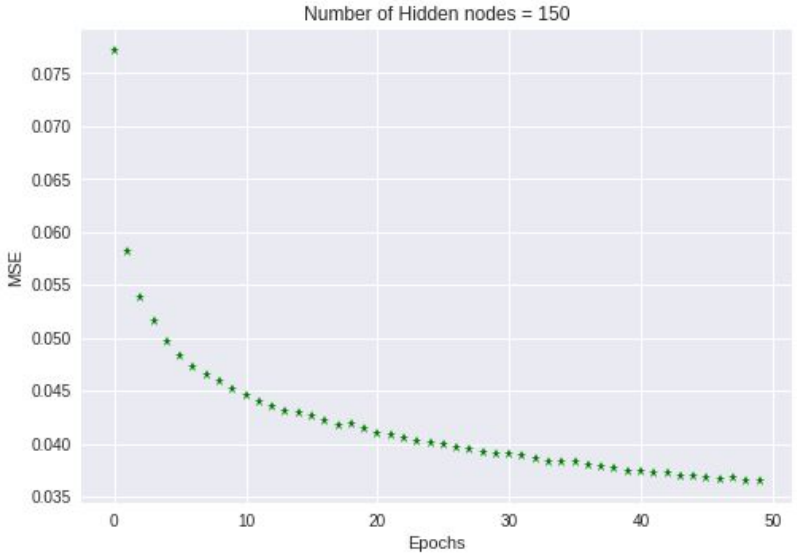
Carl Ridnert (940325-0112)
Óttar Guðmundsson (910302-0450)

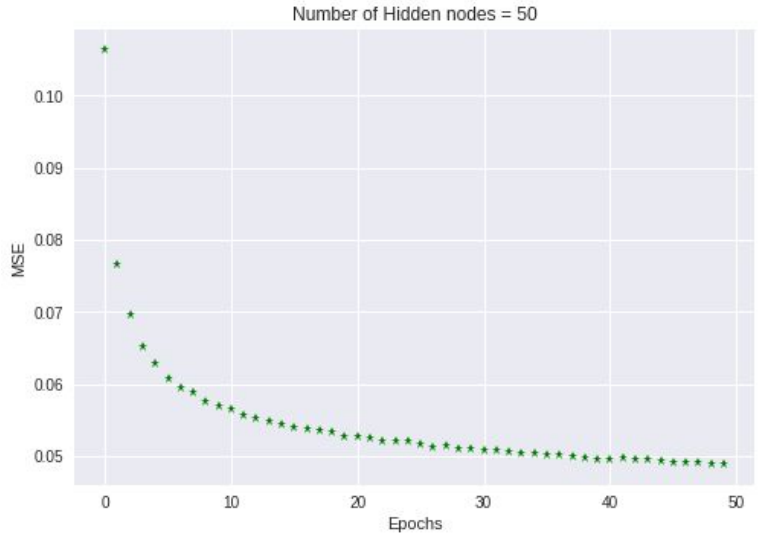
Lab assignment 4 - Deep neural network architectures with restricted Boltzmann machines and autoencoders

3.1 RBM and autoencoder features for binary-type MNIST images

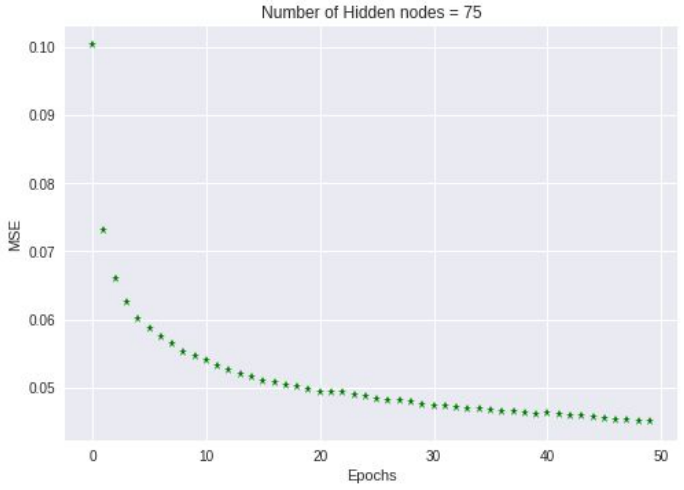
In this part we are going to perform unsupervised learning on the MINST dataset of 784 pixel pictures of handwritten digits. We will be training a restricted Bolzmann machine (RBM) and an Autoencoder (AE). For this final lab we used Tensorflow to stack the layers on top of each other.

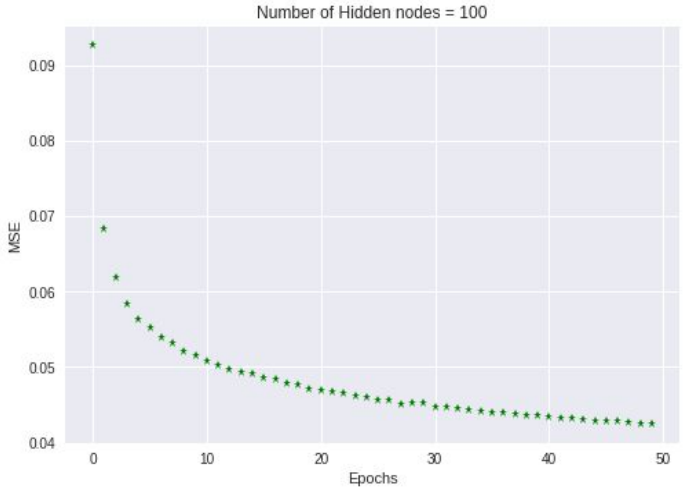
In this part we construct a one-layer RBM, we subsample the data and train it using contrastive divergence on 2000 samples which should provide a good spread and therefore bee a good representation. We presented one sample at a time and updated the weights, one run through all the samples defines one epoch and the mean squared error was computed and is depicted below as a function of epochs. The error was computed as an average over all pixels and all pictures.



