Assignment-4

3 Neural Network from scratch

I have implemented all the required layers from the scratch having some batch sizes.
I have written all the classes required for Maxpool2D, Conv2D, ReLU and the Linear layers.
In all the layers there, the forward and the backward layers are implemented.

Then, I have called all the layers with the required kernel sizes and the attributes. The cross entropy loss function is then implemented and the required functions of backward and forward layers are implemented here. During backpropagation, the gradients of the filter weights, bias terms and input tensor are computed using the chain rule and passed back to the previous layer.

Now, The adam optimiser class is implemented and then the corresponding gradients and the functions are implemented.

Now, a definition train is implemented inorder to train for the epochs given and then get the required results.
Now, giving the number of epochs=20, batch_size=32, learning rate=0.001, the data is trained on it.

4 Implementing a Pytorch based solution

4.1 Hyperparameter Tuning

4.1.1 Learning rate (LR)

I used Cross entropy as the loss function.

a) Using Adam optimiser, having batch_size=4 and number of epochs = 20,

i) With learning rate 0.001,

Validation accuracy of the test data set is ***62%***

The class wise accuracy is as follows:

```
Accuracy of plane : 71 %
Accuracy of   car : 77 %
Accuracy of  bird : 48 %
Accuracy of   cat : 47 %
Accuracy of  deer : 46 %
Accuracy of   dog : 37 %
Accuracy of  frog : 79 %
Accuracy of horse : 70 %
Accuracy of  ship : 71 %
Accuracy of truck : 77 %
```

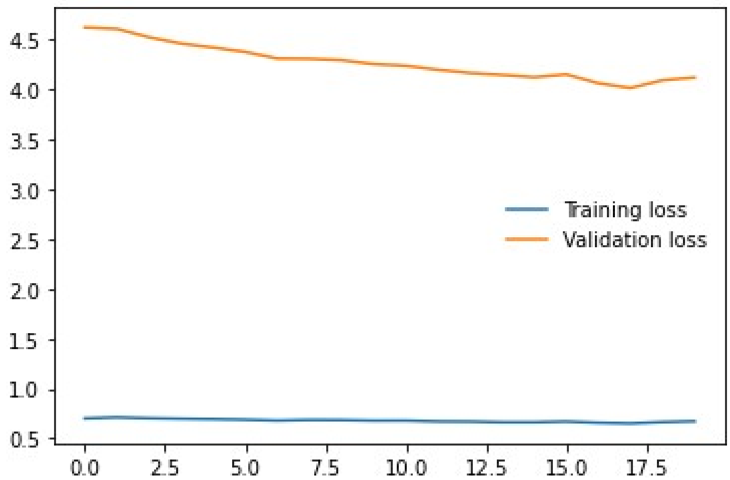
The training losses and validation losses are as given below :

Training loss :
[0.7021797132729439, 0.7130721498129249, 0.7040048102652751, 0.6989032543499241, 0.6952100029715511, 0.6896819250660156, 0.6808936719252777, 0.6870099316218173, 0.6856227309522932, 0.6801073056965156, 0.6805630419980956, 0.6718175743105929, 0.6704566083847012, 0.663705007885361, 0.6637892820416592, 0.6707398858620511, 0.6585974822867591, 0.6521715321659265, 0.665377841314218, 0.6736910927711858]

Validation loss :
[4.624181667895093, 4.609577525242471, 4.52562960401443, 4.46229753572393, 4.422490369334811, 4.378273181234462, 4.311704244067265, 4.308941039438013, 4.2950591382566445, 4.255972315760852, 4.238962346181151, 4.198524322929357, 4.166000254106791, 4.146836813454595, 4.123799853320101, 4.15182847248181, 4.063216434370691, 4.015498303175236, 4.093231013135104, 4.121051975639793]

The values of training loss vary from 0.7021 to 0.6739
The values of validation loss vary from 4.6241 to 4.121

The graph of training and validation loss vs epoch is as follows :



ii) *With learning rate 0.01,*

*Validation accuracy of the test data set is **10%***

The class wise accuracy is as follows:

```
Accuracy of plane :  0 %
Accuracy of   car :  0 %
Accuracy of  bird :  0 %
Accuracy of   cat : 100 %
Accuracy of  deer :  0 %
Accuracy of   dog :  0 %
Accuracy of  frog :  0 %
Accuracy of horse :  0 %
Accuracy of  ship :  0 %
Accuracy of truck :  0 %
```

The training losses and validation losses are as given below :

Training loss :

