8P361 Project Imaging - BIA Group 4 | Assignment 2

Myrthe van den Berg, Rick van Bergen, Nils Bodenstaff, Cris Claessens and Jeanine van Ingen

Exercise 1 The example neural network classification model in this assignment is relatively simple — it contains a single hidden layer of 64 neurons. Perform a set of experiments with more complex models, e.g. with more layers (deeper models), more neurons per layer or a combination. Describe the set of experiments that you have performed. What is the accuracy of the best model? How did you determine which model is the best?

After running the different models, with differences in the number of hidden layers and neurons the loss and accuracy for each model are evaluated. The script used to create the different models can be found in the .zip file. In table 1, an overview of the models and the loss and accuracy of each model are shown when applying it on a testing dataset.

Table 1, the different loss and accuracy by different hidden layers and neurons composition.

Number of hidden layers	Neurons	Loss	Accuracy
1	32	0.2067	0.9415
1	64	0.1864	0.9455
1	128	0.1714	0.9497
1	256	0.1653	0.9525
2	64 64	0.1413	0.9598
2	128 64	0.1294	0.9610
2	64 128	0.1392	0.9588
2	128 128	0.1290	0.9615
2	256 128	0.1138	0.9653
2	128 256	0.1283	0.9618
2	256 256	0.1220	0.9644
2	64 256	0.1504	0.9545
2	256 64	0.1121	0.9652
3	64 64 64	0.1303	0.9597
3	128 128 128	0.1088	0.9668
3	256 256 256	0.1203	0.9636
3	64 128 256	0.1258	0.9611
3	256 128 64	0.0934	0.9711
4	512 256 128 64	0.1068	0.9674

It seems that there is a correlation between the number of neurons and the loss and accuracy; more neurons give in the majority of the times a lower loss and a higher accuracy. Furthermore, when having more neurons in the first layer than the second and so on, like a pyramid structure, for example; 256, 128, 64, the model performs better since it results in lower loss and higher accuracy. This is true until a certain level of complexity where the model starts overfitting the training data resulting in a lower accuracy on the testing data as can be seen when looking at the four-layer model.

After looking at the values of the model outcome, the accuracy curves from Tensorboard are discussed. The accuracy curves from the train and the validation is shown in figure 1 for multiple amounts of layers, a 3-layer model and a 4-layer model. In figure 2 the loss curves of the train and validation, for the same layers, are shown. In appendix A and B, other loss and accuracy curves are shown.

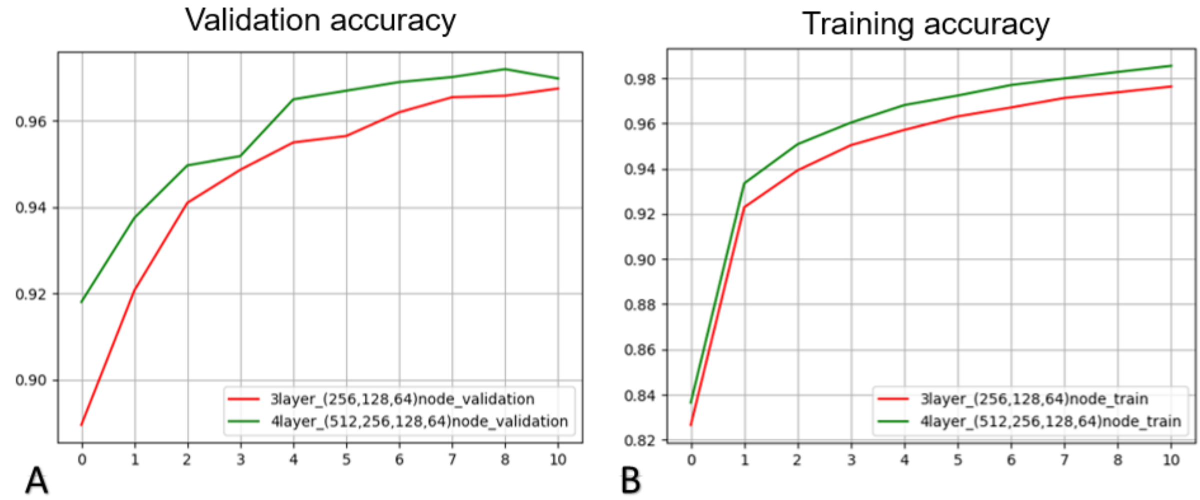


Figure 1, the accuracy curves of validation data (A) and train (B). With on the y-axis the accuracy and on the x-axis the epochs.

