Robot Navigation Project - Sequence Classification

This dataset was obtained from the UCI Machine Learning Repository.

You can find more information on this dataset here:

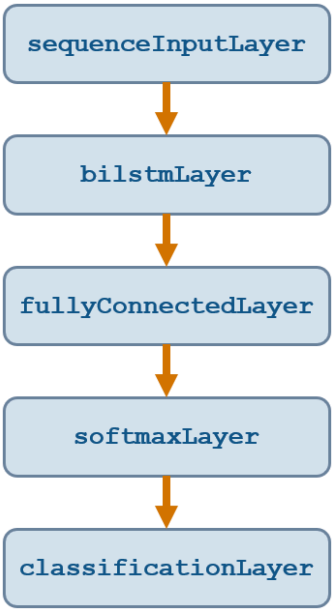
https://archive.ics.uci.edu/ml/datasets/Wall-Following+Robot+Navigation+Data

Load the processed sensor data. This mat file contains 4 variables for the sensor data and the robot's action.

- `XTrain`: A matrix with 2 columns, one for each sensor reading. The first row is the sensor reading from the front of the robot, and the second row is the sensor reading from the left of the robot.
- `XTest`: The last 1000 time steps are reserved for testing.
- `YTrain` and `YTest`: The action the robot should take at a time step. At any point, the robot can either move forward, turn slightly to the left, turn slightly to the right, or turn sharply to the right.

```
load robotDataProcessed.mat
```

Network architecture




```
layers = [
    sequenceInputLayer(2)
    bilstmLayer(100,"OutputMode","sequence")
    fullyConnectedLayer(4)
    softmaxLayer()
    classificationLayer()]
```

```
layers =
  5x1 Layer array with layers:

     1   ''   Sequence Input          Sequence input with 2 dimensions
```

```
     2   ''   BiLSTM                  BiLSTM with 100 hidden units
     3   ''   Fully Connected         4 fully connected layer
     4   ''   Softmax                 softmax
     5   ''   Classification Output   crossentropyex
```

Train the network

```
options=trainingOptions("adam","MaxEpochs",100,"InitialLearnRate",0.05,"Plots","training-progre
```

```
X = 2×4456
    1.6870    1.6870    1.6870    1.6870    1.6870    1.6860    1.6840    1.6800 ···
    0.4450    0.4490    0.4490    0.4490    0.4490    0.4460    0.4510    0.4530
Y = 1×4456 categorical
Slight-Right-Turn    Slight-Right-Turn    Slight-Right-Turn    Slight-Right-Turn ···
options =
  TrainingOptionsADAM with properties:

           GradientDecayFactor: 0.9000
       SquaredGradientDecayFactor: 0.9990
                         Epsilon: 1.0000e-08
                InitialLearnRate: 0.0500
               LearnRateSchedule: 'none'
             LearnRateDropFactor: 0.1000
             LearnRateDropPeriod: 10
                L2Regularization: 1.0000e-04
          GradientThresholdMethod: 'l2norm'
                GradientThreshold: Inf
                        MaxEpochs: 100
                    MiniBatchSize: 128
                          Verbose: 1
                  VerboseFrequency: 50
                    ValidationData: []
               ValidationFrequency: 50
                ValidationPatience: Inf
                          Shuffle: 'once'
                   CheckpointPath: ''
              ExecutionEnvironment: 'auto'
                        WorkerLoad: []
                         OutputFcn: []
                             Plots: 'training-progress'
                  SequenceLength: 'longest'
             SequencePaddingValue: 0
         SequencePaddingDirection: 'right'
             DispatchInBackground: 0
           ResetInputNormalization: 1
```

Train the LSTM network.

```
net = trainNetwork(X,Y,layers,options)
```

```
Training on single CPU.
|=========================================================================================|
|  Epoch  |  Iteration  |  Time Elapsed  |  Mini-batch  |  Mini-batch  |  Base Learning  |
|         |             |   (hh:mm:ss)   |   Accuracy   |     Loss     |      Rate       |
|=========================================================================================|
|       1 |           1 |       00:00:02 |       37.46% |       1.3853 |          0.0500 |
```

```
options2=trainingOptions("adam","MaxEpochs",200,"InitialLearnRate",0.05,"Plots","training-progr
```