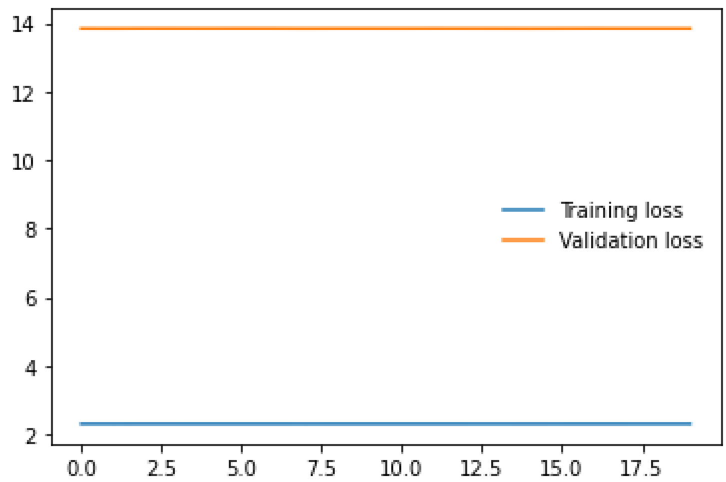
[2.306441500968933, 2.3061785318565367, 2.306817191734314, 2.3066503198051453, 2.3065257640075685, 2.3066223890686035, 2.306616963996887, 2.3061314706611635, 2.3065399529457093, 2.306496998271942, 2.3063169831848143, 2.30663291282653 8, 2.30643185338974, 2.306315396156311, 2.306009289417267, 2.3063694169425966, 2.306623875179291, 2.306436901435852, 2.306375387687683, 2.3063205846595762]

Validation loss :

[13.838887833404542, 13.83750821237564, 13.840583702278137, 13.839933949661255, 13.839286315345765, 13.839946049308777, 13.839609106826781, 13.837278133201599, 13.839384312152863, 13.83912808084488, 13.83853514213562, 13.839672940731049, 13.838850291156769, 13.83805277299881, 13.836480431938172, 13.83831839237213, 13.839690273189545, 13.83868899230957, 13.838562157535552, 13.838358086681366]

The values of training loss vary from 2.30644 to 2.30632
The values of validation loss vary from 13.8388878, 13.83835

The graph of training and validation loss vs epoch is as follows :



Observation : *When the learning rate increased to 0.01, the training losses for all the epochs is almost all the same and the validation accuracy and the class wise accuracies dropped drastically as compared to LR 0.001. The accuracies are too poor.*

iii) With learning rate 0.0001,

*Validation accuracy of the test data set is **70%***

The class wise accuracy is as follows:

```
Accuracy of plane : 75 %
Accuracy of   car : 76 %
Accuracy of  bird : 57 %
Accuracy of   cat : 56 %
Accuracy of  deer : 60 %
Accuracy of   dog : 63 %
Accuracy of  frog : 75 %
Accuracy of horse : 84 %
Accuracy of  ship : 82 %
Accuracy of truck : 73 %
```

The training losses and validation losses are as given below :

Training loss :

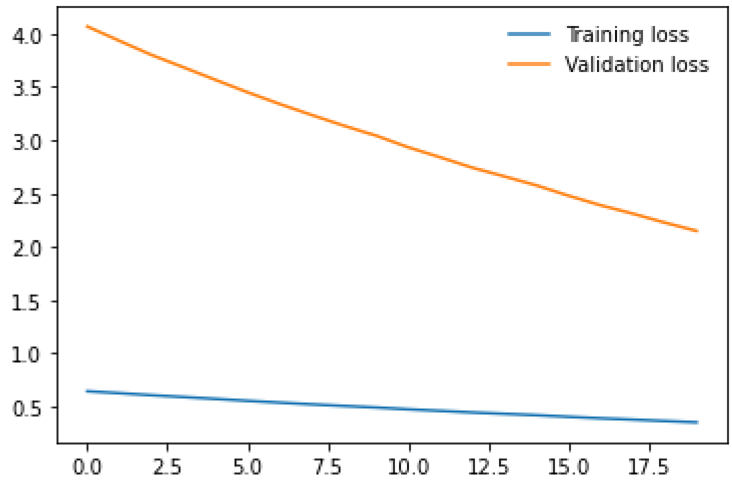
[0.6436985421931185, 0.6259312926884368, 0.6066817069961457, 0.589546145662954, 0.5719271258290625, 0.5537207170562353, 0.53695751557983, 0.5207757950703753, 0.50577585588546, 0.49111041609419975, 0.4740028419686481, 0.45941796199019763, 0.444307783245162, 0.43154777751097106, 0.41884279367562965, 0.40377681570697166, 0.38947954616199626, 0.37730551287195907, 0.364080267945372, 0.351560325909339]

Validation loss :

[4.068944336105231, 3.9344741336051374, 3.8018432223740732, 3.6852970351319296, 3.5671877560424385, 3.4504774786477443, 3.340686915445118, 3.2370629168249434, 3.1369609188026515, 3.0425847198600184, 2.933311160479253, 2.8389170496694627, 2.7416142998762747, 2.6594727559973776, 2.5751808188144873, 2.478050341088537, 2.3883680501357842, 2.310081344505692, 2.225591408233659, 2.1468537847238562]

The values of training loss vary from 0.64369 to 0.3515
The values of validation loss vary from 4.068944 to 2.146853

The graph of training and validation loss vs epoch is as follows :



