Machine Learning - Assignment 3

Meraj Ahmed

2019MCS2565

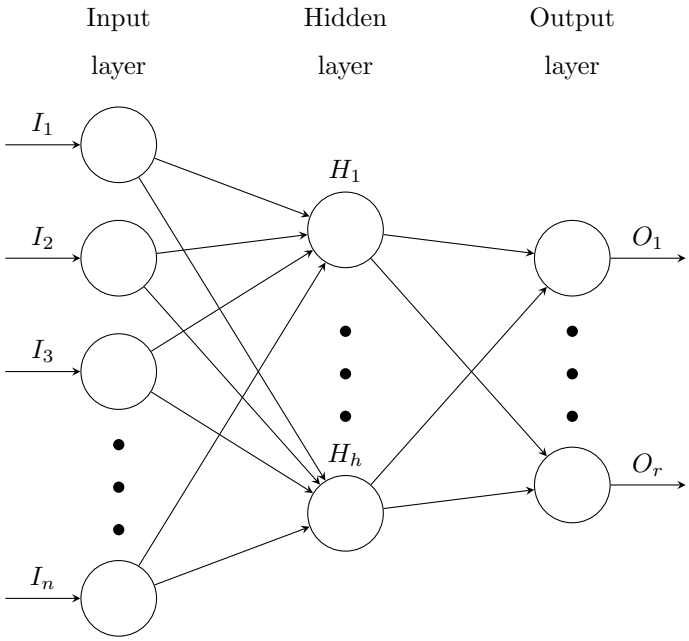
1 Implementation

We have to implement neural networks from scratch for Alphabet recognition. Input is 13,000 image of alphabets with 28*28 pixels. Each pixel is a feature of image.

The goal is to correctly classify each image from A-Z. The neural network which will implement this will have 784 size of input layer and 26 size of output layer. We will implement Neural network which will take following parameters as input

- Mini batch size
- No. of input features
- Hidden layer architecture
- Output layer
- Hidden layer's activation function
- Adaptive
- Maximum epoch
- No. of epoch no change
- Random state
- Verbose

