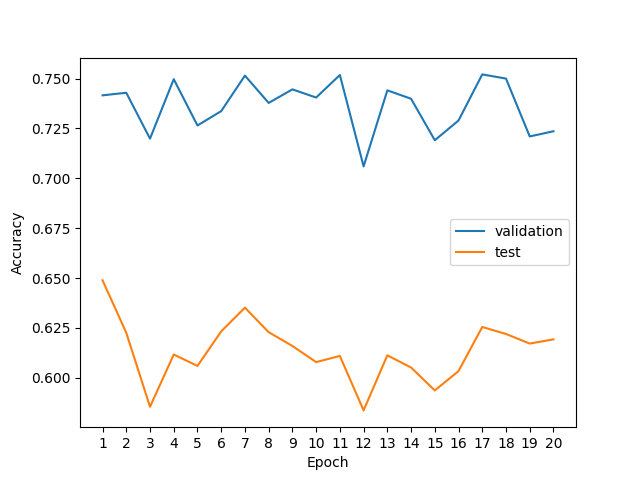
**Work Distribution**

The coding exercises were done by both students which in turn were compared, choosing the best implementation, the testing and bug solving was done as a group. Question number 3 was solved together. Both students contributed equally.

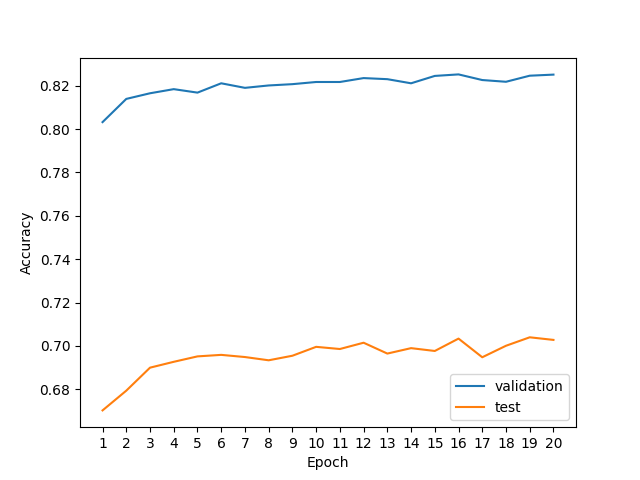
**Question 1**

1. **Performance:** 
   * **Test set:** 0.6193
   * **Validation set:** 0.7236

****

1. **Performance**

* **Test set:** 0.7028
* **Validation set:** 0.8251

****

**2.**

**a)** A simple perceptron such as the one implemented in the previous exercise is a model that cannot solve non-linearly separable problems since the VC- dimension of a line is 3 and a single line in 2 dimensions cannot discriminate it.

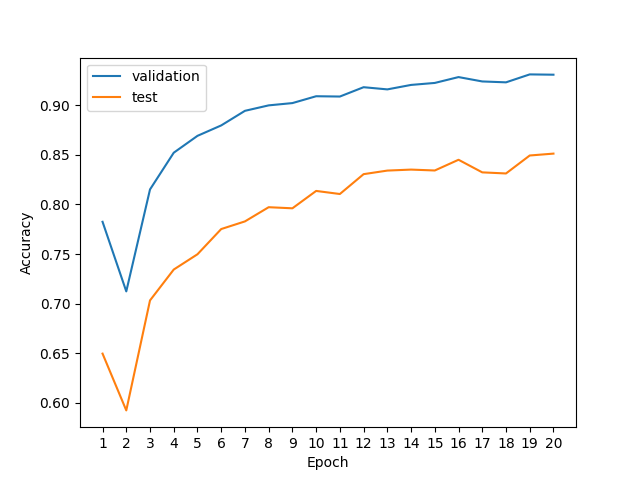
As opposed to this the multi layered perceptron can solve non-linearly separable problems like XOR.

This is possible because each hidden layer of the perceptron computes a representation of the input and propagates it forward. This in turn will increase the expressive power of the network allowing for more complex, non-linear models.

In case the activation function is linear the multi-layer perceptron will have the same results as the simple perceptron. A multi-layer perceptron with linear activations can be replaced with a simple perceptron by composing those linear activation functions into one linear function.

