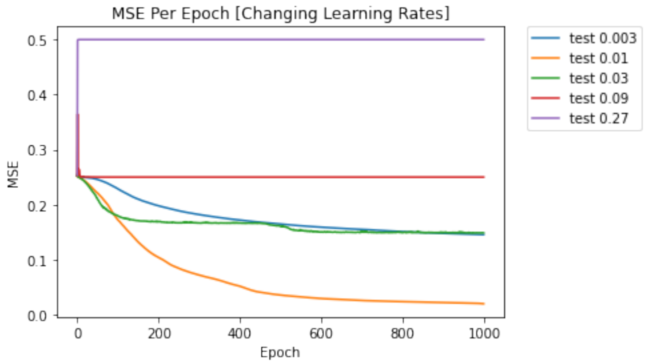
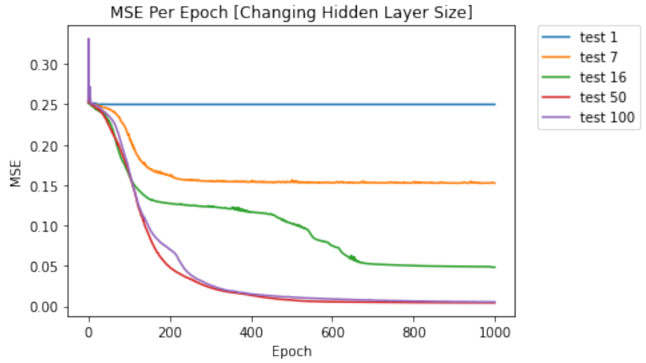
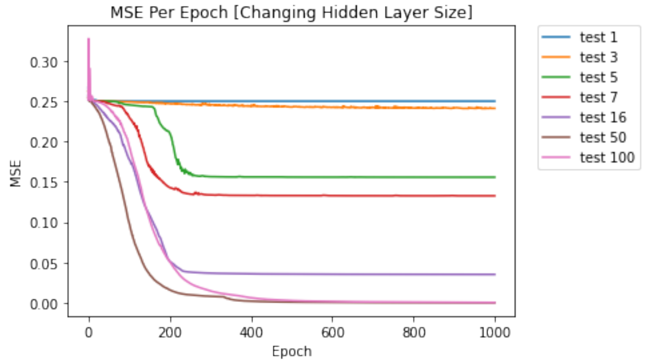
1.1)



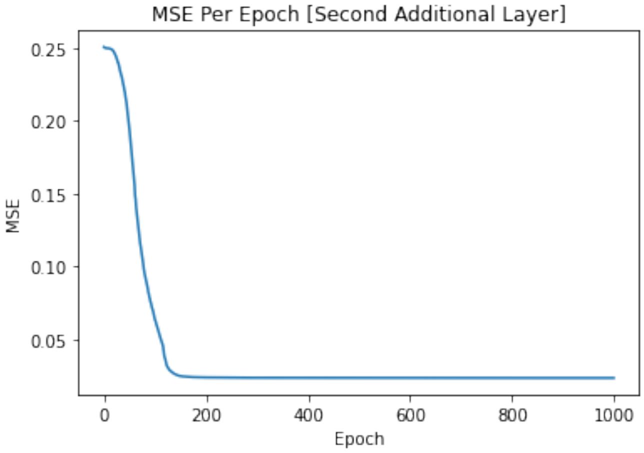
1.2)





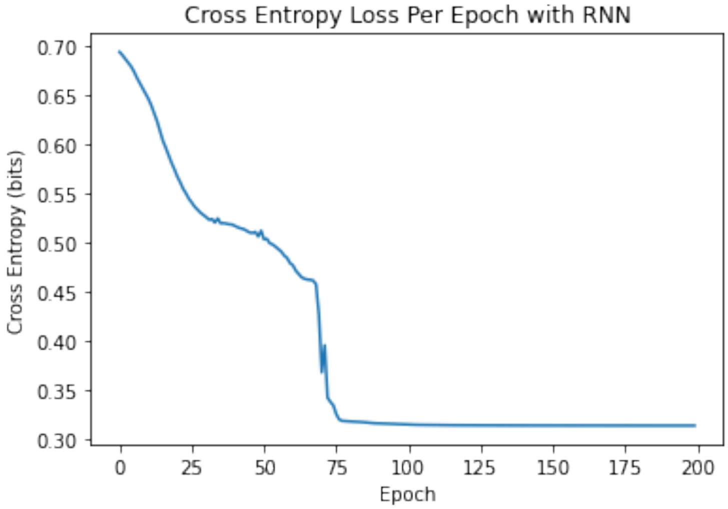
As we can see, a good hidden layer size smaller than 16 is between the input and output size, as it tapers off when the layer size is 3. After printing out the softmax results for all the hidden sizes of 1-16, it gets bad after 5.

