<u>Question 01</u>

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

1 FC MODEL

```
With only tanh activation in both convolutional layers:

[1,  2000] loss: 1.887
[1,  4000] loss: 1.631
[1,  6000] loss: 1.557
[1,  8000] loss: 1.476
[1, 10000] loss: 1.415
[1, 12000] loss: 1.417
[2,  2000] loss: 1.332
[2,  4000] loss: 1.333
[2,  6000] loss: 1.323
[2,  8000] loss: 1.274
[2, 10000] loss: 1.279
[2, 12000] loss: 1.276
Finished Training

Test Accuracy: 52 %

With only sigmoid activation in both convolutional layers:

[1,  2000] loss: 2.299
[1,  4000] loss: 2.080
[1,  6000] loss: 1.980
[1,  8000] loss: 1.969
[1, 10000] loss: 1.918
[1, 12000] loss: 1.904
[2,  2000] loss: 1.846
[2,  4000] loss: 1.821
[2,  6000] loss: 1.771
[2,  8000] loss: 1.736
[2, 10000] loss: 1.694
[2, 12000] loss: 1.660
Finished Training

Test Accuracy: 41 %

With sigmoid activation in first convolutional layer and tanh activation
in second convolutional layer:

[1,  2000] loss: 2.123
[1,  4000] loss: 1.911
```

```
[1,  6000] loss: 1.788
[1,  8000] loss: 1.686
[1, 10000] loss: 1.627
[1, 12000] loss: 1.567
[2,  2000] loss: 1.524
[2,  4000] loss: 1.478
[2,  6000] loss: 1.464
[2,  8000] loss: 1.425
[2, 10000] loss: 1.430
[2, 12000] loss: 1.430
Finished Training

Test Accuracy: 48 %
```

With tanh activation in first convolutional layer and sigmoid activation in second convolutional layer:

```
[1,  2000] loss: 2.166
[1,  4000] loss: 1.937
[1,  6000] loss: 1.801
[1,  8000] loss: 1.722
[1, 10000] loss: 1.632
[1, 12000] loss: 1.587
[2,  2000] loss: 1.540
[2,  4000] loss: 1.500
[2,  6000] loss: 1.466
[2,  8000] loss: 1.459
[2, 10000] loss: 1.414
[2, 12000] loss: 1.411
Finished Training

Test Accuracy: 49 %
```

With tanh activation in first convolutional layer and sigmoid activation in second convolutional layer and in 1 FC layer:

```
[1,  2000] loss: 2.188
[1,  4000] loss: 2.104
[1,  6000] loss: 2.084
[1,  8000] loss: 2.064
[1, 10000] loss: 2.036
[1, 12000] loss: 2.014
[2,  2000] loss: 1.988
[2,  4000] loss: 1.977
[2,  6000] loss: 1.970
[2,  8000] loss: 1.960
[2, 10000] loss: 1.955
```

```
[2, 12000] loss: 1.946
Finished Training

Test Accuracy: 43 %
```

3 FC MODEL

With only tanh activation in convolutional and 3 fully connected layer:

```
[1,  2000] loss: 1.989
[1,  4000] loss: 1.701
[1,  6000] loss: 1.615
[1,  8000] loss: 1.497
[1, 10000] loss: 1.493
[1, 12000] loss: 1.440
[2,  2000] loss: 1.365
[2,  4000] loss: 1.350
[2,  6000] loss: 1.342
[2,  8000] loss: 1.313
[2, 10000] loss: 1.315
[2, 12000] loss: 1.283
Finished Training

Test Accuracy = 55%
```

With only sigmoid activation in convolutional and 3 fully connected layer:

```
[1,  2000] loss: 2.316
[1,  4000] loss: 2.313
[1,  6000] loss: 2.311
[1,  8000] loss: 2.312
[1, 10000] loss: 2.311
[1, 12000] loss: 2.310
[2,  2000] loss: 2.310
[2,  4000] loss: 2.308
[2,  6000] loss: 2.308
[2,  8000] loss: 2.308
[2, 10000] loss: 2.308
[2, 12000] loss: 2.307
Finished Training

Test Accuracy = 10%
```

With sigmoid activation in convolutional layer and
tanh activation in 3 fully connected layer:

```
[1,  2000] loss: 2.309
```

```
[1,  4000] loss: 2.290
[1,  6000] loss: 2.116
[1,  8000] loss: 2.042
[1, 10000] loss: 1.980
[1, 12000] loss: 1.930
[2,  2000] loss: 1.873
[2,  4000] loss: 1.811
[2,  6000] loss: 1.754
[2,  8000] loss: 1.691
[2, 10000] loss: 1.646
[2, 12000] loss: 1.617
Finished Training

Test Accuracy: 42 %
```

2. Increase the depth of the given network by adding more Fully-Connected layers till the point you encounter the vanishing gradient problem. With the help of the results, mention how to identify it.