- Weight Initialization



2 Single hidden Layer - Constant Learning rate

2.1 Convergence condition

1. If cost difference of last 1000 iteration is lesser than 10^{-4} for consecutive n_epoch_no_change times(default=10)

2. If Neural Network is trained for max_epoch times(default=1500)

2.2 Parameters

2.2.1 Hidden layer unit = 100

Learning rate = 0.1

Train accuracy = 96.5384 %

Test accuracy = 90.9538 %

Time to fit = 362.437 s

No. of epoch = 1500

No. of iteration = 195000

Weight Initialization = Xavier

Final cost = 0.03481

2.2.2 Hidden layer unit = 50

Learning rate = 0.1

Train accuracy = 94.6692 %

Test accuracy = 88.5538 %

Time to fit = 261.794 s

No. of epoch = 1493

No. of iteration = 194000

Weight Initialization = Xavier

Final cost = 0.04975

2.2.3 Hidden layer unit = 10

Learning rate = 0.1

Train accuracy = 78.0769 %

Test accuracy = 71.5846 %

Time to fit = 157.556 s

No. of epoch = 1500

No. of iteration = 195000

Weight Initialization = Xavier

Final cost = 0.17811

2.2.4 Hidden layer unit = 5

Learning rate = 0.1

Train accuracy = 3.9692 %

Test accuracy = 4.0615 %

Time to fit = 3.94 s

No. of epoch = 39

No. of iteration = 5000

Weight Initialization = Xavier

Final cost = 0.4808

2.2.5 Hidden layer unit = 1

Learning rate = 0.1

Train accuracy = 3.8462 %

Test accuracy = 3.8462 %

Time to fit = 5.667 s

No. of epoch = 70

No. of iteration = 9000

Weight Initialization = Xavier

Final cost = 0.4808

2.3 Plots

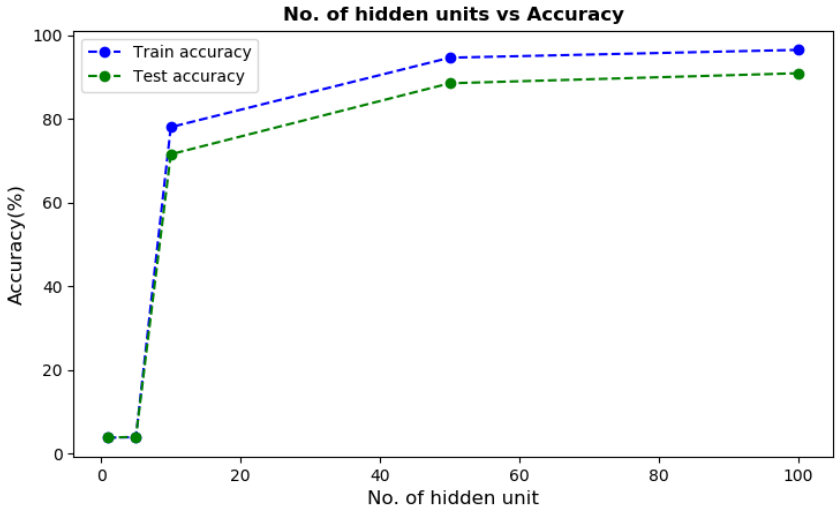


Figure 1: Hidden unit vs Accuracy

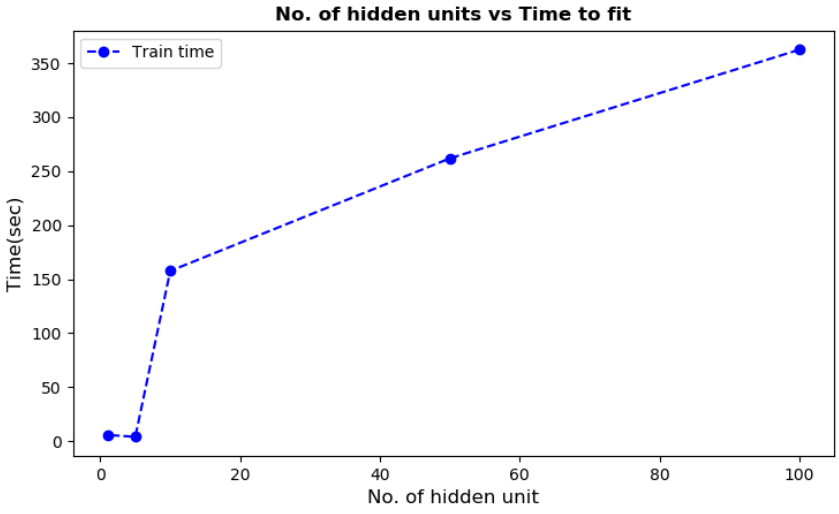


Figure 2: Hidden unit vs Train Time

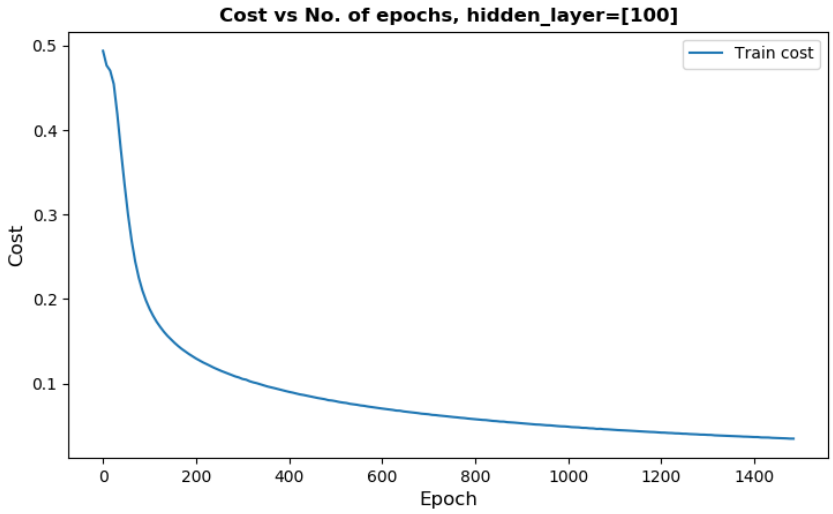


Figure 3: Cost vs Epoch(Hidden layer=100)

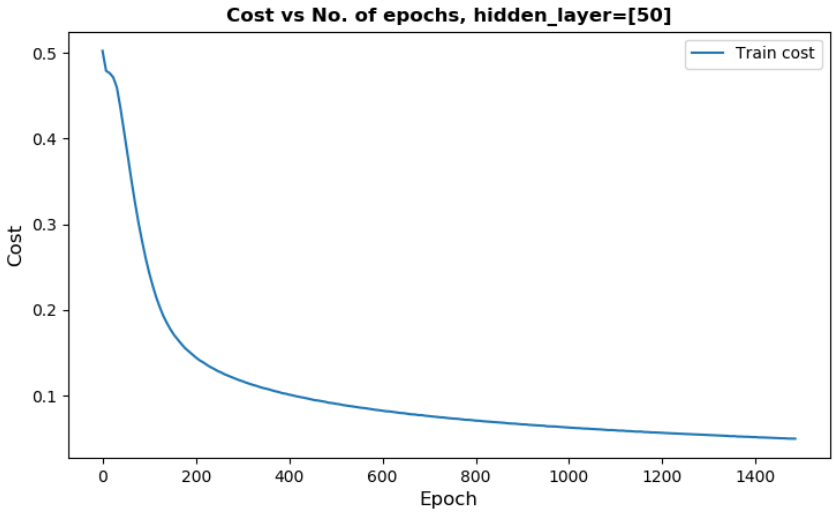


Figure 4: Cost vs Epoch(Hidden layer=50)