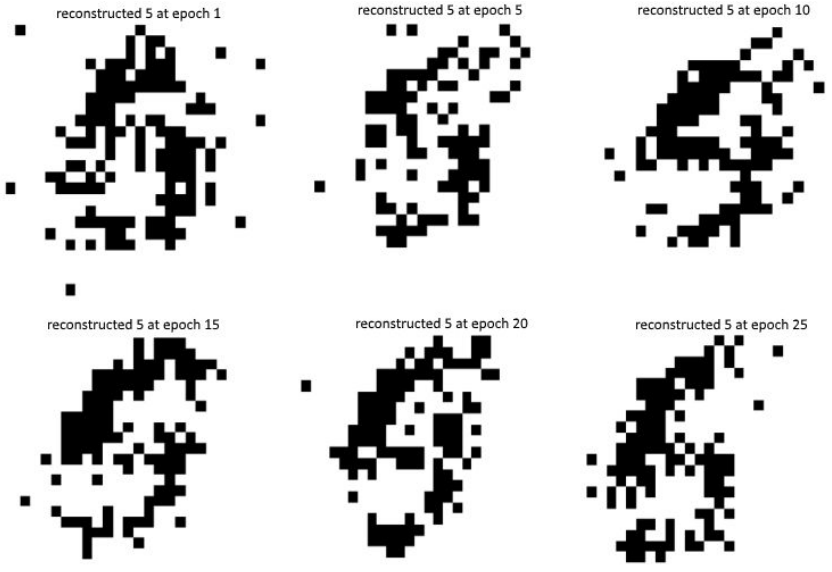


Carl Ridnert (940325-0112)
Óttar Guðmundsson (910302-0450)





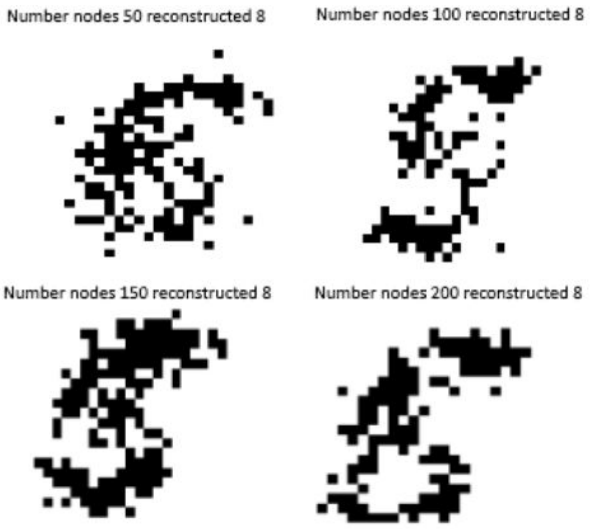
We can clearly see that it is decreasing for all number of hidden nodes and that the error decreases faster with more hidden nodes.
Next we look at how the number of epochs affect the performance of the model.



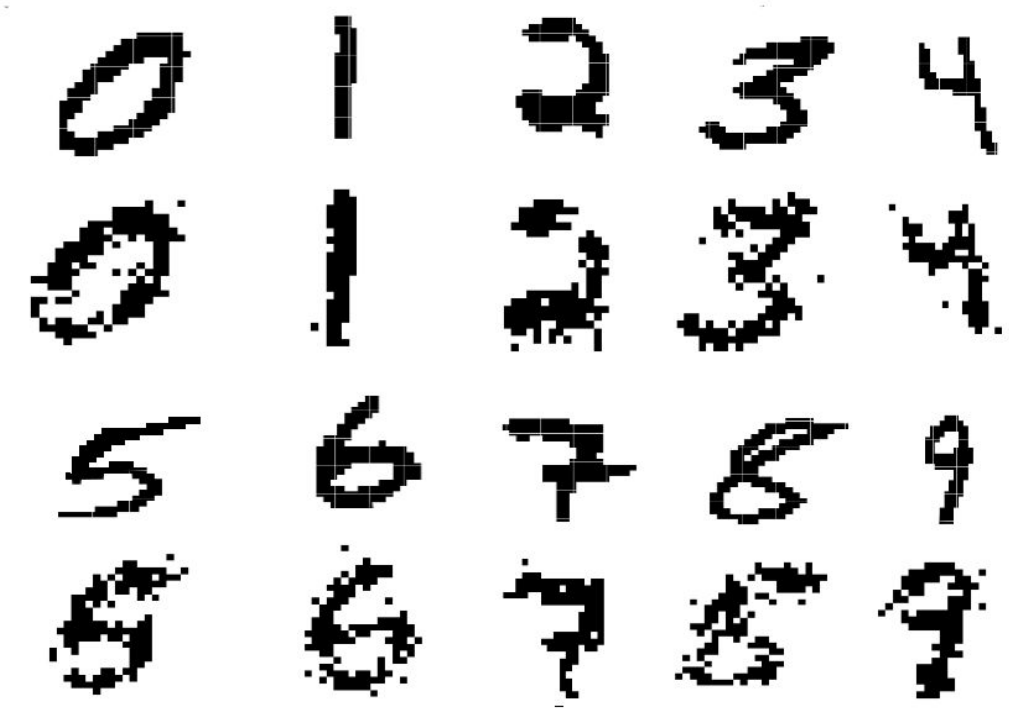
In this picture above we can see how the network slowly but surely gets better at restoring the pattern 5 in epochs 1 to 25. The hidden layer had 200 hidden nodes but doesn't really

Carl Ridnert (940325-0112)
Óttar Guðmundsson (910302-0450)

restore the pattern, next we look at how the reconstruction changes with the number of hidden layer nodes.



As we can see from the picture above the reconstruction works better with higher number of nodes, but the network still does not capture the complete form of the pattern. Maybe by adding more hidden nodes or increasing the learning rate might improve the reconstructive capability of the model.

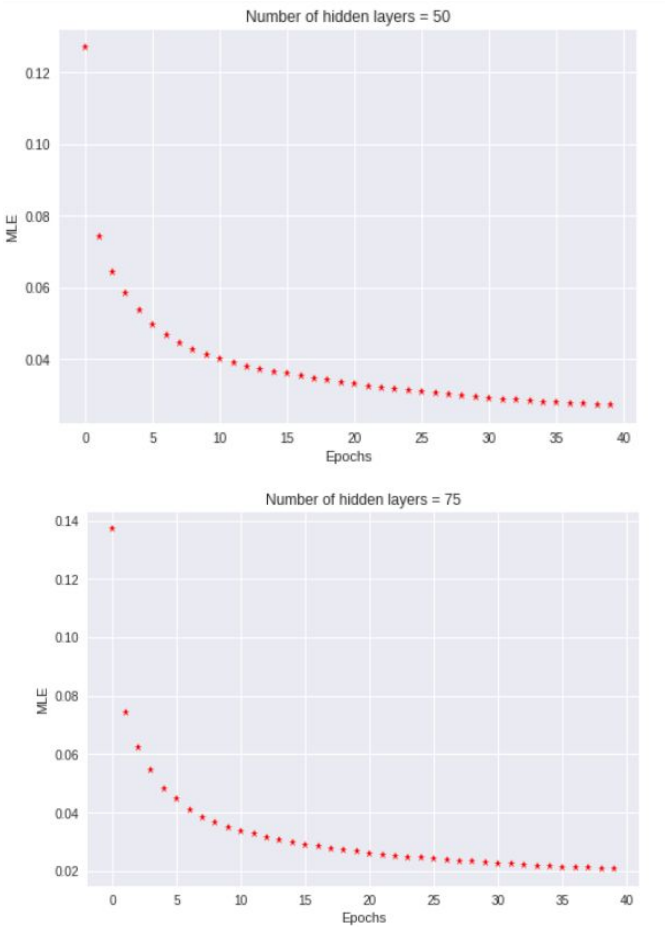


Carl Ridnert (940325-0112)
Óttar Guðmundsson (910302-0450)