**b) Performance:**

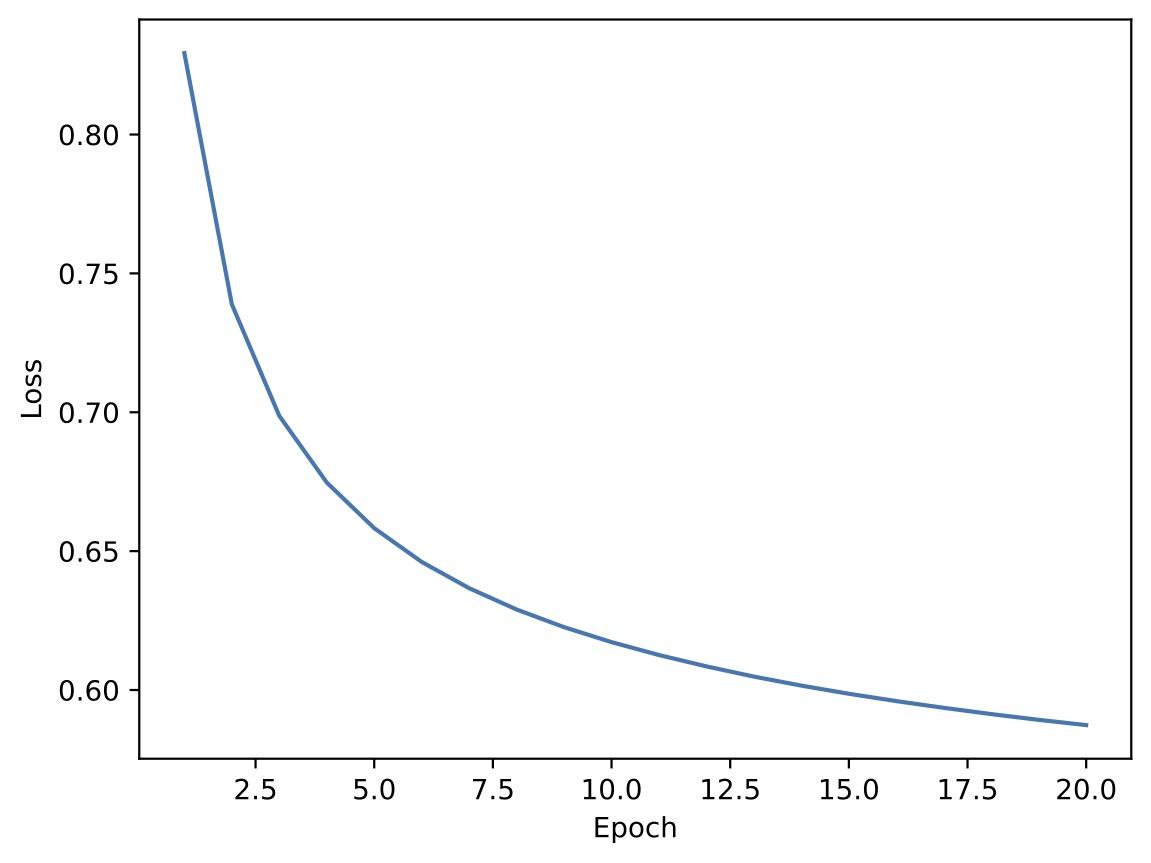
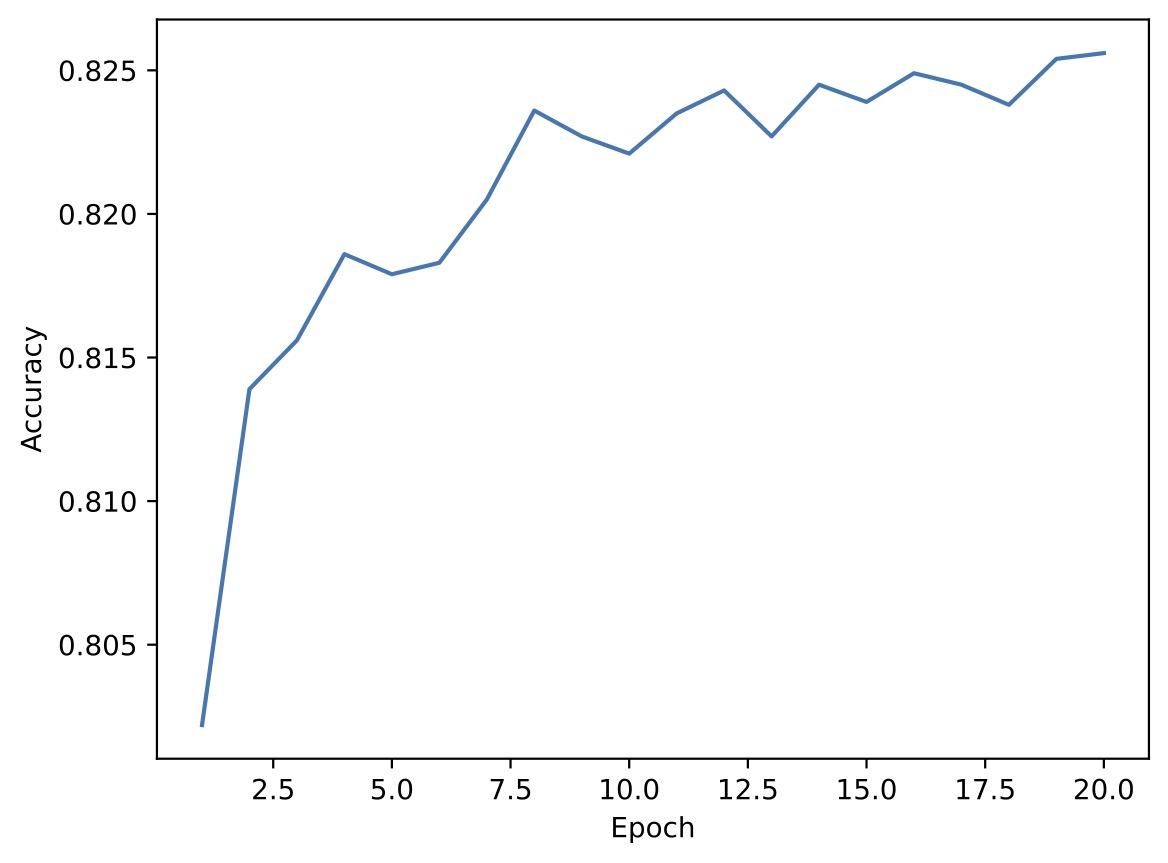
* **Test set:** 0.8512
* **Validation set:** 0.9307