***Observation :** When the learning rate decreased from 0.001 to 0.0001,all the accuracies have been increased. The class wise accuracies, the training and validation accuracies have also been increased. The training and validation losses have decreased more as compared to that of LR 0.001.*

b) Using SGD optimiser, having batch_size=4 and number of epochs = 20,

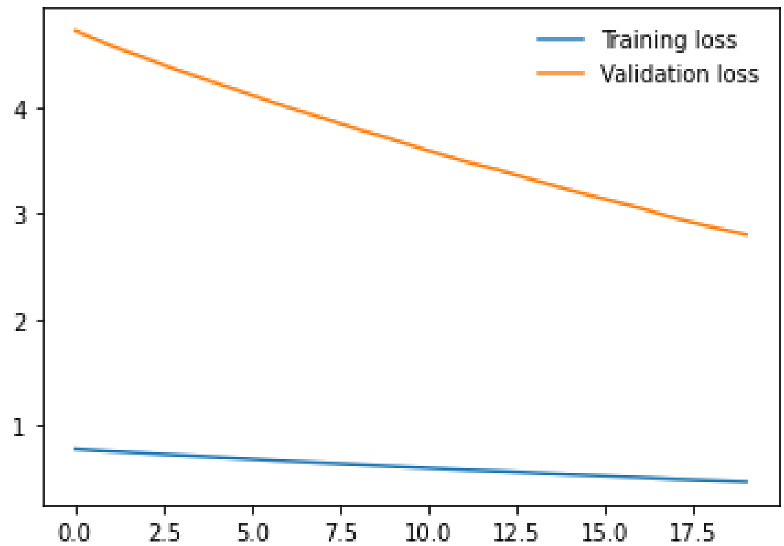
i) With learning rate 0.001,

*Validation accuracy of the test data set is **66%.***

<u>The class wise accuracy is as follows:</u>

```
Accuracy of plane : 62 %
Accuracy of   car : 73 %
Accuracy of  bird : 60 %
Accuracy of   cat : 51 %
Accuracy of  deer : 53 %
Accuracy of   dog : 52 %
Accuracy of  frog : 77 %
Accuracy of horse : 71 %
Accuracy of  ship : 85 %
Accuracy of truck : 72 %
```

<u>The graph of training and validation loss vs epoch is as follows :</u>



The values of the training losses varied from 0.89 to 0.56.
The values of the validation losses varied from 4.65 to 2.91.

ii) *With learning rate 0.01,*

Validation accuracy of the test data set is ***67%***

The class wise accuracy is as follows:

```
Accuracy of plane : 64 %
Accuracy of   car : 78 %
Accuracy of  bird : 60 %
Accuracy of   cat : 45 %
Accuracy of  deer : 62 %
Accuracy of   dog : 62 %
Accuracy of  frog : 79 %
Accuracy of horse : 77 %
Accuracy of  ship : 67 %
Accuracy of truck : 74 %
```

The training losses and validation losses are as given below :

Training loss :

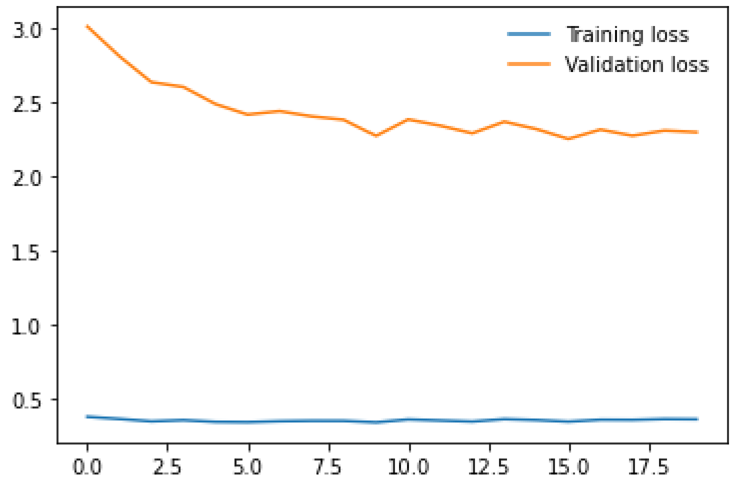
[0.38094632618626184, 0.36637299435157183, 0.350340847460945, 0.3580952224653074, 0.3466692626050148, 0.3448666298882504, 0.35113171788727005, 0.35387077036254244, 0.3534915931364878, 0.34352028204117174, 0.36296613186534, 0.3558787563815986, 0.34892887532868966, 0.3651761444209201, 0.35883759206578253, 0.34837506629374476, 0.36099889613465025, 0.36001269764163857, 0.3656468220197059, 0.36435632465051493]

Validation loss :

[3.0079189348000837, 2.8076793055678646, 2.632713185205172, 2.600347489462743, 2.485130978221591, 2.414257328102437, 2.437435589309973, 2.4022454705612057, 2.379145282334886, 2.2709028166024843, 2.381583618344252, 2.3395816164709755, 2.288085103419084, 2.366334315645527, 2.31658928672568, 2.2502640504779445, 2.3130240454313613, 2.2718271239241727, 2.3065293205194775, 2.2955220386175004]

