Single and Multi-layer perceptrons for classifications and predictions of time series

Henrik Åhl

December 4, 2015

Department of Astronomy and Theoretical Physics, Lund University

Project supervised by Mattias Ohlsson

LUND UNIVERSITY

If you can't go with the scientific argument –
go with the non-scientific argument.
— Mattias Ohlsson, 2015

1 Introduction

Neural networks can be used for a wide range of purposes, ranging from function approximation to identifying patterns in images. Although the networks themselves can be constructed to be however complicated the creator wishes, relatively simple networks can be shown to be able to solve complicated tasks. Especially single-layer perceptrons, as are the main study in this report, are able to perform surprisingly complex tasks.

2 Problems

2.1 XOR logic

In the XOR problematic, it is easy to realize that the problem cannot be solved by a perceptron without hidden nodes; the reason being best visualised graphically, where it can be seen that the decision boundary for the network is constructed by a single linear function in the x_1–x_2 plane. Since the desired outputs are not separable by a linear function, it is impossible for the network to solve the problem.

Using simple gradient descent learing, i.e. without adaptive learning rates and momentum included, the network is contrary to the intuitive idea not able to learn to circumvent the problematic. Out of 1000 simulated networks, not a single one produces a solution to the problem. When instead introducting different mechanics to the learning algorithm, the results found in table 1 display the performances measured on different version of the Gradient Descent (GD) method, as well as the Levenberg-Marquardt (LM) method.

Method	**Success rate** [%]
GD	0.0
GD + momentum	0.0
GD + adaptive learning rate	25.3
GD + both	29.1
LM	49.1

Table 1: Results when trying to solve the XOR logic with a neural net constructed by two hidden nodes. Every method experienced 1000 simulations.

When instead increasing the number of nodes, all methods but the ordinary GD algorithm, as well as the GD with a momentum modification, produce results rapidly converging towards 100 % correct classifications. Already at 4 hidden nodes, all but the original GD and the momentum method give results with >95 % correct classifications over 1000 simulations. The ordinary GD method instead only produces the correct result 2 % of the time with as many as 50 hidden nodes. Similar results apply for the momentum version of the algorithm.

2.2 Synthetic Data

Using the second given synthetic data set, a graphical visualization of the data, as can be seen in section 2.2 helps understand why the problem is solvable by a network with a single node; clearly the two classes are clustered in such a way that a linear boundary is sufficient to discern the class of the data in an adequate way. Plotting the full data set (not here depicted) further shows the nature of this specific feature of the data.

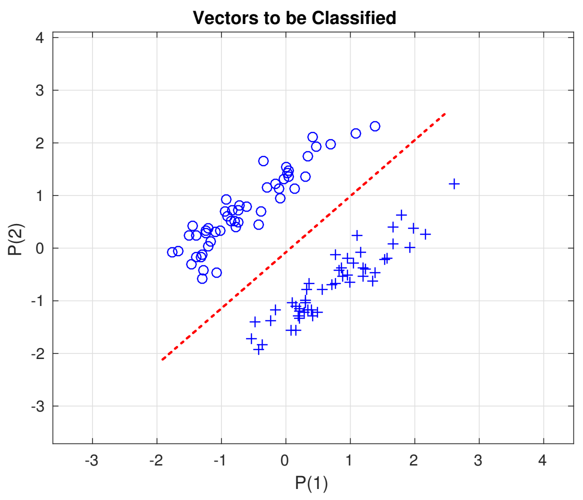

Figure 1: Graphical results of the training performance for the synthetic data set. The linear function shows the decision boundary produced by the neural network.

Also when performing the analogue analysis on the first data set, a visual representation is beneficial in understanding the problematic. In section 2.2 the validation output of a 15 node network can be seen, which highlights the significant overlap between the two classes.