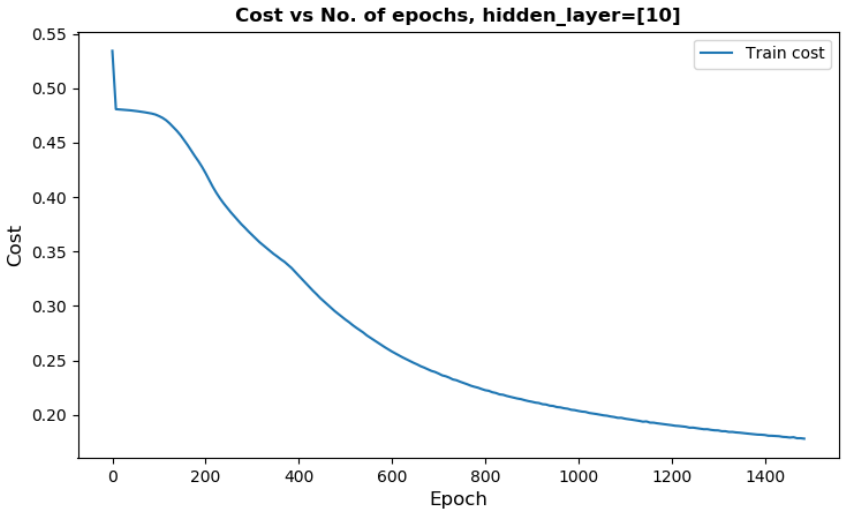


Figure 5: Cost vs Epoch(Hidden layer=10)

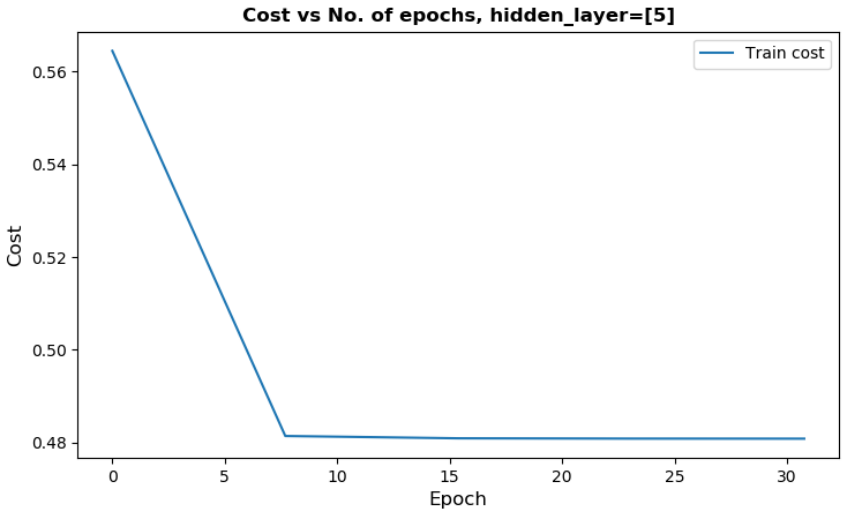


Figure 6: Cost vs Epoch(Hidden layer=5)

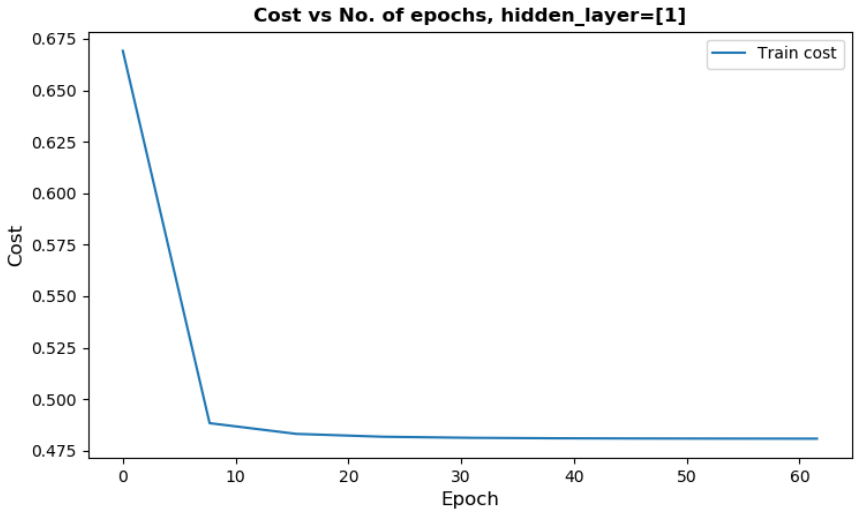


Figure 7: Cost vs Epoch(Hidden layer=1)

2.4 Comments

As we can see from the accuracy plot that, accuracy of model increases with increasing no. of unit in hidden layer.

