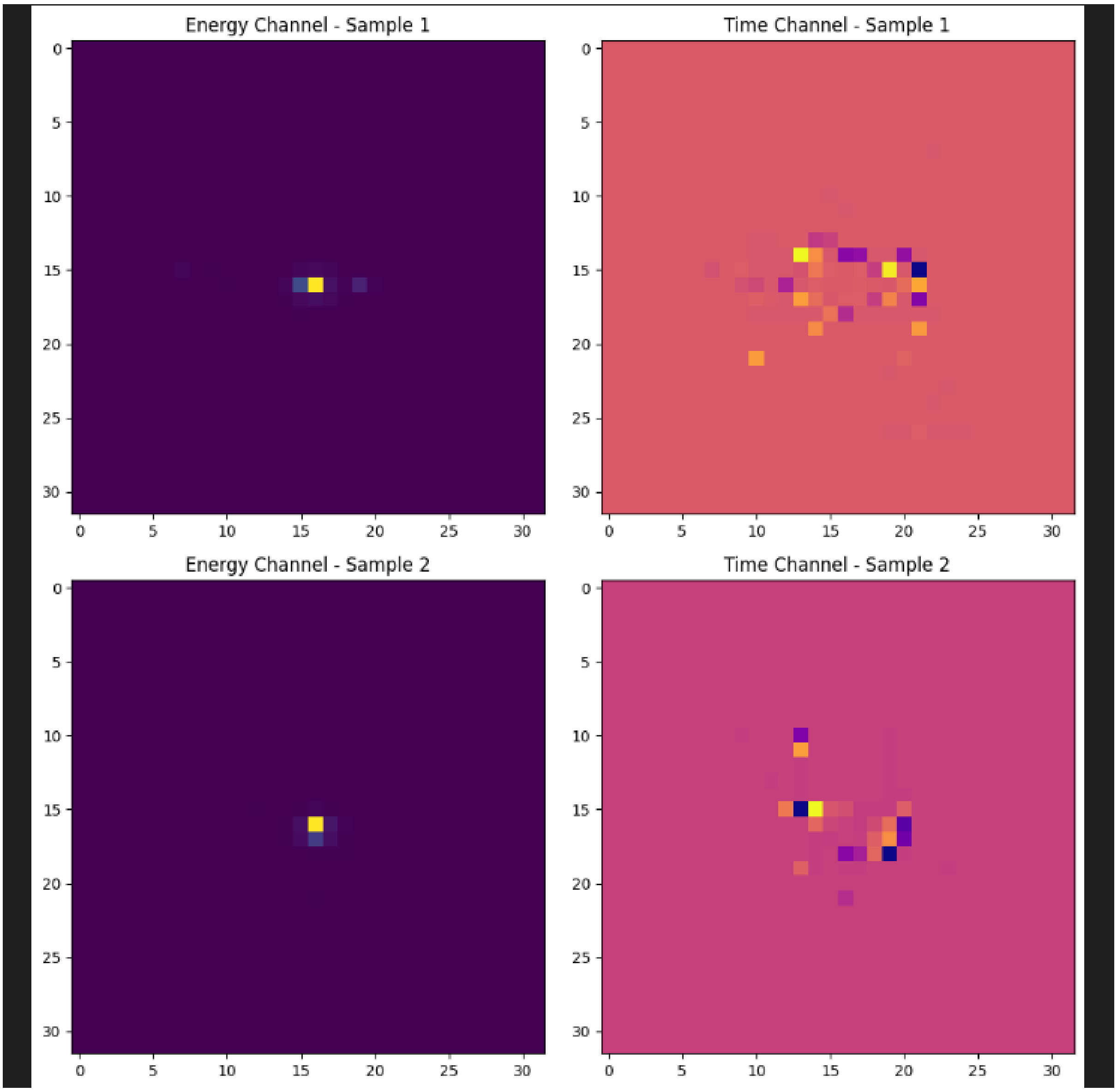
Outputs for common task1

Data visualisation of hit energy and time of the particle



### INSIGHTS FROM THE DATA

### How Can Energy and Time Be Used to Classify Electrons and Photons?

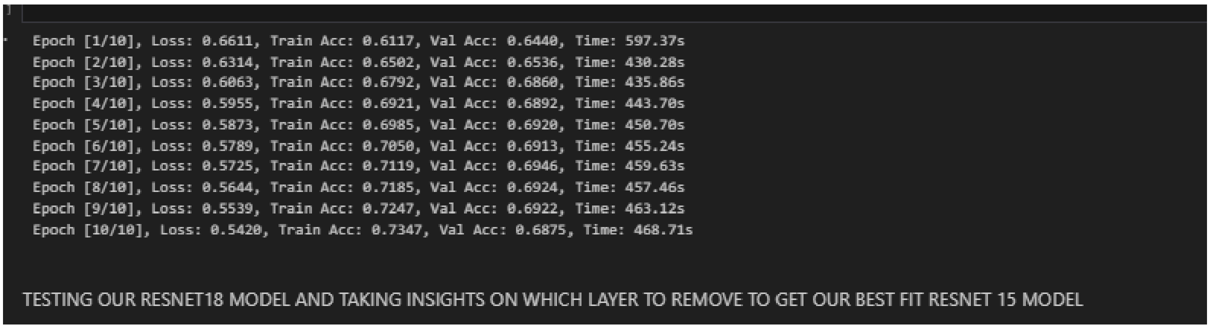
Even though the data looks like images (a grid of values), it actually captures the physics of particle interactions:

Electrons interact more frequently and leave a broader spread of energy in the detector.

Photons convert into electron-positron pairs, which means their energy deposition pattern is different from a direct electron hit.

Time Information is useful because photons tend to interact slightly differently in the detector compared to electrons.

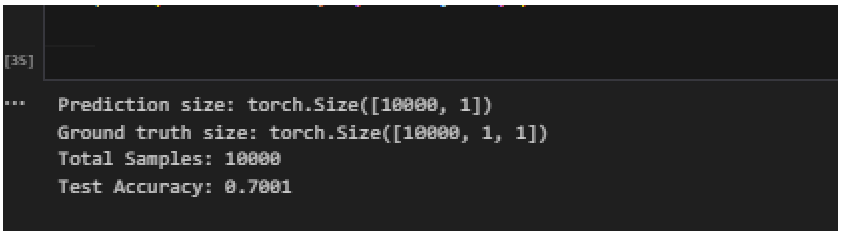
Training resnet 18 to take idea which three layers to remove and make our own Resnet15



