Artificial Neural Networks and Deep Architectures, DD2437

Short report on lab assignment 2

Radial basis functions, competitive learning and self-organisation

Maximilian Auer, Lukas Frösslund and Valdemar Gezelius

September 24, 2020

1 Main objectives and scope of the assignment

Our major goals in the assignment were to familiarize ourselves and get comfortable with the detailed implementation of RBF networks as well as SOMs for both supervised and unsupervised learning tasks. An additional goal was to utilize not only error based learning, but also competitive learning.

2 Methods

For all parts of the lab, Python 3 was used together with NumPy for mathematical operations and Matplotlib for plots. For some parts, Pandas was also used for data manipulation.

3 Results and discussion - Part I: RBF networks and Competitive Learning

3.1 Function approximation with RBF networks

We conducted experiments where we varied the number of RBF-units from 1 to 25, placed evenly spaced and studied the residual error. Figure 1 shows the resulting graphs for both batch- and sequential learning. We used σ-value 0.3 and η-value 0.1 for these experiments. On the left (batch learning) we can see that for both clean- and noisy data, we receive a lower error for approximating the sin(2x)-function, than the square(2x)-function. The same conclusion can be drawn from the sequential learning graph to the right, where we see that over 15 RBF-units doesn't seem to lower the error significantly. The conclusion we can draw from clean vs noisy data is that we generally see a higher error for the models trained with noisy data.

We can reduce the error on the square(2x)-approximation by threshold the error calculation to be 1 if the output is >=0 and -1 otherwise. We see that we need about 7 units before it reaches an error of 0. This approach can be used when dealing with discrete valued functions.

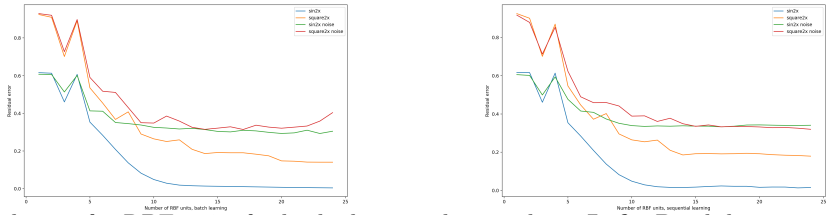
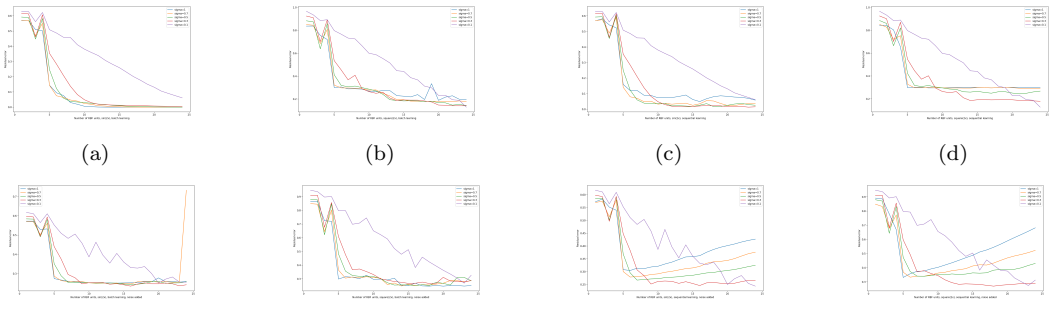


Figure 1: Residual error for RBF-units for both clean- and noisy data. **Left:** Batch learning. **Right:** Sequential learning.

For clean data with batch learning we see that if we increase the σ-value, we get a lower error when predicting the sin(2x)-function, it also converges faster in regards to number of units. With σ-value 0.7 we get an error below 0.1 for about 6 units, below 0.01 for about 12 units and below 0.001 for about 16 units, with a minimum of $\approx$2.06-05 reached around 24 units. Here we get a minimum error for square(2x)-function of $\approx$0.18 reached around 21 units.

In Figure 2 we see the the residual error for σ-values 0.1-1 as a function of number of RBF-units. The graphs show both approximation of sin(2x) and square(2x) with both clean- and noisy data. Conclusions that can be drawn is that for σ-values 0.5-1 we reach a low error with about 5 units, for both noisy- and clean data. When we have <5 units, the units spread doesn't cover the data and we are not able to make good predictions. When we have low σ-values the spread of the units are to low and doesn't cover the data well. We see some oscillations in the error curves for noisy data, Figure 2 (e-h). The sequential learning seems to have a problem with increasing errors for high σ-values with noisy data, Figure 2 (g-h). σ-value 0.1 has a decreasing linear curve and reaches about the same residual error but demands more RBF-units. We see that we reach ultimate results when the spread of the units cover the data well, without overlapping.