****

**Question 2**

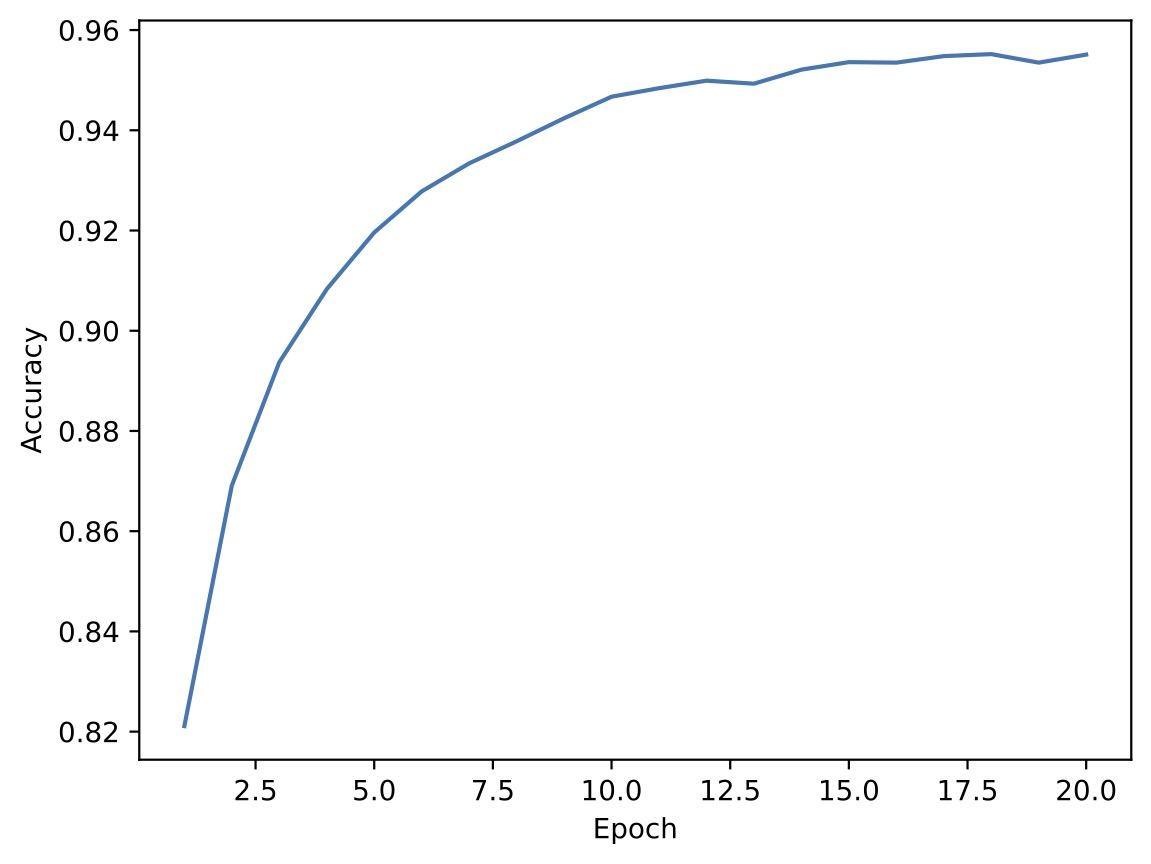
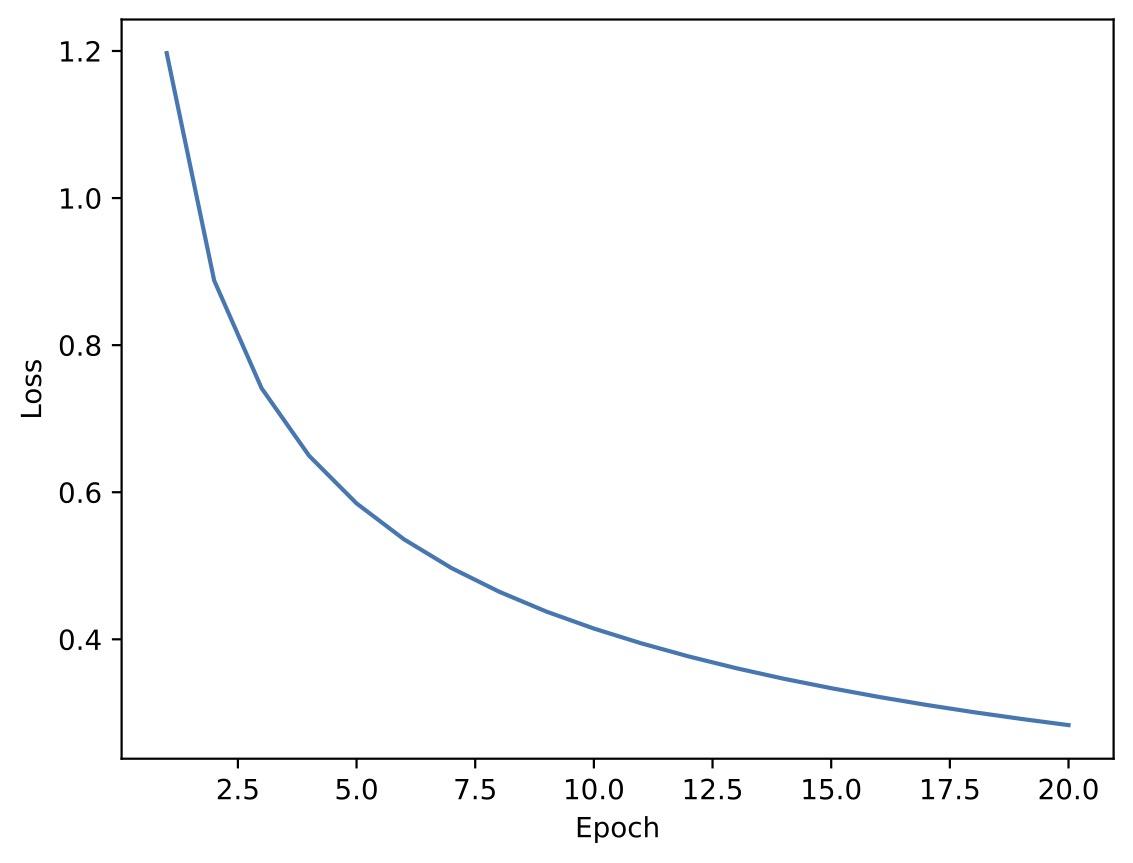
1. **Best configuration with learning rate** 0.001