We encounter the vanishing gradient problem with 5 fully connected (FC) layers. The train loss changed very minutely with 4 FC but with 5 FC the network is unable to learn. Below is the train loss observed with the 5 FC layer model:

```
[1,  4000] loss: 2.306
[1,  6000] loss: 2.306
[1,  8000] loss: 2.306
[1, 10000] loss: 2.305
[1, 12000] loss: 2.305
[2,  2000] loss: 2.306
[2,  4000] loss: 2.306
[2,  6000] loss: 2.305
[2,  8000] loss: 2.306
[2, 10000] loss: 2.306
[2, 12000] loss: 2.305
Finished Training

Test Accuracy: 10 %
```

Identification - The vanishing gradient problem can be identified by observing the training performance of the network. If the network is unable to learn or if the training loss becomes stagnant, it may be a sign of the vanishing gradient problem. In some cases, it is also possible to observe that the activations of certain neurons in the network become very small or close to zero during training, which is another indication of the vanishing gradient problem.

3. Suggest and implement methods to overcome the above problem.

We can use an activation function that will not reduce gradients like the relu activation function.
Proper utilization of free parameters such as weights and biases can also curb this problem.
Using relu activation, the train loss and accuracy were observed as follows:

```
[1,  2000] loss: 2.306
[1,  4000] loss: 2.302
[1,  6000] loss: 2.299
[1,  8000] loss: 2.170
[1, 10000] loss: 1.901
[1, 12000] loss: 1.792
[2,  2000] loss: 1.681
[2,  4000] loss: 1.628
[2,  6000] loss: 1.558
[2,  8000] loss: 1.530
[2, 10000] loss: 1.508
[2, 12000] loss: 1.428
Finished Training

Test Accuracy: 48 %
```

Question 02

1. Implement L-1 and L-2 regularization from scratch and dropout using pytorch. Compare the performance of the above techniques and mention reasons.

In the code, ‘l1’ and ‘l2’ are the hyperparameters for the L1 and L2 regularization, respectively. The regularization method calculates the regularization loss by adding up the L1 or L2 norms of the weights for the fully connected layers (l1 and l2). The regularization loss is added to the cross-entropy loss in the training loop to form the total loss that is used to update the model's parameters.

```
Epoch 1    loss with L1 regularization: 2.944
Epoch 2    loss with L1 regularization: 2.925
Epoch 3    loss with L1 regularization: 2.934
Epoch 4    loss with L1 regularization: 2.928
Epoch 5    loss with L1 regularization: 2.934
Epoch 6    loss with L1 regularization: 2.925
Epoch 7    loss with L1 regularization: 2.923
Epoch 8    loss with L1 regularization: 2.930
Epoch 9    loss with L1 regularization: 2.925
Epoch 10    loss with L1 regularization: 2.932
Epoch 1    loss with L2 regularization: 2.306
Epoch 2    loss with L2 regularization: 2.303
Epoch 3    loss with L2 regularization: 2.303
Epoch 4    loss with L2 regularization: 2.303
Epoch 5    loss with L2 regularization: 2.303
Epoch 6    loss with L2 regularization: 2.303
Epoch 7    loss with L2 regularization: 2.303
Epoch 8    loss with L2 regularization: 2.303
Epoch 9    loss with L2 regularization: 2.303
Epoch 10    loss with L2 regularization: 2.303
```

References:

[Guide to build an image classifier using CNN with CIFAR-10 dataset | by Asif Hashmi | Medium](https://towardsdatascience.com/cifar-10-image-classification-in-tensorflow-5b501f7dc77c?gi=6397906cb4ca)
https://towardsdatascience.com/cifar-10-image-classification-in-tensorflow-5b501f7dc77c?gi=6397906cb4ca