The values of training loss vary from 0.38094 to 0.36435
The values of validation loss vary from 3.00791 to 2.29552

`The graph of training and validation loss vs epoch is as follows :`



Observation: *The values of the accuracies increased here as the learning rate increased from 0.001 to 0.01 and the validation loss decrement is decreased. The class wise accuracies have also been increased. This is in contrast as compared to the Adam optimiser where all the accuracies decreased very drastically.*

iii) With learning rate 0.0001,

*Validation accuracy of the test data set is **32%***

The class wise accuracy is as follows:

```
Accuracy of plane : 45 %
Accuracy of   car : 51 %
Accuracy of  bird :  1 %
Accuracy of   cat :  2 %
Accuracy of  deer :  6 %
Accuracy of   dog : 36 %
Accuracy of  frog : 70 %
Accuracy of horse : 42 %
Accuracy of  ship : 24 %
Accuracy of truck : 44 %
```

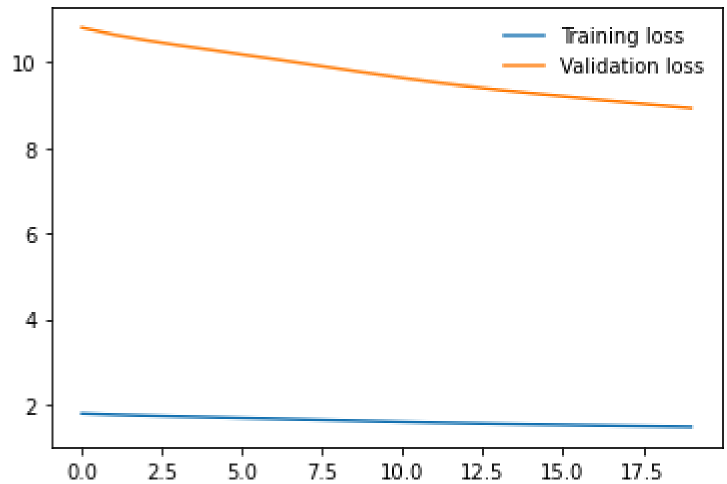
The training losses and validation losses are as given below :

Training loss :
[1.8023297538614274, 1.7750132148742677, 1.75340335231781, 1.7338893839645386, 1.7163384019231795, 1.6975744302320481, 1.6799356524896623, 1.6612767442798615, 1.6426113918685914, 1.6238345438861848, 1.6057543339681626, 1.5893942174744606, 1.574595586462021, 1.5601899522995948, 1.5474601112651825, 1.5360127938580512, 1.5237209501361846, 1.5121198535394669, 1.501150409772396, 1.4897561836862565]

Validation loss :
[10.799070443558692, 10.637779388332367, 10.508806329488754, 10.392715673851967, 10.286702152049541, 10.17459487632513, 10.067753491449356, 9.955320865166188, 9.842579279112815, 9.729872756159306, 9.621759905958175, 9.523730105233193, 9.434825832283497, 9.349099523639678, 9.272549288368225, 9.20242927417755, 9.12832522906065, 9.058409850513936, 8.991853768241405, 8.924035250294208]

The values of training loss vary from 1.8023 to 1.4897
The values of validation loss vary from 10.79907 to 8.9240

The graph of training and validation loss vs epoch is as follows :

