Assignment 1

Question 1

Utilize various activation functions like sigmoid, tanh and critique the performance in each case.

Hyperparameter:

- Number of epochs: 5
- Batch size: 32
- Learning Rate: 0.2
- Hidden layer: 10

Vanishing Gradient for Different Activation Function:

- Sigmoid:
  For sigmoid as activation function gradient vanish in 4 hidden layer neural networks with.

```
Epoch: 1 -------------------------
Train Loss: 2.3031 Train Accuracy: 0.0099
Test Loss: 2.3028 Test Accuracy: 0.0100


Epoch: 2 -------------------------
Train Loss: 2.3030 Train Accuracy: 0.0098
Test Loss: 2.3028 Test Accuracy: 0.0100


Epoch: 3 -------------------------
Train Loss: 2.3030 Train Accuracy: 0.0099
Test Loss: 2.3027 Test Accuracy: 0.0100


Epoch: 4 -------------------------
Train Loss: 2.3029 Train Accuracy: 0.0101
Test Loss: 2.3029 Test Accuracy: 0.0100


Epoch: 5 -------------------------
Train Loss: 2.3030 Train Accuracy: 0.0099
Test Loss: 2.3027 Test Accuracy: 0.0100
```

Change in weight like near to zero:

```
Weight change:
tensor([[ 0.0012,  0.0012,  0.0012,  ...,  0.0004,  0.0004,  0.0004],
        [ 0.0003,  0.0003,  0.0002,  ...,  0.0011,  0.0011,  0.0012],
        [ 0.0005,  0.0005,  0.0005,  ..., -0.0005, -0.0005, -0.0005],
        ...,
        [ 0.0001,  0.0001,  0.0001,  ..., -0.0007, -0.0007, -0.0007],
        [ 0.0004,  0.0004,  0.0004,  ..., -0.0002, -0.0002, -0.0003],
        [ 0.0004,  0.0004,  0.0005,  ..., -0.0012, -0.0012, -0.0013]])
```

- Tanh:

  For Tanh as activation function gradient vanish in 14 hidden layer neural networks with

```
epoch: 1 ------------------------
Train Loss: 2.3047 Train Accuracy: 0.0099
Test Loss: 2.3060 Test Accuracy: 0.0100


epoch: 2 ------------------------
Train Loss: 2.3048 Train Accuracy: 0.0100
Test Loss: 2.3041 Test Accuracy: 0.0100


epoch: 3 ------------------------
Train Loss: 2.3044 Train Accuracy: 0.0100
Test Loss: 2.3038 Test Accuracy: 0.0100


epoch: 4 ------------------------
Train Loss: 2.3044 Train Accuracy: 0.0099
Test Loss: 2.3044 Test Accuracy: 0.0100


epoch: 5 ------------------------
Train Loss: 2.3043 Train Accuracy: 0.0098
Test Loss: 2.3040 Test Accuracy: 0.0100
```

Change in weight like near to zero:

```
Change in weight:
tensor([[ 3.1357e-04,  3.1517e-04,  3.1672e-04,  ..., -4.3675e-04,
         -4.1288e-04, -3.8977e-04],
        [-6.3543e-04, -6.6205e-04, -6.7993e-04,  ...,  3.6509e-04,
          3.5406e-04,  3.4595e-04],
        [-7.2557e-03, -7.2810e-03, -7.3266e-03,  ...,  2.4114e-03,
          2.1089e-03,  1.8113e-03],
        ...,
        [-9.6095e-04, -9.4376e-04, -9.4006e-04,  ..., -1.1091e-04,
```

```
 -1.5619e-04, -2.0514e-04],
[ 1.5911e-03,  1.6068e-03,  1.6230e-03,  ..., -2.7549e-06,
  3.8907e-05,  8.0751e-05],
[-6.1042e-03, -6.0816e-03, -6.1048e-03,  ...,  2.7978e-04,
  6.4387e-05, -1.5265e-04]])
```

Suggest and implement methods to overcome the above problem.

