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

Ivan Stresec, Frano Rajic

September 11, 2020

1 Main objectives and scope of the assignment

Our major goals in the assignment were

- to investigate the performance of classification single-layer perceptrons (SLPs) in various settings
- to observe the capabilities of MLPs in the context of solving non-linearly separable patterns in both classification and regression problems
- to implement MLPs in a chaotic time-series prediction[1]

2 Methods

For the first assignment we used Python as a programming language and its library Numpy for data representation and matrix operations. For graphs we used the Matplotlib library. In the second assignment we used Pytorch, as well as the aforementioned libraries. As an IDE, we've used JetBrains' PyCharm for both assignments.

[1] FIX THIS ONE

3 Results and discussion - Part I

3.1 Classification with a single-layer perceptron *(ca.1 page)*

Combine results and findings from perceptron simulations on both linearly separable and non-separable datasets. Answer the questions, quantify the outcomes, discuss your interpretations and summarise key findings as conclusions.

In solving classification problem using an SLP one can easily observe its efficiency and its limitations.

Firstly, we have implemented the perceptron rule (implying a sequential learning scheme) and the Delta learning rule (batch learning) to a randomly generated, linearly separable classification dataset. Both have performed well and converged quickly...

Secondly, we have tested the differences between batch and sequential learning for the Delta rule SLP. In terms of epochs, both approaches converge relatively quickly, with ... being somewhat quicker

Thirdly, we have tested removing the bias term with the Delta rule in batch mode. Even without testing it, it is clear that the algorithm can converge only when the data is linearly separable by a line which goes through the center of the coordinate system.

3.2 Classification and regression with a two-layer perceptron *(ca.2 pages)*

3.2.1 Classification of linearly non-separable data

3.2.2 The encoder problem

3.2.3 Function approximation

4 Results and discussion - Part II *(ca.2 pages)*

For this assignment, we have used the Mackey-Glass time series to evaluate a two-layer perceptron network. The data was simply generated by using the starting conditions and sequentially calculating values using the iterative formula given in the assignment. Furthermore, the data was split into three consecutive non-overlapping blocks for training, validation, and testing using 800, 200, and 200 values, respectively. For the regularisation method we have used weight decay with the L2 norm. Early stopping is implemented in such a way that if improvement (by at least 10^{-5}) in the validation mean square error (MSE) is not visible for 50 iterations the learning stops. Also, the maximum number of epochs allowed is capped at 10000.

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

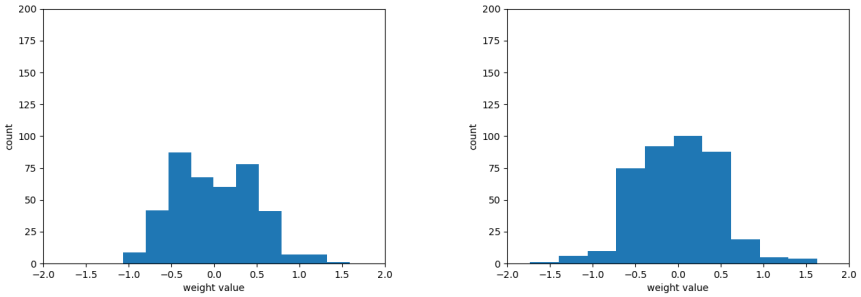
Using a rate of learning rate of 0.1 which showed good results and relatively fast convergence as well as a medium regularization coefficient $\lambda = 0.1$ several two-layer perceptrons were tested, the results of which we can see in table 1. Each configuration was tested 10 times to calculate mean values and standard deviation. We opted for the networks with 8 nodes in the hidden layer as those show the best results on the test set.

nodes	MSE mean	MSE std
1	0.01413	0.01153
2	0.00916	0.00188
4	0.00870	0.00384
6	0.00806	0.00173
8	0.00774	0.00135

Table 1: Evaluation of various network configurations

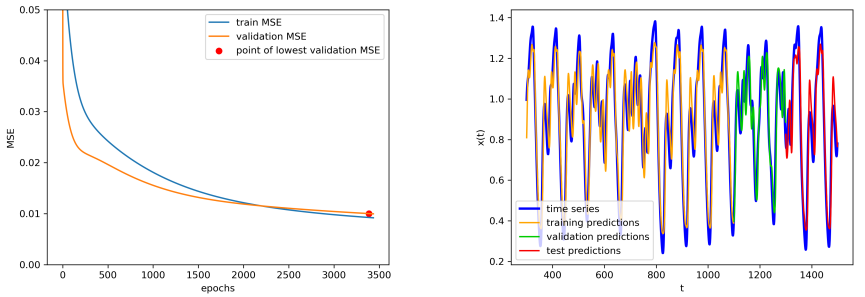
Furthermore, we have tested the impact of regularization strength on the selected network. Generally, one could expect that the weights of a network with stronger regularization tend to be closer to 0, but this effect was not obvious in our case, probably due to the fact that our networks are too simple to have problems with overfitting and larger weights. The described effect can to some extent be seen in figure 1.

Finally, figure 2 shows an example of the learning process and the outputs of our selected network with 8 nodes in the hidden layer, a learning rate of 0.1 and an L2 weight decay with $\lambda = 0.1$. The MSE on the test set of such a network can be seen in table 1. The best of the 10 tested networks had an MSE of 0.00621 which demonstrates its good generalisation capabilities.



(a) No regularization (b) L2 regularization $\lambda = 0.1$

Figure 1: Weight distributions



(a) Learning process (b) Predictions after training

Figure 2: Weight distributions

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

5 Final remarks *(max 0.5 page)*