**Final Accuracy of test set:** 0.7019

****

|  |  |
| --- | --- |
| **Best Hyperparameters and Design Choices** | |
| **Learning Rate** | 0.01 |
| **Hidden Size** | 200 |
| **Dropout Probability** | 0.3 |
| **Activation Function** | Relu |

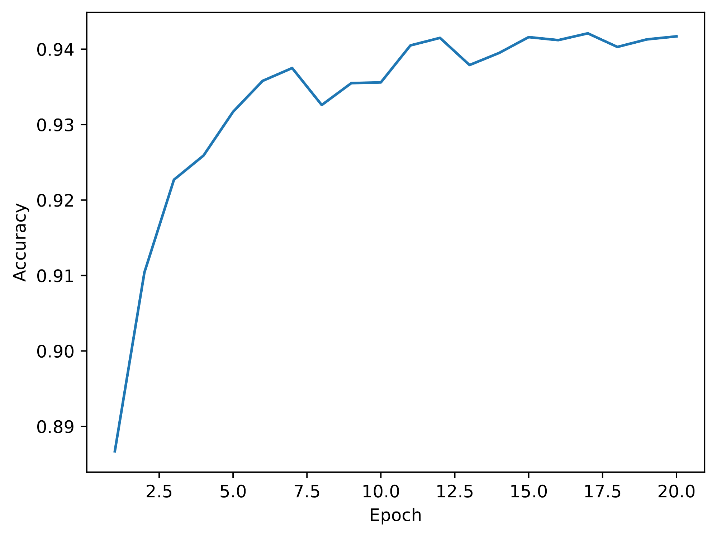
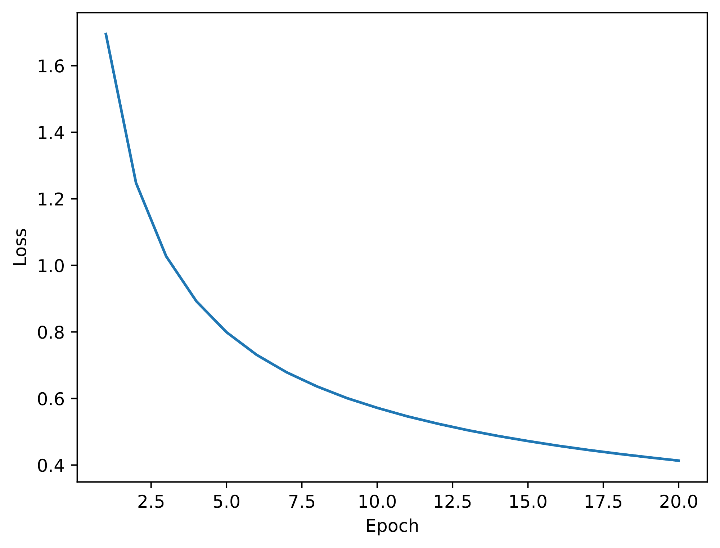
**Final Accuracy of test set:** 0.8953