Highest accuracy with 100 hidden layer unit. This is because large hidden unit can compute complex boundary.

But as the no. of unit in hidden layer increases, time to fit the model is also increases.

Model with 1 and 5 hidden unit perform worst, because it is simple model that it can't classify the data with 26 classes.

So, we need some complexity in model such that it can classify the data with good accuracy in reasonable time.

3 Single hidden Layer - Adaptive Learning rate

3.1 Introduction

In adaptive learning rate. learning rate at each epoch is inversely proportional to the no. of epoch

$$\eta_e = \frac{\eta_0}{\sqrt{e}} \tag{1}$$

where e is the current epoch no and η_0 is the initial learning rate

3.2 Convergence condition

1. If cost difference of last 1000 iteration is lesser than 10^{-12} for consecutive n_epoch_no_change times(default=10)
2. If Neural Network is trained for max_epoch times(default=1500)

3.3 Parameters

3.3.1 Hidden layer unit = 100

Learning rate = 0.1
Train accuracy = 90.1769 %
Test accuracy = 87.1692 %
Time to fit = 427.128 s
No. of epoch = 1500
No. of iteration = 195000
Weight Initialization = Xavier
Final cost = 0.0938

3.3.2 Hidden layer unit = 50

Learning rate = 0.1
Train accuracy = 88.9692 %
Test accuracy = 85.7538 %

Time to fit = 290.91 s

No. of epoch = 1500

No. of iteration = 195000

Weight Initialization = Xavier

Final cost = 0.1050

3.3.3 Hidden layer unit = 10

Learning rate = 0.1

Train accuracy = 51.0615 %

Test accuracy = 49.9846 %

Time to fit = 164.545 s

No. of epoch = 1500

No. of iteration = 195000

Weight Initialization = Xavier

Final cost = 0.3231

3.3.4 Hidden layer unit = 5

Learning rate = 0.1

Train accuracy = 3.8462 %

Test accuracy = 3.8462 %

Time to fit = 148.285 s

No. of epoch = 1500

No. of iteration = 195000

Weight Initialization = Xavier

Final cost = 0.47482

3.3.5 Hidden layer unit = 1

Learning rate = 0.1

Train accuracy = 3.8462 %

Test accuracy = 3.8462 %

Time to fit = 123.512 s

No. of epoch = 1500

No. of iteration = 195000

Weight Initialization = Xavier

Final cost = 0.4807

3.4 Plots

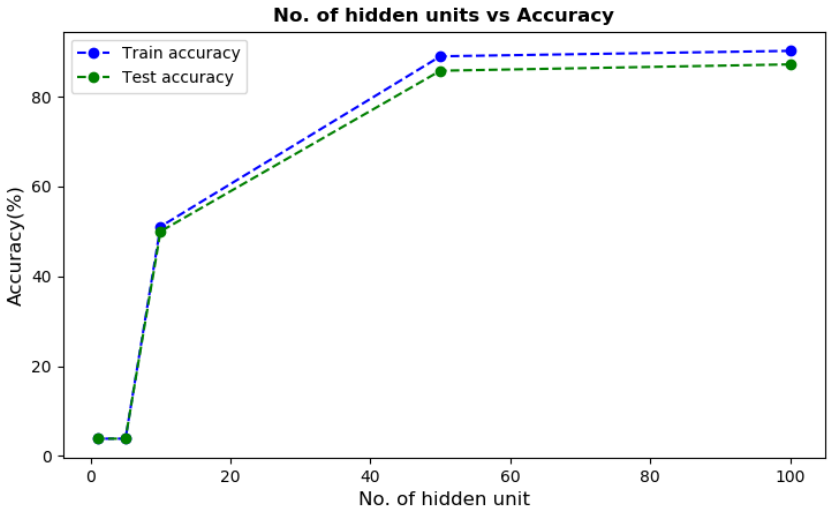


Figure 8: Hidden unit vs Accuracy

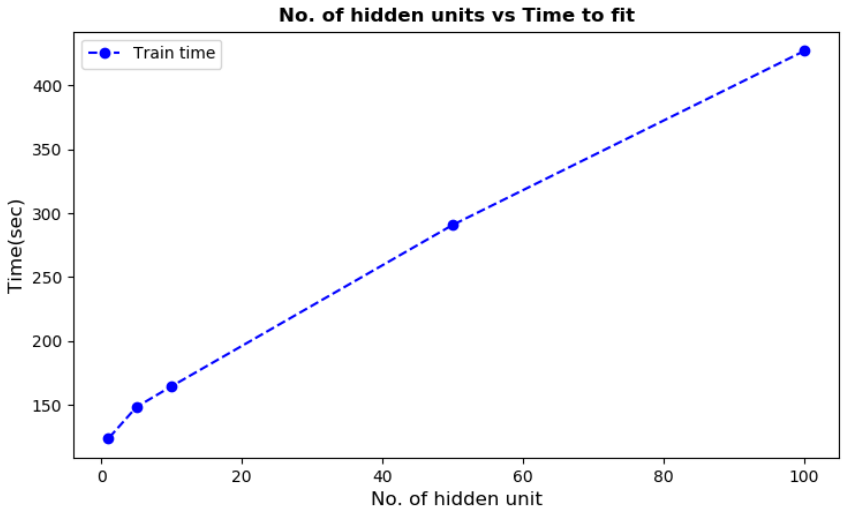


Figure 9: Hidden unit vs Train Time

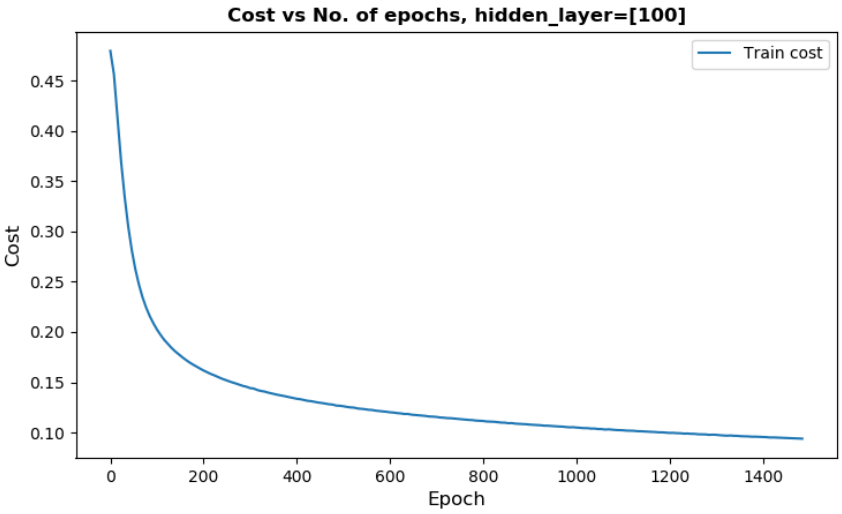


Figure 10: Cost vs Epoch(Hidden layer=100)

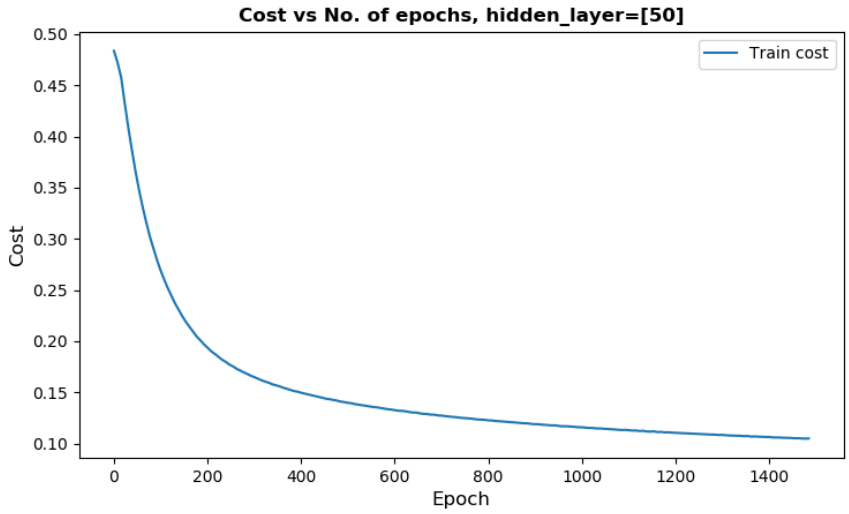


Figure 11: Cost vs Epoch(Hidden layer=50)

