Deep Learning
Fall 2024
Homework 2
Name: Anna Kompan (安娜)
ID: 313540009

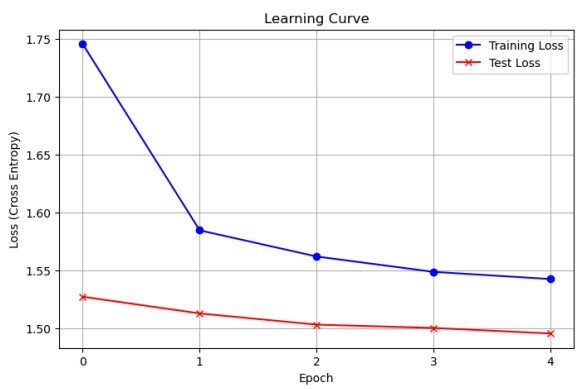
1. MNIST dataset

(1-1) Network architecture:

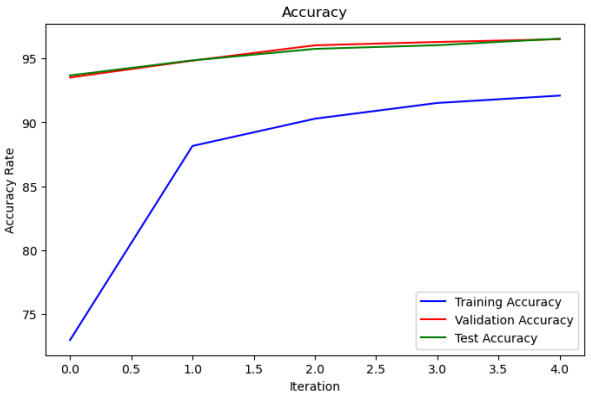
conv1 layer: Extract 10 feature maps from input image.
max_pool2d: Reduce image dimensions, and extracts sharpest features.
conv2 layer: Extract 20 feature maps and apply dropout to prevent overfitting.
fc1 layer: Reduce features to 50.
output layer: Outputs 10 probabilities for 10 handwritten digits
train_size = 55000
val_size = 5000
Hyperparameters:
batch_size = 100
kernel_size = 5
learning_rate = 0.001
epochs = 5
stride(default) = 1 (for convolutional layers)
stride = 2 (for max-pooling layers)

```
Train Epoch 0: Training loss: 1.7453, Training Accuracy: 72.95%
Validation Accuracy: 93.5200
Test set: Average Loss: 1.5275, Test Accuracy: 9368/10000 (93.68%)
Train Epoch 1: Training loss: 1.5847, Training Accuracy: 88.16%
Validation Accuracy: 94.8400
Test set: Average Loss: 1.5130, Test Accuracy: 9486/10000 (94.86%)
Train Epoch 2: Training loss: 1.5621, Training Accuracy: 90.29%
Validation Accuracy: 96.0400
Test set: Average Loss: 1.5034, Test Accuracy: 9576/10000 (95.76%)
Train Epoch 3: Training loss: 1.5489, Training Accuracy: 91.52%
Validation Accuracy: 96.3000
Test set: Average Loss: 1.5005, Test Accuracy: 9605/10000 (96.05%)
Train Epoch 4: Training loss: 1.5427, Training Accuracy: 92.10%
Validation Accuracy: 96.5200
Test set: Average Loss: 1.4957, Test Accuracy: 9656/10000 (96.56%)
```

