Artificial Neural Networks and Deep Architectures, DD2437

Short report on lab assignment 2

Radial basis functions, competitive learning and self-organisation

Widad Farhat, Xiaolei Liu and Saly Al Masri

14 mars 2025

1 Main objectives and scope of the assignment

Our major goals in the assignment were

- Training RBF networks and evaluating their characteristic properties.
- Implementing competitive learning paradigms and analyzing their key influencing factors.
- Exploring SOM architectures and their adaptability to diverse application scenarios.

2 Methods

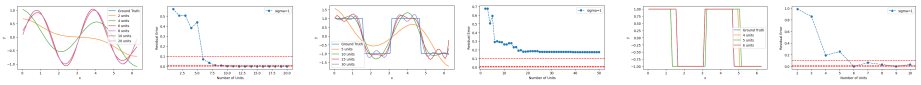
The experiment was implemented in Python using Jupyter Notebook. The network was built manually with NumPy for numerical computations, and Matplotlib was used for visualizing network behavior and performance metrics.

3 Results and discussion - Part I: RBF networks and Competitive Learning *(ca. 2.5-3 pages)*

3.1 Function approximation with RBF networks

Batch For the sine function 4b, increasing the number of RBF units improves accuracy: six nodes reduce the error below 0.1, eight below 0.01, and ten to

0.0015âjust above the 0.001 threshold. This confirms that greater model complexity enhances smooth function approximation. For the square wave 4c, the network struggles due to its discontinuities, with residual error remaining high at 0.2644 even with 30 nodes. However, applying a sign function transformation eliminates the error entirely with just six nodes by mapping continuous outputs to discrete ± 1 values. This effectively converts the regression task into classification, demonstrating the necessity of postprocessing 1c for discontinuous functions. The findings highlight RBF networks' strength in smooth function approximation and the importance of domain-specific transformations for classification problems.

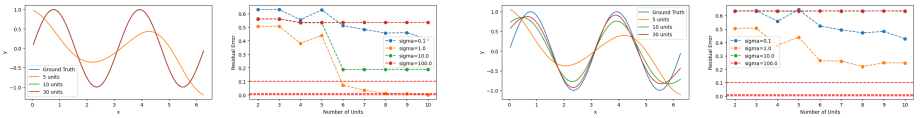


(a) Sine (b) Square (c) Postprocessing

Figur 1: Comparison of Sine, Square, and Postprocessing Results.

3.1.1 RBF

The results indicate that RBF width (σ) strongly impacts accuracy. Small σ (0.1) leads to underfitting, while large values (10, 100) cause over-smoothing. The optimal σ (1.0) achieves the best performance, with batch learning reducing the error to 0.0015 at 10 nodes. Batch learning outperforms online learning by directly finding the global optimum and being less sensitive to noise. For $\sigma = 1.0$ with 10 nodes, batch learning reaches 0.0015 residual error, while online learning remains at 0.2475.

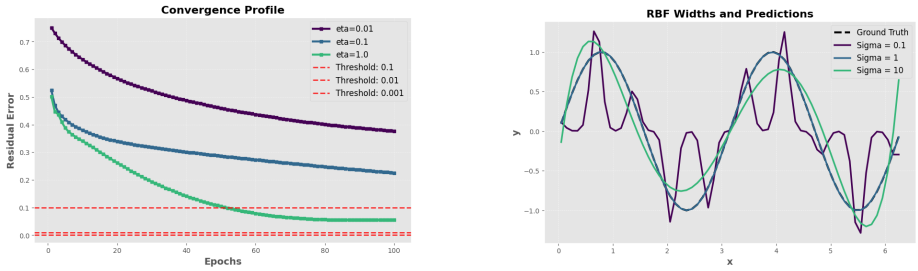


(a) Least Squares (b) Delta Rule

Figur 2: Comparison of Least Squares and Delta Rule approaches.

3.1.2 Convergence and RBF Width Effects

The rate of convergence in online learning 3a depends on the learning rate η. Higher η accelerates learning, with $\eta = 1.0$ achieving the fastest error reduction, while $\eta = 0.01$ converges slowly. No divergence was observed, confirming $\eta = 1.0$ as optimal for this problem. The RBF width σ 3b impacts the bias-variance trade-off. A small $\sigma = 0.1$ leads to high variance and large prediction oscillations. A large $\sigma = 10$ oversmooths, causing underfitting and high bias. The optimal $\sigma = 1.0$ balances adaptability and generalization, achieving accurate predictions.



