Artificial Neural Networks and Deep Architectures, DD2437

Short report on lab assignment 1b

Learning with backpropagation and generalisation in multi-layer perceptrons

Widad Farhat, Xiaolei Liu and Saly Al Masri

January 28, 2025

1 Main objectives and scope of the assignment

Our major goals in the assignment were:

- to implement and understand the behavior of Multi-Layer Perceptron (MLP) networks for classification and function approximation tasks
- to investigate how different network architectures and hyperparameters affect learning performance and generalization capabilities
- to explore the effects of noise and regularization techniques on network performance in time series prediction
- to gain practical experience in designing, training, and validating neural networks using modern tools and best practices

The scope of this assignment was limited to feed-forward neural networks trained with the backpropagation algorithm. We focused on supervised learning scenarios with an emphasis on classification and time series prediction tasks. The main assumptions included working with batch learning approaches and using standard activation functions (sigmoid for hidden layers and linear for output layers). We assumed that our training data was representative of the underlying patterns we aimed to learn.

2 Methods

For this assignment, we utilized Python as our primary programming language within the Google Colab environment. We leveraged key libraries including

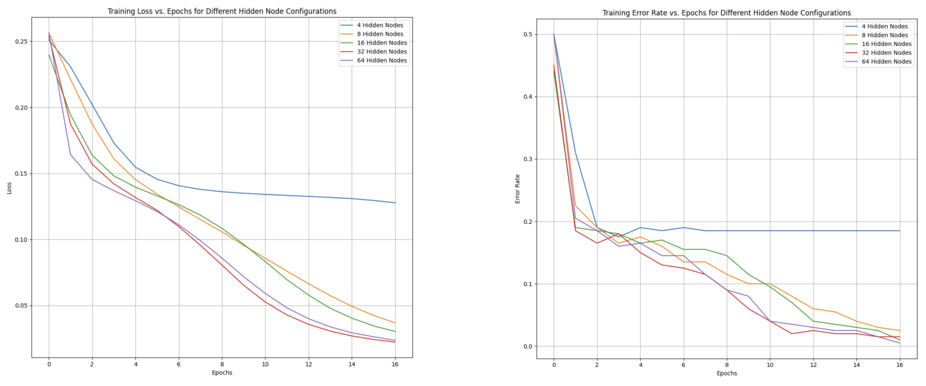
NumPy for numerical computations, scikit-learn's MLPRegressor for implementing neural networks, and Matplotlib for visualization. The performance of our models was primarily evaluated using the mean squared error (MSE) metric, which provides a quantitative measure of the difference between predicted and actual values. For classification tasks, we additionally monitored the misclassification rate to assess model accuracy.

3 Results and discussion

3.1 Classification and regression with a two-layer perceptron

3.1.1 Classification of linearly non-separable data

In our experiments with classifying linearly non-separable data, we investigated the impact of network architecture and training strategies on classification performance. We implemented a two-layer perceptron with varying numbers of hidden nodes (4, 8, 16, 32, and 64) and evaluated its performance under different training scenarios.



(a) Training loss curves.

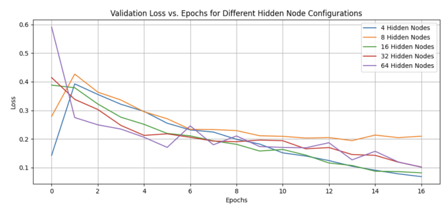
(b) Training error rate curves.

Figur 1: Training loss and error rate vs. epochs for different hidden node configurations.

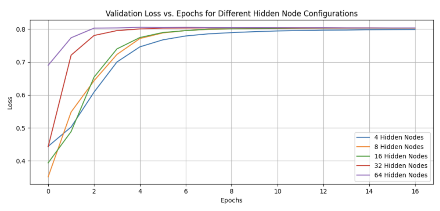
Effect of Hidden Layer Size Our initial experiments focused on how the number of hidden nodes affects the network's learning capability and classification performance. The results showed that:

Networks with 4 hidden nodes struggled to separate the classes effectively, achieving only modest improvement in training loss (from 0.234 to 0.140) and maintaining a relatively high error rate of 21.5% after 16 epochs. Increasing the number of hidden nodes to 8 or 16 significantly improved performance, with both configurations achieving error rates below 1% by the end of training. The 8-node network reduced its training loss from 0.244 to 0.026, while the 16-node network achieved a final loss of 0.020. Larger networks (32 and 64 nodes) showed similar final performance but exhibited different learning dynamics, with slower initial convergence but eventually reaching comparable error rates (1.5% and 1% respectively).

Training and Validation Performance We then investigated the network's generalization capabilities using three different data splitting scenarios. The learning curves revealed several interesting patterns:



(a) Random 50% from class A.



(b) 20%/80% split from A subsets.

Figur 2: Validation loss vs. epochs for different data splitting scenarios.