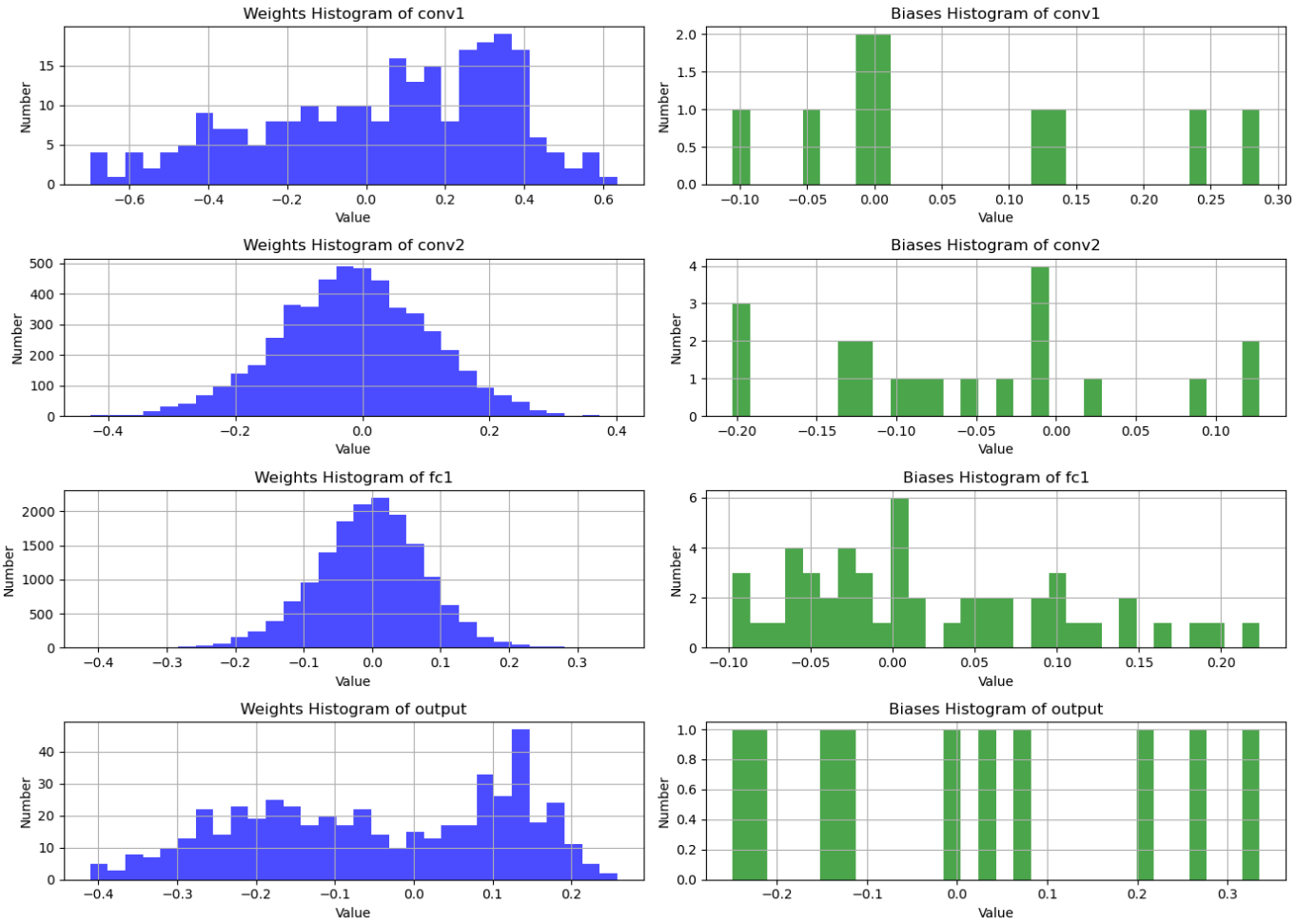
Learning Curve



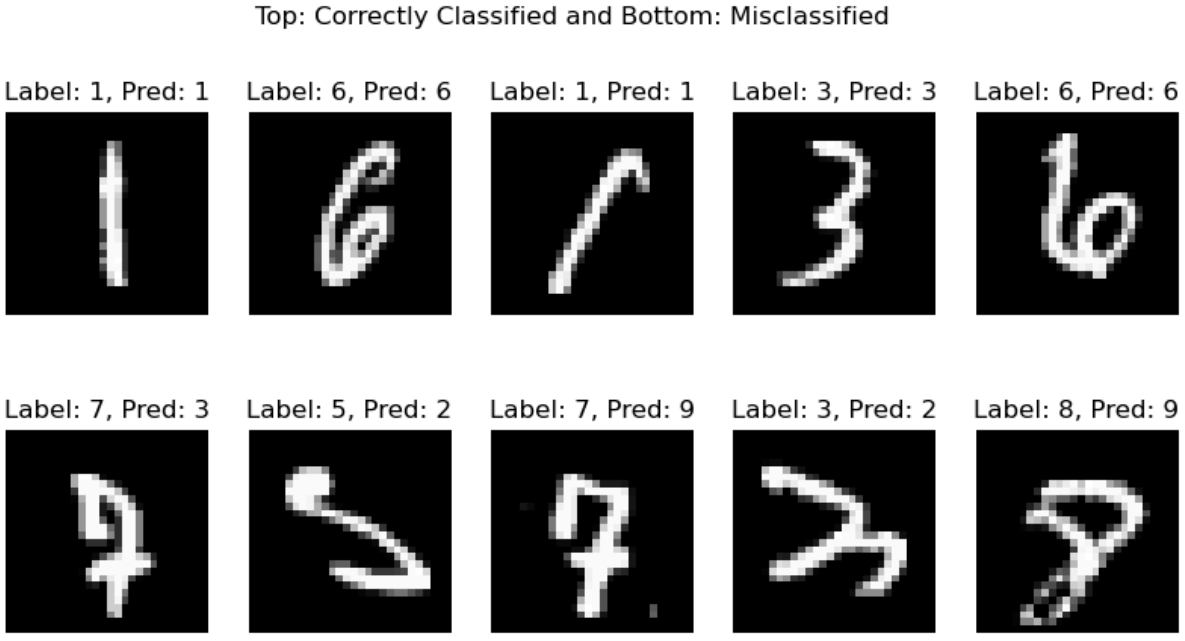
Accuracy of training and test sets



Distribution of weights and biases (Histograms of layers)



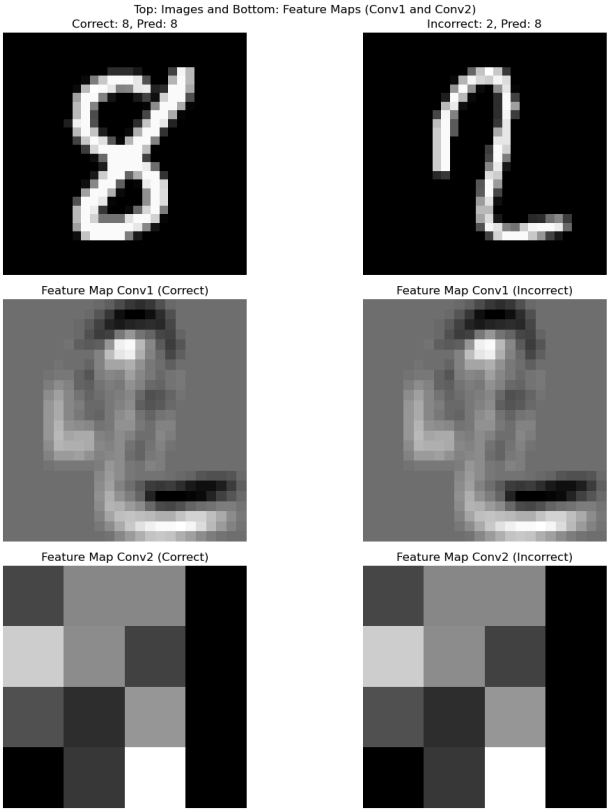
(1-2) Example of correctly classified and miss-classified images



We have a grid of 5 correctly and 5 misclassified images. In the list of misclassified images we can see digits that have shapes that could be mistakenly

classified as other digits. For instance, we have a number “3” that was predicted as “2”, the reason behind this is that the handwritten type of “3” has certain resembling features with 2, lower semi-circle is smaller than upper one, which led to misclassification. It means that our model can classify handwritten digits that have distinct features, but struggle to predict correctly those digits that have less distinct or ambiguous handwritten style.