****

* **With 2 Layers**

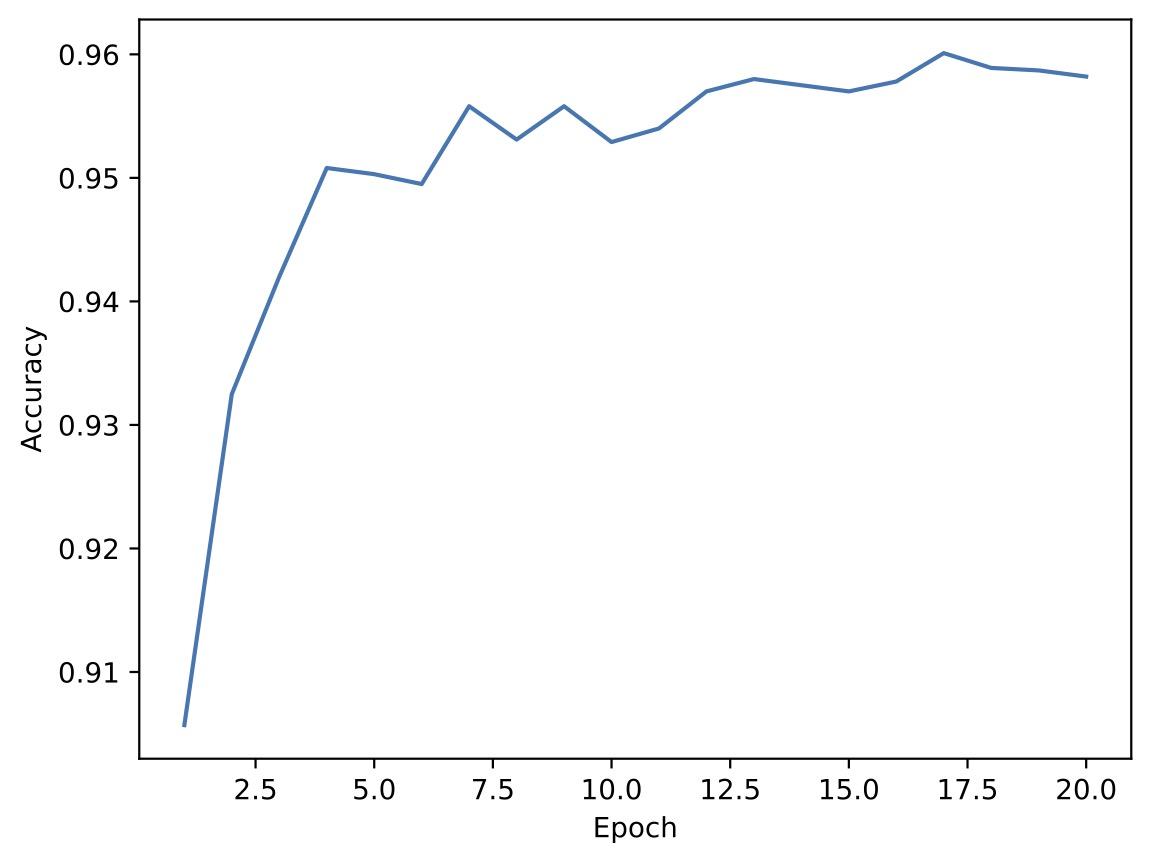
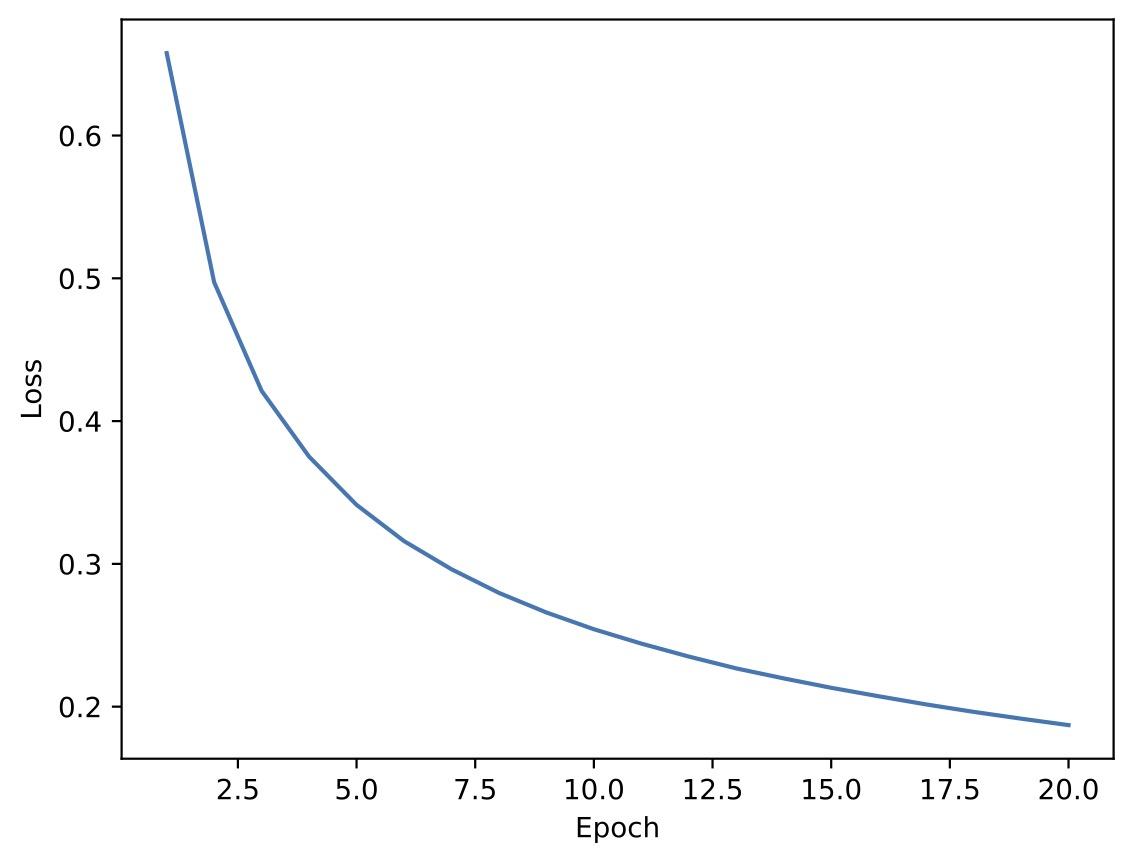
**Original Parameters (Shown in the provided table):**