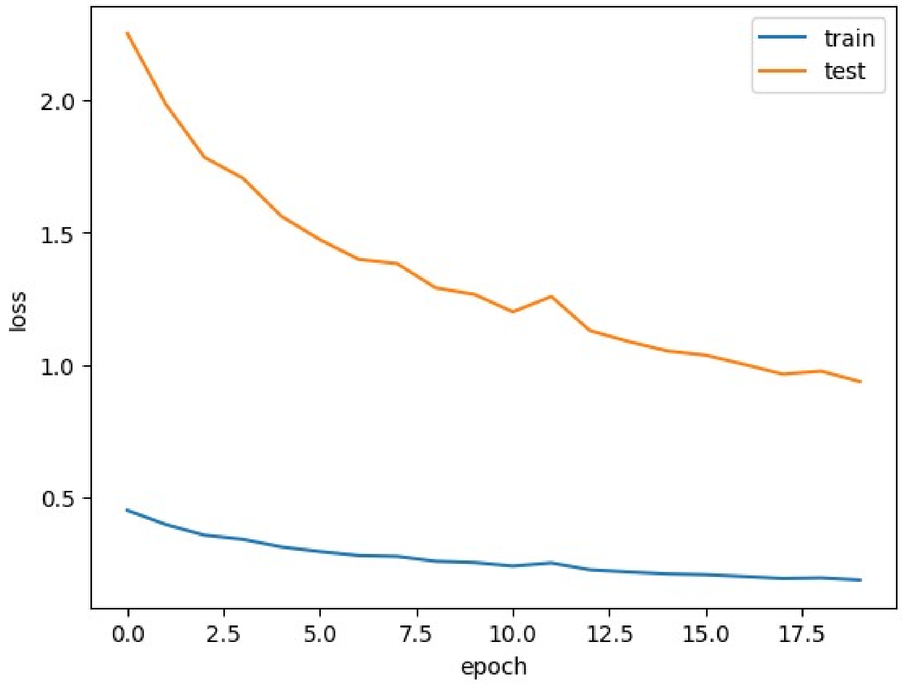


Observation : *The values of the validation and training losses increased as compared to that of LR 0.001. The validation accuracy value dropped to 32%. The class wise accuracies also have been decreased.*
This occurred completely opposite as compared to the Adam optimiser as compared to the LR used there.

4.1.2 Variation in LR

Here, I have used LinearLR here.
The line which I have added here extra is simply
```
scheduler = LinearLR(optimizer, start_factor=0.4, total_iters=10
```
The LR used here is 0.001. It starts from 0.0004 and has 10 iterations in total.
The number of epochs used here is 20. The batch size is 4.

The graph of training and validation loss curves is as follows :



The observation is that using a learning rate scheduler to vary the learning rate over epochs can lead to better training results compared to using a fixed learning rate. When we used the varying LR, the observation is that the values of the training losses and validation losses are decreasing more as compared to that of fixed LR.
The values of the validation losses and training losses are less.
I have seen a faster convergence and lower validation loss compared to using a fixed learning rate.

Also, what I have observed in class wise accuracies is that it became biased towards one of the classes.

4.1.3 Number of Training Epochs

I used Adam optimiser varying the epoch values from 20, 40, 60, 80.
The Learning rate is fixed to 0.001. *The batch size is also kept to 4.*

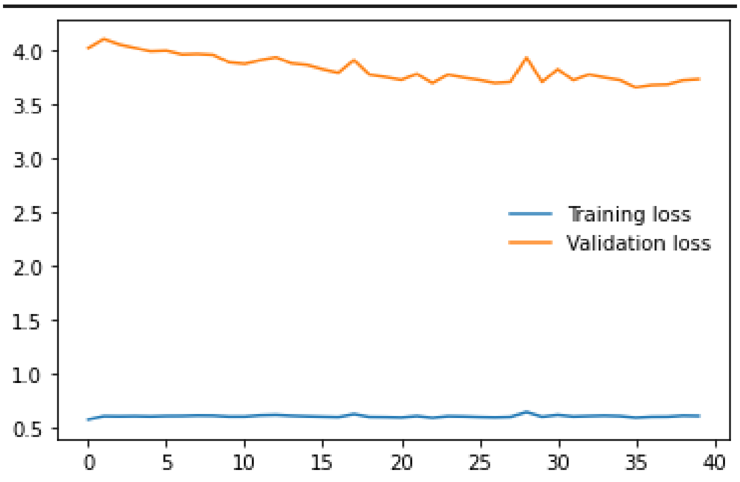
When we have the number of epochs to 20 with Adam optimiser, we have seen the values as they were above.
When the number of epochs is increased to 40,
The training accuracy is ***82%*** *and the validation accuracy is* ***64%.***

The class wise accuracy is as follows:

```
Accuracy of plane : 68 %
Accuracy of   car : 79 %
Accuracy of  bird : 50 %
Accuracy of   cat : 42 %
Accuracy of  deer : 54 %
Accuracy of   dog : 57 %
Accuracy of  frog : 72 %
Accuracy of horse : 67 %
Accuracy of  ship : 79 %
Accuracy of truck : 71 %
```

The values of training loss vary from 0.5687 to 0.6030105
The values of validation loss vary from 4.0205 to 3.7306

The graph of training and validation loss vs epoch is as follows :



Observation as compared to 20 epochs: *The training accuracy increased a bit and validation accuracy decreased a little. The validation losses and training losses decreased a bit as compared to the 20 epochs. Here what I feel is that the accuracies were good enough that the model is not overfiiting the data. That's what we can know from the graphs.*

When the number of epochs is increased to 60,

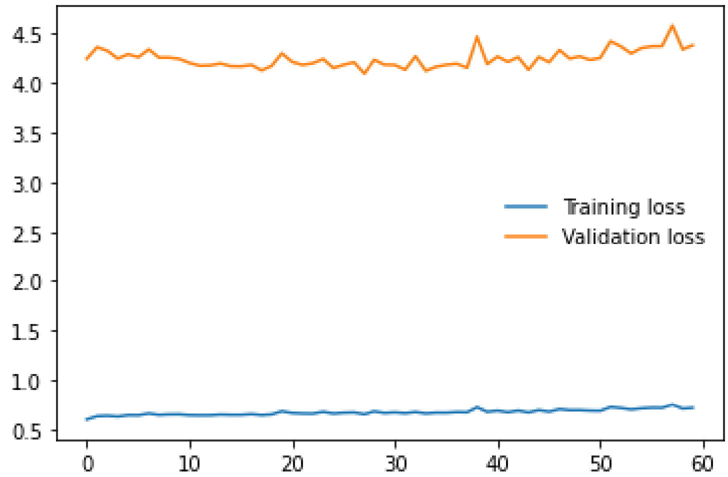
The training accuracy is ***81%*** *and the validation accuracy is* ***62%.***