(1-3) Feature maps from different convolutional layers



Feature map becomes more abstract as we move deeper through layers. In Conv1, feature maps capture edges, textures of digits. In Conv2, we go deeper and feature maps that focus on deeper, more complex patterns, capturing loops, curves of handwritten digits, and highlighting important parts that would help in prediction. The image in Conv2 has less resemblance with the original image as we go deeper into details.

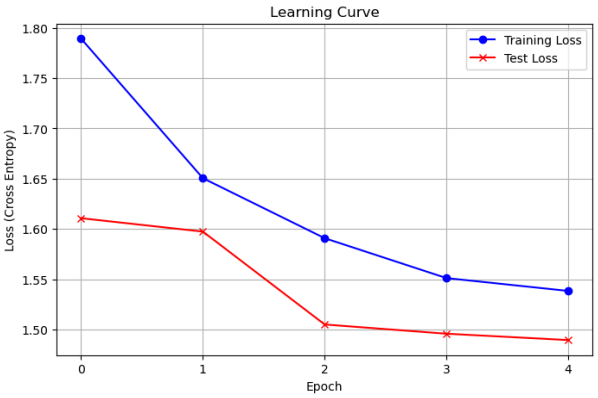
(1-4) Added L2 regularisation to the CNN

I’ve added weight_decay = 1e-4 to the model optimizer(Adam), which prevents the model from learning too large weights, which helps avoid overfitting. In my case I used 1e - 4 which is 10^{-4} which equals 0.0001, this value is small to not overly influence on weights, so that they won’t be too small, but still it helps to prevent the model from overfitting by setting slightly smaller weights.

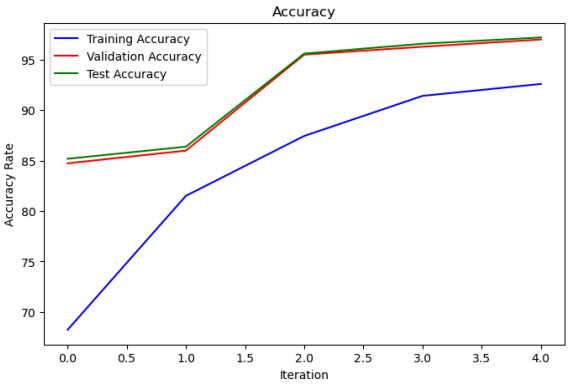
We can compare the final epoch with and without L2 regularizer. Without it, our train loss was 1.5455 and with l2 it is 1.5383, which is slightly smaller, so that our training accuracy is higher 92.61%, without l2 regularizer it is 91.83%. Test accuracy is also higher 97.22%, compared to previous 96.58%. And average loss in the test set is 1.4894, without l2 it is 1.4956.

```
Train Epoch 0: Training loss: 1.7893, Training Accuracy: 68.23%
Validation Accuracy: 84.7400
Test set: Average Loss: 1.6107, Test Accuracy: 8519/10000 (85.19%)
Train Epoch 1: Training loss: 1.6505, Training Accuracy: 81.52%
Validation Accuracy: 86.0000
Test set: Average Loss: 1.5974, Test Accuracy: 8639/10000 (86.39%)
Train Epoch 2: Training loss: 1.5908, Training Accuracy: 87.45%
Validation Accuracy: 95.5200
Test set: Average Loss: 1.5049, Test Accuracy: 9562/10000 (95.62%)
Train Epoch 3: Training loss: 1.5511, Training Accuracy: 91.43%
Validation Accuracy: 96.3000
Test set: Average Loss: 1.4957, Test Accuracy: 9660/10000 (96.60%)
Train Epoch 4: Training loss: 1.5383, Training Accuracy: 92.61%
Validation Accuracy: 97.0200
Test set: Average Loss: 1.4894, Test Accuracy: 9722/10000 (97.22%)
```