(a) (b) (c) (d)

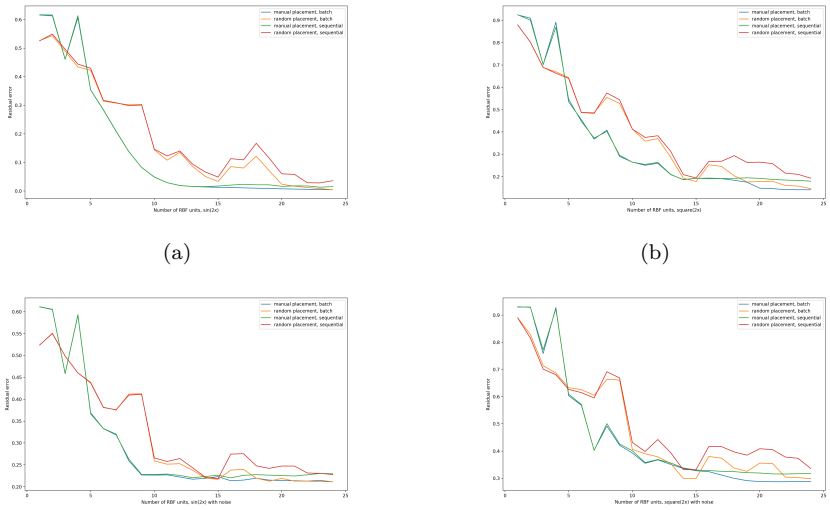
(e) (f) (g) (h)

Figure 2: Residual error for RBF-units with different σ-values. **a-d:** no noise. **e-h:** noise added.

When we studied the how the selection of *eta*-value impacted the training error in sequential learning, we saw that, for both sin(2x) and square(2x), the error curve converges in 2-5 epochs for *eta*-values 0.05-0.8 with no significant difference between them.

Figure 3 shows the residual error as a function of RBF-units regarding manual- or random placement of the RBF-centers, the μ-values. Our approach to manually distribute the units was to place them with even space between. We can see that, for both approximation of the sin(2x)-function and square(2x)-function, generally a manual distribution of the units is favorable. This also seem to hold true for noisy data. A conclusion would be that we want the units to cover the range of the input data and since our inputs are evenly distributed this approach works best. A random approach could see a majority of the

units ending up in a limited range of our input space.



(a) (b)

(c) (d)

Figure 3: Residual error for RBF-units for manual- and random initialization of parameter μ. **Left:** (a) sin(2x) - no noise, (c) sin(2x) - noise. **Right:** (b) square(2x) - no noise, (d) square(2x) - noise.

3.2 Competitive learning for RBF unit initialisation

Here we saw that using competitive learning when initialising the RBF units for the sin(2x) and square(2x) function approximation there was a small decrease in performance. This was mostly due to the data being evenly spaced and there not existing any clusters, therefore moving the already appropriately evenly spaced RBF units, and consequently not moving to any clusters but instead just moving in a random fashion. There was however a small increase in performance when using noisy data, even though it was minuscule. This was probably caused by the data being shifted from the noise, creating clusters.

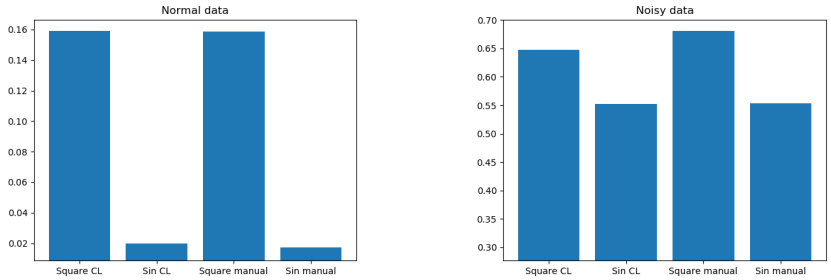


Figure 4: Residual error with manual vs competitive learning RBF unit placement.

Using either a leaky learning approach and conscience learning approach we saw no big difference in performance in the clean data, though this dead unit prevention technique improved the performance on the noisy data even further even if by a small amount.

Looking at the ballistical data set where both input and output was in $\mathbb{R}^2$ and the data unevenly spaced, a manual placement of the nodes was hard and the competitive learning approach performed better than previous data sets.