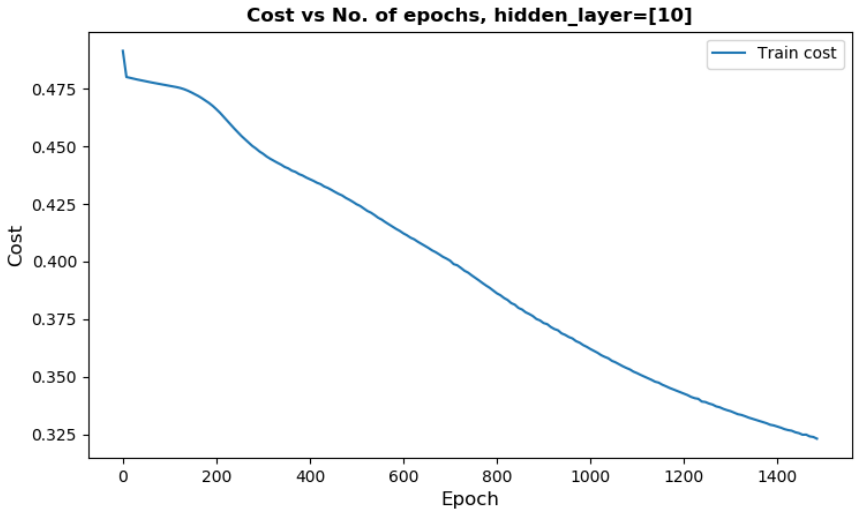


Figure 12: Cost vs Epoch(Hidden layer=10)

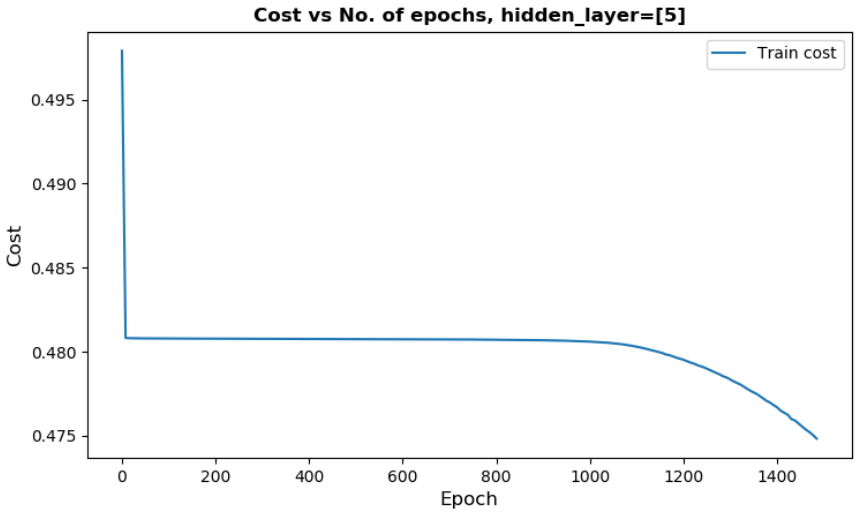


Figure 13: Cost vs Epoch(Hidden layer=5)

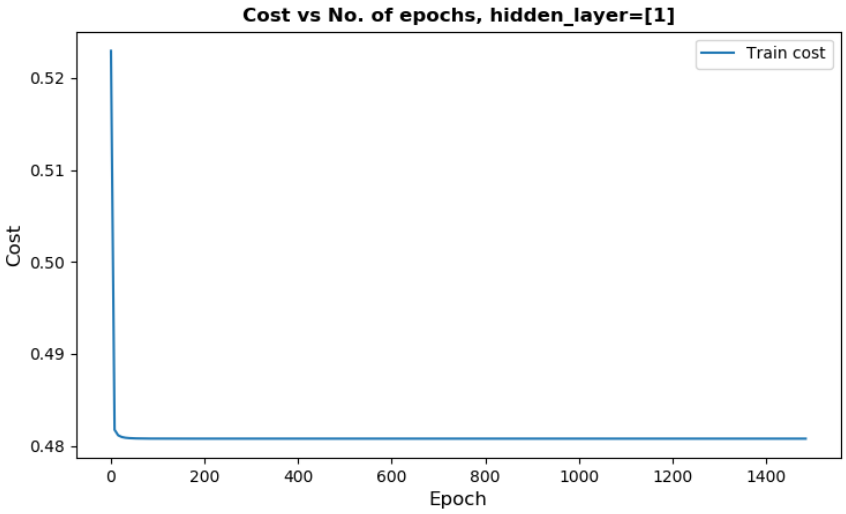


Figure 14: Cost vs Epoch(Hidden layer=1)

3.5 Comments

As we can see from the accuracy plot that, accuracy of model increases with increasing no. of unit in hidden layer.

But as the no. of unit in hidden layer increases, time to fit the model is also increases.

Accuracy with constant learning rate is better the adaptive learning rate.

