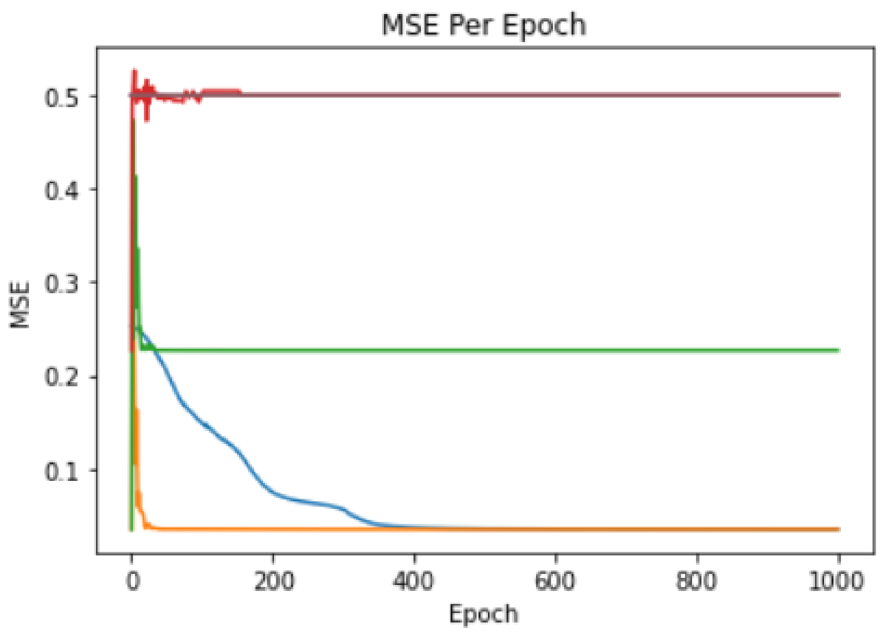
Homework 4

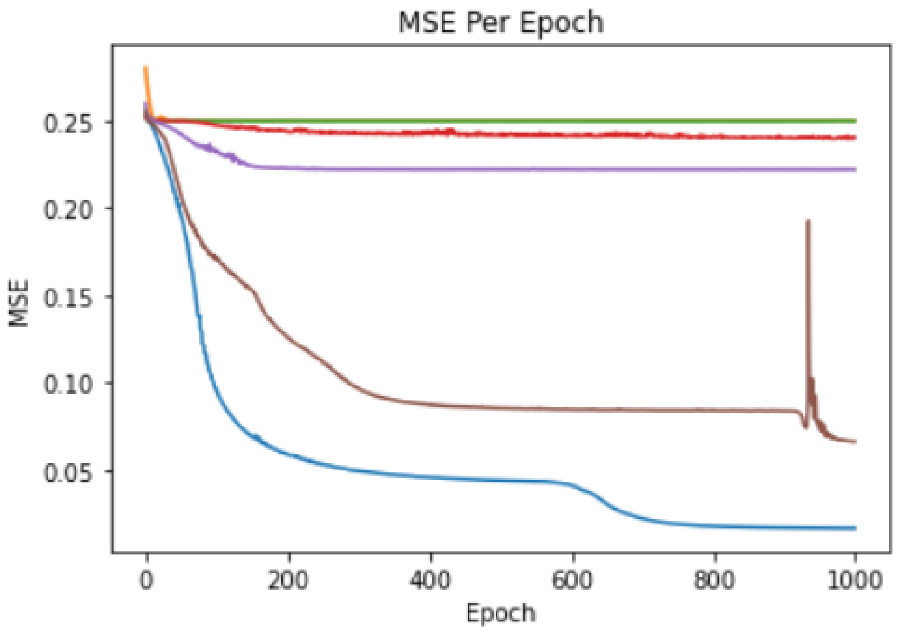
Feed Forward Tasks:

Graph of loss curve for 5 different learning rates



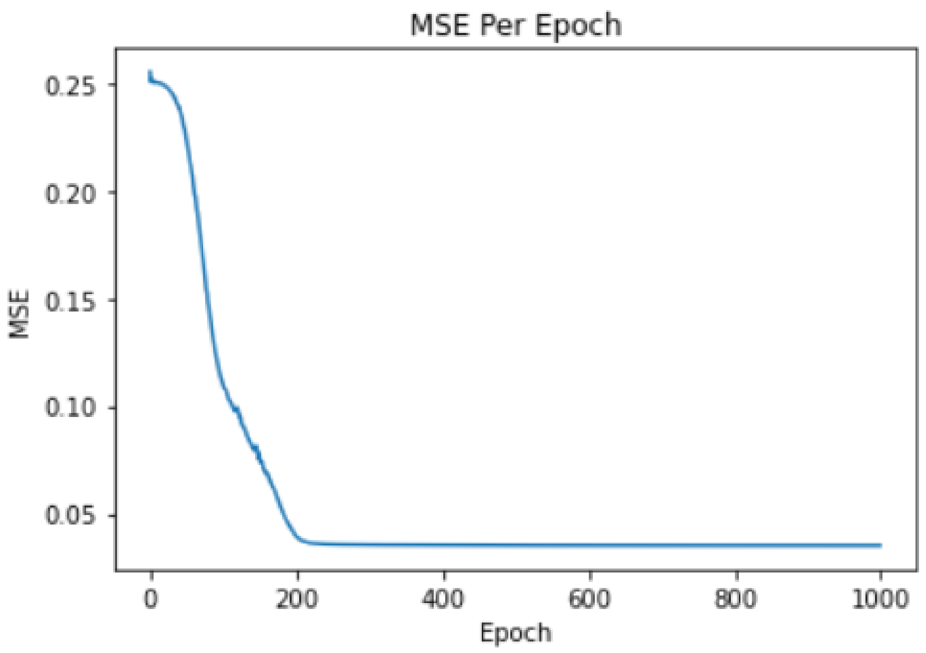
Learning rates used: 0.03 (original), 0.09, 0.27, 0.81, 0.01, and 0.0033.