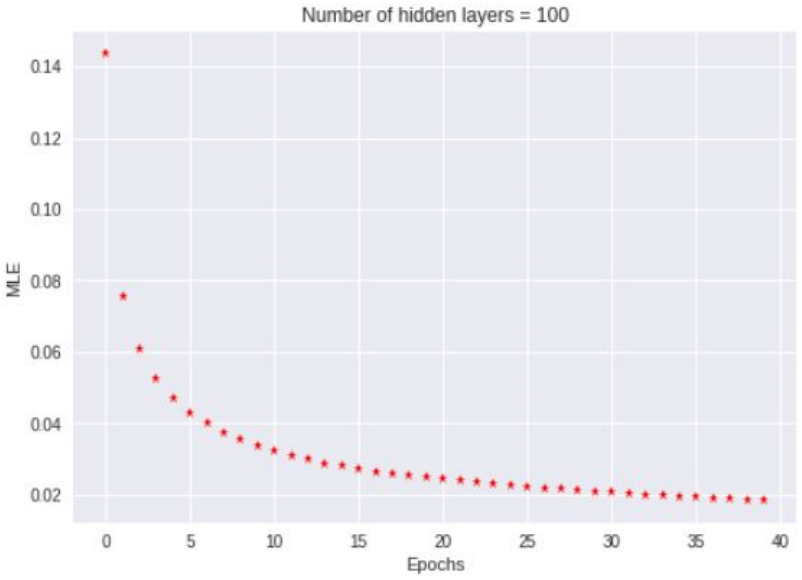
In the picture above we show the reconstructed images of all numbers for the case of 200 hidden nodes and 40 epochs. The network manages to restore the patterns but they are quite noise and distorted.
We agreed that the RBM's didn't quite live up to the expectation. Thus, we had higher hopes for the auto encoder.

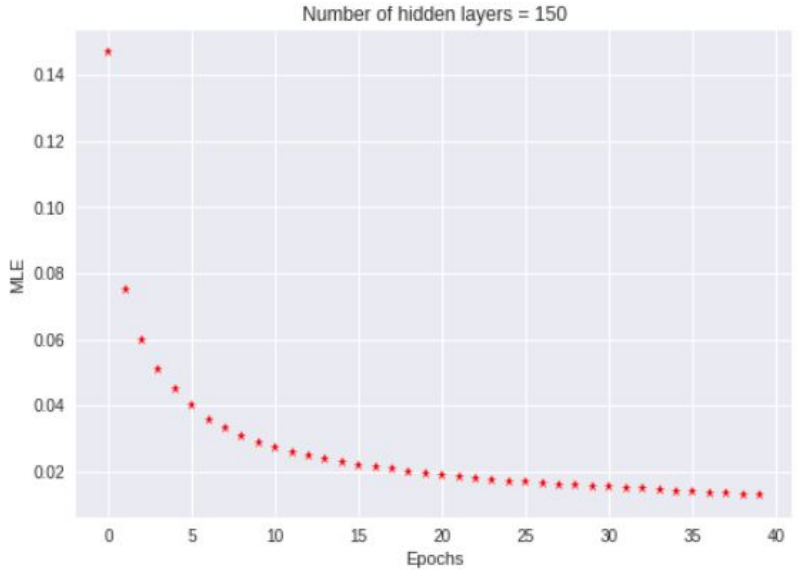
AE

The autoencoder was trained using backpropagation, the number of hidden nodes was varied and plots of the error rates for different number of hidden nodes as function of the epoch in the same manner as for the RBM are shown below.



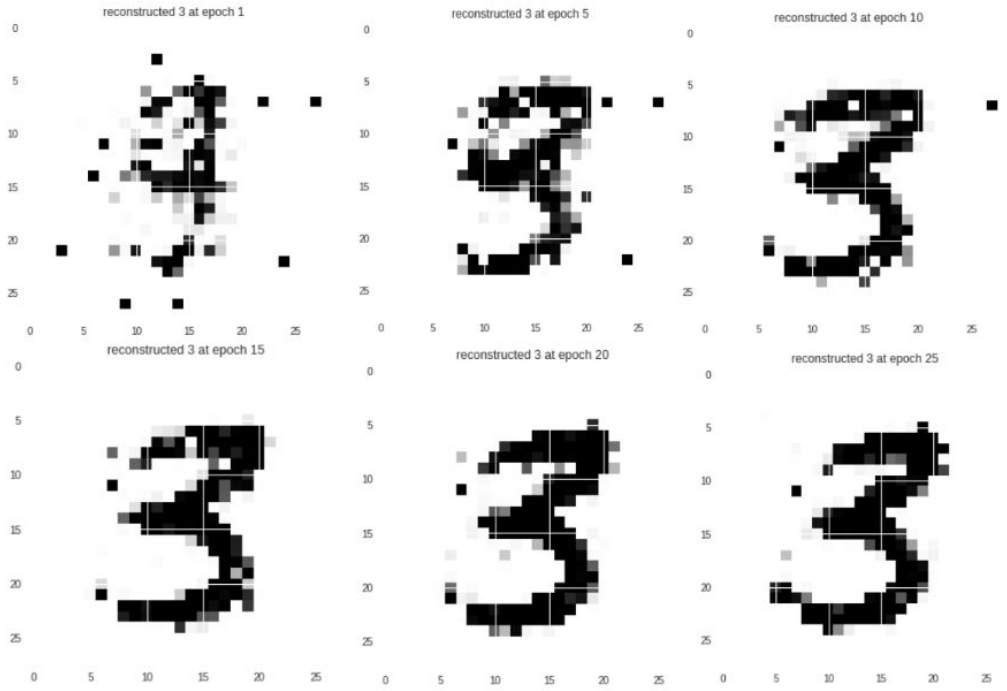
Carl Ridnert (940325-0112)
Óttar Guðmundsson (910302-0450)



