EE40098 Computational Intelligence - Coursework A

Seb Hall samh25@bath.ac.uk, 24th October 2025
Department of Electronic & Electrical Engineering, University of Bath

1. Exercise 1 - Logical Operators

Perceptrons[1] can be used to perform logical operations on binary and discrete data inputs. Truth tables, state space graphs, and neural network weights and biases for the AND, NAND, OR and XOR logical operators are shown below. In the graphs, red indicates a value of 0 and blue indicates a value of 1.

1.1. AND Operator

A	B	AND(A,B)
0	0	0
1	0	0
0	1	0
1	1	1

Table 1: AND Truth Table

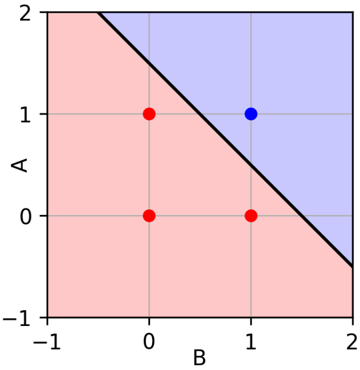

Figure 1: AND State Space

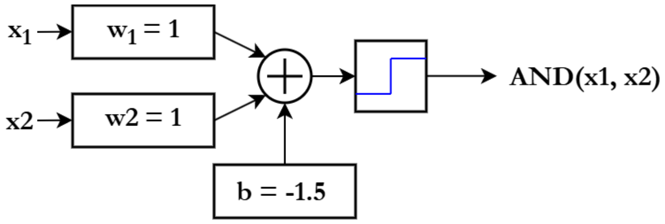

Figure 2: AND network diagram, showing weights ($w_1 = 1$, $w_2 = 1$) and bias ($b = -1.5$).

1.2. NAND Operator

A	B	NAND(A,B)
0	0	1
1	0	1
0	1	1
1	1	0

Table 2: NAND Truth Table

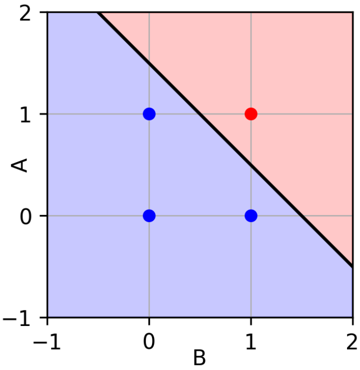

Figure 3: NAND State Space

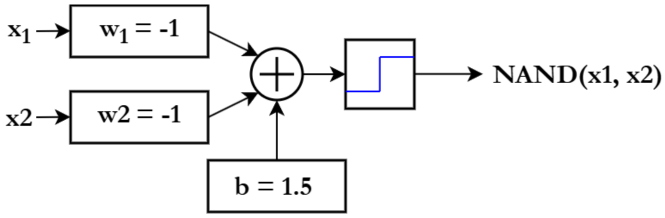

Figure 4: NAND network diagram, showing weights ($w_1 = -1$, $w_2 = -1$) and bias ($b = 1.5$).

1.3. OR Operator

A	B	OR(A,B)
0	0	0
1	0	1
0	1	1
1	1	1

Table 3: OR Truth Table

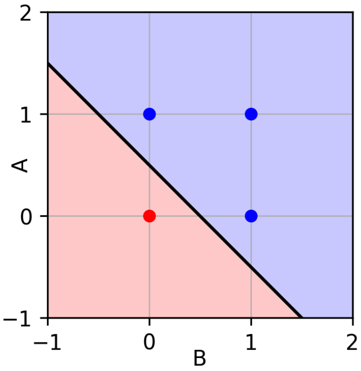

Figure 5: OR State Space

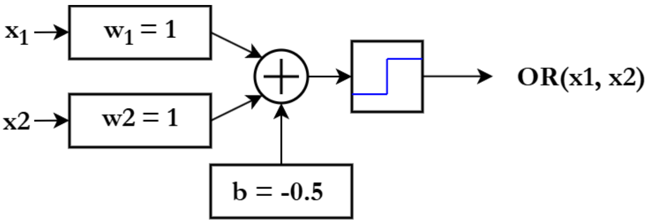

Figure 6: OR network diagram, showing weights ($w_1 = 1$, $w_2 = 1$) and bias ($b = -0.5$).