```
Epoch [1/10], Loss: 0.6611, Train Acc: 0.6117, Val Acc: 0.6440, Time: 597.37s
Epoch [2/10], Loss: 0.6314, Train Acc: 0.6502, Val Acc: 0.6536, Time: 430.28s
Epoch [3/10], Loss: 0.6063, Train Acc: 0.6792, Val Acc: 0.6860, Time: 435.86s
Epoch [4/10], Loss: 0.5955, Train Acc: 0.6921, Val Acc: 0.6892, Time: 443.70s
Epoch [5/10], Loss: 0.5873, Train Acc: 0.6985, Val Acc: 0.6920, Time: 450.70s
Epoch [6/10], Loss: 0.5789, Train Acc: 0.7050, Val Acc: 0.6913, Time: 455.24s
Epoch [7/10], Loss: 0.5725, Train Acc: 0.7119, Val Acc: 0.6946, Time: 459.63s
Epoch [8/10], Loss: 0.5644, Train Acc: 0.7185, Val Acc: 0.6924, Time: 457.46s
Epoch [9/10], Loss: 0.5539, Train Acc: 0.7247, Val Acc: 0.6922, Time: 463.12s
Epoch [10/10], Loss: 0.5420, Train Acc: 0.7347, Val Acc: 0.6875, Time: 468.71s
```