In figure 1, it is shown that the more complex a model is, often results in higher accuracy. For example when comparing the 1 layer model to the 3 layer model the latter would reach a higher accuracy on testing and training.

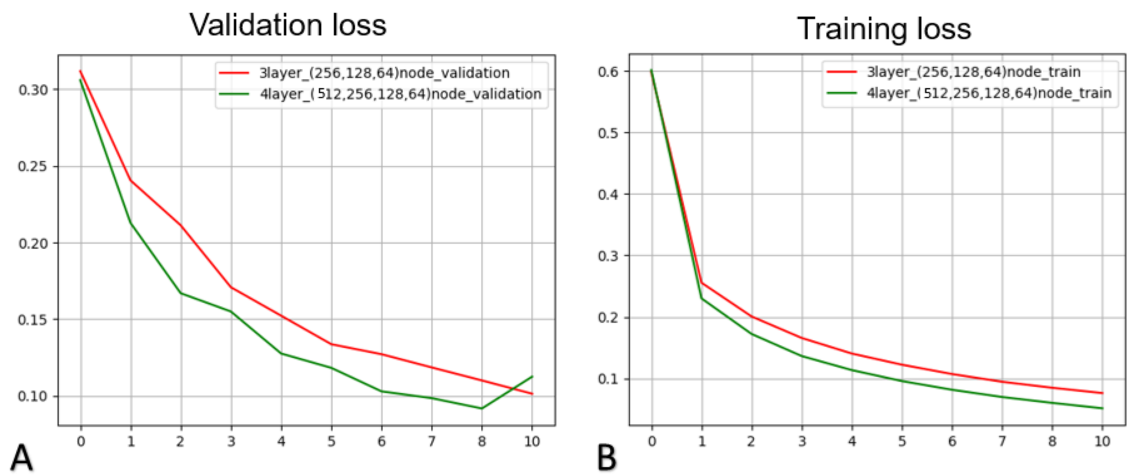


Figure 2, the loss curves of validation data (A) and of train data (B). With on the y-axis the loss and on the x-axis the epochs.

Figure 2, shows that the 4-layer model is overfitted, because of the loss curve that increases.

It can be concluded that the best model, a model is with multiple hidden layers with a decreased number of neurons towards the outcome layer, in a pyramid-like structure. The best amount of hidden layers is three for this problem.
This is concluded since these models yield the highest accuracy on training, validation and testing data.

Exercise 2 Compare the performance of the following three models:

1. **Neural network without any hidden layers (the input layer connects directly to the output layer).**
2. **Neural network with 3 hidden layers with ReLU activations.**
3. **Neural network with 3 hidden layers with linear activations (i.e. without nonlinearities between the layers).**

Analyze the performance of the three models. What is the reason behind the difference in performance between the second and third models?

The script used to create the 3 models can be found in the .zip file. The two models containing hidden layers had an equal amount of neurons in each layer. The number of neurons in each layer where: 256 in the first layer, 128 in the second layer and 64 in the last layer. It was chosen to use this configuration since this model yielded the highest accuracy and loss in exercise 1. The results of the loss and accuracy of the three models are shown in table 2. In figure 2, the curves from Tensorflow are shown.

Table 2, The loss and accuracy of different activation models.

Number of hidden layers	Activation	Loss	Accuracy
None	N/A	0.3108	0.9155
3	ReLU	0.1002	0.9691
3	linear	0.2732	0.9206

As can be seen in the loss and accuracy a model without nodes yields the lowest accuracy compared to the model using ReLU and linear activations. From the latter two, the ReLU results in the highest accuracy on the testing data.