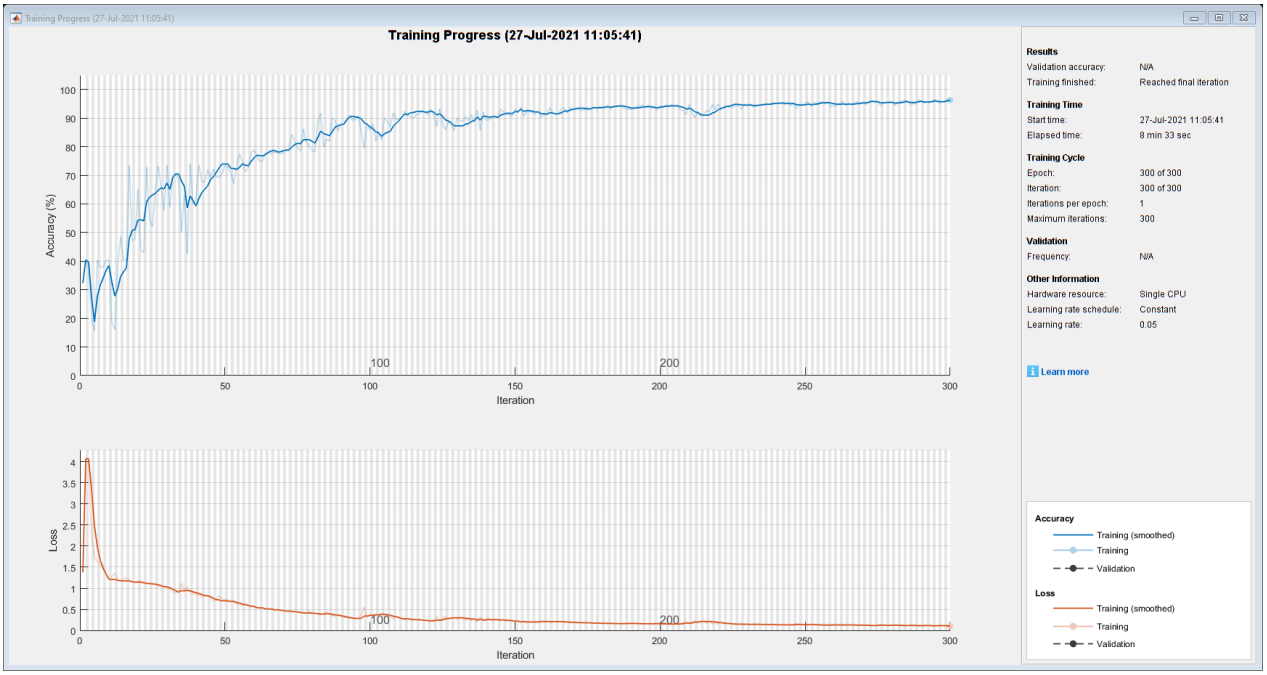
```
net2 = trainNetwork(X,Y,layers,options2)
```

```
options3=trainingOptions("adam","MaxEpochs",300,"InitialLearnRate",0.05,"Plots","training-progr
```

```
pred = 1×1000 categorical
Move-Forward    Sharp-Right-Turn    Move-Forward    Move-Forward    Move-Forward ···
options3 =
  TrainingOptionsADAM with properties:

           GradientDecayFactor: 0.9000
       SquaredGradientDecayFactor: 0.9990
                         Epsilon: 1.0000e-08
                InitialLearnRate: 0.0500
               LearnRateSchedule: 'none'
             LearnRateDropFactor: 0.1000
             LearnRateDropPeriod: 10
                L2Regularization: 1.0000e-04
          GradientThresholdMethod: 'l2norm'
                GradientThreshold: Inf
                        MaxEpochs: 300
                    MiniBatchSize: 128
                          Verbose: 1
                  VerboseFrequency: 50
                    ValidationData: []
               ValidationFrequency: 50
                ValidationPatience: Inf
                          Shuffle: 'once'
                   CheckpointPath: ''
             ExecutionEnvironment: 'auto'
                       WorkerLoad: []
                        OutputFcn: []
                            Plots: 'training-progress'
                   SequenceLength: 'longest'
              SequencePaddingValue: 0
          SequencePaddingDirection: 'right'
              DispatchInBackground: 0
           ResetInputNormalization: 1
```

```
net3= trainNetwork(X,Y,layers,options3)
```

```
Training on single CPU.
|=========================================================================================|
|  Epoch  |  Iteration  |  Time Elapsed  |  Mini-batch  |  Mini-batch  |  Base Learning  |
|         |             |   (hh:mm:ss)   |   Accuracy   |     Loss     |      Rate       |
|=========================================================================================|
|       1 |           1 |       00:00:02 |       32.29% |       1.3758 |          0.0500 |
|      50 |          50 |       00:01:31 |       73.68% |       0.7236 |          0.0500 |
|     100 |         100 |       00:03:02 |       85.86% |       0.3807 |          0.0500 |
|     150 |         150 |       00:04:24 |       93.18% |       0.2103 |          0.0500 |
|     200 |         200 |       00:05:48 |       94.19% |       0.1587 |          0.0500 |
|     250 |         250 |       00:07:10 |       95.42% |       0.1275 |          0.0500 |
|     300 |         300 |       00:08:33 |       96.30% |       0.1036 |          0.0500 |
|=========================================================================================|
```



