Artificial Neural Networks and Deep Architectures, DD2437

Short report on lab assignment 2

Radial basis functions, competitive learning and self-organisation

Mustafa Al-Janabi, Einar Lennelöv - Group 15

February 11, 2020

1 Main objectives and scope of the assignment

Our major goals in the assignment were

- Investigate the properties of RBF networks without CL on 1d data.
- Investigate the effect of using CL in an RBF network on 1d and 2d data.
- Investigate the applicability of SOMs on the tasks of ordering animals, the travelling salesman problem, and clustering MPs.

We have focused on qualitative findings that help in the understanding of the algorithms, as opposed to more rigorous numerical results.

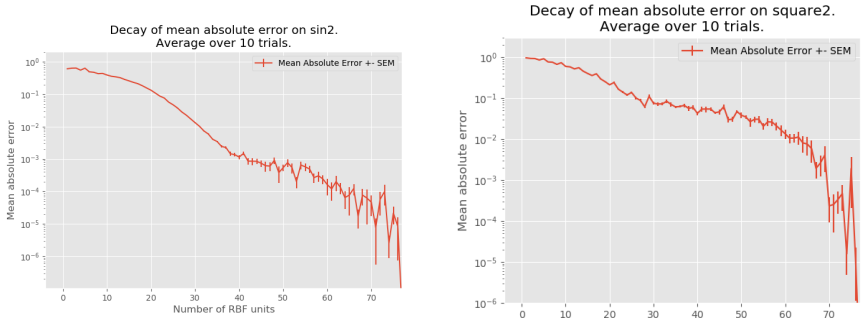
2 Methods

All of the code was written in Python. The data sets used were those provided in the assignment.

3 Results and discussion - Part I

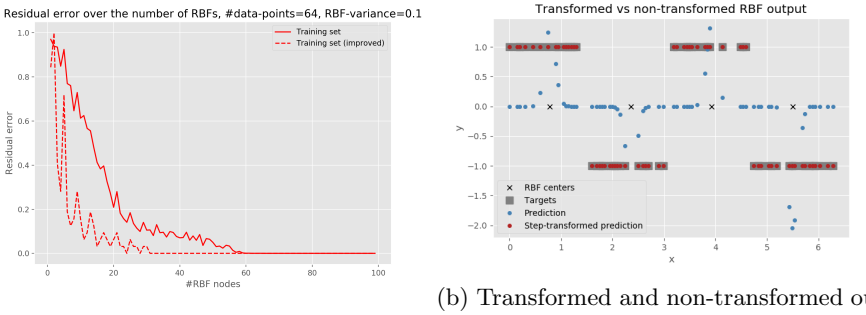
3.1 Batch mode training using least squares - supervised learning of network weights

A critical parameter when designing the RBF network is the number of RBF nodes. We begin by analysing how the mean squared error (MSE) on the training data depends on the number of RBF nodes. Figure 1a shows the plot for



(a) (b)

Figure 1: Mean squared error plotted against number of RBF nodes.



(a)

(b) Transformed and non-transformed outputs on the square data set.

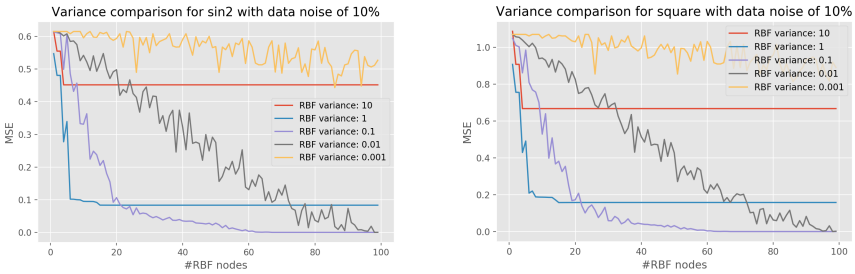
Figure 2: Mean squared error plotted against number of RBF nodes.

the sin2 function, and Figure 1b shows the corresponding plot for the square function. The centers are linearly spaced.

We focus now on approximating the square function. One simple yet effective trick is to transform the output with the sign function. Figure 2a shows the effect of this transformation with linearly spaced RBF's, showing some improvement. However, by manually placing the RBF's we can get significantly better performance. Figure 2b clearly shows this by plotting the target, output, transformed output, and RBF centers. When the centers are placed intelligently we are able to achieve 0 MSE with only four RBF nodes. This might be useful for classification when domain knowledge provides good placements for the RBF's.

3.2 Regression with noise

Another important parameter when designing RBF networks is the width. Figure 3 shows the MSE on the training data plotted against the number of nodes for various widths of the RBFs. With too large width, the model is unable to properly fit to the data. We can also clearly see that smaller widths lead to slower convergence.



(a) (b)

Figure 3: Comparison of RBF widths.

(a) (b)

Figure 4: Comparison of RBF widths.

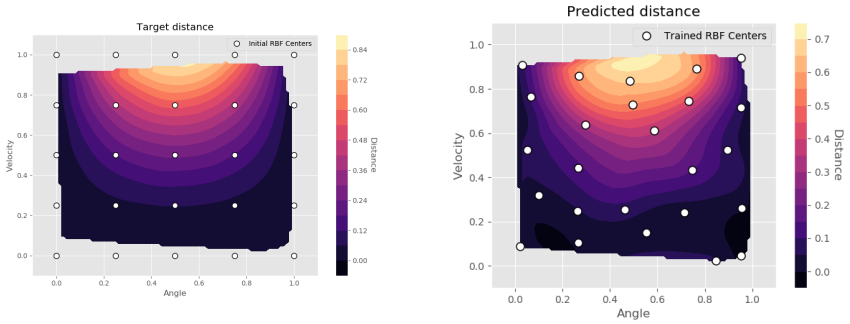
3.3 Competitive learning to initialise RBF units

So far we have only considered training the output weights of the RBF networks. However, we have previously seen that manually placing the RBF centers can result in massive improvements. Therefore it makes sense to also learn the locations of the RBF centers. In this section we apply competitive learning to this task.