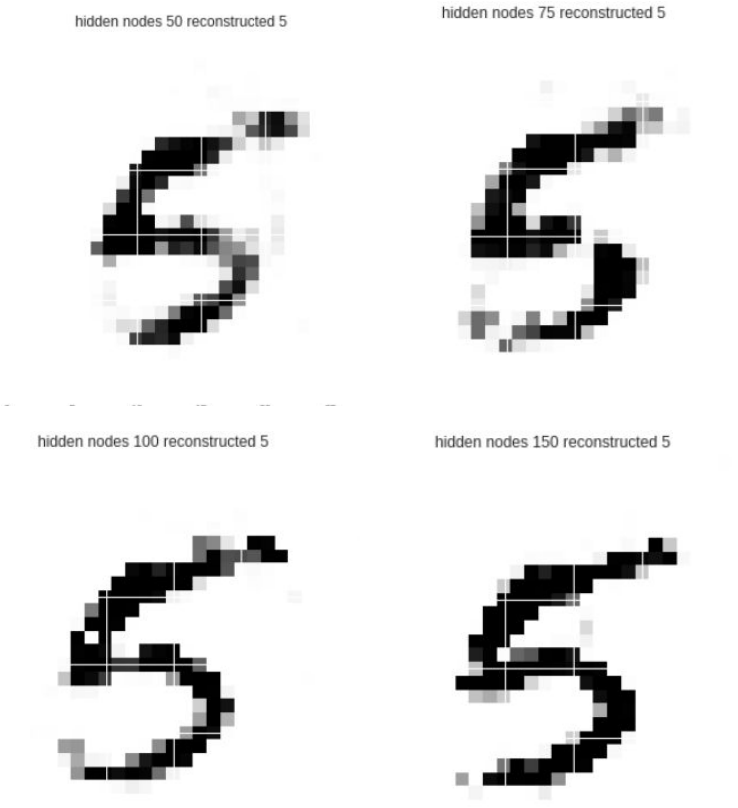


We see that the error decreases faster with an increasing number of hidden nodes, thus our best model would be the Autoencoder with 150 hidden nodes. Next we plot reconstructed images from every class in the same way as for the RBM.

Carl Ridnert (940325-0112)
Óttar Guðmundsson (910302-0450)



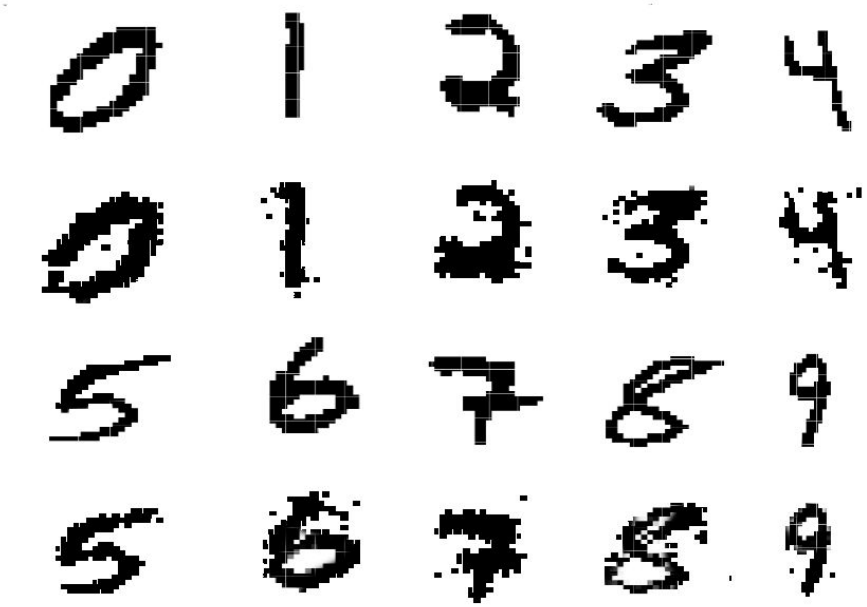
In the picture above it is depicted how the change in number of epochs affects the performance. The number of hidden nodes is fixed at 100 and the epoch is varied from 1 to 25 and we see the increase in performance clearly when using more epochs. Next we show the effects of varying the number of hidden nodes when it comes to reconstructing images.



One can observe that the reconstruction works better with higher number of nodes, just as we discovered from the error plots. However the difference is not as big as when the number

Carl Ridnert (940325-0112)
Óttar Guðmundsson (910302-0450)

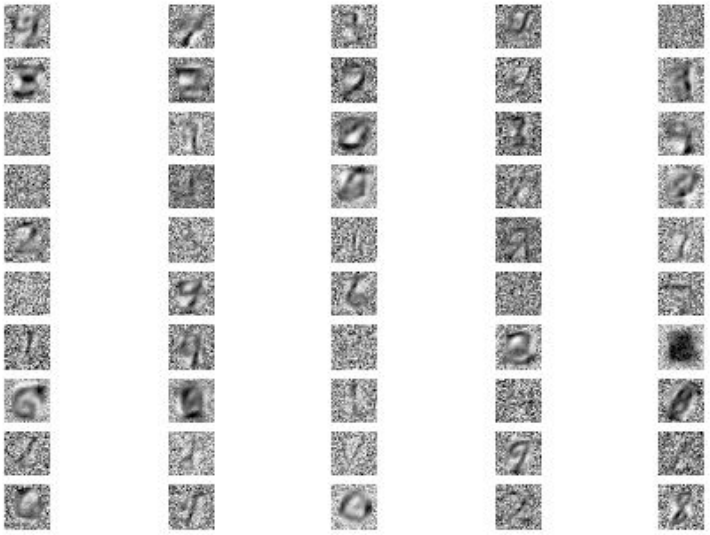
of epochs were varied. In the following picture we show the reconstructed images of numbers for the case of 200 hidden nodes and 40 epochs.



We see that the reconstructed images have some distortions in them but they are clearly recognizable.

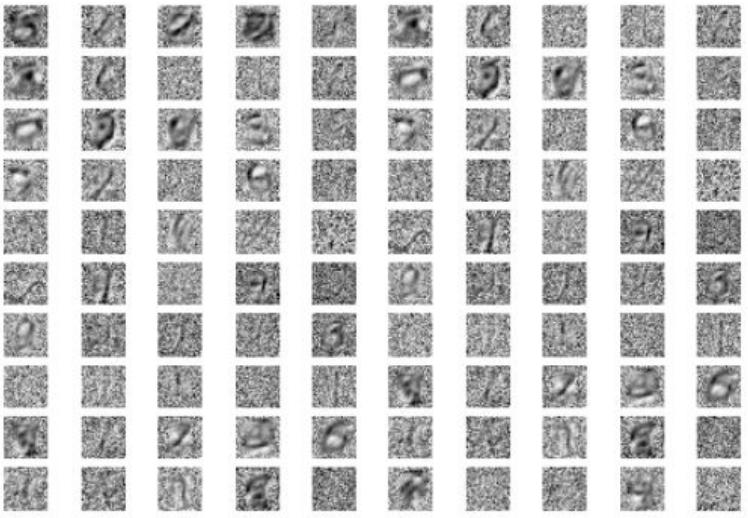
Next, we plot the 784 bottom-up nal weights for each hidden unit, using one different figure for each hidden unit.