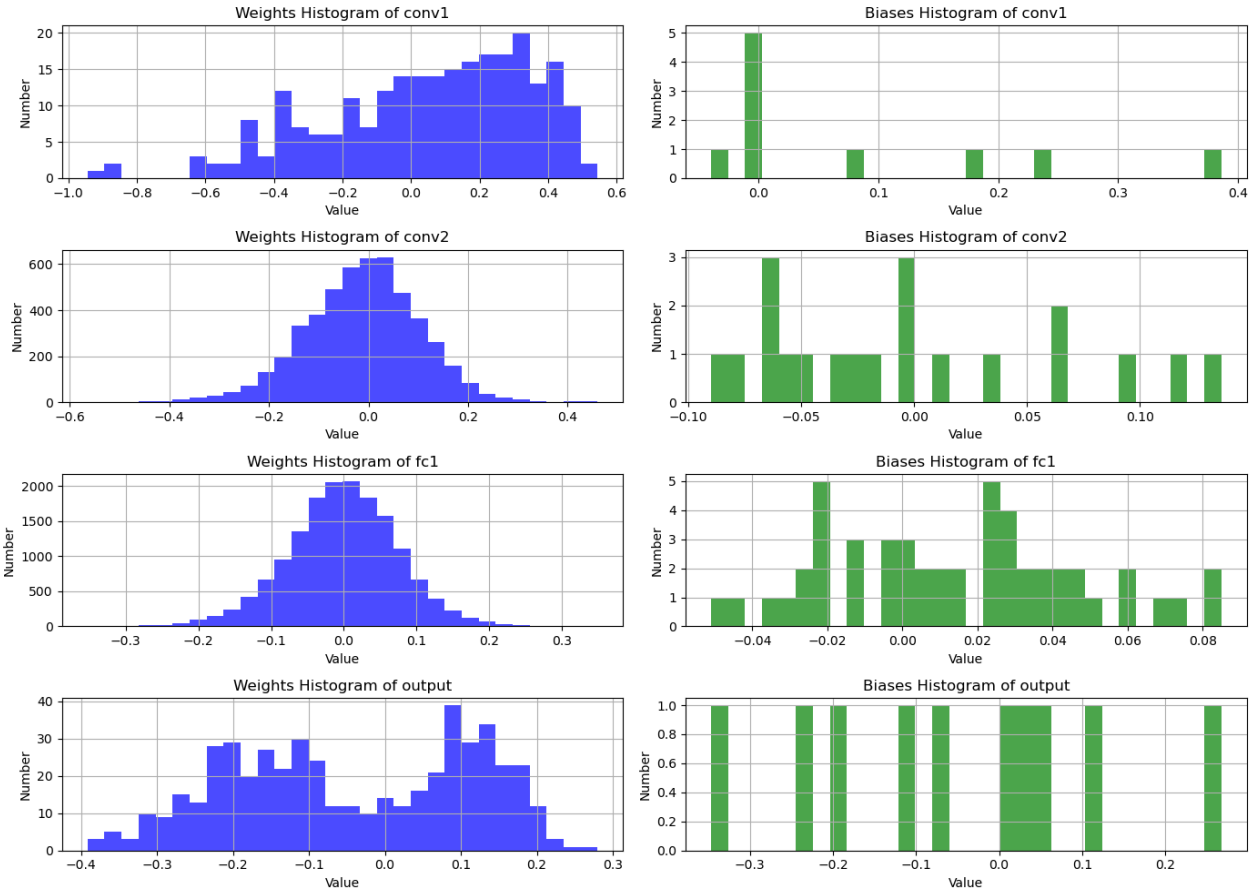
Learning Curve



Accuracy of training and test sets



Distribution of weights and biases (Histograms of layers)



2. **CIFAR10 dataset**

(2-1) Network architecture:

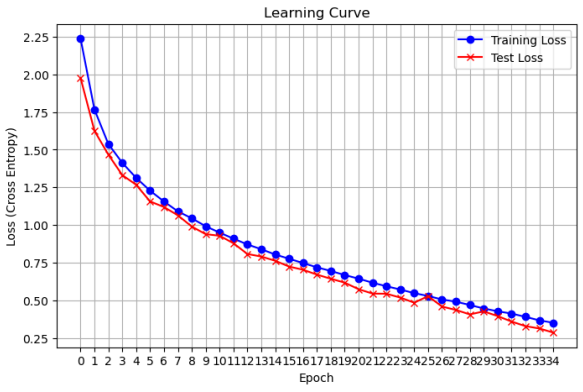
conv1 layer: Extract 3 feature maps from input image, 12 channels and 5*5 kernel.
pool (max_pool2d): Reduce image dimensions, and extracts sharpest features (2*2)
conv2 layer: 12 from previous layer to 24 channels and 5*5 kernel.
fc1 layer: 24 channels*5*5(kernel) = 600 to 120 neurons
fc2 layer: 120 neurons reduced to 84
fc3 layer: 84 to 10 output, that are 10 classes from dataset.

Hyperparameters:

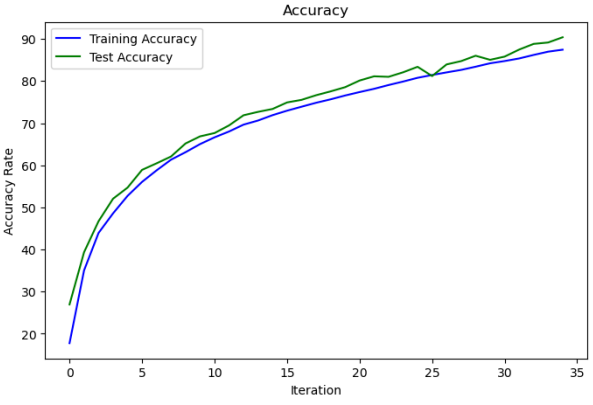
batch_size = 32
kernel_size = 5
learning_rate = 0.001
epochs=35
stride(default) = 1 (for convolutional layers)
stride = 2 (for max-pooling layers)

```
Train Epoch 1: Training loss: 2.2377, Training Accuracy: 17.75%, Test set: Average Loss: 1.9765, Test Accuracy: 13475/50000 (26.95%)
Train Epoch 2: Training loss: 1.7668, Training Accuracy: 35.02%, Test set: Average Loss: 1.6253, Test Accuracy: 19672/50000 (39.34%)
Train Epoch 3: Training loss: 1.5378, Training Accuracy: 43.93%, Test set: Average Loss: 1.4672, Test Accuracy: 23354/50000 (46.71%)
Train Epoch 4: Training loss: 1.4128, Training Accuracy: 48.58%, Test set: Average Loss: 1.3315, Test Accuracy: 26029/50000 (52.06%)
Train Epoch 5: Training loss: 1.3129, Training Accuracy: 52.73%, Test set: Average Loss: 1.2669, Test Accuracy: 27348/50000 (54.70%)
Train Epoch 6: Training loss: 1.2278, Training Accuracy: 56.05%, Test set: Average Loss: 1.1565, Test Accuracy: 29455/50000 (58.91%)
Train Epoch 7: Training loss: 1.1561, Training Accuracy: 58.79%, Test set: Average Loss: 1.1192, Test Accuracy: 30231/50000 (60.46%)
Train Epoch 8: Training loss: 1.0904, Training Accuracy: 61.33%, Test set: Average Loss: 1.0641, Test Accuracy: 31044/50000 (62.09%)
Train Epoch 9: Training loss: 1.0442, Training Accuracy: 63.11%, Test set: Average Loss: 0.9894, Test Accuracy: 32591/50000 (65.18%)
Train Epoch 10: Training loss: 0.9916, Training Accuracy: 65.03%, Test set: Average Loss: 0.9390, Test Accuracy: 33434/50000 (66.87%)
Train Epoch 11: Training loss: 0.9493, Training Accuracy: 66.64%, Test set: Average Loss: 0.9286, Test Accuracy: 33828/50000 (67.66%)
Train Epoch 12: Training loss: 0.9093, Training Accuracy: 68.03%, Test set: Average Loss: 0.8785, Test Accuracy: 34739/50000 (69.48%)
Train Epoch 13: Training loss: 0.8706, Training Accuracy: 69.66%, Test set: Average Loss: 0.8074, Test Accuracy: 35941/50000 (71.88%)
Train Epoch 14: Training loss: 0.8389, Training Accuracy: 70.63%, Test set: Average Loss: 0.7902, Test Accuracy: 36345/50000 (72.69%)
Train Epoch 15: Training loss: 0.8038, Training Accuracy: 71.91%, Test set: Average Loss: 0.7624, Test Accuracy: 36689/50000 (73.38%)
Train Epoch 16: Training loss: 0.7757, Training Accuracy: 72.97%, Test set: Average Loss: 0.7236, Test Accuracy: 37464/50000 (74.93%)
Train Epoch 17: Training loss: 0.7473, Training Accuracy: 73.91%, Test set: Average Loss: 0.7028, Test Accuracy: 37771/50000 (75.54%)
Train Epoch 18: Training loss: 0.7187, Training Accuracy: 74.85%, Test set: Average Loss: 0.6702, Test Accuracy: 38319/50000 (76.64%)
Train Epoch 19: Training loss: 0.6946, Training Accuracy: 75.66%, Test set: Average Loss: 0.6430, Test Accuracy: 38776/50000 (77.55%)
Train Epoch 20: Training loss: 0.6674, Training Accuracy: 76.57%, Test set: Average Loss: 0.6169, Test Accuracy: 39274/50000 (78.55%)
Train Epoch 21: Training loss: 0.6436, Training Accuracy: 77.39%, Test set: Average Loss: 0.5733, Test Accuracy: 40065/50000 (80.13%)
Train Epoch 22: Training loss: 0.6164, Training Accuracy: 78.16%, Test set: Average Loss: 0.5443, Test Accuracy: 40566/50000 (81.13%)
Train Epoch 23: Training loss: 0.5931, Training Accuracy: 79.06%, Test set: Average Loss: 0.5432, Test Accuracy: 40507/50000 (81.01%)
Train Epoch 24: Training loss: 0.5712, Training Accuracy: 79.89%, Test set: Average Loss: 0.5178, Test Accuracy: 41043/50000 (82.09%)
Train Epoch 25: Training loss: 0.5472, Training Accuracy: 80.78%, Test set: Average Loss: 0.4838, Test Accuracy: 41695/50000 (83.39%)
...
Train Epoch 32: Training loss: 0.4114, Training Accuracy: 85.37%, Test set: Average Loss: 0.3585, Test Accuracy: 43740/50000 (87.48%)
Train Epoch 33: Training loss: 0.3899, Training Accuracy: 86.21%, Test set: Average Loss: 0.3282, Test Accuracy: 44415/50000 (88.83%)
Train Epoch 34: Training loss: 0.3664, Training Accuracy: 86.99%, Test set: Average Loss: 0.3125, Test Accuracy: 44582/50000 (89.16%)
Train Epoch 35: Training loss: 0.3505, Training Accuracy: 87.46%, Test set: Average Loss: 0.2853, Test Accuracy: 45203/50000 (90.41%)
```