Graph of loss curve for 5 different numbers of hidden nodes



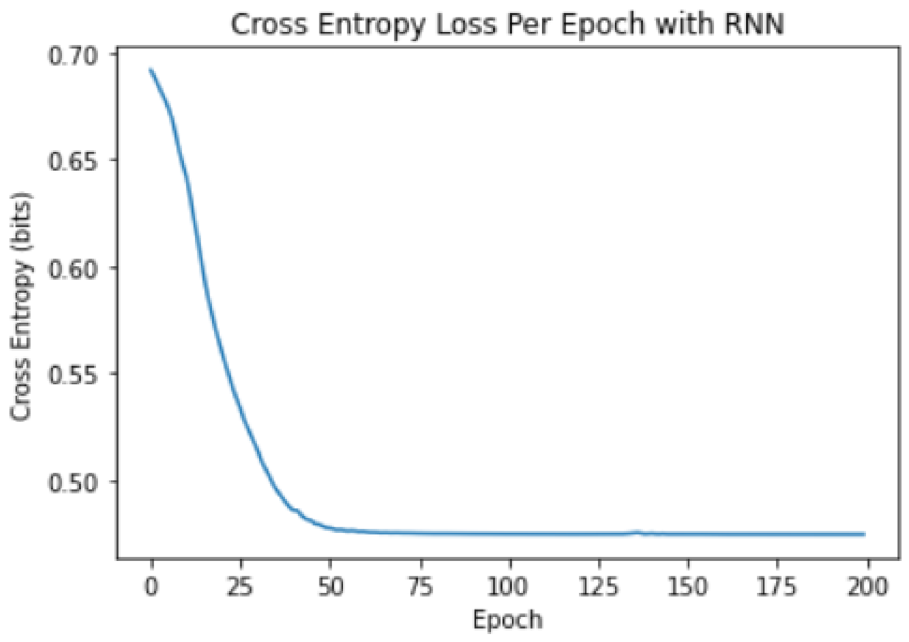
Hidden state value: 1, 2, 4, 8, 12, and 16 (original). From testing, twelve seemed to be the lowest state value you could use before learning became really poor (and even here it becomes weird towards the end).

Graph of loss curve with an additional layer

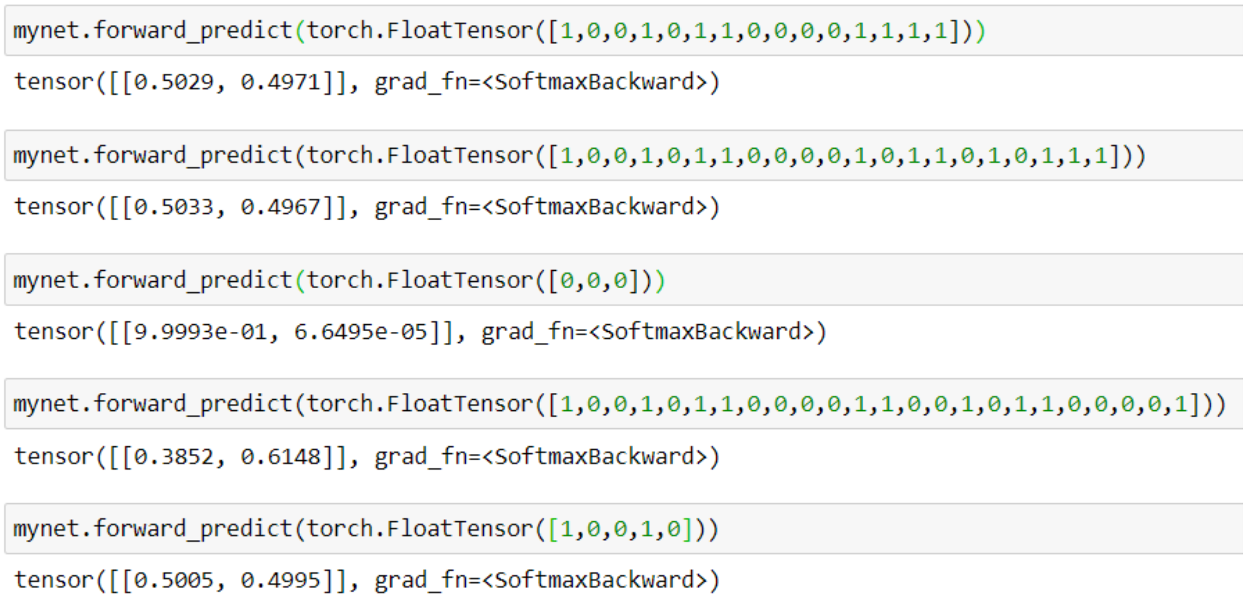


RNN Tasks:

RNN Loss Curve



RNN Classification of 5 Different Length Sequences



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

```
tensor([[0.5029, 0.4971]], grad_fn=<SoftmaxBackward>)
```

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

```
tensor([[0.5033, 0.4967]], grad_fn=<SoftmaxBackward>)
```

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

```
tensor([[9.9993e-01, 6.6495e-05]], grad_fn=<SoftmaxBackward>)
```

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

```
tensor([[0.3852, 0.6148]], grad_fn=<SoftmaxBackward>)
```

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

```
tensor([[0.5005, 0.4995]], grad_fn=<SoftmaxBackward>)
```