RBM



50 nodes, 40 epochs

Carl Ridnert (940325-0112)
Óttar Guðmundsson (910302-0450)



100 nodes, 40 epochs

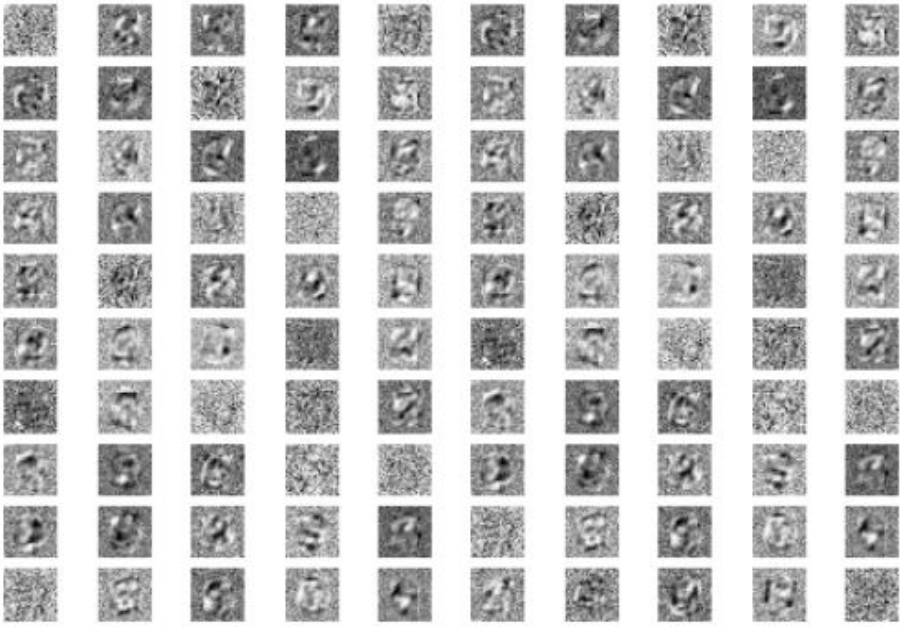
We can see that with both 50 and 100 nodes the network is able to capture features such as the numbers 8 and 6. We note that in the case of 100 nodes we observe more line-shaped weights and also more weights which seems to be randomly distributed. Perhaps it is the case that 100 nodes is enough, that the network already has enough flexibility and just starts to put weights randomly. However, we did observe increase in performance for the 100-node network. We can argue that this increase was marginal and since we double the complexity of the system the performance boost might not be worth it.

Auto Encoders

Both of the networks manage to capture circly lines and sphere that can represent 8 and 9 for example. Some nodes even manage to capture the outlines of whole patterns by itself. But for the former network, it seems like it catches more distinct edges while the latter captures small lines and part of the numbers. According to our pictures, there are way more useless nodes that simply catches noise in the 100 node network.

Carl Ridnert (940325-0112)
Óttar Guðmundsson (910302-0450)

50 nodes 40 epochs



100 nodes 40 epocs

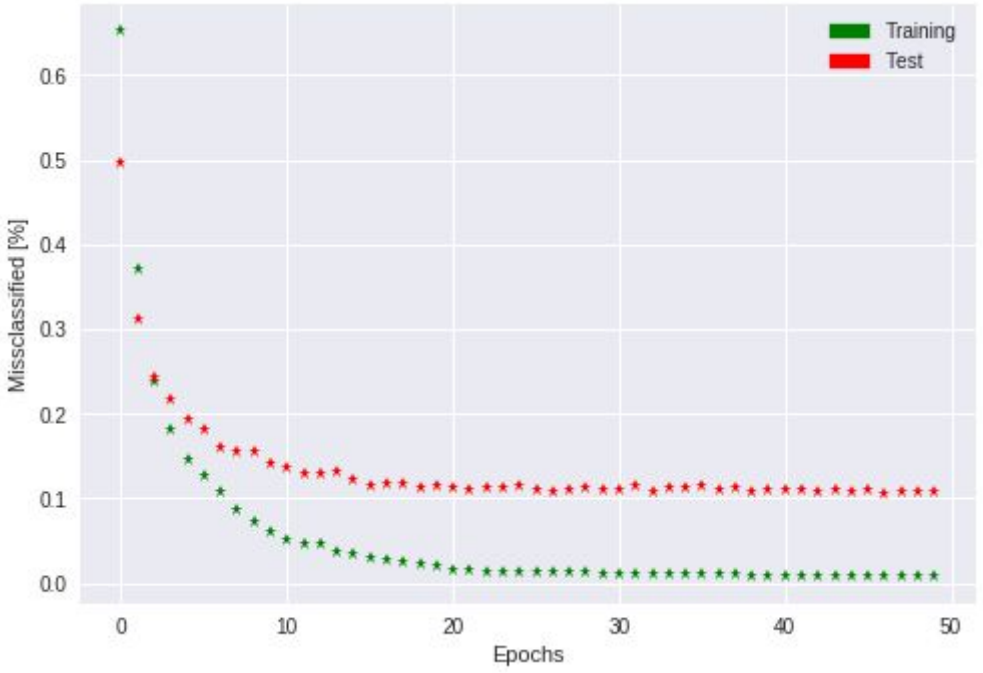
Carl Ridnert (940325-0112)
Óttar Guðmundsson (910302-0450)

3.2 DBN and stacked autoencoders for MNIST digit classification

Stacked RBM's

We choose to use 100 nodes in our first hidden layer, since the mean squared error shown in the first part of the assignment was only marginally better with 150 nodes. For the second layer we choose 80 and finally the last one had 60 nodes. We trained the networks in a greedy layer-wise fashion using the output from the previous layer as input to the next layer. Afterwards we included a final output layer of 10 nodes with a sigmoid transfer function representing the different classes of numbers. For a given image input the winning node (the one with highest output) decided the prediction. We adjusted the network using backprop after initial unsupervised training to minimize the squared error between predictions and the labels. For each epoch we test the predictive performance on a hold-out test set of data, the following plots show our results for different number of layers.