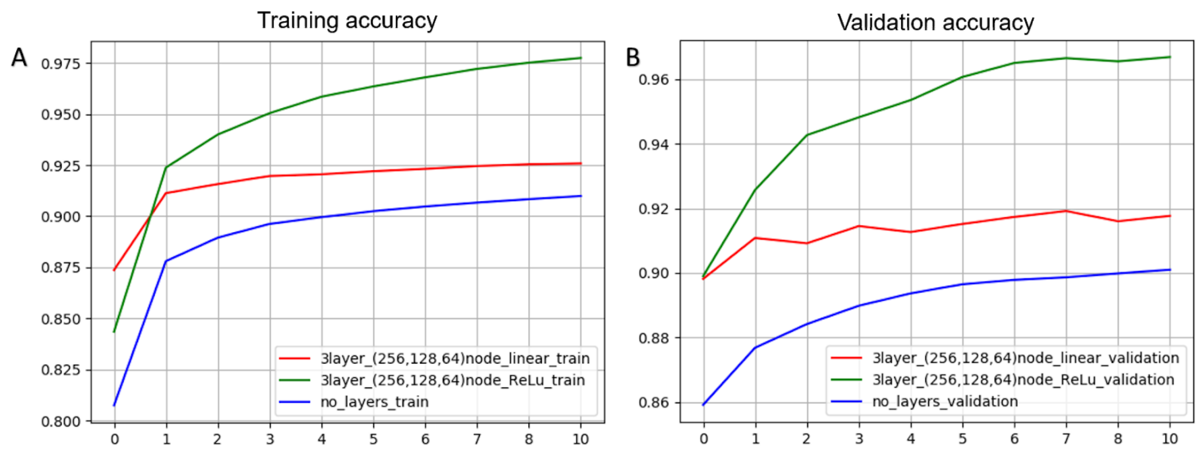


Figure 2, the accuracy curves of the train data (A) and the validation data (B) of the three different models. With on the y-axis the accuracy and on the x-axis the epochs.

During training, similar behavior is observed. Yet interestingly the accuracy on the validation is equal for both the ReLU and the linear activation at the starting point, as shown in figure 2, B. Furthermore, figure 2 shows that the accuracy of the training and the validation with linear activation increases much more than the model using ReLU activation.

The difference between the second and third models is due to the different characteristics of ReLU and linear activation. A multi-layer network in which all units have linear activation functions can always be collapsed to an equivalent network with only two layers of units. The neural network will not reduce the error since the gradient function is the same for every iteration of a linear activation function [1]. The advantage of a ReLU function is that it does not activate all neurons at the same time. The neurons will only be deactivated if the output of the linear transformation is less than zero.

Exercise 3 Train a neural network model (the specific architecture is up to you) for a four-class classification problem derived from MNIST in the following way:

- "vertical digits": 1, 7
- "loopy digits": 0, 6, 8, 9
- "curly digits": 2, 5
- "other": 3, 4

The implemented code can be found in the .zip file. The data collected from the code is shown in table 3.

Table 3, the data derived from the code of the four-class classification problem.

Number of hidden layers	Neurons	Neurons output	Loss	Accuracy
2	64 64	10	0.1413	0.9598
2	128 64	10	0.1294	0.9610
3	256 128 64	10	0.0934	0.9711
2	64 64	4	0.1092	0.9634
2	128 64	4	0.1008	0.9690
3	256 128 64	4	0.0876	0.9717

When looking at table 3 it can be derived that a model with less output has higher accuracy and a lower loss compared to a model with more output nodes. This is mostly true when looking at the loss, namely the model with four outputs yield a lower loss compared to the model with 10 outputs. Yet when looking at the accuracy this is not the case. Here the model with 10 outputs most of the time yields higher accuracy, except for the case of 2 layers of 64 nodes. An explanation for this is that due to the less complex output overfitting happens more easily. Thus resulting in lower accuracy.