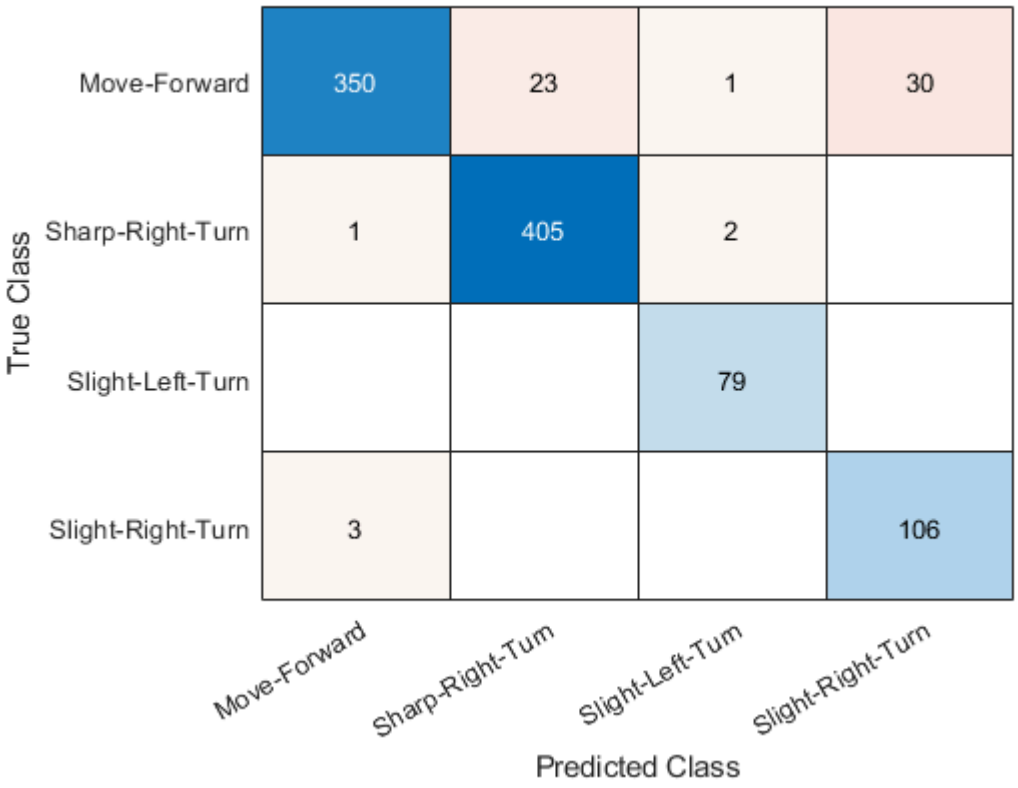

```
net3 =
  SeriesNetwork with properties:

         Layers: [5×1 nnet.cnn.layer.Layer]
     InputNames: {'sequenceinput'}
    OutputNames: {'classoutput'}
```

Plot confusion matrix

```
pred3=classify(net3,XTest)
```

```
pred3 = 1×1000 categorical
Slight-Left-Turn    Slight-Left-Turn    Slight-Left-Turn    Slight-Left-Turn    ...
```

```
confusionchart(YTest,pred3)
```


Move-Forward
Sharp-Right-Turn
Slight-Left-Turn
Slight-Right-Turn
True Class
350
23
1
30
1
405
2
79
3
106
Move-Forward
Sharp-Right-Turn
Slight-Left-Turn
Slight-Right-Turn
Predicted Class