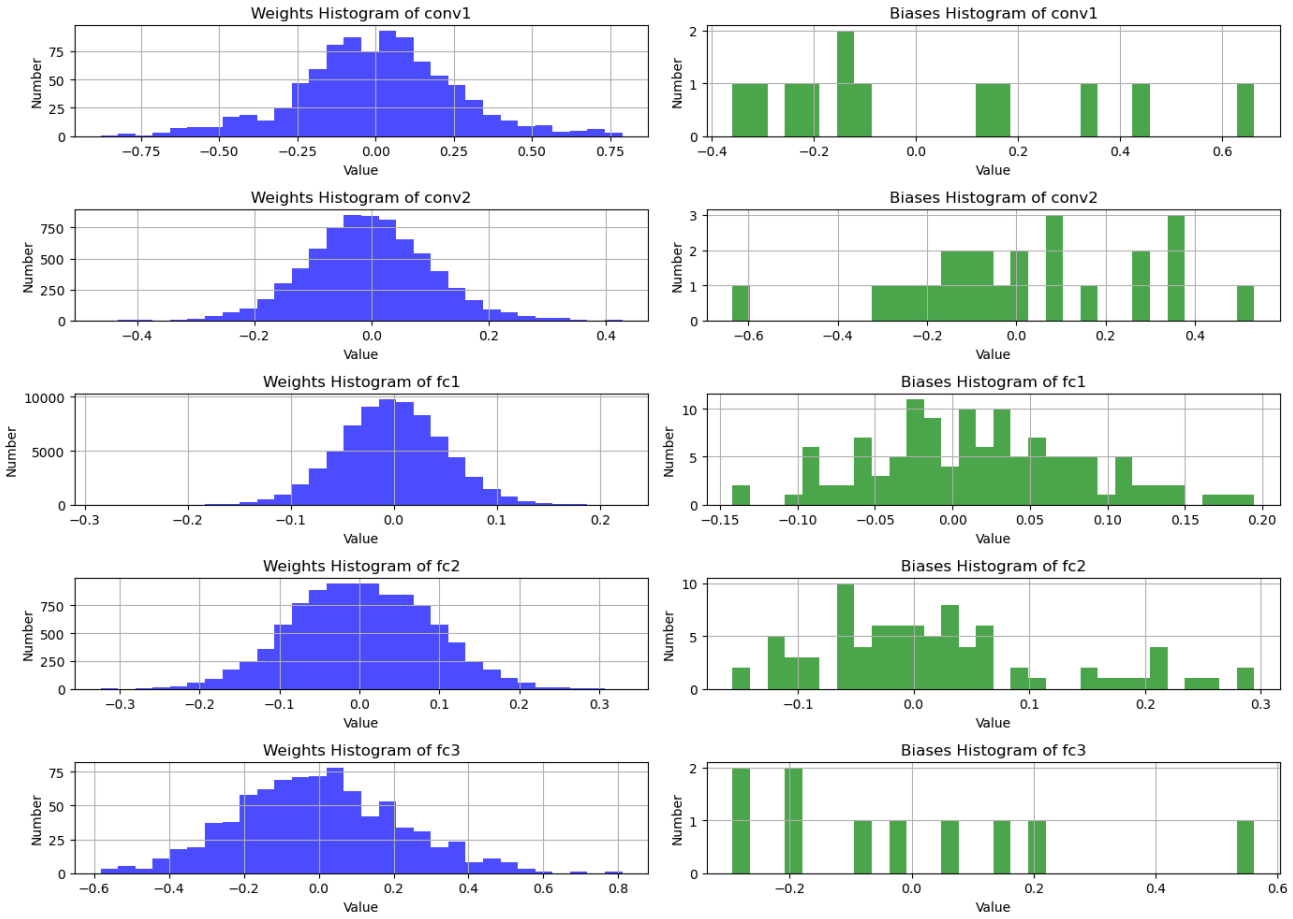
Learning Curve



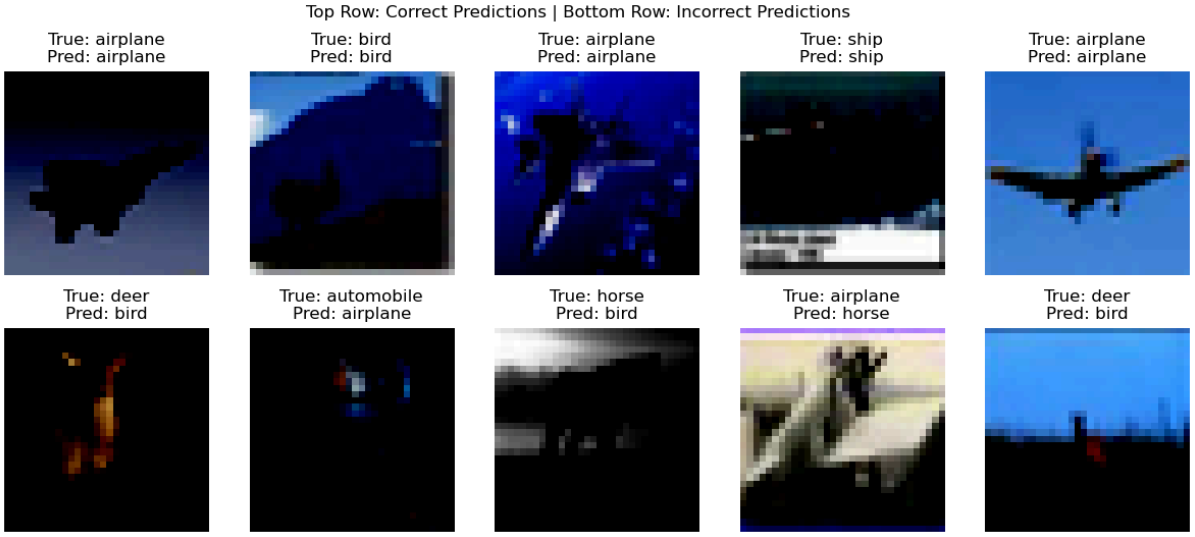
Accuracy of training and test sets



Distribution of weights and biases (Histograms of layers)



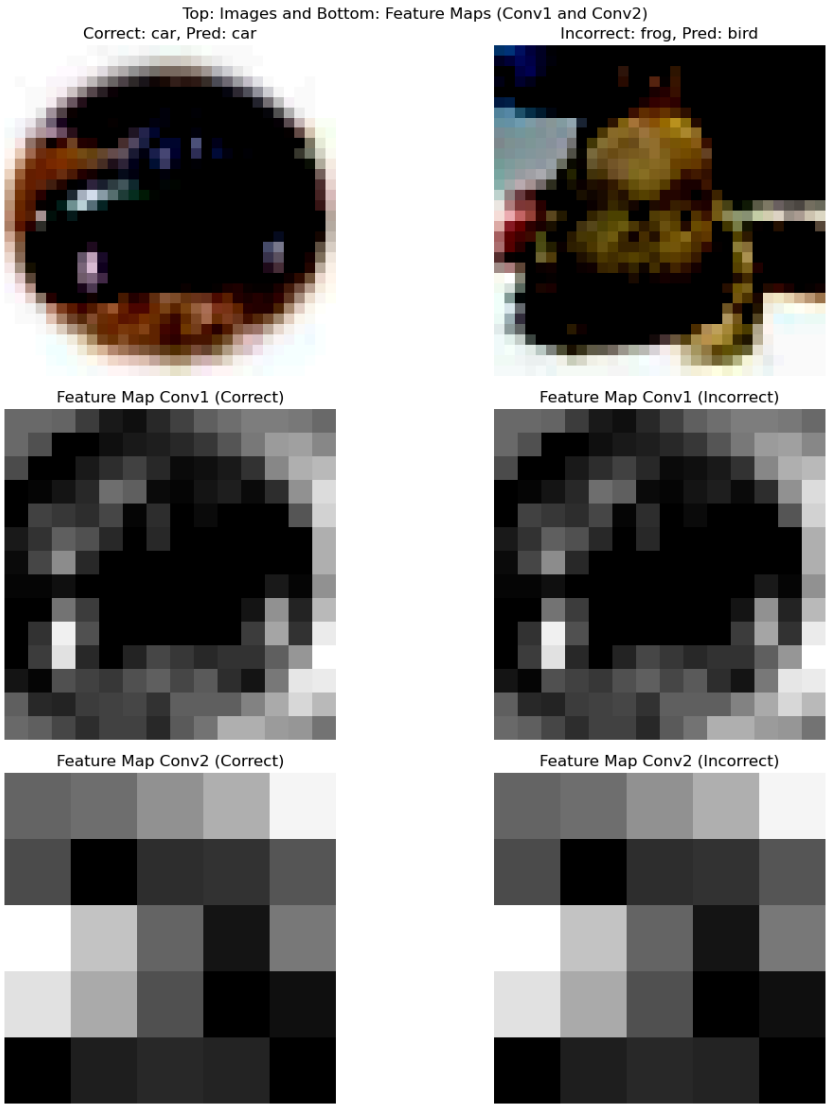
(2-2) Example of correctly classified and misclassified images