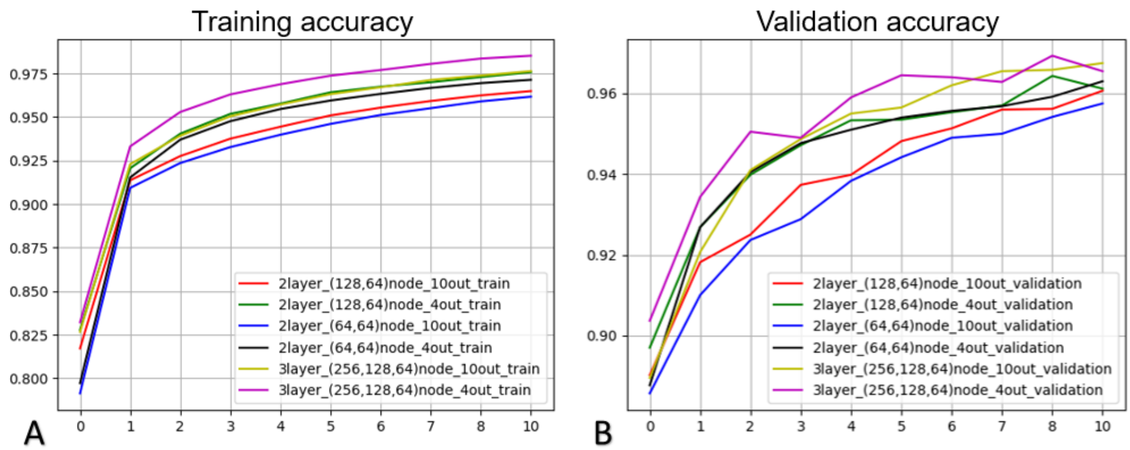


Figure 3, two accuracy curves during training (A) and validation (B). A 2-layer and 3-layer model is shown for a neuron output of 4 and 10 neurons. With on the y-axis the accuracy and on the x-axis the epochs.

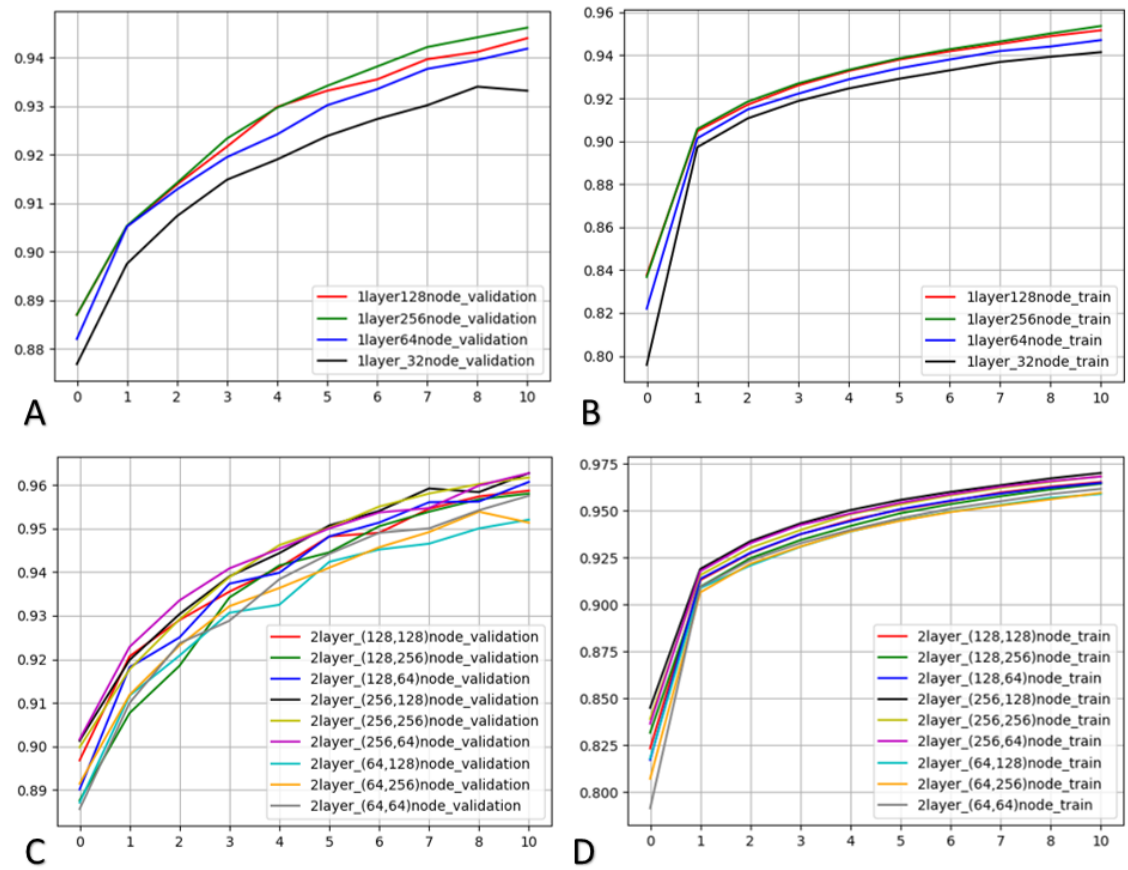
When looking at the curves of the accuracy during training and validation, in figure 3, a similar behavior is observed. Namely that the most complex model with 4 outputs yields the highest accuracy on the training data yet the validation accuracy seems to be lower than the 4 outputs. This could point towards overfitting.