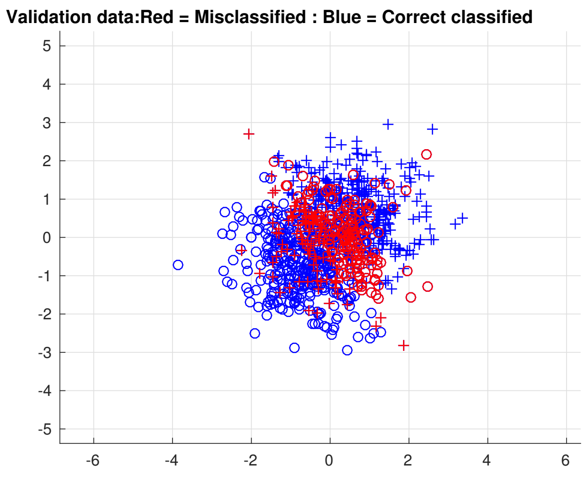

Figure 2: Depiction of the problematic in trying to discern the two classes from each other. The graph itself shows the validation performance of a 15 node network.

In examining the occurence of overtraining, as can be investigated in comparison between fig. 3 and fig. 4, it can somewhat unintuitively be seen that although it might be so that the network indeed adapts to individual data points more and more as additional nodes are added to the network, it does not seem to be biased so heavily that the approximation turns downright bad. Indeed, single data points can be noted to have greater weight with more nodes, although since the number of weights are increased so much between the two graphs, it would seem natural if more overtraining were present.

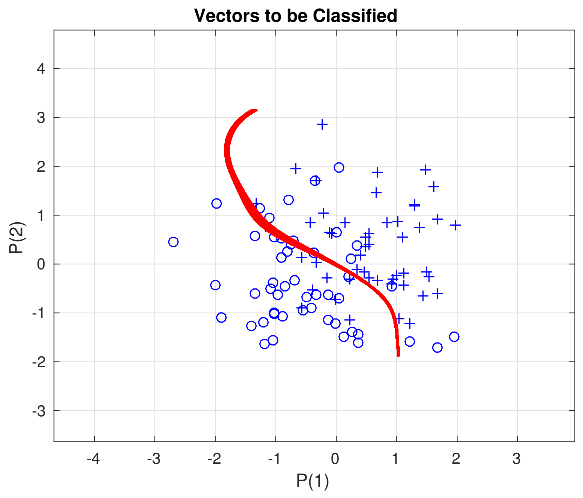

Figure 3: Classification output for synthetic data set 1, with 12 hidden nodes.

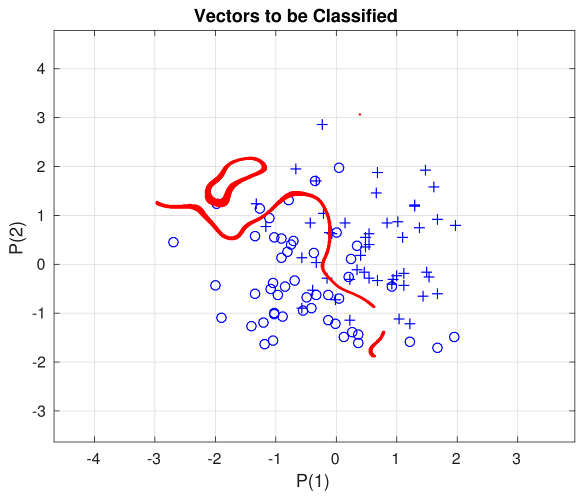

Figure 4: Classification output for synthetic data set 1, with 100 hidden nodes. Note that there are vague signs of overtraining relative to fig. 3.