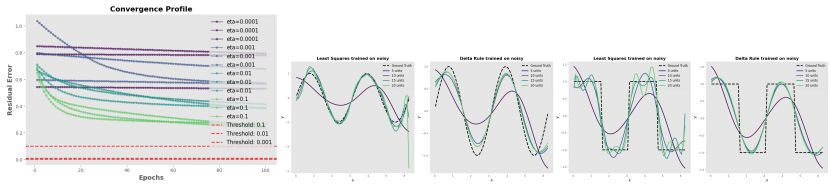
(a) Convergence (b) RBF Width Effect

Figur 3: Comparison of Convergence and RBF Width Effects

3.1.3 RBF Node Positioning and Test Performance

While increasing the learning rate (η) 4a improves stability, residual errors remain high. The best performance is observed at $\eta = 0.1$, though structured placement remains superior.

For test performance on clean data, models with 5 nodes underfit, while 10 nodes improved accuracy but retained bias. The optimal balance between generalization and accuracy occurred at 15 nodes, whereas 20 nodes led to overfitting. Least Squares produced smoother approximations, while Delta Rule exhibited greater variability, confirming 15 nodes as the most effective configuration.



(a) Node Positioning (b) Sine (c) Square

Figur 4: Comparison of RBF Node Positioning and Function Approximations

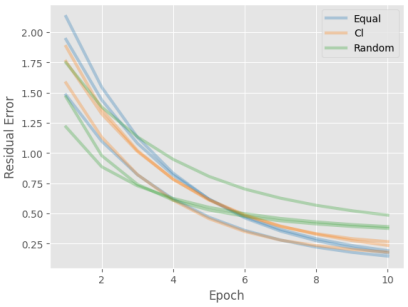
3.2 Competitive learning for RBF unit initialisation *(ca. 1 page)*

The comparison 5a shows that **equal spacing achieves the lowest residual errors**, while **CL converges faster but with slightly higher final errors**. **Random initialization performs the worst**, with high variability and poor convergence.

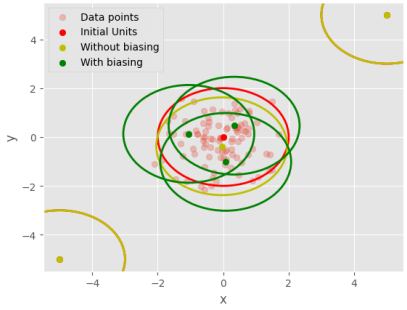
In 5b In vanilla CL, units at $(-5, -5)$ and $(5, 5)$ never updated, while a single unit at $(-0.0448, -0.3794)$ dominated, leading to poor generalization. In contrast, the multi-winner approach distributed updates across three RBF units,

with final positions at $[0.0744, -1.0183]$, $[0.3337, 0.4638]$, and $[-1.0520, 0.1286]$, resulting in better function coverage.

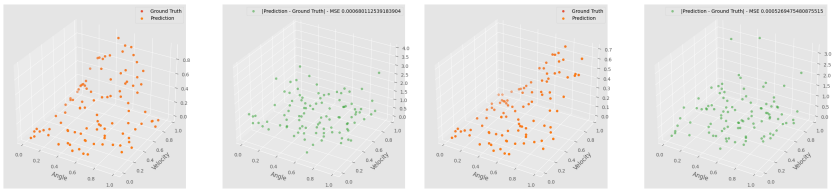
The RBF network trained with Competitive Learning (CL) effectively approximates the ballistic function, mapping angle and velocity to distance 5c and height 5d with high accuracy. The mean squared error (MSE) for distance is 0.00068, while for height, it is 0.00053, with errors mostly within the range of 0.0001 to 0.001. The low error values indicate strong generalization and minimal overfitting. Competitive Learning ensured optimal placement of RBF centers, preventing dead units and capturing the underlying function behavior.



(a) CL



(b) Standard CL (yellow) vs. Multi-winner CL (green)



(c) Distance

(d) Height

Figur 5: Comparison of CL, Multi-winner CL, and Function Approximations

4 Results and discussion - Part II: Self-organising maps *(ca. 2 pages)*

4.1 Topological ordering of animal species

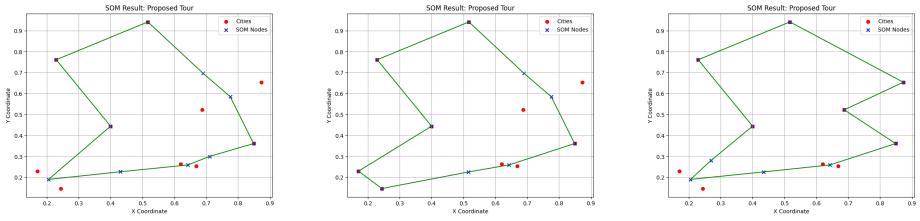
The 1d embedding sequence of animals is as follows:

