Artificial Neural Networks and Deep Architectures, DD2437

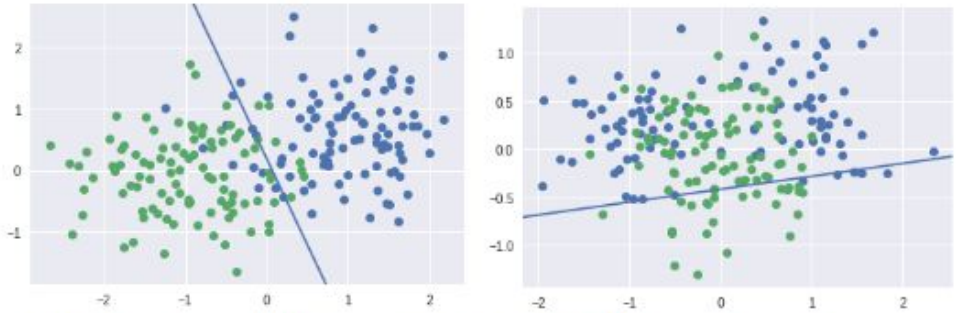
Short report on lab Assignment 1

Learning and generalisation in feed-forward networks — from perceptron learning to backprop

Victor Castillo, Balint Kovacs, and Jun Zhang

January 28, 2019

3.1 Classification with a single-layer perceptron

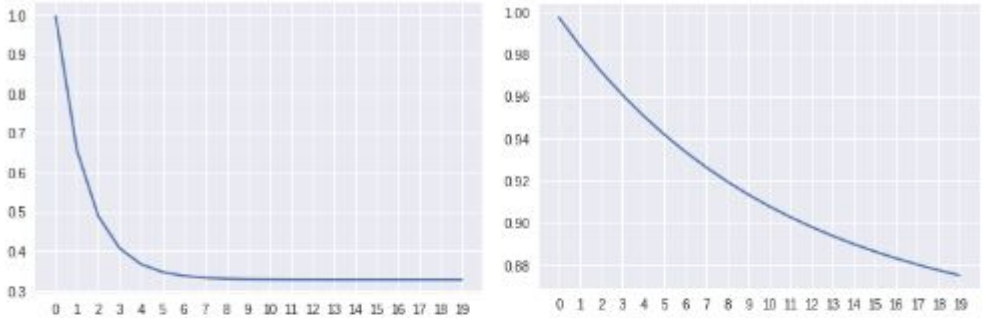


(a) Plot of linearly separable data and boundary of batch learning (delta rule)
(b) Predictions for non linearly separable data and boundary of batch learning (perceptron learning)

- The initial size of the weights is relevant for the algorithms to converge
- Using the delta rule, convergence is achieved faster (less epochs) than with perceptron rule
- Sequential and Batch learning methods have similar convergence rates if the learning rate is very small, but in the first case weights are updated with each sample and the boundary changes constantly.
- If data is not shuffled after each epoch, there is a bias in the weights after each epoch toward the last values in the dataset in the sequential case, while batch is unaffected by the order of the data.

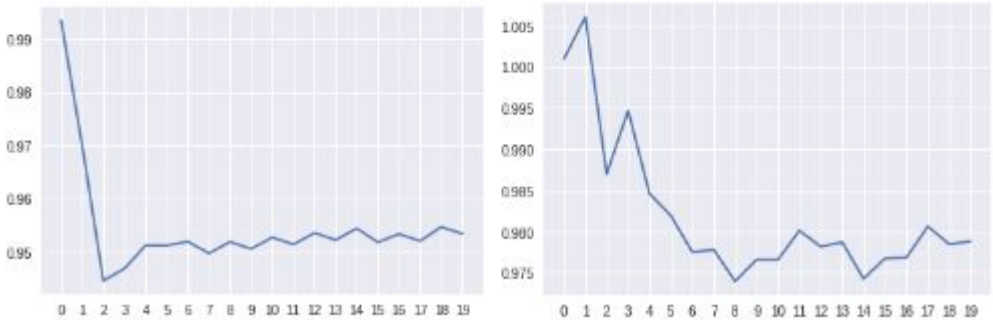
3.1 Classification with a single-layer perceptron

- While the accuracy for batch and sequential is similar, the MSE for the second has a decreasing trend, but it oscillates more.
- If the learning rate is too large (1 or 0.1), batch will end up changing from all positive predictions to all negative predictions, and weights that are very large in magnitude, with accuracy of around 50%, but not converge. Sequential, however, can converge fast being more stochastic.
- •By removing the bias, we found that the perceptron converges in the cases when the means of the two classes are on the same line with the origin and in the same quadrant