In this case, as we decreases the η at each epoch, so the learning will be slower than the previous case.

But, as we are using SGD with 100 batch size, larger leaning rate will diverges more from the ideal learning. Slower learning rate will help in converging more near to optimum point. By adaptive learning rate, we decreases it at each iteration.

As we have fixed the max_epoch to 1500, both constant and adaptive learning rate takes full 1500 epoch. So, time taken is almost similar, but accuracy after 1500 epoch of training is better with constant learning rate as compared to adaptive learning rate.

4 ReLU - Adaptive Learning rate

4.1 Introduction

ReLU(Rectified Linear Unit) is a simple activation function composed of two linear pieces.
We used ReLU as a activation function in the hidden layer units and sigmoid at output layer units.

$$ReLU(z) = max(0, z)$$

Practically, ReLU can learn converge faster than the sigmoid.

4.2 Parameter

4.2.1 ReLU

Hidden layer = (100, 100)
Hidden activation = ReLU
Out activation = Sigmoid
Learning rate init = 0.5
Train accuracy = 94.9 %
Test accuracy = 90.2462 %
Time to fit = 472.901 s
No. of epoch = 1500
No. of iteration = 195000
Weight Initialization = Uniform(-0.01, 0.01)
Tol = 10^{-12}
Convergence condition = max_epoch is 1500 or change in cost is lesser than tol for consecutive n_epoch_no_change times(default=10)
Final cost = 0.04289

4.2.2 Sigmoid

Hidden layer = (100, 100)

Hidden activation = Sigmoid

Out activation = Sigmoid

Learning rate init = 0.5

Train accuracy = 84.1 %

Test accuracy = 81.5077 %

Time to fit = 416.406 s

No. of epoch = 1500

No. of iteration = 195000

Weight Initialization = Uniform(-0.01, 0.01)

Tol = 10^{-12}

Convergence condition = max_epoch is 1500 or change in cost is lesser than tol for consecutive n_epoch_no_change times(default=10)

Final cost = 0.1443

4.3 Plot

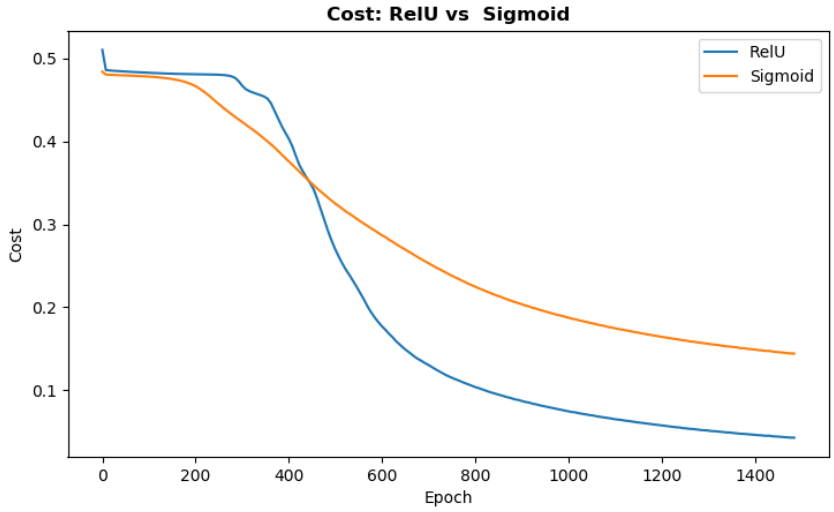


Figure 15: Cost of ReLU and Sigmoid

4.4 Comments

ReLU perform better than the sigmoid activation unit. Accuracy with ReLu is better than sigmoid

Test accuracy of ReLU is 90 % whereas test accuracy of Sigmoid is 81 %.

So ReLU has better test accuracy than the sigmoid unit. It generalizes better than the sigmoid. Comparing to part 2(b), test accuracy there was also 90 %. So both generalizes better. One issue with (100, 100) hidden layers may be with over-fit. As we complex our model, it may over-fit.

As we can see from the cost curve, ReLU converges faster than the sigmoid activation unit. Sigmoid learns slower. As we can see from the plot that both curve are constant initially, looks like some plateau on the cost curve at cost = 0.48 around. After that, ReLU decreases its cost faster than Sigmoid.

Time taken to train network is almost same in both cases.

Weight initialization is important to converge faster. We try with both He and

Xavier technique. Finally, we initialize weight from uniform in range of -0.01 to +0.01 distribution.

As compared to single hidden layer with 100 units with sigmoid activation and adaptive learning rate, ReLU accuracy is better after 1500 epoch of training.

As compared to single hidden layer with 100 units with sigmoid activation and constant learning rate, ReLU accuracy is almost similar after 1500 epoch of training.