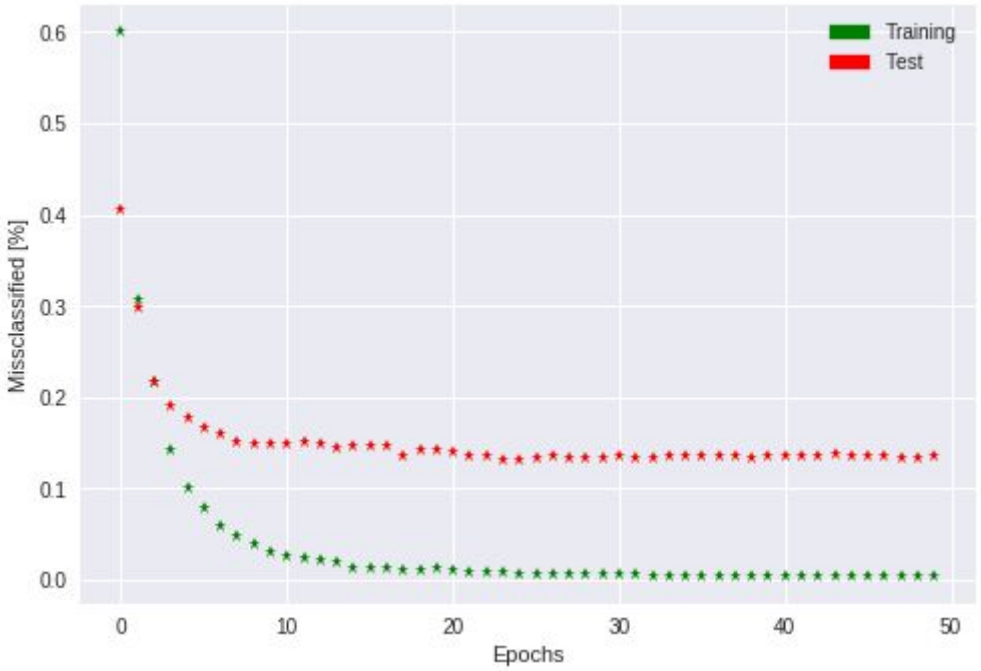
1 Layer



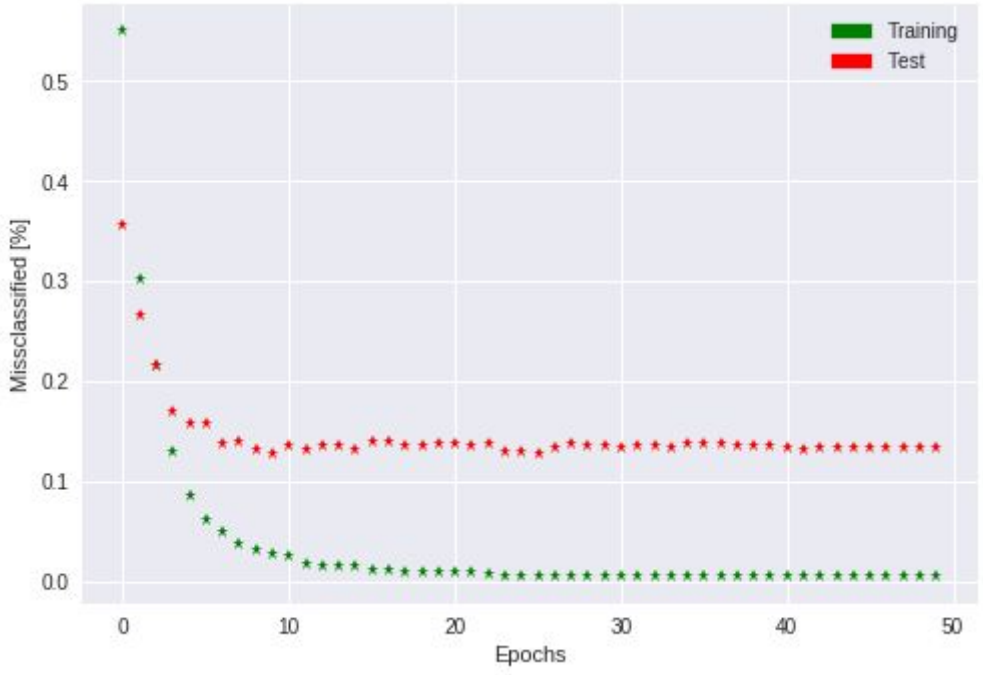
This is the model that performs the best with a misclassification rate of just above 10%. NExt we test for 2 layers.

Carl Ridnert (940325-0112)
Óttar Guðmundsson (910302-0450)

2 Layers

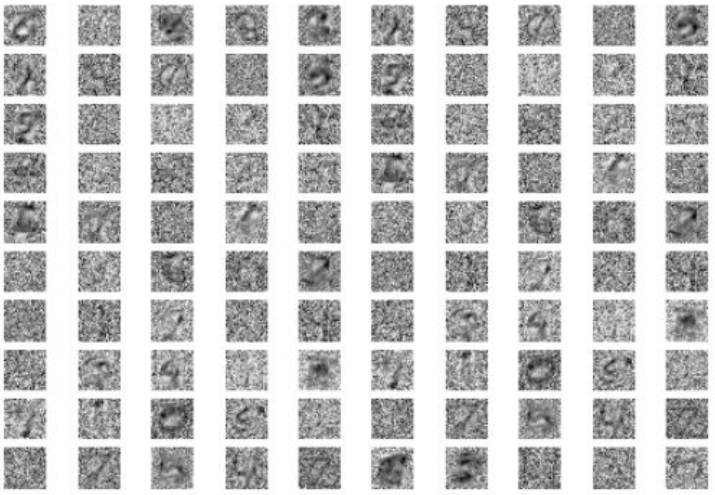


3 Layers



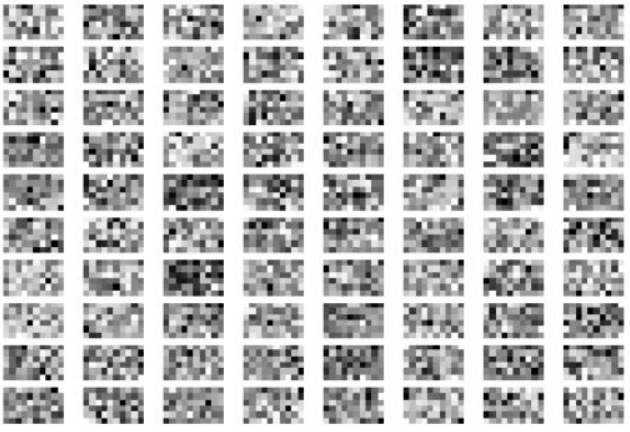
Note that the test error is roughly the same for three layer network as for the two-layer network. And that both are performing significantly worse than the one-layer network, this suggest that we are dealing with some kind of overfitting issue.

Carl Ridnert (940325-0112)
Óttar Guðmundsson (910302-0450)