1.3)

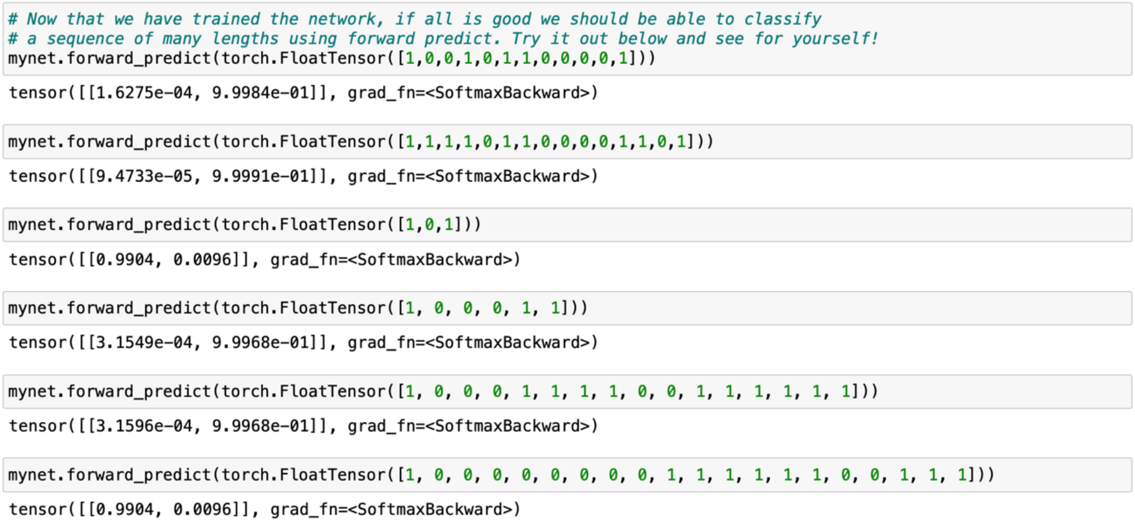


z

2.1)



2.2)



```
# Now that we have trained the network, if all is good we should be able to classify
# a sequence of many lengths using forward predict. Try it out below and see for yourself!
mynet.forward_predict(torch.FloatTensor([1,0,0,1,0,1,1,0,0,0,0,1]))
```

```
tensor([[1.6275e-04, 9.9984e-01]], grad_fn=<SoftmaxBackward>)
```

```
mynet.forward_predict(torch.FloatTensor([1,1,1,1,0,1,1,0,0,0,0,1,1,0,1]))
```

```
tensor([[9.4733e-05, 9.9991e-01]], grad_fn=<SoftmaxBackward>)
```

```
mynet.forward_predict(torch.FloatTensor([1,0,1]))
```

```
tensor([[0.9904, 0.0096]], grad_fn=<SoftmaxBackward>)
```

```
mynet.forward_predict(torch.FloatTensor([1, 0, 0, 0, 1, 1]))
```

```
tensor([[3.1549e-04, 9.9968e-01]], grad_fn=<SoftmaxBackward>)
```

```
mynet.forward_predict(torch.FloatTensor([1, 0, 0, 0, 1, 1, 1, 1, 0, 0, 1, 1, 1, 1, 1, 1]))
```

```
tensor([[3.1596e-04, 9.9968e-01]], grad_fn=<SoftmaxBackward>)
```

```
mynet.forward_predict(torch.FloatTensor([1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 0, 0, 1, 1, 1]))
```

```
tensor([[0.9904, 0.0096]], grad_fn=<SoftmaxBackward>)
```