**Final Accuracy of test set: 0.8633**

****

|  |  |
| --- | --- |
| **Best Hyperparameters and Design Choices** | |
| **Learning Rate** | 0.1 |
| **Hidden Size** | 200 |
| **Dropout Probability** | 0.3 |
| **Activation Function** | Relu |

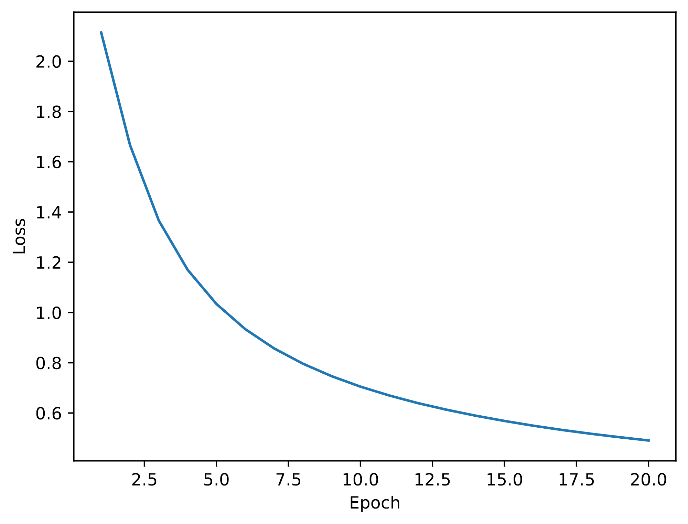
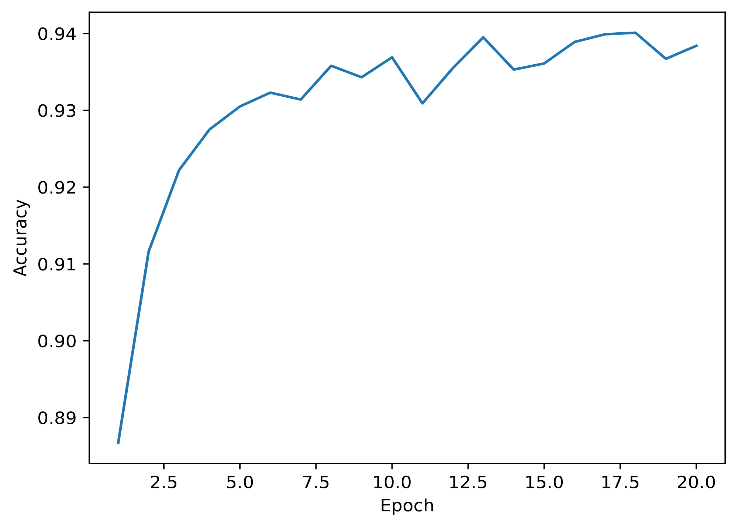
**Final Accuracy of test set:** 0.9011