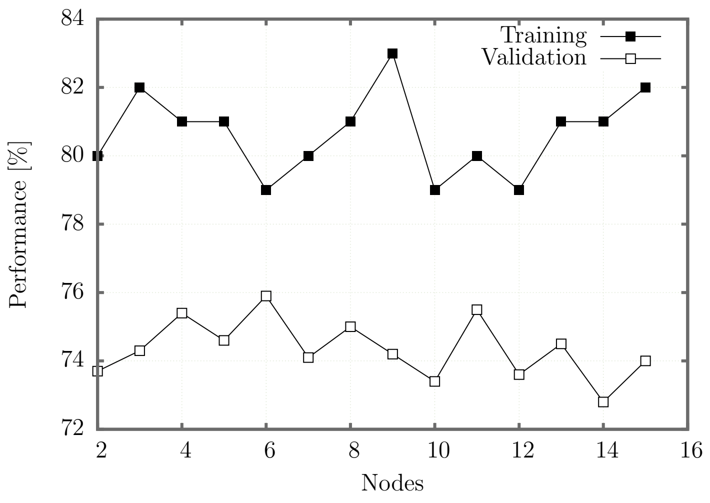

Figure 5: Training versus validation performance (fraction of correct classifications) on synthetic data set 1. Note that the validation performance is consistently lower than the training one.

Increasing the number of nodes used seems not to affect the overall performance in any determinable way, as fig. 5 signifies. Instead, the performance fluctuates stadily around the same output as the linear MLP manages to produce.

For the final synthetic data set, one can at observing the data set itself realize that it ought to require at least three hidden nodes in order to classify the network properly. This is because of the enclosedness that forelie, where one class is wholly enclosed by the other in the data plane; since it would require three straight lines in order to enclose an area in a plane, it would seem natural that at least three nodes are required. The conclusion is solely based on the fact that a node in a network principally represents a linear delimiter in the input space.

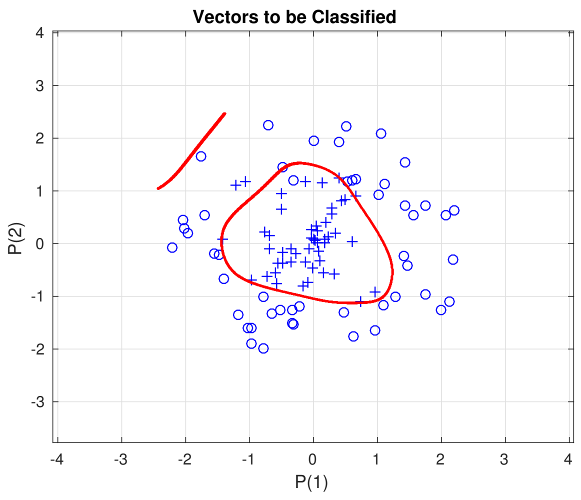

Figure 6: Training sample of synthetic data set 3. Graph shows training classification for a 15 node network.

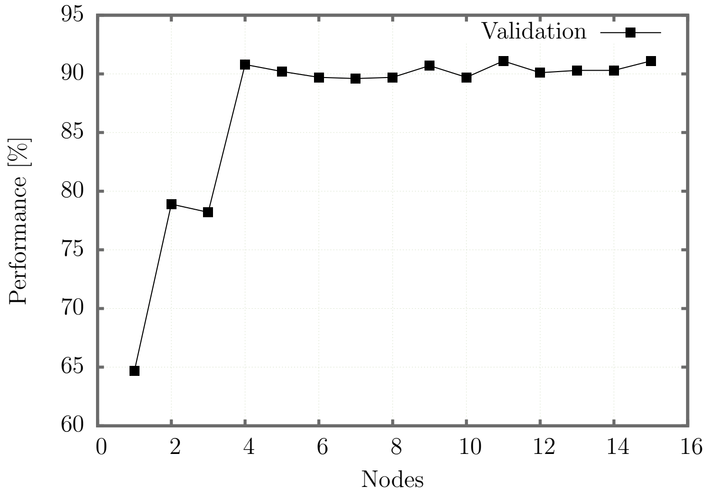

Figure 7: Validation performance for different sizes of a single-layer MLP on synthetic data set 3.

It can be noted from the validation performance chart in fig. 7 that as long as at least 4 nodes are present, the network is able to correctly classify the data to a large extent. It can principally be assumed that there would be an increase in performance the more nodes are added to the network (over a sample of simulations), although some error is bound to be present due to the slight overlap between the data sets. It ought to be probable that the convergence value inherent to the problem is close to the plateau visible in the performance graph however.

