Neural Networks - Task (2) Report

MLP Back-Propagation

ID	NAME
2022170447	منة ياسر حامد مصطفى
2022170162	رؤى محمد لطفي محمد علي
2022170340	مايا سامح احمد سمير بيبرس
2022170629	ميسون حلمي عبدالرحمن علي شعبان
2022170441	منة الله محروس السيد عيد محروس
2022170611	منى خالد عثمان عواد عثمان

Introduction

The goal of this experiment is to implement a multi-layer perceptron (MLP) using back-propagation and evaluate its performance on a dataset with three classes. We investigate how the choice of activation function, learning rate, number of epochs, and network architecture affects classification accuracy.

Experimental Setup

Network Parameters:

- Hidden layers
- Neurons per layer
- Activation function
- Learning rate
- Epochs
- Training mode (Stochastic / Batch)

Experiments:

1. Sigmoid Activation

Experiment 1

Network Configuration:

```
Number of hidden layers: 2
Neurons per hidden layer: [8, 6]
Learning rate (eta): 0.1
epochs: 200
Activation function: Sigmoid
Use bias: True
training_mode: Stochastic
```

Results:

```
Training Accuracy: 0.9222 (92.22%)
```

```
Test samples: 60
Correct predictions: 52
Overall Accuracy: 0.8667 (86.67%)
```

Confusion Matrix:

Actual \ Predicted	Class 1	Class 2	Class 3
Class 1	20	0	0
Class 2	7	13	0
Class 3	0	1	19

Observations:

- The network achieved high overall accuracy (86.67%), with Classes 1 and 3 well classified, while Class 2 had more misclassifications, indicating it's slightly harder to separate.

Experiment 2

Network Configuration:

```
Number of hidden layers: 2
Neurons per hidden layer: [8, 6]
Learning rate (eta): 0.1
epochs: 500
Activation function: Sigmoid
Use bias: True
training_mode: Stochastic
```

Results:

```
Training Accuracy: 1.0000 (100.00%)
```

```
Test samples: 60
Correct predictions: 58
Overall Accuracy: 0.9667 (96.67%)
```

Confusion Matrix:

Actual \ Predicted	Class 1	Class 2	Class 3
Class 1	19	1	0
Class 2	0	20	0
Class 3	0	1	19

Observations:

- Increasing the number of epochs significantly improved performance, achieving 96.67% overall accuracy and nearly perfect classification for all classes.

Experiment 3

Network Configuration:

```
Number of hidden layers: 2
Neurons per hidden layer: [8, 6]
Learning rate (eta): 0.1
epochs: 200
Activation function: Sigmoid
Use bias: False
training_mode: Stochastic
```

Results:

```
Training Accuracy: 0.9444 (94.44%)
```

```
Test samples: 60
Correct predictions: 54
Overall Accuracy: 0.9000 (90.00%)
```

Confusion Matrix:

Actual \ Predicted	Class 1	Class 2	Class 3
Class 1	20	0	0
Class 2	4	15	1
Class 3	0	1	19

Observations:

- Removing the bias slightly reduced performance, lowering overall accuracy to 90% and slightly affecting Class 2 predictions.

Experiment 4

Network Configuration:

```
Number of hidden layers: 1
Neurons per hidden layer: [10]
Learning rate (eta): 0.05
epochs: 200
Activation function: Sigmoid
Use bias: True
training_mode: Stochastic
```

Results:

```
Training Accuracy: 0.9889 (98.89%)
```

```
Test samples: 60
Correct predictions: 57
Overall Accuracy: 0.9500 (95.00%)
```

Confusion Matrix:

Actual \ Predicted	Class 1	Class 2	Class 3
Class 1	19	1	0
Class 2	1	19	0
Class 3	0	1	19

Observations:

- A single hidden layer with 10 neurons and learning rate 0.05 achieves 95% test accuracy, demonstrating good generalization across all classes even though the architecture is simple.