<u>*The class wise accuracy is as follows:*</u>

```
Accuracy of plane : 62 %
Accuracy of   car : 72 %
Accuracy of  bird : 38 %
Accuracy of   cat : 53 %
Accuracy of  deer : 51 %
Accuracy of   dog : 53 %
Accuracy of  frog : 75 %
Accuracy of horse : 74 %
Accuracy of  ship : 69 %
Accuracy of truck : 78 %
```

The values of training loss vary from 0.6011 to 0.71955
The values of validation loss vary from 4.24805 to 4.28313

<u>*The graph of training and validation loss vs epoch is as follows :*</u>



Observation as compared to 40 epochs: *The training accuracy and validation accuracy decreased a little. The validation losses and training losses are fluctuating and increasing. What I am feeling is that there is a slight chance of overfitting the data as the accuracies are decreasing.*

When the number of epochs is increased to 80,

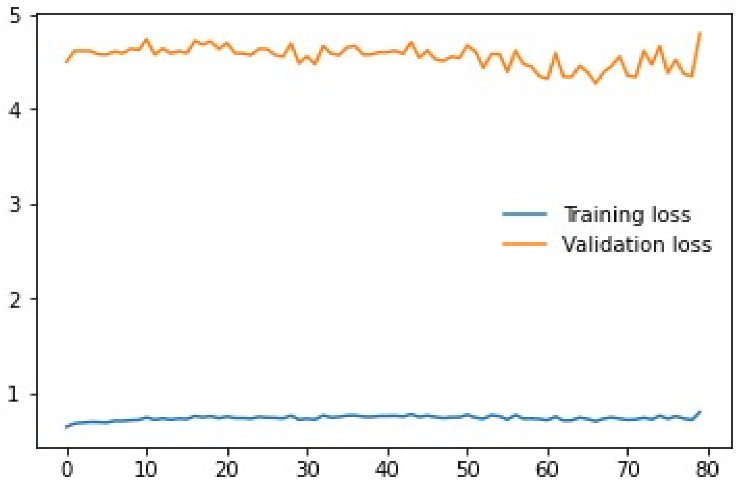
The training accuracy is ***80%*** *and the validation accuracy is* ***60%.***

<u>*The class wise accuracy is as follows:*</u>

```
Accuracy of plane : 65 %
Accuracy of   car : 81 %
Accuracy of  bird : 45 %
Accuracy of   cat : 43 %
Accuracy of  deer : 47 %
Accuracy of   dog : 49 %
Accuracy of  frog : 74 %
Accuracy of horse : 60 %
Accuracy of  ship : 72 %
Accuracy of truck : 64 %
```

The values of training loss vary from 0.63929 to 0.79433
The values of validation loss vary from 4.49886 to 4.79689

<u>*The graph of training and validation loss vs epoch is as follows :*</u>



Observation as compared to 60 epochs: *The training accuracy and validation accuracy decreased a little. The validation losses and training losses are fluctuating and increasing a lot here. This is a clear indication of overfitting the data as the accuracies are decreasing and the values of the losses are too fluctuating which means these are the signs of overfitting the data.*

Final Analysis : *Increasing the number of training epochs does not always help, as there is a risk of overfitting the model to the training data. Overfitting occurs when a model becomes too complex and starts to memorize the training data instead of learning the underlying patterns that can generalize to new, unseen data.*

To decide on the appropriate number of epochs, it is important to monitor the loss and accuracy on both the training and validation sets. Typically, as the number of epochs increases, the training loss decreases while the validation loss initially decreases and then starts to increase due to overfitting.

4.1.4 Batch Size

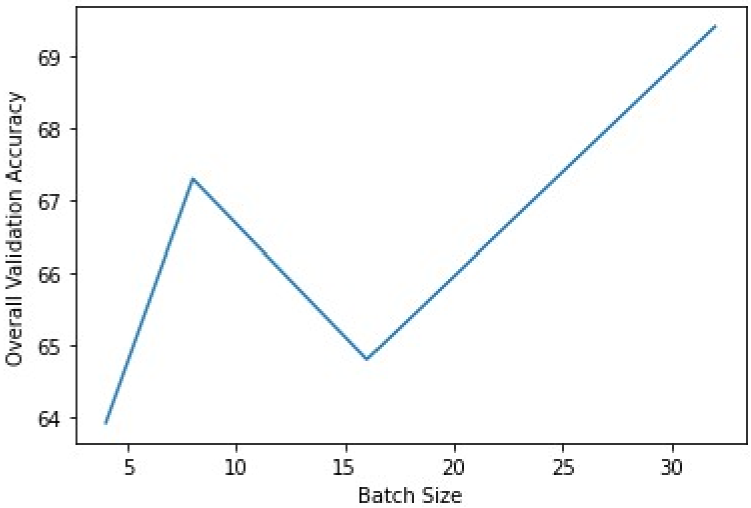
Here, the batch size is varied from {4, 8, 16, 32}. Here, we have learning rate 0.001 and number of epochs as 20.