We begin by also addressing the issue of dead units. Dead units occur when an RBF center lays outside the range of the data, causing it to never be updated. We apply the simple strategy of using leaky updates. While the winning center is moved an η-scaled step in the direction of the data, all other units are also moved an $\frac{\eta}{100}$-scaled step.

Figure 4a shows the resulting predictions when the ordinary CL update rule is used. There are two dead units on the sides, and the remaining two units lack the capacity to properly represent the function. Figure 4b shows the predictions when the leaky update rule is used. All centers are now moved into the range of the data, leading to a much better prediction.

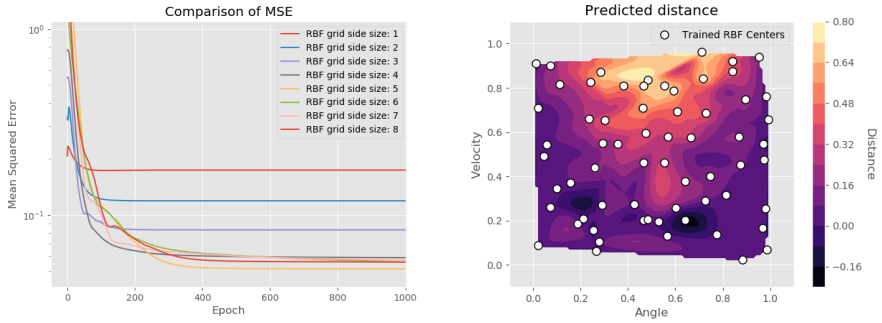
We now turn to a 2-dimensional problem, with 2d inputs and 2d outputs. In



(a) Non-leaky CL.

(b) Leaky CL.

Figure 5



(a)

(b)

Figure 6

order to make plotting feasible we here examine only one of the two outputs, although the model was trained using both.

Figure 5a shows the contour plot of the test data of the target function, along with the initial placements of the RBF centers. Figure 5b shows the prediction by the trained model on the test data, as well as the trained RBF centers. The prediction is reasonably good, but it is sensitive to the values of the hyper parameters.

Figure 6a shows the convergence of the MSE on the test data when the RBF centers are spaced on a $N \times N$ grid.

An example for a poorly performing arrangement can be found in Figure 6b. In this case, the width of the RBF is too low, despite and the increased number of nodes makes the problem even worse.

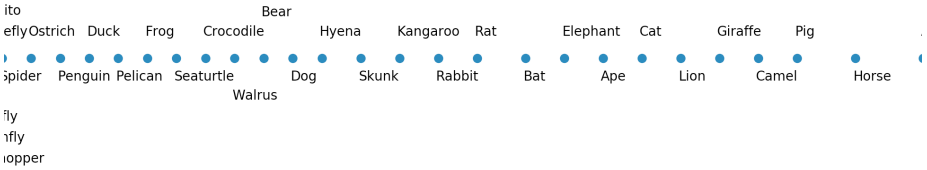


Figure 7: Ordering of animal species.

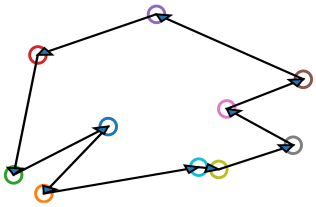


Figure 8

4 Results and discussion - Part 2

The concept of moving nodes is also useful for simplifying high-dimensional data into lower, visualizable dimensions. We examine three examples in this section.

4.1 Topological Ordering of Animal Species

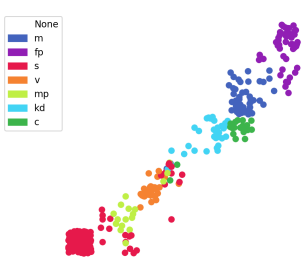
Figure 7 shows a one-dimensional linear topology applied to an 84-dimensional dataset of animals. The results intuitively make sense, with insects and non-mammals on the left side, and mammals on the right hand side.

4.2 Cyclic Tour

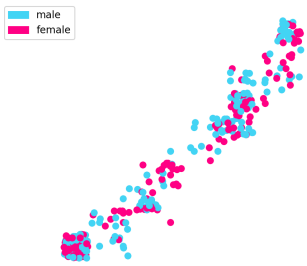
Figure 8 shows a one-dimensional cyclical topology applied to the traveling salesman problem. As expected, the SOM resulted in a decent, although not necessarily optimal tour (as made evident by the detour to the blue node). There were some issues in avoiding repeated node visits, and they were circumvented by simply redoing the SOM when the result was deemed invalid.

4.3 Data Clustering: Votes of MPs

Figure 9 shows a two-dimensional grid topology applied to a 31-dimensional dataset of MP votes. The left hand figure shows the party of each node, while the right hand figure shows the gender of each node. There is an obvious clustering



(a) Nodes colored by party.



(b) Nodes colored by gender.

Figure 9: Clustering of MPs based on voting patterns.

with regard to party (as expected), while there is no obvious clustering with regard to gender.

5 Final remarks

We enjoyed the assignment, especially the questions that required some additional thought to complete (such as the 2d part of 3.3, and 4.3).