(a) for linearly separable data (b) for non-linearly separable data

Figure 2: Plot MSE with delta learning (Batch)



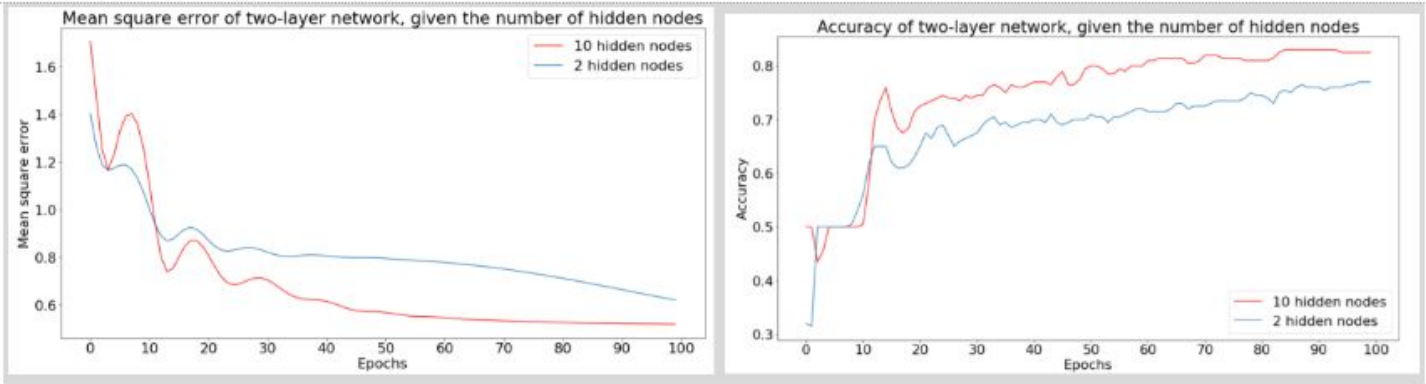
(a) for linearly separable data (b) for non-linearly separable data

Figure 3: Plot MSE with perceptron learning (Batch)

3.2 Classification and regression with a two-layer perceptron

3.2.1 Classification of linearly non-separable data

- We found that a higher number of hidden nodes can make the model reach the convergence faster, and the more complex our classifier can be, thus the more complex data it can fit.
- If we use too many hidden nodes for a simple problem, it can cause overfitting.
- When 50% of class B will be the training set, the performance is high with a simple model on the test set but low on the training set(f.e. hidden_nodes=5, epochs=20, learning_rate=0.001, alpha=0.9), with a complex model(f.e. hidden_nodes=5, epochs=100, learning_rate=0.001, alpha=0.9), the accuracy is high on the training set and low on the testset.



(a) MSE of the two-layer networks, 10 nodes (red) and 2 nodes (blue)
(b) Accuracy of the two-layer networks, 10 nodes (red) and 2 nodes (blue)