5 MLPClassifier

5.1 Parameter

5.1.1 ReLU

Hidden layer = (100, 100)

Hidden activation = ReLU

Out activation = Sigmoid

Learning rate init = 0.5

Train accuracy = 89.9077 %

Test accuracy = 86.4615 %

Time to fit = 625.683 s

No. of iteration = 1500

Tol = 1e-7

Final cost = 0.7217

5.1.2 Sigmoid

Hidden layer = (100, 100)

Hidden activation = Sigmoid

Out activation = Sigmoid

Learning rate init = 0.5

Train accuracy = 41.7769 %

Test accuracy = 41.4615 %

Time to fit = 560.931 s

No. of iteration = 1500

Tol = 1e-7

Final cost = 2.7388

5.2 Plot

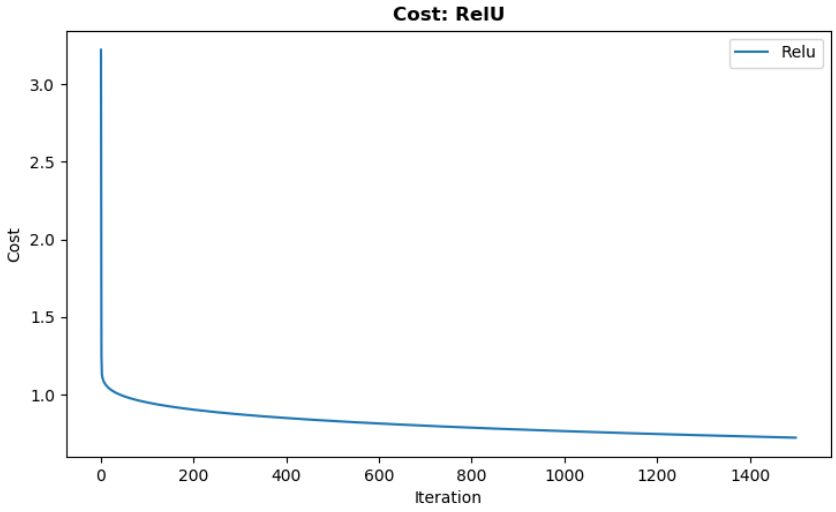


Figure 16: Cost: ReLU vs Iteration

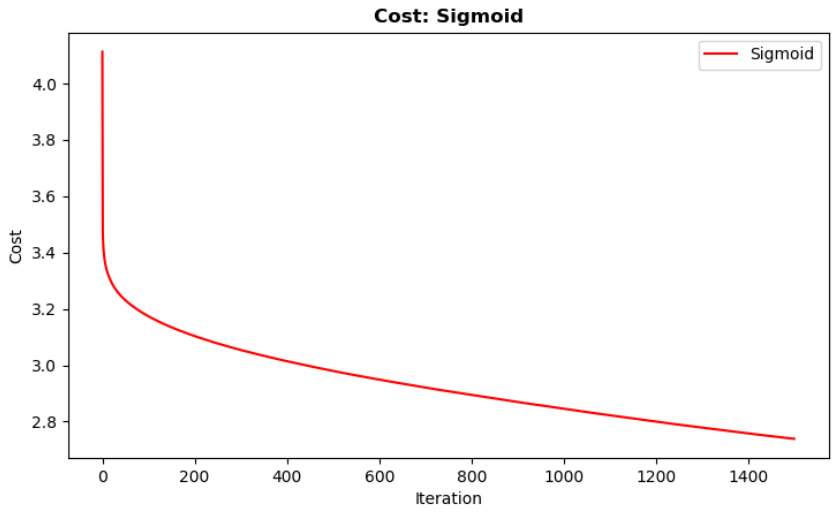


Figure 17: Cost: Sigmoid vs Iteration

5.3 Comments

We passed the one hot encoded train_y to the MLPClassifier. So, we are using MLPClassifier as multi label classifier. MLPClassifier predict all those label which has probability has greater than 0.5.

But, we have to output only single label for each example. So, we write our own prediction function which output label which has the highest probability.

In part (d), we use MSE loss function where MLPClassifier uses binary cross entropy loss function.

Accuracy is lesser than the previous part 2(d). In previous part we got 94.9 % train accuracy where here we got 89.9 % train accuracy.

In previous part we got 90.24 % train accuracy where here we got 86.4 % train accuracy.

In this part, using sigmoid activation function fails to converge to optimum poin after 1500 epoch of training.

As we can see from the above cost plot, RelU cost reach around 0.7 whereas sigmoid cost reach around 2.7 after 1500 epoch. Clearly, ReLU converges faster than the sigmoid.