From the grid of correctly and misclassified images we can see that the model makes good predictions on such categories as aeroplanes and ships, because these images have distinctive features that are easier for the model to perform well. But the model incorrectly classifies some animals, deer, birds, horses. It can be due to background, similar colours, different textures of images. For instance, the model

incorrectly predicts the image of a horse as a bird, it might be due to colour similarities.

(2-3) Feature maps from different convolutional layers



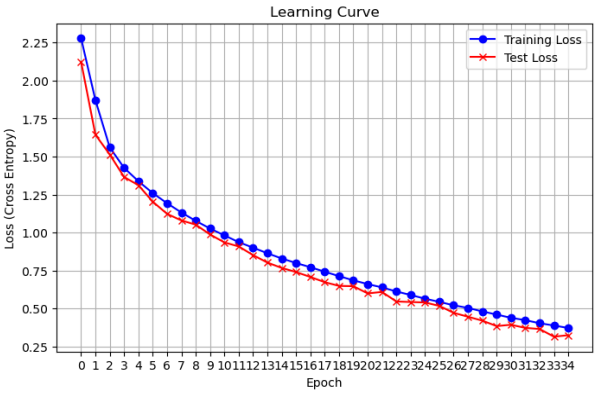
With increasing depth of feature map, model is able to capture more complex details of images, In first conv1 layer model captures edges, corners of image, than in conv2 we can see how it takes small portion of image and captures smaller details, of smaller regions of images.

(2-4) Added L2 regularisation to the CNN

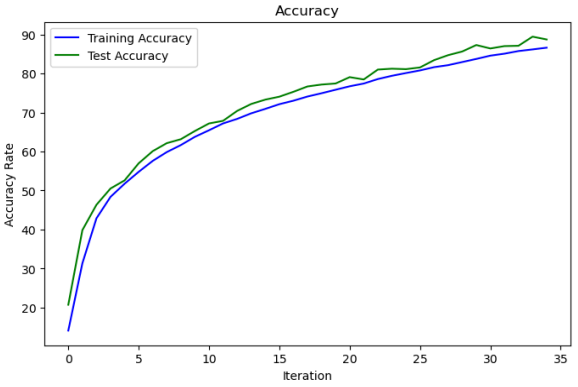
Similarly to MNIST model training, I've added weight_decay = 1e-4, but in the optimizer I've used is SGD (Stochastic Gradient Descent) , although Adam is faster, but had lower accuracy results for Cifar10 dataset, SGD with momentum here led to higher accuracy results, better optimization for the model. Momentum is equal to 0.9, which means that 90% of the previous update's direction will be kept. It helps the optimizer to progress faster.

```
Train Epoch 1: Training loss: 2.2787, Training Accuracy: 14.11%, Test set: Average Loss: 2.1234, Test Accuracy: 10365/50000 (20.73%)
Train Epoch 2: Training loss: 1.8716, Training Accuracy: 31.35%, Test set: Average Loss: 1.6436, Test Accuracy: 19924/50000 (39.85%)
Train Epoch 3: Training loss: 1.5600, Training Accuracy: 42.88%, Test set: Average Loss: 1.5135, Test Accuracy: 23174/50000 (46.35%)
Train Epoch 4: Training loss: 1.4256, Training Accuracy: 48.38%, Test set: Average Loss: 1.3656, Test Accuracy: 25262/50000 (50.52%)
Train Epoch 5: Training loss: 1.3375, Training Accuracy: 51.75%, Test set: Average Loss: 1.3113, Test Accuracy: 26302/50000 (52.60%)
Train Epoch 6: Training loss: 1.2594, Training Accuracy: 54.79%, Test set: Average Loss: 1.2030, Test Accuracy: 28488/50000 (56.98%)
Train Epoch 7: Training loss: 1.1927, Training Accuracy: 57.60%, Test set: Average Loss: 1.1227, Test Accuracy: 30050/50000 (60.10%)
Train Epoch 8: Training loss: 1.1317, Training Accuracy: 59.86%, Test set: Average Loss: 1.0793, Test Accuracy: 31075/50000 (62.15%)
Train Epoch 9: Training loss: 1.0768, Training Accuracy: 61.66%, Test set: Average Loss: 1.0517, Test Accuracy: 31581/50000 (63.16%)
Train Epoch 10: Training loss: 1.0261, Training Accuracy: 63.78%, Test set: Average Loss: 0.9869, Test Accuracy: 32635/50000 (65.27%)
Train Epoch 11: Training loss: 0.9812, Training Accuracy: 65.45%, Test set: Average Loss: 0.9348, Test Accuracy: 33598/50000 (67.20%)
Train Epoch 12: Training loss: 0.9366, Training Accuracy: 67.20%, Test set: Average Loss: 0.9083, Test Accuracy: 33936/50000 (67.87%)
Train Epoch 13: Training loss: 0.9004, Training Accuracy: 68.37%, Test set: Average Loss: 0.8514, Test Accuracy: 35206/50000 (70.41%)
Train Epoch 14: Training loss: 0.8634, Training Accuracy: 69.80%, Test set: Average Loss: 0.8021, Test Accuracy: 36096/50000 (72.19%)
Train Epoch 15: Training loss: 0.8291, Training Accuracy: 70.93%, Test set: Average Loss: 0.7668, Test Accuracy: 36652/50000 (73.30%)
Train Epoch 16: Training loss: 0.7992, Training Accuracy: 72.14%, Test set: Average Loss: 0.7398, Test Accuracy: 37031/50000 (74.06%)
Train Epoch 17: Training loss: 0.7717, Training Accuracy: 73.03%, Test set: Average Loss: 0.7073, Test Accuracy: 37656/50000 (75.31%)
Train Epoch 18: Training loss: 0.7420, Training Accuracy: 74.10%, Test set: Average Loss: 0.6731, Test Accuracy: 38342/50000 (76.68%)
Train Epoch 19: Training loss: 0.7127, Training Accuracy: 74.94%, Test set: Average Loss: 0.6485, Test Accuracy: 38586/50000 (77.17%)
Train Epoch 20: Training loss: 0.6853, Training Accuracy: 75.84%, Test set: Average Loss: 0.6467, Test Accuracy: 38722/50000 (77.44%)
Train Epoch 21: Training loss: 0.6605, Training Accuracy: 76.73%, Test set: Average Loss: 0.5992, Test Accuracy: 39535/50000 (79.07%)
Train Epoch 22: Training loss: 0.6386, Training Accuracy: 77.43%, Test set: Average Loss: 0.6087, Test Accuracy: 39239/50000 (78.48%)
Train Epoch 23: Training loss: 0.6119, Training Accuracy: 78.58%, Test set: Average Loss: 0.5459, Test Accuracy: 40505/50000 (81.01%)
Train Epoch 24: Training loss: 0.5888, Training Accuracy: 79.42%, Test set: Average Loss: 0.5425, Test Accuracy: 40618/50000 (81.24%)
Train Epoch 25: Training loss: 0.5645, Training Accuracy: 80.12%, Test set: Average Loss: 0.5395, Test Accuracy: 40560/50000 (81.12%)
...
Train Epoch 32: Training loss: 0.4230, Training Accuracy: 85.08%, Test set: Average Loss: 0.3723, Test Accuracy: 43513/50000 (87.03%)
Train Epoch 33: Training loss: 0.4029, Training Accuracy: 85.76%, Test set: Average Loss: 0.3648, Test Accuracy: 43556/50000 (87.11%)
Train Epoch 34: Training loss: 0.3877, Training Accuracy: 86.18%, Test set: Average Loss: 0.3144, Test Accuracy: 44725/50000 (89.45%)
Train Epoch 35: Training loss: 0.3717, Training Accuracy: 86.62%, Test set: Average Loss: 0.3239, Test Accuracy: 44365/50000 (88.73%)
```

