Contents

Problem 1: Feed Forward Neural Network

Verification of gradient using gradcheck() function:

```
(base) saammmy@saammmy-xps15:~/Deep_Learning/4$ /home/saammmy/anaconda3/bin
 The Gradient check on 5 training examples gives the error as shown below:
1.0302307469074127e-06
```

Best Hyperparameters:

```
Validation loss and Accuracy: 0.37 , 87.98 %
Hyperparameters Updated
Best Hyperparameter uptill now: [50, 3, 35, 1, 0.01, 16]

-----------------------------------------------------------------------
 Grid Search Completed
-----------------------------------------------------------------------
 Results after performing Grid Search:
 Best Hyperparameters:
   Hidden Inputs in Each Layer=  50
   Hidden Layers=  3
   Epochs=  35
   Alpha=  1
   Learning Rate=  0.01
   Mini Batch Size=  16
```

- Training Loss (on training + validation dataset) and Test Accuracy:

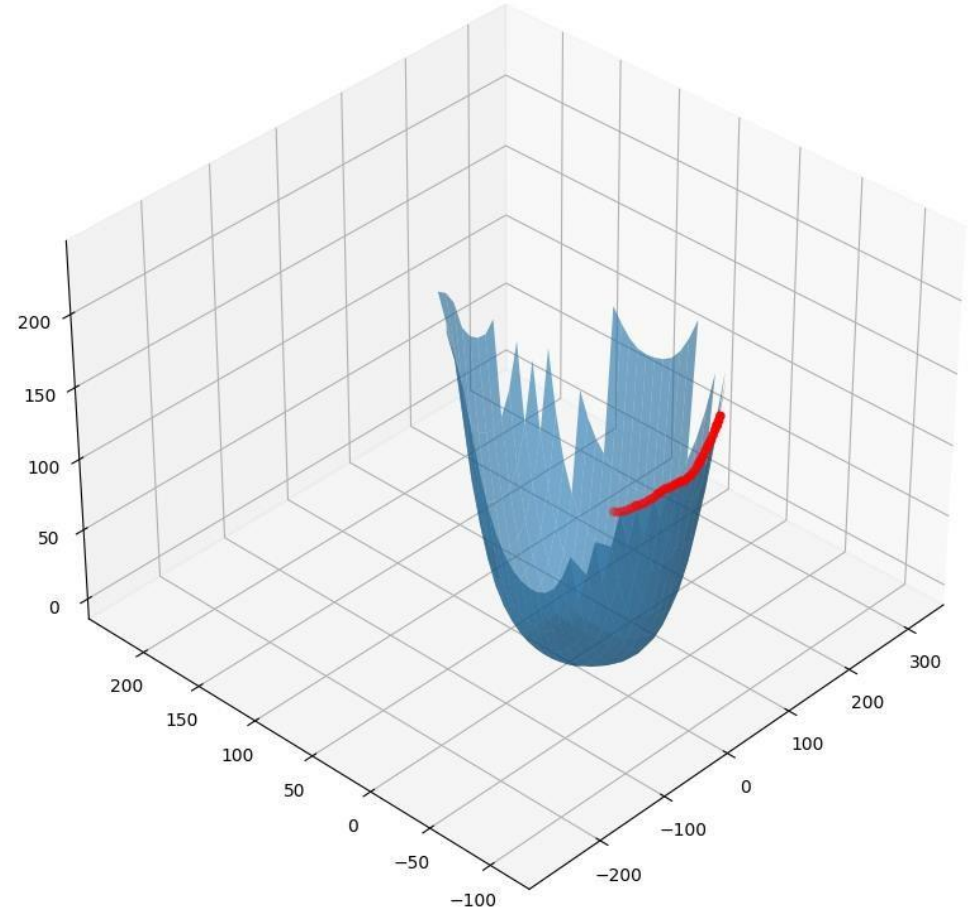
```
------------------------------------------------------------------------
Training on Training + Validation Dataset:
------------------------------------------------------------------------
Training loss and Accuracy at Epoch  1 : 0.60 , 78.99 %
Training loss and Accuracy at Epoch  2 : 0.48 , 83.34 %
Training loss and Accuracy at Epoch  3 : 0.42 , 85.04 %
Training loss and Accuracy at Epoch  4 : 0.40 , 85.83 %
Training loss and Accuracy at Epoch  5 : 0.38 , 86.49 %
Training loss and Accuracy at Epoch  6 : 0.36 , 87.06 %
Training loss and Accuracy at Epoch  7 : 0.35 , 87.48 %
Training loss and Accuracy at Epoch  8 : 0.34 , 87.70 %
Training loss and Accuracy at Epoch  9 : 0.32 , 88.18 %
Training loss and Accuracy at Epoch  10 : 0.32 , 88.36 %
Training loss and Accuracy at Epoch  11 : 0.31 , 88.62 %
Training loss and Accuracy at Epoch  12 : 0.30 , 88.84 %
Training loss and Accuracy at Epoch  13 : 0.30 , 89.05 %
Training loss and Accuracy at Epoch  14 : 0.29 , 89.29 %
Training loss and Accuracy at Epoch  15 : 0.28 , 89.61 %
Training loss and Accuracy at Epoch  16 : 0.28 , 89.72 %
Training loss and Accuracy at Epoch  17 : 0.27 , 89.92 %
Training loss and Accuracy at Epoch  18 : 0.27 , 90.14 %
Training loss and Accuracy at Epoch  19 : 0.26 , 90.37 %
Training loss and Accuracy at Epoch  20 : 0.26 , 90.44 %
Training loss and Accuracy at Epoch  21 : 0.25 , 90.49 %
Training loss and Accuracy at Epoch  22 : 0.25 , 90.66 %
Training loss and Accuracy at Epoch  23 : 0.25 , 90.75 %
Training loss and Accuracy at Epoch  24 : 0.25 , 90.83 %
Training loss and Accuracy at Epoch  25 : 0.24 , 90.91 %
Training loss and Accuracy at Epoch  26 : 0.24 , 91.08 %
Training loss and Accuracy at Epoch  27 : 0.24 , 91.13 %
Training loss and Accuracy at Epoch  28 : 0.24 , 91.08 %
Training loss and Accuracy at Epoch  29 : 0.24 , 91.12 %
Training loss and Accuracy at Epoch  30 : 0.23 , 91.17 %
Training loss and Accuracy at Epoch  31 : 0.23 , 91.27 %
Training loss and Accuracy at Epoch  32 : 0.23 , 91.43 %
Training loss and Accuracy at Epoch  33 : 0.23 , 91.50 %
Training loss and Accuracy at Epoch  34 : 0.22 , 91.56 %
Training loss and Accuracy at Epoch  35 : 0.22 , 91.70 %

------------------------------------------------------------------------
Performance Evaluation
------------------------------------------------------------------------
Testing loss and Accuracy: 0.37 , 87.67 %
```