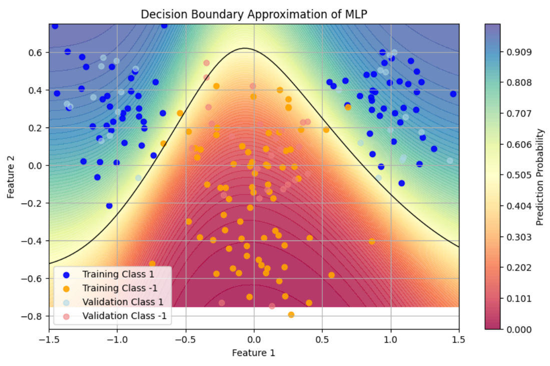
Random 50% from class A: The asymmetric sampling led to more pronounced differences between training and validation performance. Networks with 8-16 hidden nodes showed the most robust generalization, while larger networks exhibited higher variance in validation performance.

Biased sampling (20%/80% split): This scenario revealed the most significant discrepancies between training and validation performance, highlighting the importance of representative training data. All network configurations showed relatively stable convergence, but with higher final validation error rates compared to the other sampling strategies.

Decision Boundary Analysis The decision boundary visualization demonstrates that the network successfully learned a non-linear decision boundary to separate the classes. The boundary appears smooth and follows the natural separation between the classes, with appropriate curvature in regions where the classes overlap. The color gradient indicates the network's classification confidence, with darker regions representing stronger predictions.

3.1.2 Function approximation

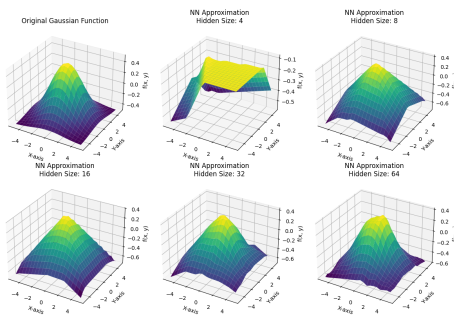
We investigated the capability of a two-layer perceptron network to approximate a 2D Gaussian function $f(x, y) = e^{-(x^2+y^2)/10} - 0.5$ defined over the domain



Figur 3: Decision boundary visualization showing the network's classification regions and confidence levels. Blue and yellow points represent different classes, with darker regions indicating stronger prediction confidence.

$[-5, 5] \times [-5, 5]$. Our analysis focused on four key aspects: the effect of hidden layer size, generalization performance, training data size impact, and noise robustness.

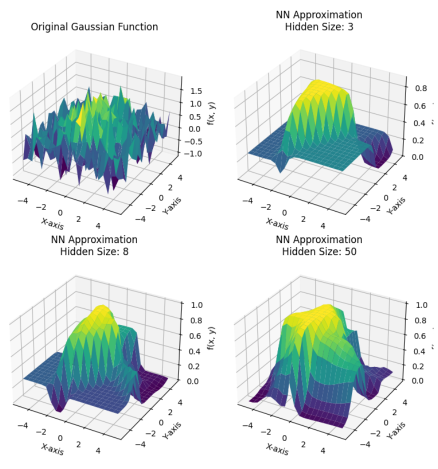
Effect of Hidden Layer Size The visual comparison (Figure 4) demonstrates the network's approximation capability across different hidden layer sizes (4, 8, 16, 32, and 64 nodes). With 4 hidden nodes, the network produces a crude approximation with visible discontinuities and sharp transitions. Increasing to 8 nodes shows significant improvement, while 16 nodes achieve a remarkably smooth approximation closely resembling the original Gaussian function. Further increases to 32 and 64 nodes show diminishing returns, suggesting that 16 nodes provide an optimal balance between model complexity and approximation accuracy.



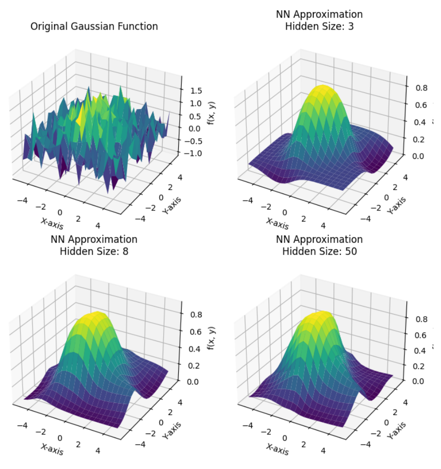
Figur 4: Comparison of neural network approximations of the 3D Gaussian function with varying hidden layer sizes.

Noise Robustness Figure 6 demonstrates the network's resilience to Gaussian noise:

- Larger networks show tends to overfit when the noise is added
- Using mini-batch processing can smooth the approximated function and build up robustness for models.



(a) Sequential Processing.



(b) Mini-batch processing.

Figur 5: Approximations of noised 3D Gaussian function.

3.2 Multi-layer perceptron for time series prediction

We conducted an analysis of three-layer perceptron architectures for predicting the Mackey-Glass time series. The investigation comprised 50 independent trials with different architectures, followed by experiments with noise and regularization.