Using Adam optimiser,

```
Training with batch size 4...
Files already downloaded and verified
Files already downloaded and verified
Test Accuracy: 10.0%
Training with batch size 8...
Files already downloaded and verified
Files already downloaded and verified
Test Accuracy: 65.98%
Training with batch size 16...
Files already downloaded and verified
Files already downloaded and verified
Test Accuracy: 66.12%
Training with batch size 32...
Files already downloaded and verified
Files already downloaded and verified
Test Accuracy: 66.83%
```

The test accuracies increased with the batch sizes is what can be observed from the above.

The graph of the validation accuracies for 20 epochs vs the batch size is as follows :

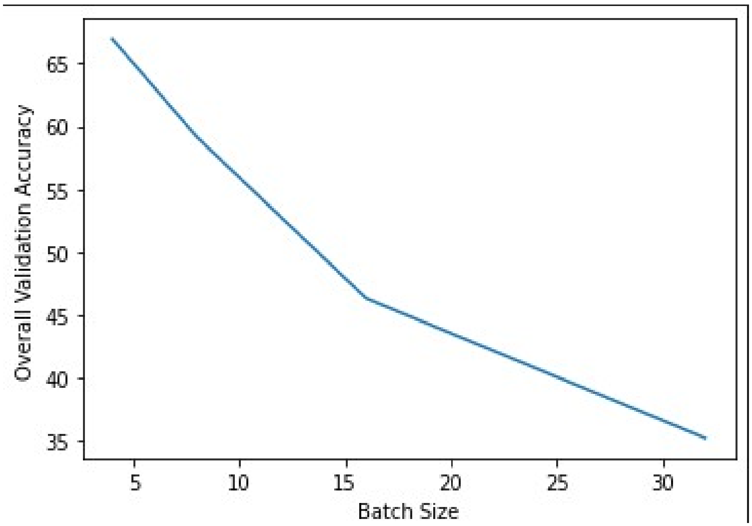


Using SGD optimiser,

```
Training with batch size 4...
Files already downloaded and verified
Files already downloaded and verified
Test Accuracy: 66.78%
Training with batch size 8...
Files already downloaded and verified
Files already downloaded and verified
Test Accuracy: 57.67%
Training with batch size 16...
Files already downloaded and verified
Files already downloaded and verified
Test Accuracy: 46.4%
Training with batch size 32...
Files already downloaded and verified
Files already downloaded and verified
Test Accuracy: 38.65%
```

The test accuracies decreased with the increase batch sizes is what can be observed from the above.

The graph of the validation accuracies for 20 epochs vs the batch size is as follows :



Observations :

As far as what we have observed till now, the SGD optimiser is completely opposite as compared to Adam optimiser. In the Adam optimiser, the validation accuracies are fluctuating and in SGD, the values are decreasing.

4.2 Effect of Loss Function

When there is change in loss function to KL divergence, there won't be much change.

The cross entropy loss measures the difference between the predicted probability distribution and the true probability distribution of the classes. It is given by:

Cross Entropy Loss = $-\sum(y*\log(p))$

where y is the true probability distribution and p is the predicted probability distribution.

KL divergence, on the other hand, measures the divergence between two probability distributions, typically between the predicted and true probability distributions. It is given by:

KL Divergence Loss = $\sum(y*\log(y/p))$

where y is the true probability distribution and p is the predicted probability distribution.

Since, this is the case, there won't be much change in the training and validation losses as well as the accuracies.

4.3 Effect of data augmentation

With the SGD optimiser having learning rate 0.001, batch size=4, number of epochs=20,

Applying this, i.e., having data augmentation,

```
transform = transforms.Compose([
    transforms.RandomCrop(size=32, padding=4),
    transforms.RandomHorizontalFlip(),
    transforms.ToTensor(),
    transforms.Normalize((0.5, 0.5, 0.5), (0.5, 0.5, 0.5))
])
```

The training accuracy and validation accuracy are ***68% and 67%*** *respectively.*
<u>The class wise accuracies are as follows :</u>

```
Accuracy of plane : 71 %
Accuracy of   car : 70 %
Accuracy of  bird : 62 %
Accuracy of   cat : 53 %
Accuracy of  deer : 65 %
Accuracy of   dog : 42 %
Accuracy of  frog : 77 %
Accuracy of horse : 72 %
Accuracy of  ship : 77 %
Accuracy of truck : 87 %
```