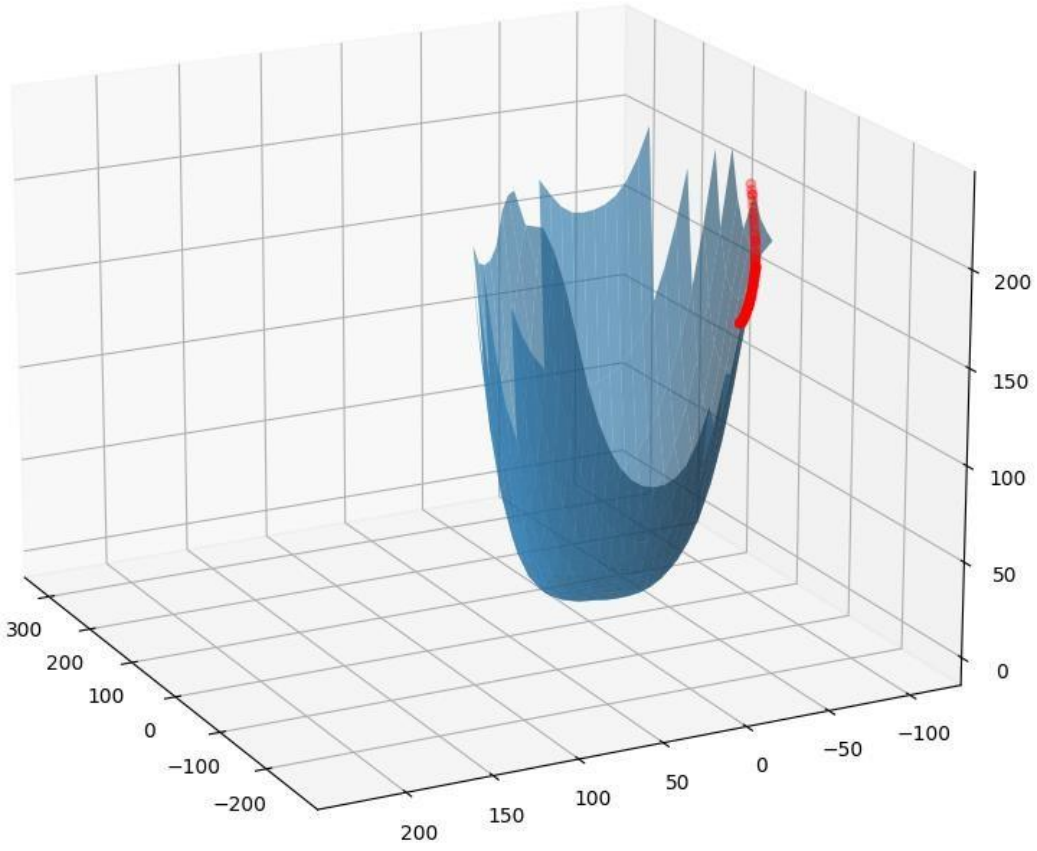
Problem 2: Mountains and Valleys

- Visualize the SGD trajectory of our network (with 3 hidden layers and 50 neurons in each hidden layer) when trained on Fashion MNIST dataset.

  The procedure for generating this visualization is as follows -

- We first performed Principal Component Analysis (PCA) and convert the original high dimensional weight-space to into a 2-D space (X-Y).
- Next, we find the min and max components in each of the X and Y directions.
- We then generate 100 samples between min and max components in both the directions. This gives us a weight space of 100*100 = 10000 components.
- After, taking the inverse PCA of all the 10000 points to generate a weight space of the original dimensions.
- We then calculate Loss on training data for all the 10000 points using these weights and then plot this Loss on the Z-axis corresponding to X and Y.
- By applying interpolation between these points, we get the surface plot as shown below.
- Now, we again perform the inverse PCA for all the 2-D weight space components we got during the training (the number of 2-D weight space components will be equal to num_of_epoch.)
- And again, calculated Loss on these weights, which is then shown as a “BLUE” scatter plot to represent how the weights and biases of our network vary in the weight space.
- The following two image represent the same plot form two different viewing angles.


200
150
100
50
0
200
150
100
50
0
−50
−100
300
200
100
0
−100
−200


300
200
100
0
−100
−200
200
150
100
50
0
−50
−100
200
150
100
50
0