'grasshopper' 'beetle' 'dragonfly' 'butterfly' 'moskito' 'housefly' 'spider'
'pelican' 'duck' 'ostrich' 'penguin' 'frog' 'seaturtle' 'crocodile'
'walrus' 'bear' 'hyena' 'dog' 'skunk' 'ape'
'lion' 'cat' 'bat' 'rat' 'rabbit' 'elephant'
'kangaroo' 'horse' 'antelop' 'pig' 'giraffe' 'camel'

, in which we can see that insects, birds, Felid animals (lion and cat) and Artiodactyla animals (from atelop to camel) are respectfully clustered. Some other clustering criteria are also relevant, for example amphibious and aquatic animals (from penguin to walrus) are near, and that the similarity in appearance may also serve as a criteria, in the case of hyena-dog and bat-rat-rabbit.

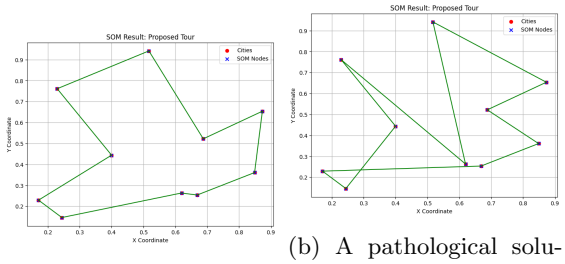
4.2 Cyclic tour

figure6 shows the result of a classical SOM algorithm, with epochs 200 and learning rate 0.1. Also wo apply a Gaussian neighborhood function as elaborated in slides. The main problem is that since some cities are pretty close, it is possible that one SOM node between is winner in both case, resulting some other nodes dead. This can not be perfectly addressed by parameter tuning.



Figur 6: Solutions of tour problem, where SOM nodes do not perfectly coincide with City points.

To address this problem, we refine the algorithm. Since the output nodes correspond to cities in this scenario, we force that one node can only win once in an epoch. Figure 8 shows an ideal solution yielded through this, together with a pathological solution, depending on random seed. Though SOM can be used to solve planning problems, it does not guarantee to result in the optimal solution.



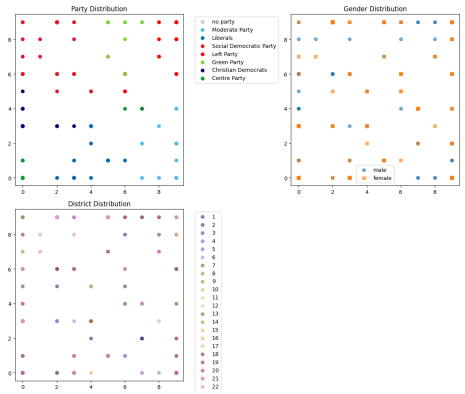
(a) An deal solution.

(b) A pathological solution.

Figur 7: Solutions of tour problem with node alignment.

4.3 Clustering with SOM

Figure 8 reveals Sweden's two major opposing parties, the Social Democratic Party (S) and Moderate Party (M), positioned farthest apart at the upper-left and lower-rght corners respectively. Progressive left-wing parties (S, Green Party, Left Party) dominate the upper section, while center-right conservative parties (Christian Democrats, Liberals, M) cluster in the lower area. The horizontal dimension reflects economic policy divergence: Christian Democrats emphasize social welfare on the left side versus Moderates' market liberalism on the right. The Centre Party splits into two clusters - one aligning with conservatives and another near the center, likely reflecting its transition from agricultural focus to broader environmental policies. No distinct patterns emerge in gender or district distributions.
Also, it is notable that roughly half of SOM nodes are activated, showing the dead node problem as mentioned in section 4.2.



Figur 8: Visualization of embedding of MPs w.r.t. different aspects of backgrounds.

5 Final remarks *(max 0.5 page)*

The philosophical foundations of RBF, competitive learning, and SOMs differ significantly from the gradient descent-driven, scalability-focused paradigms of modern deep learning. RBF networks emphasize activation function diversity over network depth, while SOMs tackle manifold learning through unique topological structures. Both methods use non-gradient-based training, particularly in hidden layers. While lectures and slides provide theoretical insights, they often lack practical depth. This lab experiment offers hands-on experience to better understand these alternative neural network approaches.