2.3 Liver Data

For the liver data set, all different methods available for the exercise were tested, simulated with nodes ranging from one to 15. As the data was assumed to be supposed to classify healthy/non-healty individuals, the specificity, i.e. the measure corresponding to the ability to find as many (again: assumed) non-healhty individuals, was chosen. Of all the methods, the Scaled Conjugate Gradient method with a single layer and 11 nodes was the must successful in this regards, being able to find 45.45 % of the data belonging to class 0 (non-healthy). This should however be put in relation to the sensitivity, which assumed a value of 84.06 %. In other words, the network did not wrongly classify all too many “healthy” individuals. It must nontheless be stressed that the variance for all the methods was high in relation to the performance, so that further results would be needed to draw realistical conclusions about the preferable algorithm.

2.4 Time Series

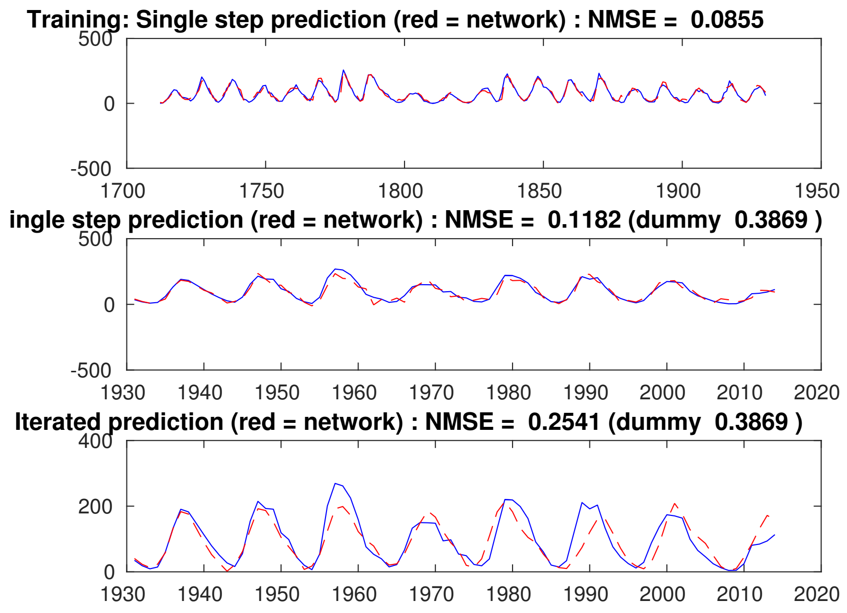

Figure 8: Performance for single-layer network with 7 hidden nodes and the LM algorithm on the sunspot data.

In finding an underlying pattern for the sunspots, who seem to exhibit an oscillating behaviour, peaking roughly every eleven years, all methods were again tested on the data set. The most efficient proved to be the Levenberg method, which managed to achieve a single step error of 0.12, as well as an iterated step error of 0.25 (although many other nets came close to this performance). Indeed, as can be seen in fig. 8, the output corresponds to a sinusoidal function with rougly uniform peaks and valleys, as ought to be expected. Interestingly, from other simulations, the most prominent trouble for the networks proved not to be the ability to exhibit oscillating traits, but rather to be stable over time.

Again, however, like in several of the previously investigated topics, the variance between the nets proved to be significant over several simulations. In order to rigorously determine the best architecture and algorithm for a net, it ought therefore to be necessary to perform ensamble analyses instead of only single-network comparisons such as here.