First layer

Second layer



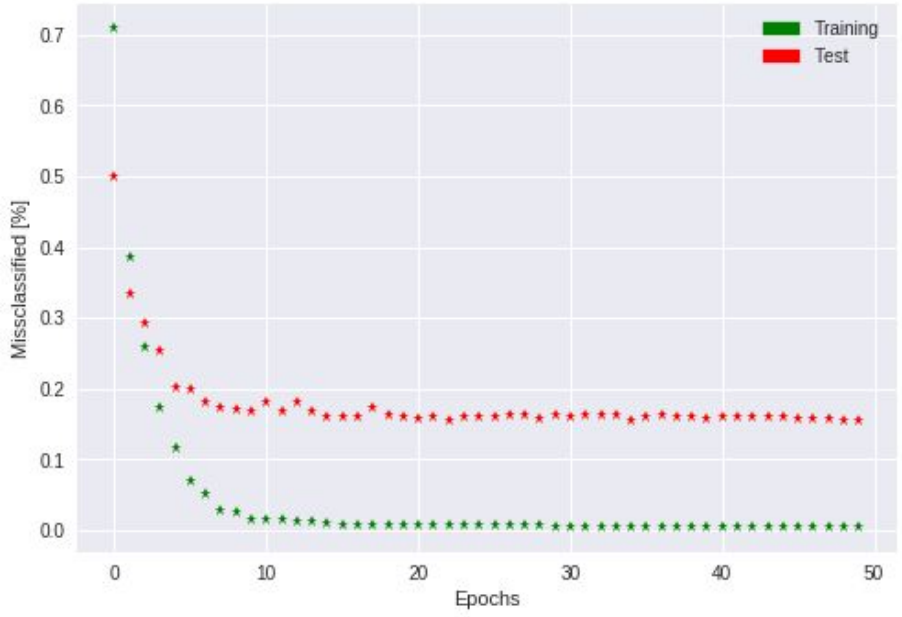
Third layer

The plots of the weights in the layers suggests that the second and third layer lack the capability of representing meaningful information about the pictures, they seem to be completely random. In contrast the first layer seems to be a good candidate for meaningful feature representation, this might be an indicator towards why we get the kind of result that

Carl Ridnert (940325-0112)
Óttar Guðmundsson (910302-0450)

we did in terms of misclassification error. Adding more random layers does not seem to increase performance on the test set.

Next we we make a comparison with a regular three-layer MLP with the weights randomly initialized, the error plot is depicted below where we can see that the performance is worse than with pre training but still quite close to the two and three layer networks.

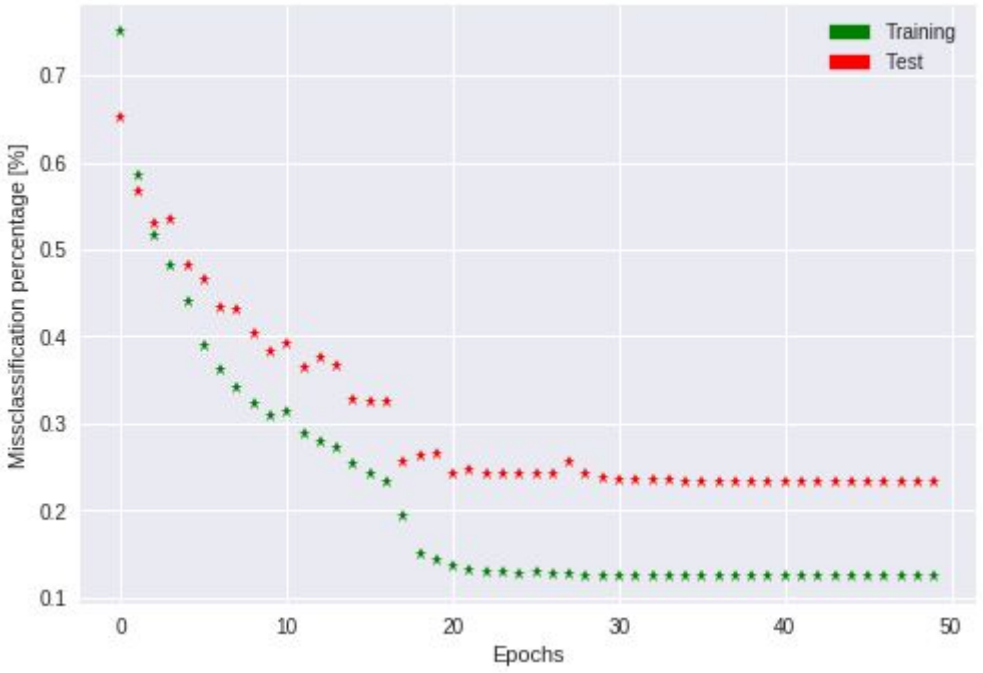


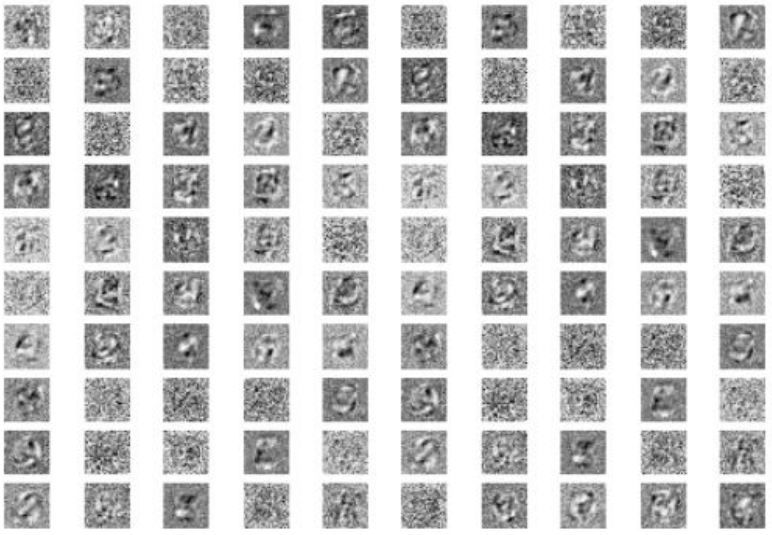
Carl Ridnert (940325-0112)
Óttar Guðmundsson (910302-0450)

Stacked Auto Encoders

Next we do a similar analysis for the stacked autoencoder. We had 100 nodes in the first layer and
We used eta 0.005 and trained for 80 epochs