****

* **With 3 Layers**

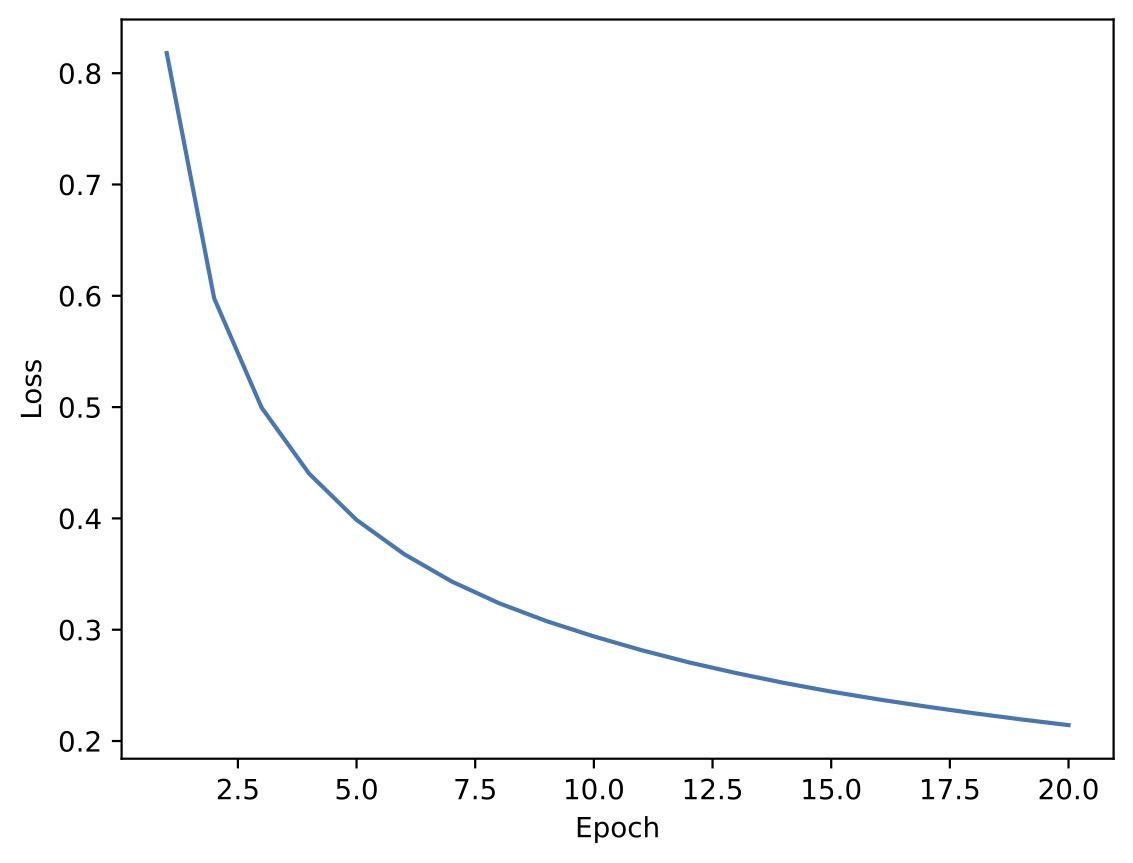
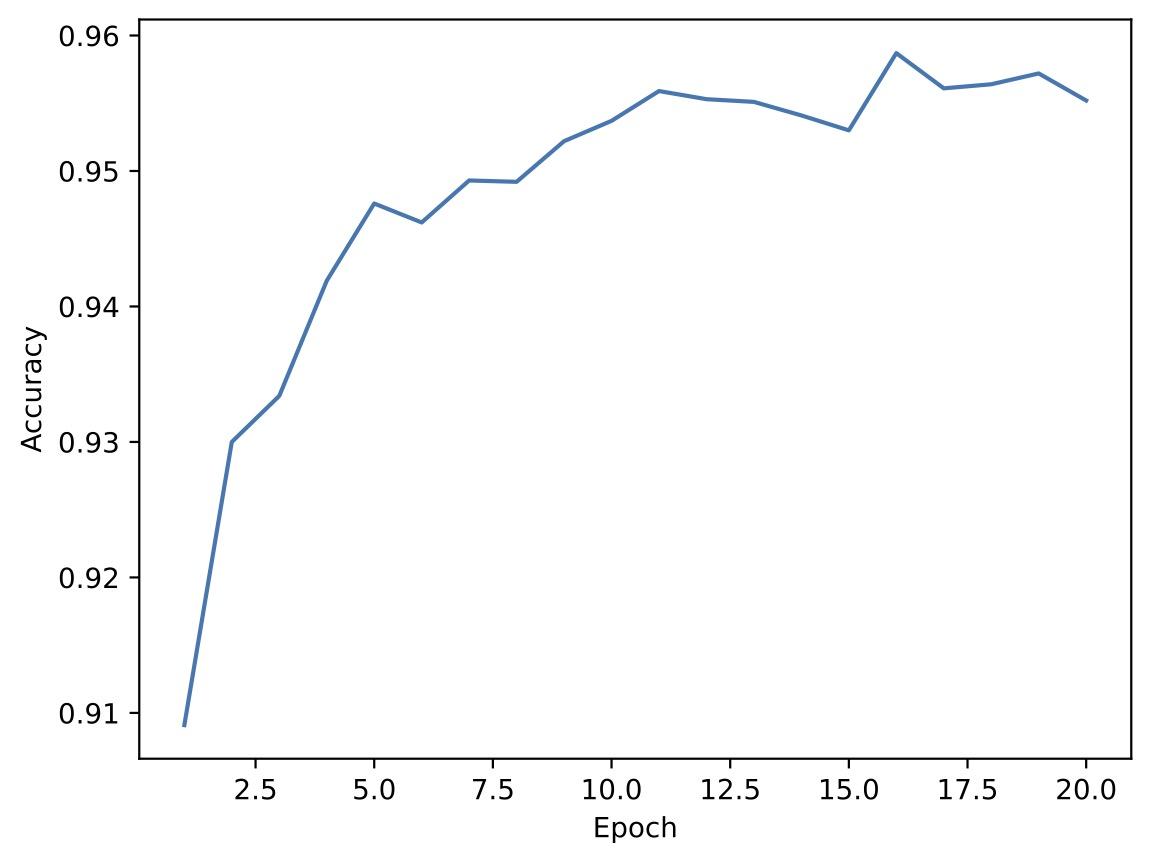
**Original Parameters (Shown in the provided table):**

