidden layer is in principle capable of learning arbitrarily complex patterns. The fact that their number of parameters are nearly the same also supports this claim.

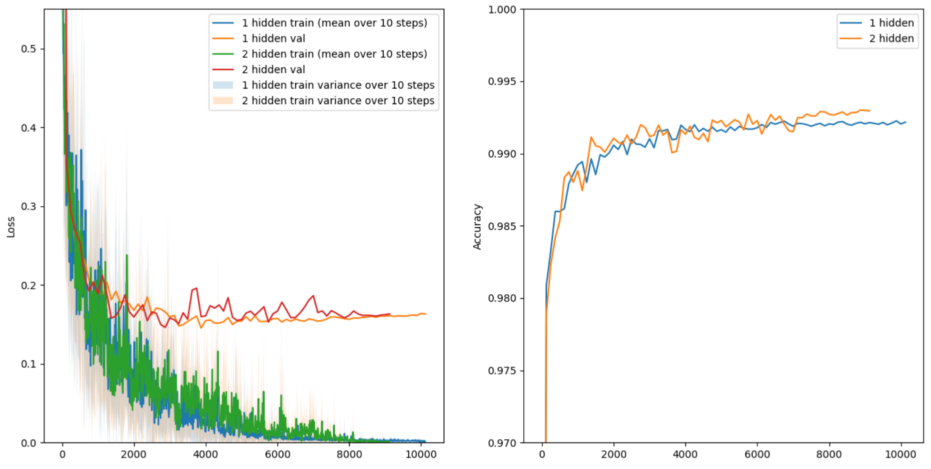


Figure 4: 1 hidden vs 2 hidden layers

e) This is an example of the vanishing gradient problem. That is when the network is too deep in conjunction with an activation function such as a sigmoid, which causes the gradients to become so small that the optimizer is unable to use them to update the parameters. In practice it leads to very slow convergence and can even render the model unable to learn anything. It can be seen from Figure 5 that the model with 10 hidden layers has higher variance, higher loss and lower accuracy than the other two.

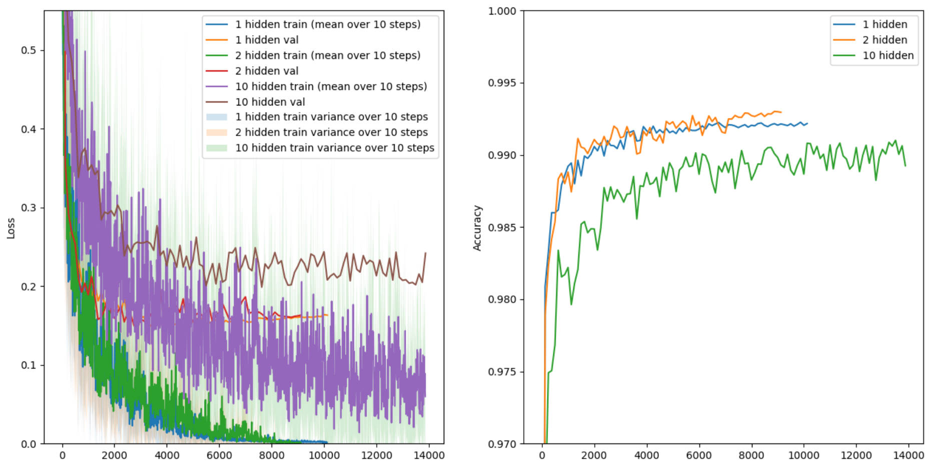


Figure 5: 1, 2 & 10 hidden layers

f) Using ReLU significantly increases performance because it fixes the problem of vanishing gradients. Here ReLU is only used on the network with 10 hidden layers. Figure 6 shows that accuracy is now on par with the two previous models. The fact that it doesn't outperform them could mean the task simply does not require a network this deep.

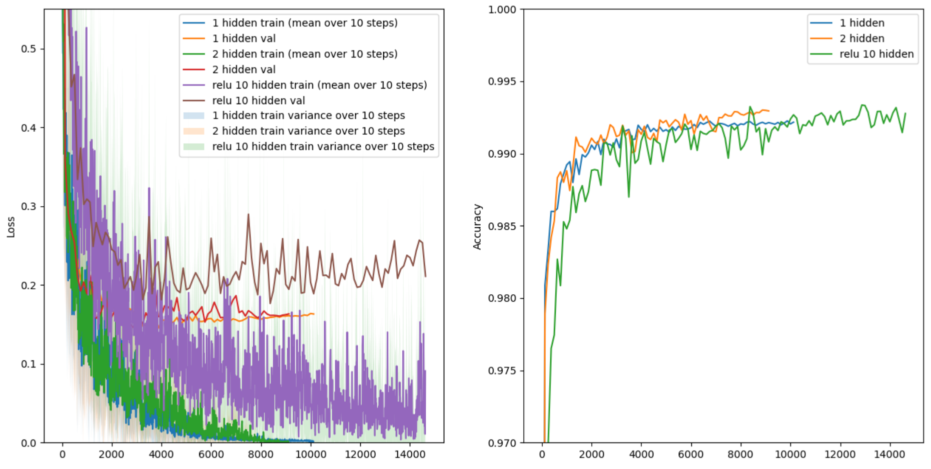


Figure 6: 1, 2 & 10 hidden layers with ReLU