1.4. XOR Operator

A	B	XOR(A,B)
0	0	0
1	0	1
0	1	1
1	1	0

Table 4: XOR Truth Table

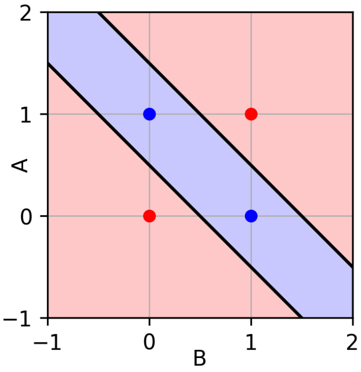

Figure 7: XOR State Space

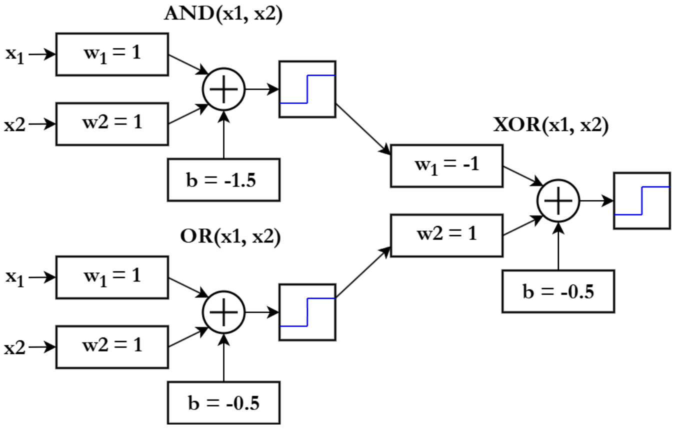

Figure 8: XOR network diagram, consisting of a hidden layer with AND and OR weights, followed by an output layer, with weights ($w_1 = -1$, $w_2 = 1$) and bias ($b = -0.5$).

2. Exercise 2 - Implementation of a Neural Network for MNIST

A perceptron-based neural network can be implemented to classify images into discrete categories. For this exercise, a network was implemented to classify MNIST[2] datasets; the first a set of handwritten numeric digits, and the second a collection of items of clothing.

2.1. Neural Network 1 - Numeric Dataset

2.1.1. Network Structure

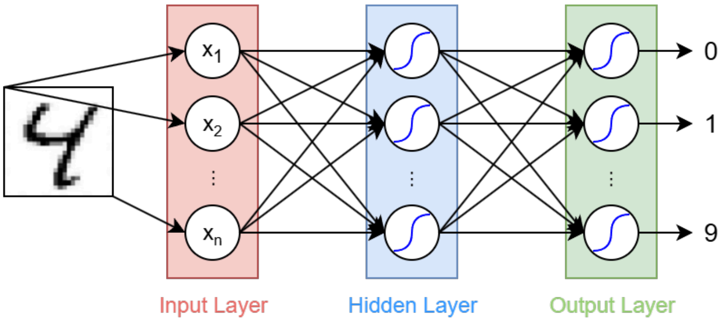

Figure 9: Artificial Neural Network (ANN) Structure for Classifying MNIST Digits.

An artificial neural network (ANN) was implemented as shown in Figure 9, consisting of an input layer with 784 input nodes corresponding to the 28x28 grayscale pixels in each image, a hidden layer with a variable number of nodes, and an output layer with 10 nodes corresponding to the 10 output digit classes (0-9).

2.1.2. Hyperparameter Search

To optimise the performance of the network, two parameters were explored in detail - the learning rate and number of hidden nodes. This was achieved with a Monte Carlo search method[3], which involves defining a parameter space, and evaluating random combinations of parameters within this space to produce an optimal region (see Figure 10).

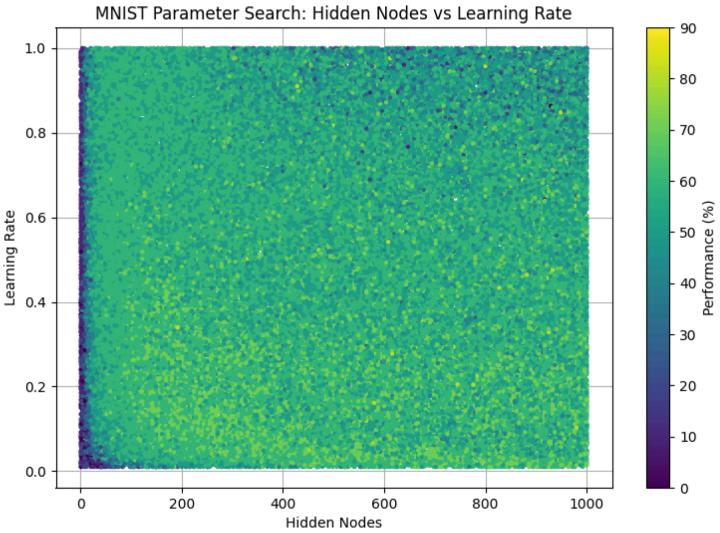

Figure 10: Monte Carlo Search for Hyperparameters of Hidden Nodes and Learning Rate.