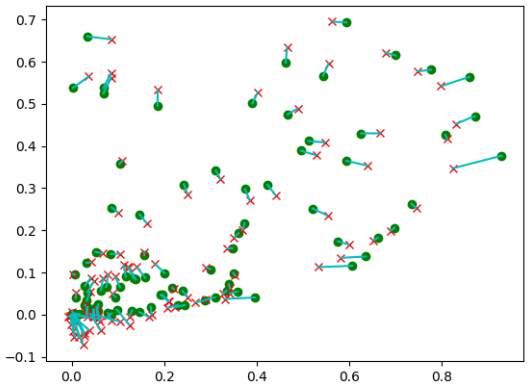


Figure 5: Ballistical data predictions (red) vs ground thruth (green), where the cyan colored lines are the error between each pair of these.

4 Results and discussion - Part II: Self-organising maps

4.1 Topological ordering of animal species

In the figure below we can see the topological ordering of animal species determined by our SOM. In large parts, the ordering makes perfect sense. Insects are grouped together at one end of the spectrum, water animals are grouped together and so are the two felines, cats and lions. The output is not logically perfect though, which if you inspect the attributes, makes sense. Where is the boundary between a "very small" animal and an "extremely small" animal? Uncertainties like this in the various attributes and the relations between them certainly could result in some positions that on the surface does not make sense. Overall though, we believe the topological ordering here is successful.

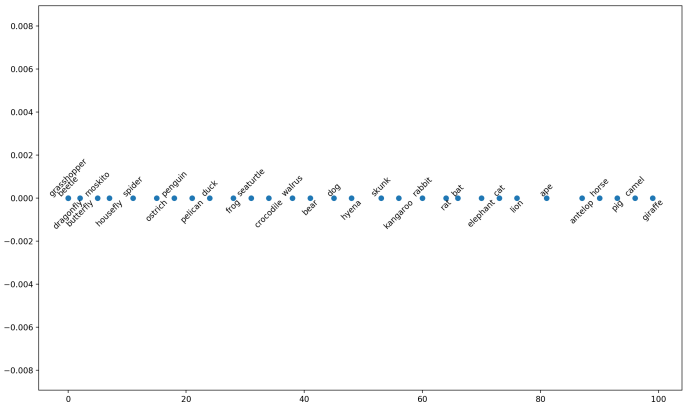


Figure 6: Output of the topological ordering of animal species.

4.2 Cyclic tour

This assignment, by virtue of being a variant of the Travelling Salesman Problem, can be seen as an NP-complete problem. Hence, it is quite unlikely that the actual optimal route is found. However, we wanted to aim for a route that was very close to optimality. After analysing the output graphs of the suggested route fitted with the data points, the output was somewhat underwhelming in terms of the performance of our SOM. The route did not pass any point more than once, and it does fit some of the data points, but certainly not all of them with the greatest precision. We tried to optimize the performance by varying several parameters. The results varied quite substantially across iterations. The ones below were achieved with an initial neighbourhood size of 2, 100 epochs of training and a learning rate of 0.3. Learning rate decay was also utilized.

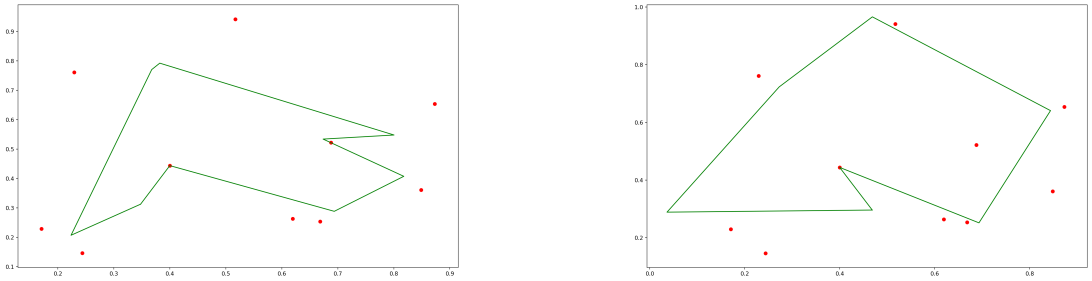


Figure 7: Two of the suggested routes from the cyclic tour SOM experiment. Both achieved with the parameter settings above.