1 Layers

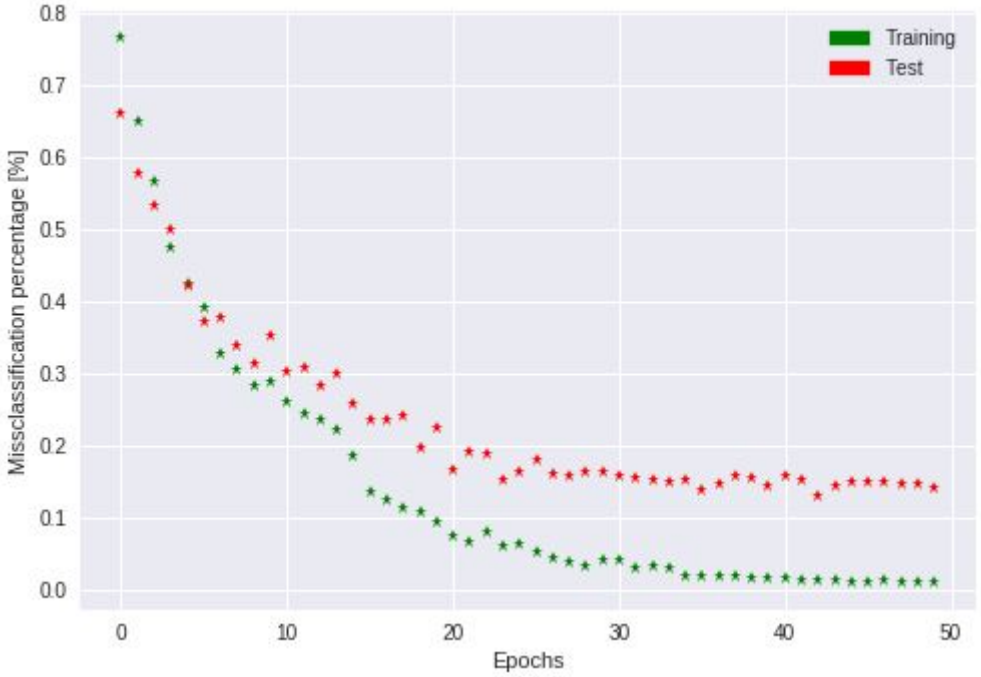


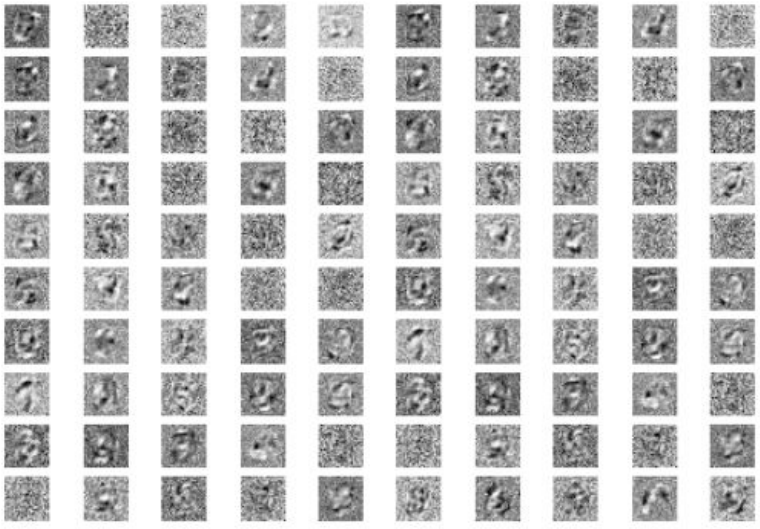


Hidden layer 1

Carl Ridnert (940325-0112)
Óttar Guðmundsson (910302-0450)

2 Layers





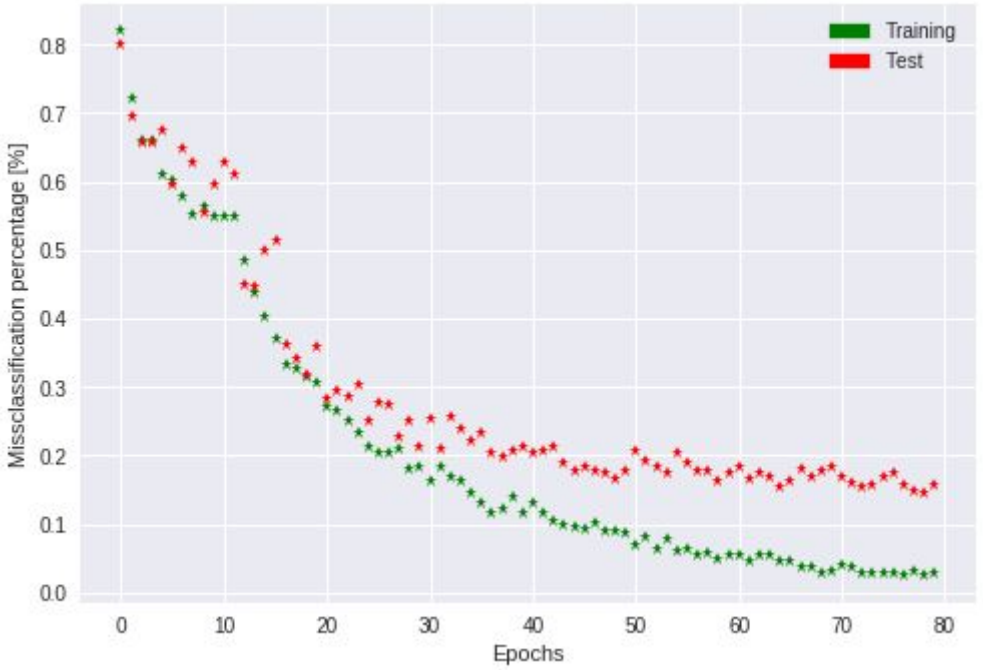
Hidden layer 2

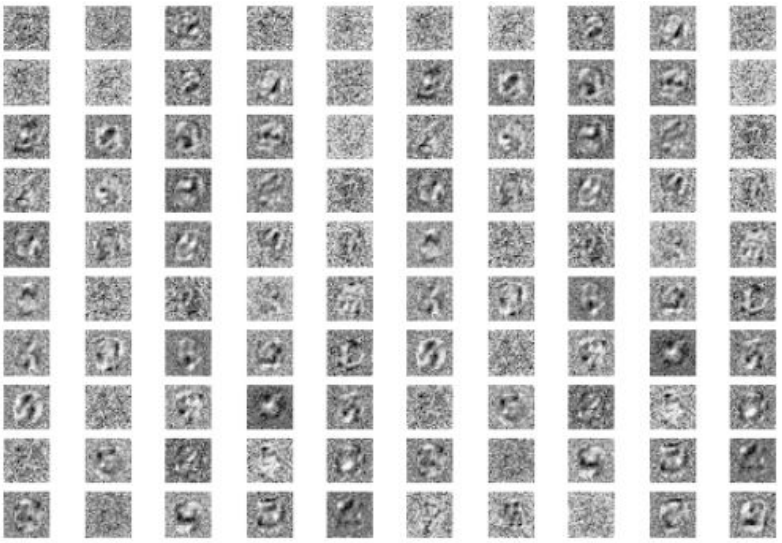
This model seems to perform a lot better than using one hidden layer misclassifying roughly 15% compared to 23%, and we see from the weight plots of the second layer that it too manages to capture something meaningful.

Carl Ridnert (940325-0112)
Óttar Guðmundsson (910302-0450)

3 Layers

Observe the effect of images representing different digits on hidden units in the hidden layers.





Hidden layer 3

We see that adding a third layer does not improve test error rates, however it does not really make it worse either. Note that the plots of weights still seem to indicate that the network is representing meaningful features. Of all these three networks, the two layer autoencoder is the best one. Next we train a MLP using backpropagation with the same architecture but using no pretraining and make a comparison.

Carl Ridnert (940325-0112)
Óttar Guðmundsson (910302-0450)

As one clearly can see from the last plot, the model without pretraining does not make a good job at all. Furthermore, the result differed and seemed to be highly dependent on how the weights were initialized suggesting that the network ended up in different local minimas which clearly are not good at all, Our conclusion is that pre training is very important for this architecture.