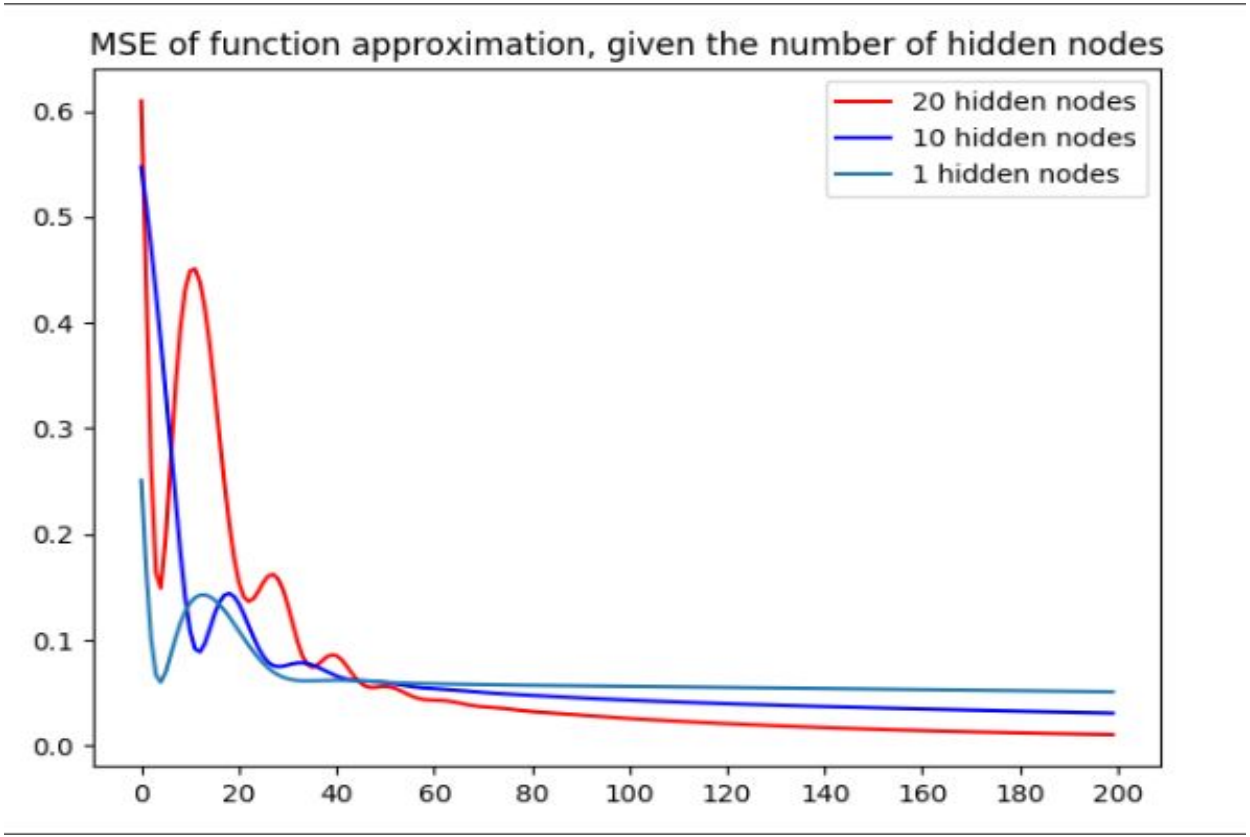
3.2.2 The encoder problem

- The hour-glass (8-3-8) shaped network is used in this case to train the network with input and output patterns that are identical, so the network itself represents a projection from the 8 dimensional space of the inputs into the three dimensional space of the internal neurons, and then back to 8 dimensional space.

- When using a hidden layer with a size of only two, we need more epochs to train a network to fit our data, or bigger step size, but it's also possible to encode the information.

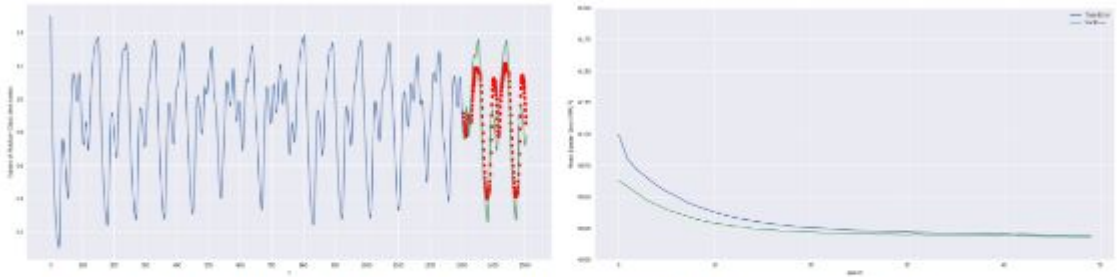
- Autoencoders seem really efficient for compressing data, or can be useful for encrypting messages as well, being able to recover the original information with the trained model, dinding compact coding of mostly sparse data.

3.2.3 Function approximation



4.1 Two-layer perceptron for time series prediction - model selection, regularisation and validation

- The best parameters found for the time-series are a 8-3-1 network training using gradient descent and sigmoid activation function.
- The chosen regularization method is weight decay (L2 regularization) and early stopping was defined for 10 epochs without improvement on the validation set.
- When we use a high value for weight decay (very strong regularization, e. g. lambda=10), the validation performance is constant, the value of the weights stay close to zero and the prediction approximates a constant around the mean of the function
- If we have few nodes, it is difficult for the network to represent the true function. With 8 nodes, convergence is faster, but overfitting on the training data is bigger. More nodes allows the model to have higher capacity, but needs more regularization to prevent overfitting on the training data and generalize better.