**Final Accuracy of test set: 0.8600**

****

|  |  |
| --- | --- |
| **Best Hyperparameters and Design Choices** | |
| **Learning Rate** | 0.1 |
| **Hidden Size** | 200 |
| **Dropout Probability** | 0.3 |
| **Activation Function** | Relu |

**Final Accuracy of test set:** 0.8990

****

**Question 3**

1. Considering , we can say that:

Considering the below equation from the first row of the expressions above, we want to know what is and

Therefore, we could consider that

,

So, we can conclude that:

and

1. We can say that since we previously showed that for a certain .

Therefore, we can conclude that .

Even though is a linear combination of the rows in , the entries in this matrix are quadratic, as exemplified above, so is not a linear function of Θ = (W , v). Because of this, the resulting model is **not linear** in terms of the original parameters Θ, but **quadratic** in terms of **W**.

1. From the previous exercises we have:

and

Let denote de Frobenius inner product .

We also know that .

With this and inner product properties we can keep manipulating the expression:

also knowing that is a scalar we can obtain:

which will be the product between 2 matrixes.

* will be a symmetric matrix since is a diagonal matrix and are symmetric.
* will belong to for a

What we pretend to do now is collect the initial parameters Θ = (W , v) expressed in the matrix and put them in a vector which will be equal to .

This vector will be obtained with since the question states that and we will be able fit all the parameters in our new model in terms of . This together with question shows us that it will be a linear model.

Lastly in case we get a matrix with and because of this we might not be able to capture all of the original parameters Θ = (W , v) in terms of .

1. Since is linear we will be able to find a closed form solution to .

Let X be a matrix with as rows, and . D = training data with

We then have which we can write as which is the L2 norm of . We now want to minimize this expression regarding . This is trivial because it is the known Least Squares problem.

The result will be .

This is a global minimum which is usually hard to find for activation function such as tanh or Relu but in this case we have a polynomial activation function. The main reason why it is possible is because Least Squares is a convex problem with a closed form solution for the optimal parameters. This means that there will be a global minimum that can be found by using mathematical techniques.