Without data augmentation,
The training accuracy is ***78%*** *and validation accuracy is* ***60%.***
<u>The class wise accuracies are as follows :</u>

```
Accuracy of plane : 62 %
Accuracy of   car : 73 %
Accuracy of  bird : 60 %
Accuracy of   cat : 51 %
Accuracy of  deer : 53 %
Accuracy of   dog : 52 %
Accuracy of  frog : 77 %
Accuracy of horse : 71 %
Accuracy of  ship : 85 %
Accuracy of truck : 72 %
```

Observations :
Yes, the training accuracy is increased and the validation accuracy is decreased when the data augmentation is turned off. That is it is trying to memorise the data and in which results in the decrease in the validation accuracy and the class wise accuracies. This is the clear indication of overfitting when the data augmentation is turned off.
When data augmentation is turned off, the model is trained on the original dataset without any transformations. This is leading to overfitting and lower generalization performance on the validation set, as the model is not exposed to different variations of the same image during training.

As a result, we observe a decrease in both the class-wise and overall validation accuracy when compared to training with data augmentations. The training accuracy may be higher, as the model is memorizing the training data, but this is not a good indicator of generalization performance.

5 Improving the CNN model

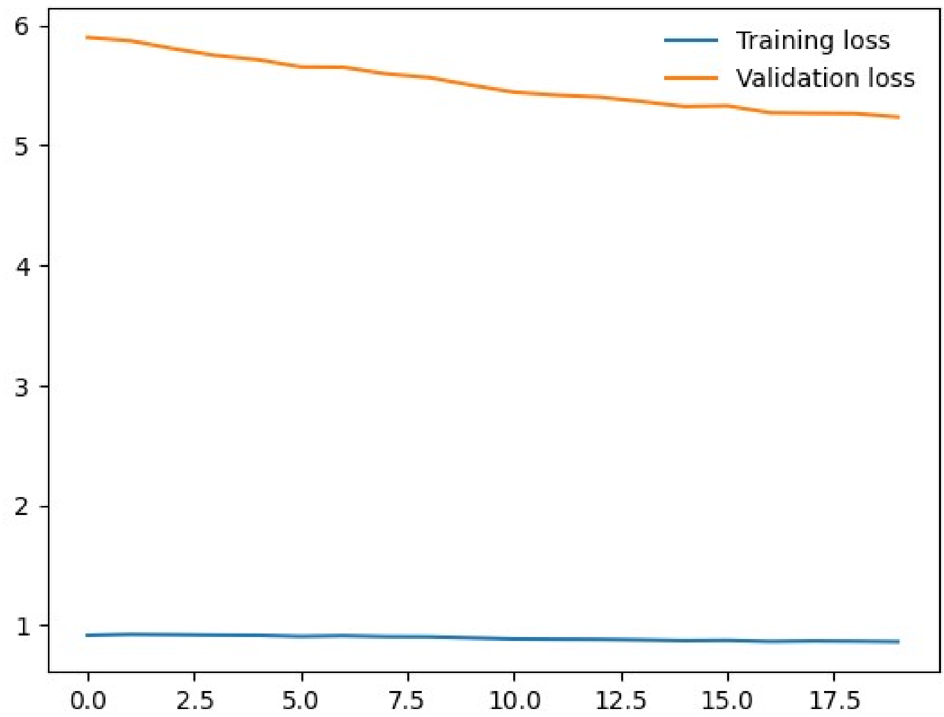
I had experimented with removing the relu application on the layers before.
The batch_size=4, epochs=20, Adam optimiser.

Then the validation accuracy reduced to 56%.

The class wise accuracies are as follows:

```
Accuracy of plane : 67 %
Accuracy of   car : 80 %
Accuracy of  bird : 41 %
Accuracy of   cat : 33 %
Accuracy of  deer : 45 %
Accuracy of   dog : 47 %
Accuracy of  frog : 55 %
Accuracy of horse : 70 %
Accuracy of  ship : 69 %
Accuracy of truck : 54 %
```

The graphs of validation losses and training losses are as follows:



The accuracy decreased when we removed relu function and the validation losses increased which tells the accuracies got reduced. The class wise accuracies decreased as well. This is as compared to the previous one.

Now, ***I am experimenting by applying relu to all the layers*** *which gives me improved accuracy for the validation and also the decrement in the training and validation losses. It also happened that the accuracy got improved in the classification as well.*

The training accuracy is 79% and the validation accuracy is 71%.

The class wise accuracies are as follows:

```
Accuracy of plane : 81 %
Accuracy of   car : 84 %
Accuracy of  bird : 51 %
Accuracy of   cat : 55 %
Accuracy of  deer : 72 %
Accuracy of   dog : 54 %
Accuracy of  frog : 78 %
Accuracy of horse : 81 %
Accuracy of  ship : 72 %
Accuracy of truck : 80 %
```

Justification: *The* model learns the complex and non-linear relationships between input features and output targets. This means that the model doesn't overfit the data and validates the test data set more correctly. This makes the model more efficient to compute and reducing overfitting.
So, overall the performance of the CNN got increased.