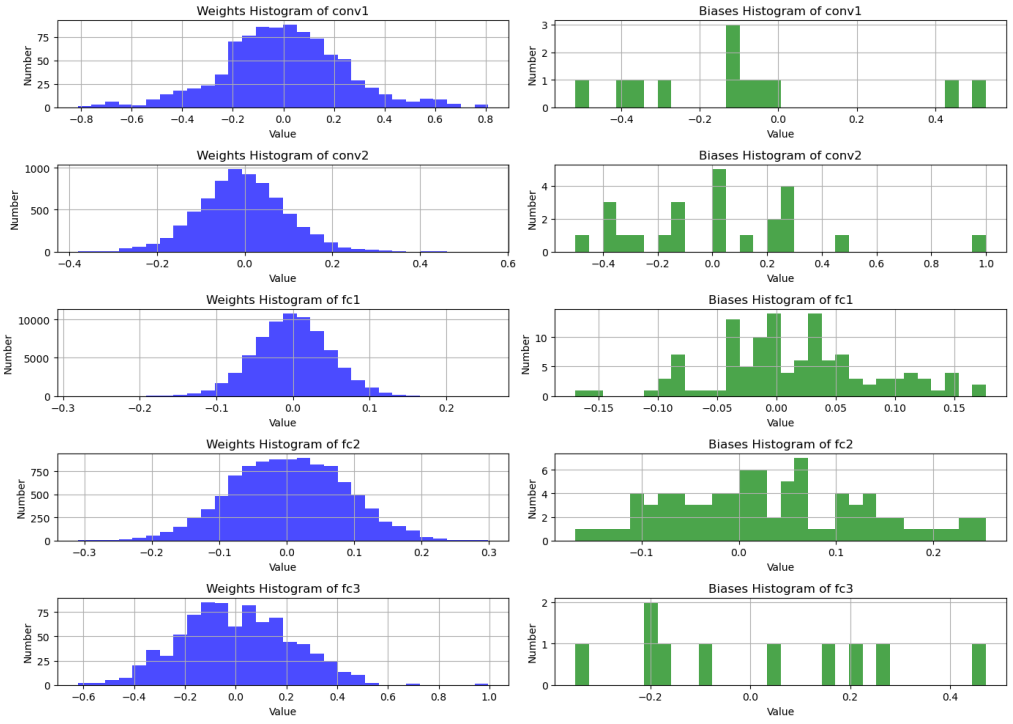
Learning Curve



Accuracy of training and test sets



Distribution of weights and biases (Histograms of layers)



(2-5) Preprocessing of CIFAR10 dataset

The dataset is preprocessed in 2 steps:

1) tensor conversion
2) normalisation

```
transform = transforms.Compose([

    transforms.ToTensor(),

    transforms.Normalize((0.5,0.5,0.5),(0.5,0.5,0.5))

])
```

1) For tensor conversion, we convert PIL (Python Imaging Library) image format to PyTorch tensors, so that model would be able to process images. This results in output of 3 dimensional tensors (C,H,W), c is number of channels (we have 3 for RGB image), H is height (32px), W is width (32px). During conversion px values scale down from 0, 255 range for each channel to 0,1 range (px / 255).

2) For normalisation, we standardise px values for 3 channels(R,G,B). Formula is x(px value) - mean/STD (Standard Deviation). So that for each px in each channel the mean is 0.5, STD is 0.5. After normalisation px values ranged from -1 to 1 for 3 channels (RGB).