3.2.1 Three-layer perceptron for time series prediction - model selection, validation

Our initial investigation focused on identifying optimal network architectures through systematic evaluation of different node configurations. Table **??** summarizes the frequency of best and worst performances across all trials.

The architecture with 5 nodes in the first hidden layer and 6 in the second (5,6) emerged as the most successful configuration, achieving the best performance in 30% of trials. Notably, architectures with 5 nodes in the first hidden layer generally performed better, suggesting that increased complexity in the initial feature extraction layer benefits prediction accuracy.

The mean squared errors (MSE) for the best and worst architectures showed significant separation:

Architecture (nh1, nh2)	Best Performance Count	Worst Performance Count
(3,2)	1	15
(3,4)	2	6
(3,6)	1	3
(4,2)	2	10
(4,4)	9	4
(4,6)	5	0
(5,2)	6	8
(5,4)	9	0
(5,6)	15	4

- Best architecture average MSE: 0.0014 ($\sigma = 0.0005$)
- Best architecture average MSE: 0.0513 ($\sigma = 0.0002$)

Figure **??** demonstrates the prediction quality difference between the best and worst architectures on the test set. The best architecture (5,6) closely tracks the actual values, while the worst architecture (3,2) shows significant deviation, particularly at the extrema of the time series.

3.2.2 Three-layer perceptron for noisy time series prediction with penalty regularisation

We extended our investigation to examine the network's robustness under noisy conditions with weight decay regularization. Two noise levels were tested (Ï = 0.05 and Ï = 0.15) with varying architecture sizes and regularization strengths. Table 1 presents the validation MSE for different configurations.

Tabell 1: Validation MSE Under Different Noise Conditions

Noise Level	Architecture	Weight Decay	Validation MSE
Ï = 0.05	(5,3)	0.1	0.0017
	(5,6)	0.1	0.0092
	(5,9)	0.1	0.0024
Ï = 0.15	(5,3)	0.1	0.0027
	(5,6)	0.1	0.0100
	(5,9)	0.1	0.0017

The impact of weight decay regularization was investigated using three different decay rates (1.0, 0.1, 0.01) with Ï = 0.05 noise level:

- Decay = 1.0: MSE = 0.0082
- Decay = 0.1: MSE = 0.0016 (optimal)
- Decay = 0.01: MSE = 0.0092

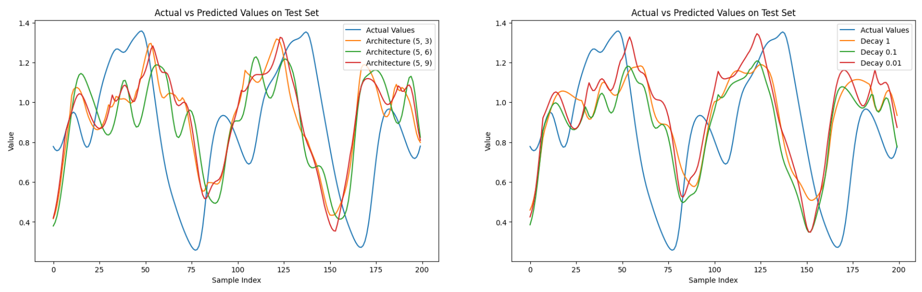
Key findings from the noise and regularization experiments:

1. Larger architectures showed better resilience to noise, with the (5,9) configuration performing best under high noise conditions (Ï = 0.15).

2. The optimal weight decay rate of 0.1 provided the best balance between model flexibility and overfitting prevention.

3. Increased noise levels generally led to higher validation MSE values, but this effect was mitigated by appropriate regularization.

The results demonstrate that successful time series prediction with MLPs requires careful consideration of both architecture size and regularization strength, particularly in the presence of noise. The superior performance of larger architectures, especially with appropriate regularization, suggests that the additional complexity aids in capturing the underlying patterns of the chaotic time series while maintaining generalization capability.



(a) Different architecture. (b) Different weight decay.

Figur 6: Approximations of time-series under noise with Sigma=0.1.

4 Final remarks *(max 0.5 page)*

The laboratory experiments provided valuable practical insights into multilayer perceptron behavior across different applications. The most enlightening aspects were observing how network architecture significantly impacts performance, particularly in our time series prediction where the (5,6) configuration proved optimal. The function approximation task with the 2D Gaussian function demonstrated the principle of diminishing returns, showing that bigger isn't always better in neural network design. The noise robustness experiments yielded practical insights about dealing with real-world data imperfections, while our work with weight decay regularization (finding 0.1 as optimal) helped bridge theory and practice. Key questions for future investigation include exploring the impact of different activation functions and the potential benefits of adaptive learning rates. These hands-on experiences have strengthened our understanding of neural networks and provided practical experience in balancing various design trade-offs, making theoretical concepts tangible through real implementation.