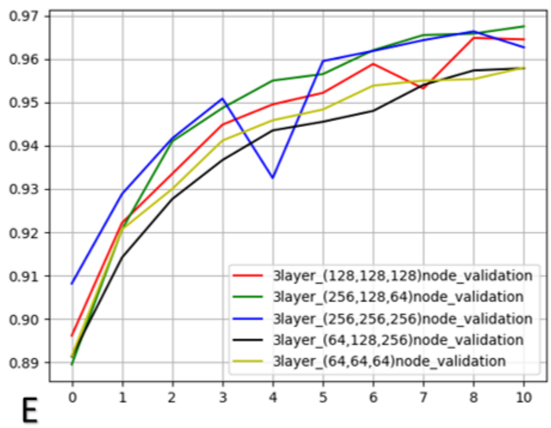
References

[1] Sharma V. A. (2017) Understanding Activation Functions in Neural Networks, retrieved from https://medium.com/the-theory-of-everything/understanding-activation-functions-in-neural-networks-9491262884e0

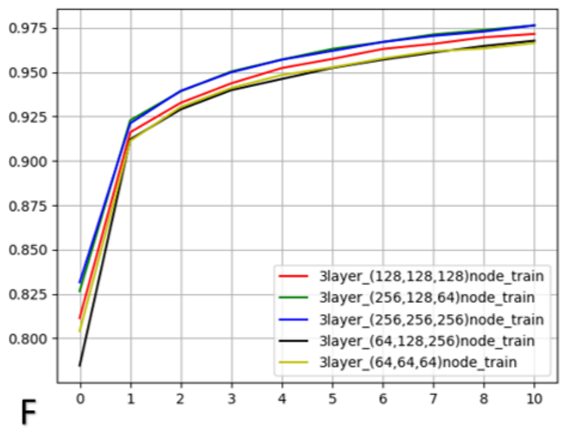
Appendix

A, Accuracy curves of different multiple hidden layers for train and for validation.




0.97
0.96
0.95
0.94
0.93
0.92
0.91
0.90
0.89
0 1 2 3 4 5 6 7 8 10
3layer_(128,128,128)node_validation
3layer_(256,128,64)node_validation
3layer_(256,256,256)node_validation
3layer_(64,128,256)node_validation
3layer_(64,64,64)node_validation

E


0.975
0.950
0.925
0.900
0.875
0.850
0.825
0.800
0 1 2 3 4 5 6 7 8 10
3layer_(128,128,128)node_train
3layer_(256,128,64)node_train
3layer_(256,256,256)node_train
3layer_(64,128,256)node_train
3layer_(64,64,64)node_train

F

B, Loss curves of different multiple hidden layers for train and for validation.