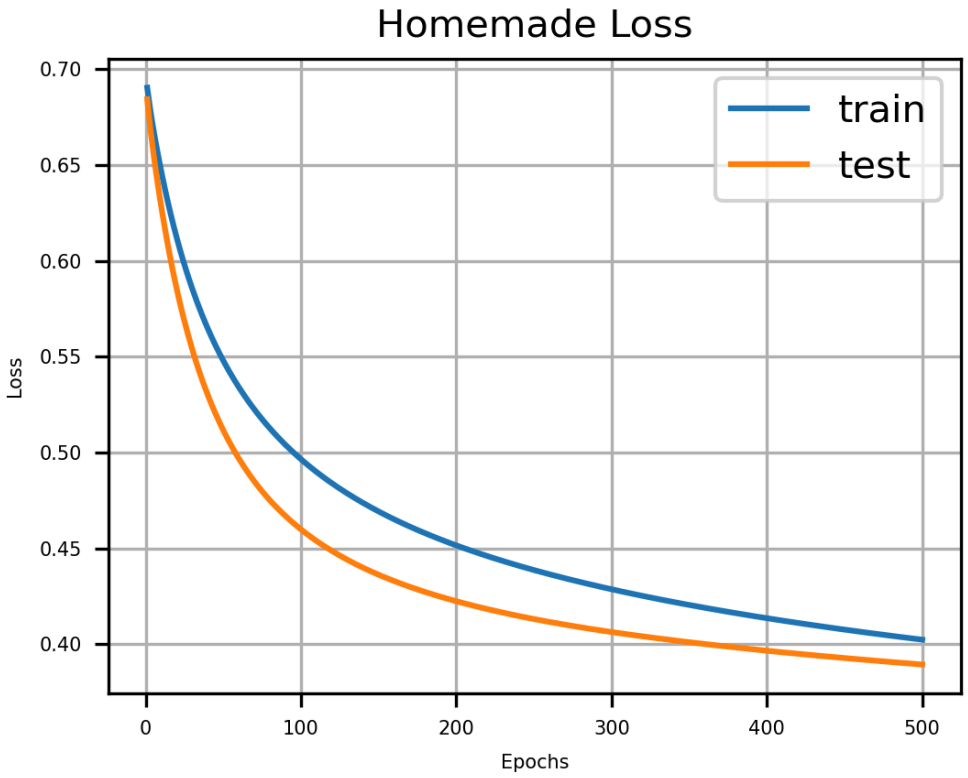
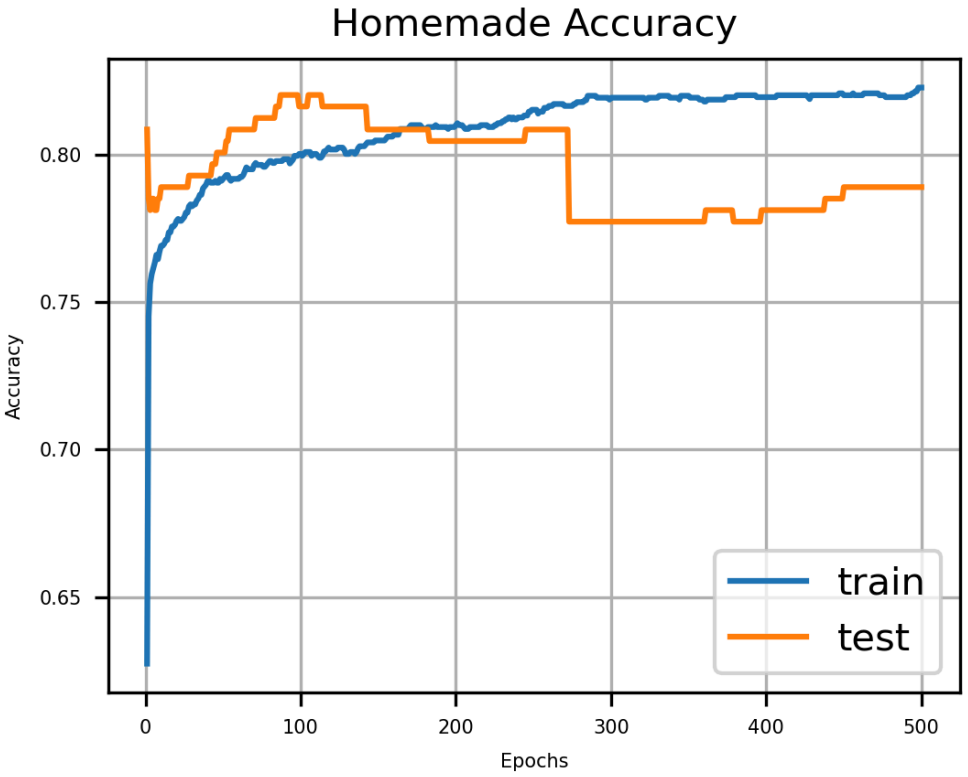
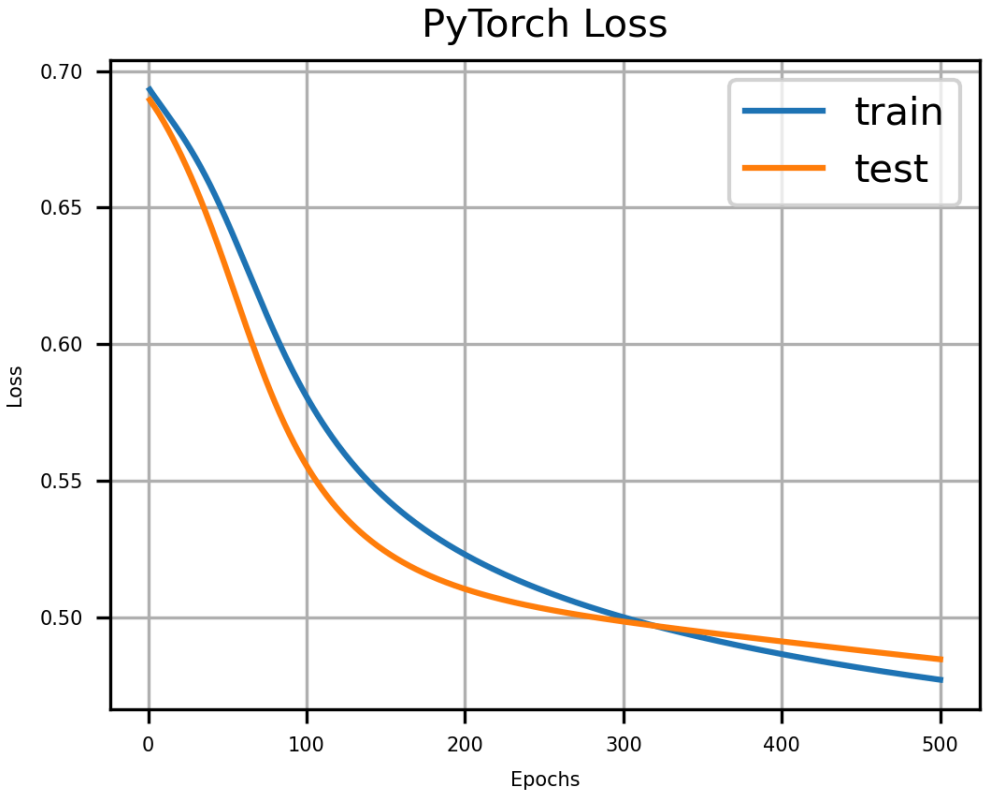
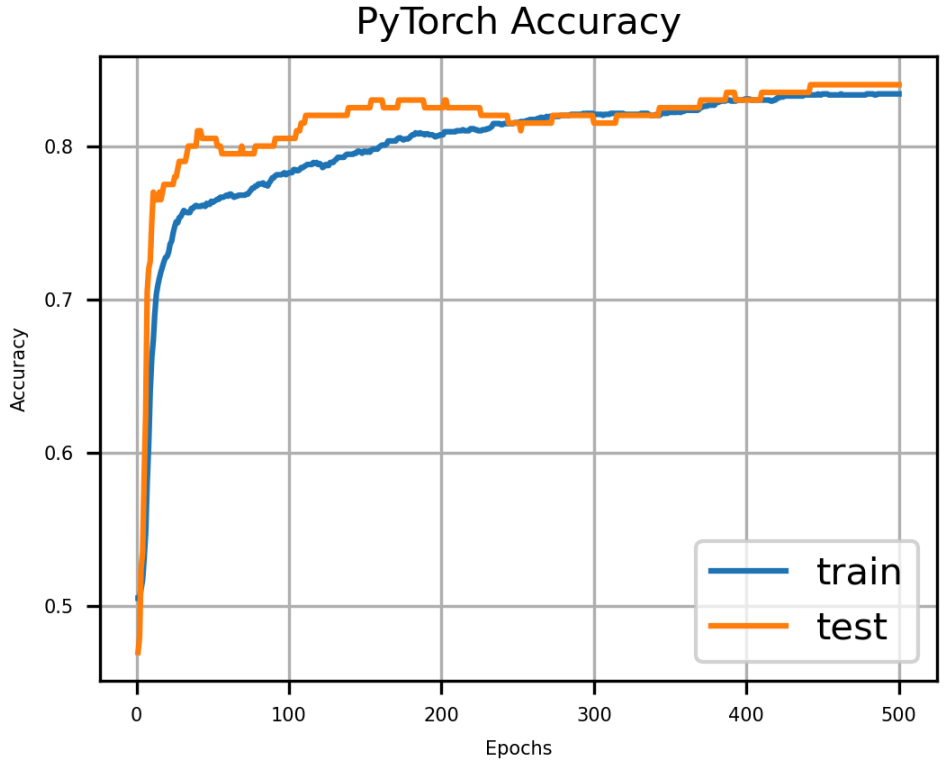

Homemade Loss
train
test
Loss
Epochs
0.70
0.65
0.60
0.55
0.50
0.45
0.40
0
100
200
300
400
500


Homemade Accuracy
train
test
Accuracy
Epochs
0.80
0.75
0.70
0.65
0
100
200
300
400
500


PyTorch Loss
0.70
0.65
0.60
0.55
0.50
Loss
0
100
200
300
400
500
Epochs
train
test


PyTorch Accuracy
0.8
0.7
0.6
0.5
Accuracy
0
100
200
300
400
500
Epochs
train
test

Based on the graphs, I noticed that the loss rates for the homemade neural network function were slightly lower than the PyTorch implementation, which is likely caused by how the homemade function was coded or how PyTorch handles loss functions, as the PyTorch graph appears to have more of a sharper downward angle than the other loss graph, which could mean that further epochs could reduce the loss at a faster rate than the homemade function.

In regards to the accuracy, both functions hit around the same accuracy of 82%, however, the homemade function does something odd with the test accuracy. A little more than halfway through the epochs, it makes a very sudden decline of around 5% as the training accuracy remains the same pattern of growth. In regards to the PyTorch function, both training and testing functions remain around the same level, with the testing accuracy ebbing up and down but still remaining around the same level.