- Use ReLU as activation function because derivation of weight itself if weights greater than zero otherwise zero.
- Hyperparameter tuning
    - like learning rate (make it small but not too small)
    - Reduce network layer or reduce layer nodes

ReLU as activation function:

For ReLU as activation function gradient vanish in 8 hidden layer neural networks with

```
epoch: 1 ------------------------
Train Loss: 2.3029 Train Accuracy: 0.0100
Test Loss: 2.3026 Test Accuracy: 0.0100


epoch: 2 ------------------------
Train Loss: 2.3026 Train Accuracy: 0.0100
Test Loss: 2.3026 Test Accuracy: 0.0100


epoch: 3 ------------------------
Train Loss: 2.3026 Train Accuracy: 0.0100
Test Loss: 2.3026 Test Accuracy: 0.0100


epoch: 4 ------------------------
Train Loss: 2.3026 Train Accuracy: 0.0100
Test Loss: 2.3026 Test Accuracy: 0.0100


epoch: 5 ------------------------
Train Loss: 2.3026 Train Accuracy: 0.0100
Test Loss: 2.3026 Test Accuracy: 0.0100
```

Change in weight like near to zero:

```
Change in weight:
tensor([[ 2.7274e-05,  2.6988e-05,  2.9447e-05,  ..., -3.0675e-05,
         -3.1985e-05, -3.2416e-05],
        [-7.4186e-05, -7.2685e-05, -7.4183e-05,  ..., -1.0392e-05,
         -1.0166e-05, -1.1549e-05],
        [-5.9903e-06, -5.7435e-06, -6.2417e-06,  ..., -1.2015e-05,
         -1.0278e-05, -9.6764e-06],
        ...,
        [ 7.2141e-05,  7.3321e-05,  7.4940e-05,  ...,  1.4943e-05,
          1.4231e-05,  1.4236e-05],
        [ 2.5760e-05,  2.4946e-05,  2.4638e-05,  ...,  1.0150e-05,
          1.0584e-05,  1.0999e-05],
        [ 2.2046e-04,  2.2185e-04,  2.2175e-04,  ...,  3.3787e-05,
          3.4152e-05,  3.7011e-05]])
```

Question 2

Train Model without any Regularization

Hyperparameters:

- Hidden units: 13
- Hidden Layers: 3
- Learning Rate: 0.003
- Epochs: 25
- Batch size: 32

Result After 25 epochs:

```
Train Loss: 1.0215 Train Accuracy: 0.5371
Test Loss: 1.1277 Test Accuracy: 0.1837
```

Train Model with L1 Regularization

Hyperparameters:

- Hidden units: 13
- Hidden Layers: 3
- Learning Rate: 0.003
- Epochs: 25
- Batch size: 32
- Regularization lambda: 0.0018 (not too small and to big)

Result After 25 epochs:

```
Train Loss: 1.4341 Train Accuracy: 0.6543
Test Loss: 1.1389 Test Accuracy: 0.2012
```

Train Model with L2 Regularization

Hyperparameters:

- Hidden units: 13
- Hidden Layers: 3
- Learning Rate: 0.0033
- Epochs: 25
- Batch size: 32
- Regularization lambda: 0.0019 (not too small and to big)

Result After 25 epochs:

```
Train Loss: 1.0614 Train Accuracy: 0.8176
Test Loss: 1.0029 Test Accuracy: 0.2077
```

Train Model with Dropout

Hyperparameters:

- Hidden units: 13
- Hidden Layers: 3
- Learning Rate: 0.0033
- Epochs: 25
- Batch size: 32
- Dropout Prob: 0.25

Result After 25 epochs:

```
Train Loss: 1.3450 Train Accuracy: 0.5411
Test Loss: 1.2807 Test Accuracy: 0.2079
```

- Here In L2 regularization model perform overfitting because compared to other train accuracy is high while test accuracy is low.
- L1 regularization and dropout perform well with respect to the normal model with.
- It is true that all models perform overfitting but L2 regularization gives more overfitting results due to the lambda hyperparameter.

Gradient Checking

Hyperparameters:

- Hidden units: 13
- Hidden Layers: 3
- Learning Rate: 0.003
- Epochs: 25
- Batch size: 32

Difference: `0.9999999999998282`

For the correct gradient we need to make our difference less or equal than 1e-5 or 1e-7 but here we get 1e-1.

Hence, our gradient is not proper.

References:

Pytorch: https://pytorch.org/docs/stable/index.html
Neural Networks: Sir's Slides
Online resource
Vanishing Gradient: Problem
Solutions

L1 and L2 Regularization: medium
Dropout: medium
Gradient Check: YouTube
Towards Data Science