4.3 Clustering with SOM

When plotting the voting patterns of the two gender distributions, we see that they are essentially identical, telling us that if our ambition is to separate the data based on some given attribute, gender is not a very good choice. It is also worth noting that the 349 members of parliament consists of 53.3%

males and 46.7% females, so it's quite even between the two distributions.

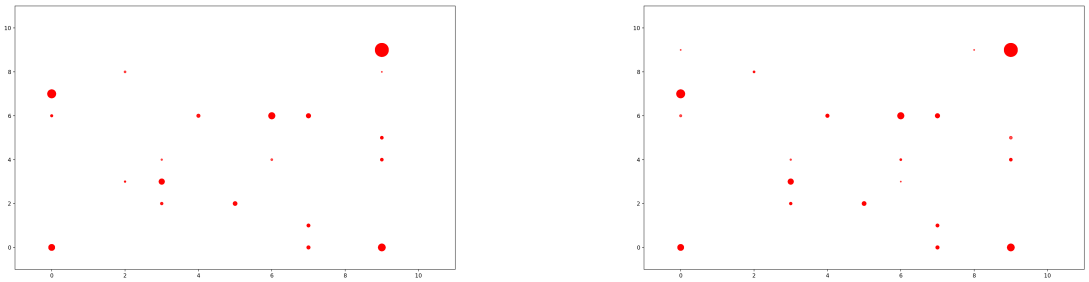


Figure 8: Voting patterns of MPs separated by the gender attribute. **Left:** Male MPs. **Right:** Female MPs.

When plotting the voting patterns across all the 29 districts represented in the parliament, one obvious visual analysis is that the plot is quite untidy as a result of the multitude of districts. Despite this though, it is easy to see that there is no clear separation in the districts and their respective voting patterns. The more popular voting patterns, which we could examine in the gender plots above, are visible here as well, and we can clearly see that a significant amount, if not most of the districts, are represented in these positions.

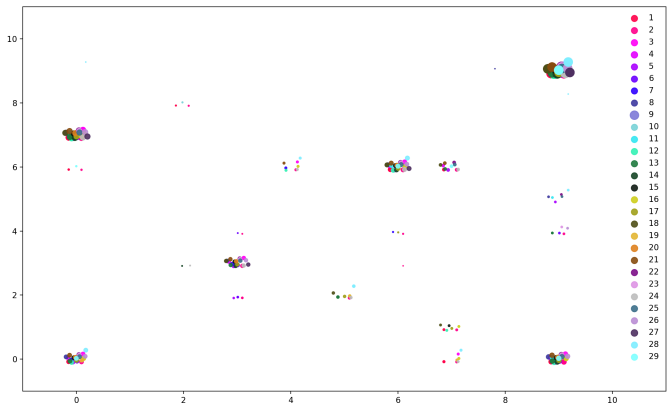


Figure 9: Voting patterns separated by districts.

When separating the input data, the members of parliament, into their respective parties, we can quite clearly see that this seems to be the definitive criterion when we aim to determine how a member will vote. The graphs tells us that despite their being a small minority of outliers, the majority of members in a given party have very similar voting patterns. We can also see that parties which at this point in time engaged in political collaborations, and tended to share several aspects of their political ideologies, are generally positioned close to each other in the output space. This tells us that classifying the data based on this attribute not only gives us the distinct voting patterns of the individual parties, but is also able to give us insights into which parties that shows similarity to one another.

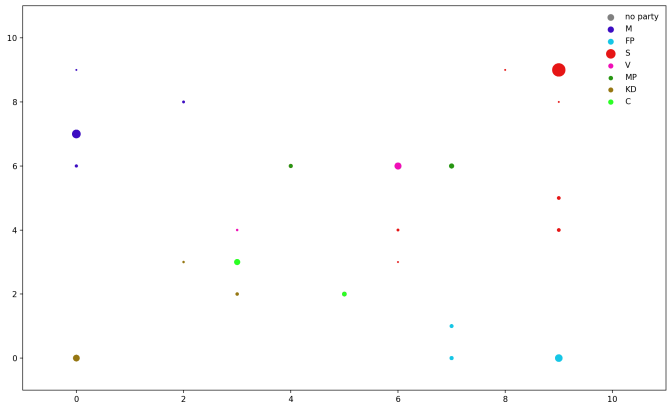


Figure 10: Voting patterns separated by political parties.

5 Final remarks

The content of the lab was overall stimulating. In comparison to the previous lab, this one did not feel as exhaustive and more varied in terms of content, creating a more enjoyable experience.

Some of the results felt somewhat inconclusive, and did not lend themselves to good reflection.