A search was performed on reduced train and test datasets for increased speed. 100,000 iterations were taken with learning rates between 0.01 and 1.0, and hidden layers between 1 and 1000. This was a wide search space, so a large number of samples were required to find optimal parameters. The output of this search was represented as a scatter plot, displaying the test performance of each sample as a colour gradient, from 0-90%. This image was then edited to increase contrast and find a clearer optimal region, shown in Figure 11.

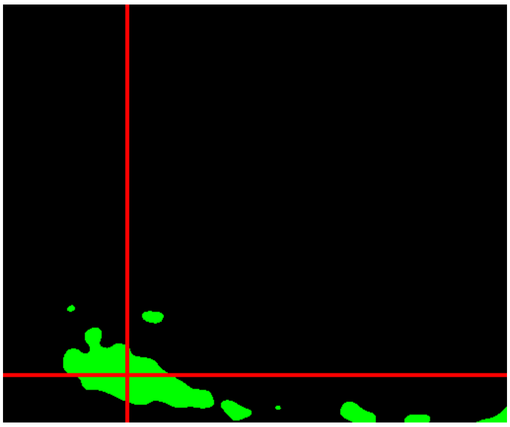

Figure 11: High Contrast Monte Carlo Output with Optimal Point Selected.

Taking a sample in the center of the largest optimal region produced a point with **254 hidden nodes** and a **learning rate of 0.11**.

2.1.3. Full Dataset Performance

After selecting optimal values for hidden layer nodes and learning rate, the network was trained over a number of iterations on the full dataset. The results for this are shown below in Figure 12.

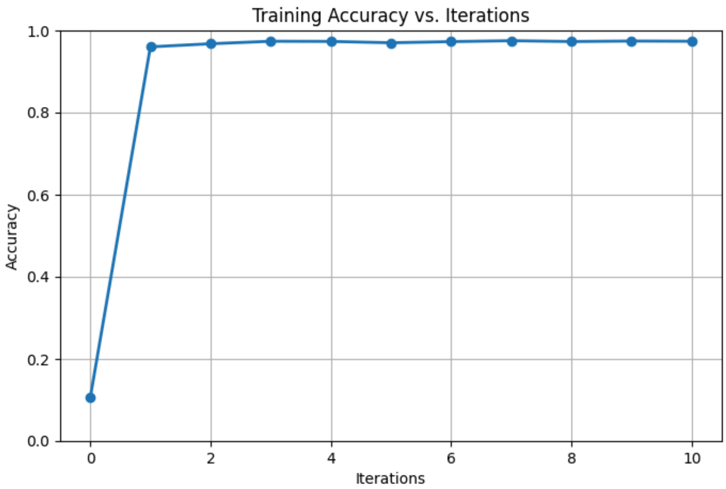

Figure 12: Test Performance vs Training Iterations for Optimal Hyperparameters.

The network reached a high level of performance after three training iterations, achieving **97.4% accuracy**. Beyond this, the performance fluctuated slightly before gradually decreasing (possibly due to overfitting of the data).

2.2. Neural Network 2 - Fashion Dataset

The second dataset used was the Fashion MNIST dataset, which consists of 28x28 grayscale images of clothing items instead of numeric digits. A structure identical to that shown in Figure 9 was used, due to the images being the same size, and the number of distinct classes also being 10.

2.2.1. Hyperparameter Search

A similar approach was taken to identify optimal hyperparameters for this dataset, however the reduced size training dataset was significantly larger than before (1000 vs 100 samples), so each training iteration took significantly longer to complete. Therefore, a new Monte Carlo search was performed with 25,000 iterations and a smaller parameter space; learning rates between 0.01 and 0.5, and hidden layers between 1 and 500. The results from this are shown below:

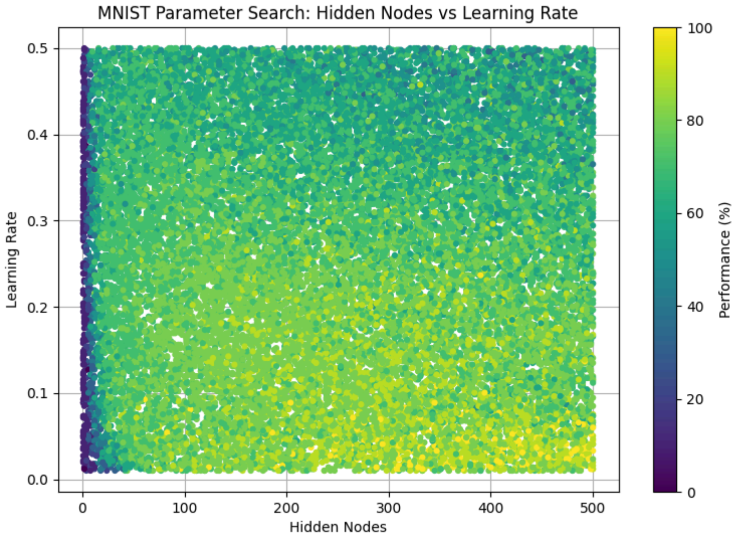

Figure 13: Monte Carlo Search for Hyperparameters of Hidden Nodes and Learning Rate (Fashion MNIST).

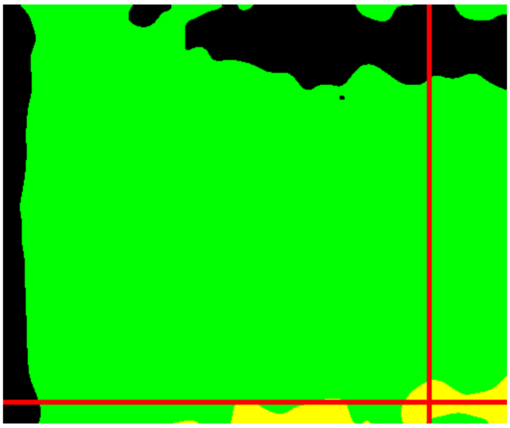

Figure 14: High Contrast Monte Carlo Output with Optimal Point Selected (Fashion MNIST).

As with the previous dataset, the output image was edited for increased contrast to determine an optimal region, shown in Figure 14. A sample in the center of the optimal region was selected to give a resulting count of **420 hidden nodes** and a **learning rate of 0.03**.

2.2.2. Full Dataset Performance

As before, the hyperparameters identified with the Monte Carlo search were then applied to training of the full dataset over a number of iterations (Figure 15).

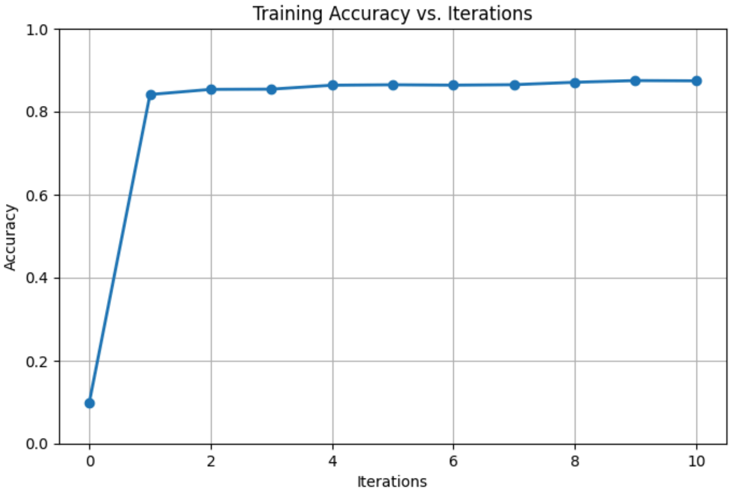

Figure 15: Test Performance vs Training Iterations for Optimal Hyperparameters (Fashion MNIST).

The performance took longer to stabilise than previously, eventually reaching a peak of **87.5% accuracy** after 9 training iterations.

2.2.3. Comparison to Digit Dataset

The final performance of the fashion network was significantly lower than that of the digit network, with error rates of 12.5% and 2.6% respectively, an increase of approximately 5x.

An additional observation is that the fashion network performed best with a higher number of hidden nodes and a lower learning rate compared to the digit network. This may be indicative of a dataset with more complex features, requiring more nodes to represent them.

2.3. Possible Improvements

A wide range of network architectures could be explored to provide higher performance, such as different activation functions or multiple hidden layers. Convolutional Neural Networks (CNNs) are also frequently used for image classification tasks, and have additional features such as pooling and convolutional layers. These are especially effective at extracting complex features, and would likely be the most beneficial for the fashion dataset which struggled with a simpler ANN architecture.

3. References

[1] H.D Block, "The Perceptron: A Model for Brain Functioning," 1962.

[2] "MNIST." [Online]. Available: http://yann.lecun.com/exdb/mnist/

[3] J. M. Hammersley and D. C. Handscomb, *Monte Carlo Methods*. 1964.