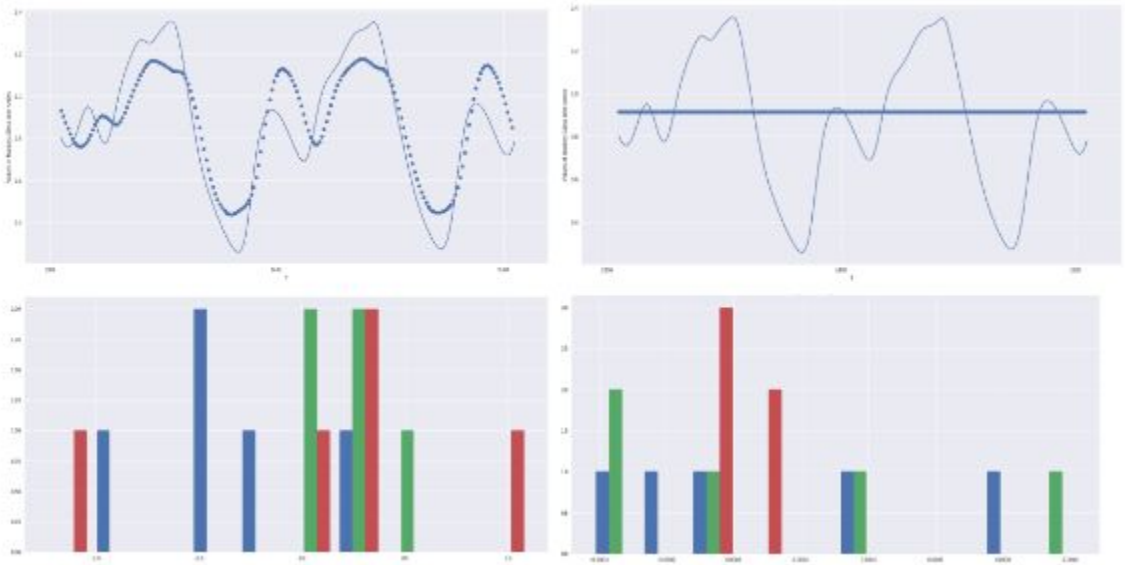


(a) Training (+Validation 0.2%), Testing data and Predictions, Two layers, no noise, no regularization

(b) Training (blue) and Validation MSE over epochs

Figure 8

4.1 Two-layer perceptron for time series prediction - model selection, regularisation and validation



(a) Two layers, no noise, no regularization
(b) Two layers, no noise, high regularization (lambda = 10)
(c) Weight histogram with no regularization (range -1 to 1)
(d) Weight histogram with lambda = 10 (range -0.0004 to 0.0010)

Figure 9

4.2 Comparison of two- and three-layer perceptron for noisy time series prediction

- In this part, we appreciate the effects of noise for generalization of the network. If the noise is high (0.18) and there is no regularization, the prediction tries to follow the noise very closely, meaning a higher variance.
- With enough regularization (more is needed as the noise increases), the prediction becomes smoother and similar to the true function, even if the MSE increases and the weights are kept small. As is the case with two layers, too much regularization and the prediction is almost a flat line around the mean of the function.
- The intuition here is that regularization could be helpful for noisy data sets. A two-layer network have a natural limit to the complexity that it can handle which helps it prevent overfitting with the noise and regularization becomes much more important in the three-layer network.
- Computational cost (time) increases with the number of nodes*layer.

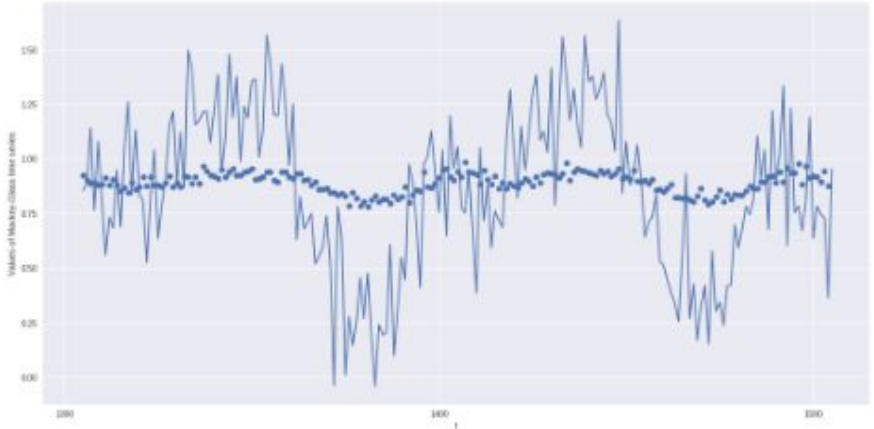


Figure 10: Three layers (5-5-3), noisy data (0.18), low regularization (0.01)

5 Final remarks

The insights gained in this lab from understanding how different parameters affect neural networks was invaluable. Some of the most interesting points to learn were how to closely approximate a true function without overfitting by using regularization, how noise can affect predictions and how important the representation of the data is for the algorithms to perform better.