2.5 Mackey-Glass

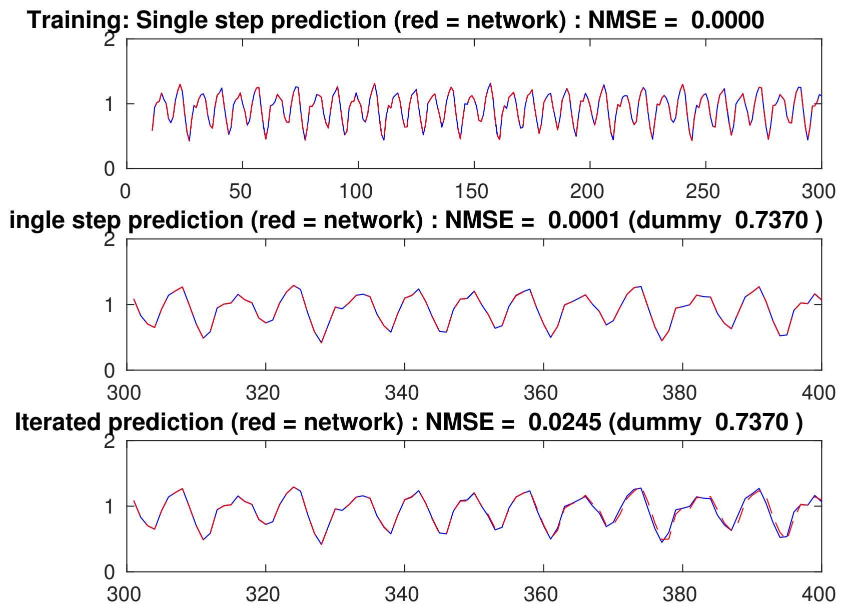

Figure 9: Performance for single-layer network with 21 hidden nodes and the LM algorithm on the Mackey-Glass data.

For the Mackey-Glass data set, the output proved to be significantly easier to predict than in previous cases. Naturally, this ought to be because of the underlying function here being definite, which results in the fact that the training set is not only periodically the same, but also identical to the test data set. It is thereby of no great surprise that the error achieved is as small as it happens to be. The architecture does furthermore not seem to play a significant role in this case, as many investigated nets proved to be equally able to predict the test data.

2.6 Challenge Exercise

With respect only to the ability to minimize the fraction of missed classifications, a 4-fold cross validation was implemented. Uses of a larger amount of folds rendered results not as good as the 4-fold, which ought to be because of the slim size of the data set (only 384 data points in total, rendering a validation set of 96 points in a 4-fold cross-validation). It is however questionable which measure of performance should be used in evaluating the networks;

in principle, the area under the ROC curve can be calculated, although it proved practically a hard feat to do so. In general however, it would seem as if the back-propagating learning algorithms provide better networks on average.

Out of all the algorithms and networks simulated, the best was the 8-node network utilizing the Resilient Backpropagation method, with a correct match performance of 76.3 %. (Data included in attached file output.dat.)

Finally, when evaluating the correlation between the different input factors and the targets in the training set given, one can see that almost all dimensions seem to have very small direct effect on the target result, aside from one single factor (which like all the others is unknown), that instead has a clear linear trend relative to the target data. This however requires contextual consideration, as several of the non-correlated factors might have a combined effect on the outcome. Nontheless, the direct correlation between the single factor mentioned ought to be of importance.

3 Concluding remarks

It is fascinating to see how a fairly simple neural network can perform so well at many complex tasks. Nevertheless, the results here shows that it is perhaps more than anything important to evaluate an ensemble of networks in order to be able to draw conclusions about an architecture's or algorithm's performance.

Although many of the more advanced learning algorithms were significantly slower from a computational perspective, the ability of many of them to predict as well time series as functions provided a large incentive to indeed use them for many of the tasks.

In summary, the capabilites of neural nets are nothing less than extraordinary – especially in the way that such a vast variety of different problems can be solved by so similar structures and architectures.

References

[1] Mattias Ohlsson, *Lecture notes, Artificial Neural Networks*, Department of Astronomy and Theoretical Physics, Lund University, 2015.

[2] Mattias Ohlsson, *Exercise 1: Perceptrons and Multi-layer perceptrons for classification and prediction of time series*, Department of Astronomy and Theoretical Physics, Lund University, 2015.