2. Hyperbolic Tangent Activation

Experiment 1

Network Configuration:

```
Number of hidden layers: 2
Neurons per hidden layer: [8, 6]
Learning rate (eta): 0.01
epochs: 100
Activation function: Hyperbolic Tangent
Use bias: True
training_mode: Stochastic
```

Results:

```
Training Accuracy: 0.6778 (67.78%)
```

```
Test samples: 60
Correct predictions: 44
Overall Accuracy: 0.7333 (73.33%)
```

Confusion Matrix:

Actual \ Predicted	Class 1	Class 2	Class 3
Class 1	20	0	0
Class 2	16	4	0
Class 3	0	0	20

Observations:

- With a low learning rate (0.01) and 100 epochs, the network struggles to learn Class 2, resulting in poor overall accuracy (73.3%) despite perfect performance on Classes 1 and 3. This shows that too small a learning rate can slow convergence and hurt minority class performance.

Experiment 2

Network Configuration:

```
Number of hidden layers: 2
Neurons per hidden layer: [8, 6]
Learning rate (eta): 0.01
epochs: 200
Activation function: Hyperbolic Tangent
Use bias: True
training_mode: Stochastic
```

Results:

```
Training Accuracy: 0.9778 (97.78%)
```

```
Test samples: 60
Correct predictions: 55
Overall Accuracy: 0.9167 (91.67%)
```

Confusion Matrix:

Actual \ Predicted	Class 1	Class 2	Class 3
Class 1	19	1	0
Class 2	3	17	0
Class 3	0	1	19

Observations:

- Reducing the learning rate to 0.01 with Tanh activation lowers both training (97.78%) and testing accuracy (91.67%), especially affecting Class 2, indicating slower learning can reduce overfitting but may slightly hurt overall performance.

Experiment 3

Network Configuration:

```
Number of hidden layers: 2
Neurons per hidden layer: [8, 6]
Learning rate (eta): 0.01
epochs: 200
Activation function: Hyperbolic Tangent
Use bias: False
training_mode: Stochastic
```

Results:

```
Training Accuracy: 0.9667 (96.67%)
```

```
Test samples: 60
Correct predictions: 55
Overall Accuracy: 0.9167 (91.67%)
```

Confusion Matrix:

Actual \ Predicted	Class 1	Class 2	Class 3
Class 1	19	1	0
Class 2	3	17	0
Class 3	0	1	19

Observations:

- Removing the bias with Tanh activation and low learning rate does not affect the overall accuracy (91.67%) compared to the previous case, suggesting that bias has minimal impact for this dataset at these parameters.

Experiment 4

Network Configuration:

```
Number of hidden layers: 1
Neurons per hidden layer: [10]
Learning rate (eta): 0.1
epochs: 200
Activation function: Hyperbolic Tangent
Use bias: True
training_mode: Stochastic
```

Results:

```
Training Accuracy: 1.0000 (100.00%)
```

```
Test samples: 60
Correct predictions: 58
Overall Accuracy: 0.9667 (96.67%)
```

Confusion Matrix:

Actual \ Predicted	Class 1	Class 2	Class 3
Class 1	19	1	0
Class 2	0	20	0
Class 3	0	1	19

Observations:

- Using a single hidden layer with Tanh activation achieves the same high accuracy (96.67%) as two layers, suggesting that for this dataset, one hidden layer is sufficient for effective classification.

Best Accuracies:

Activation Function	Train Accuracy (%)	Test Accuracy (%)	Learning Rate	Epochs	#Layers	Neurons per Layer
Sigmoid	100	96.67	0.1	500	2	8,6
Tanh	100	96.67	0.1	200	1	10

Observation:

- Both Sigmoid and Tanh reached similar test accuracy. Tanh achieves high accuracy with fewer epochs and a simpler architecture.

Conclusion

- The network's performance is significantly influenced by hyperparameters such as learning rate, number of epochs, network architecture, and use of bias.

- Increasing the number of epochs generally improves accuracy, especially for networks with more complex architectures.

- Including bias in the network helps improve classification performance, particularly for underrepresented classes.

- A higher learning rate can speed up convergence but may require careful tuning to avoid instability.

- Comparing activation functions, both Sigmoid and Hyperbolic Tangent (Tanh) achieved similar best accuracies; however, Tanh reached high accuracy with fewer epochs and a simpler network.

- Overall, careful selection of activation function, learning rate, number of epochs, and network size is essential to achieve optimal performance.