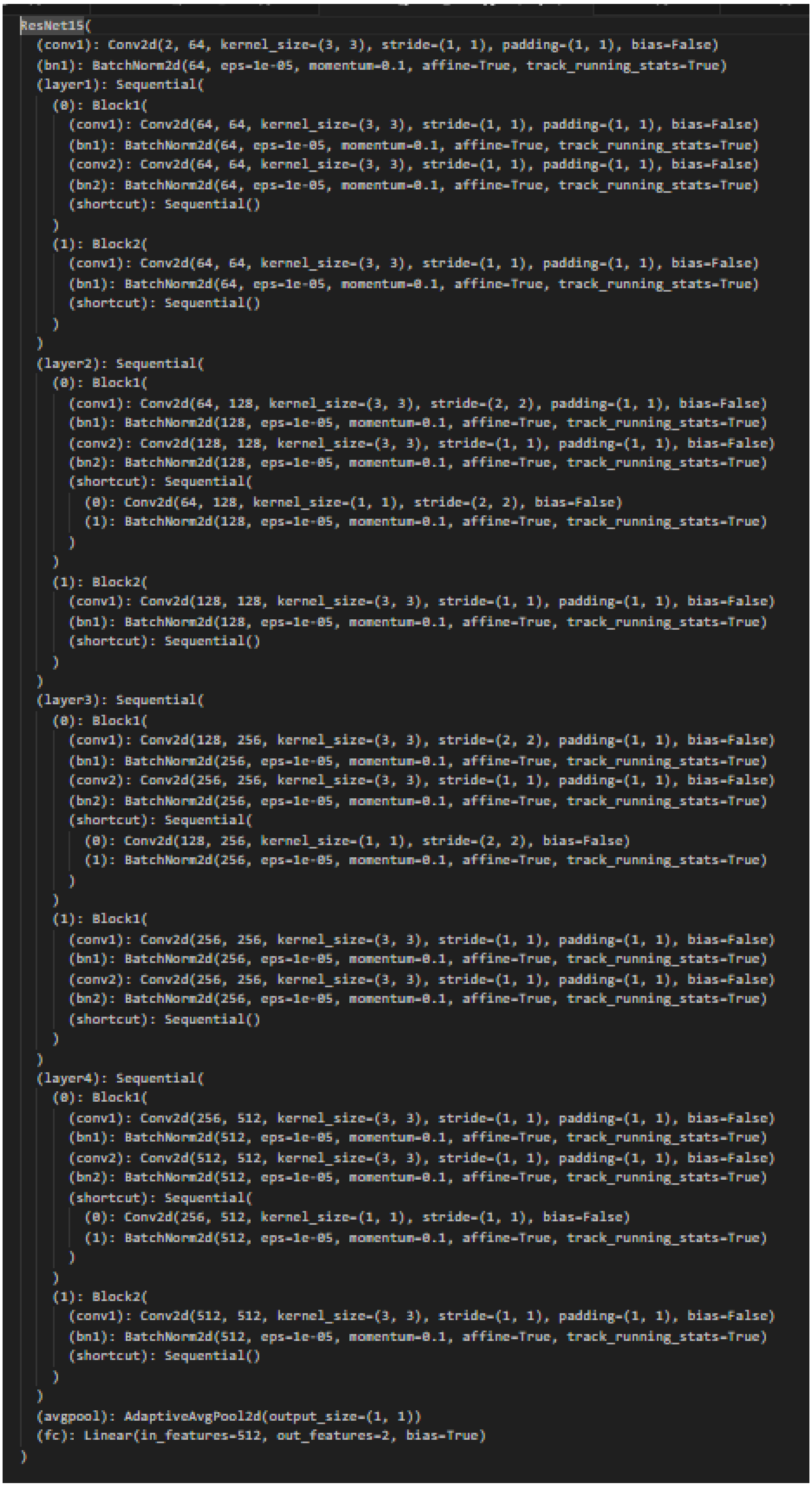
TESTING OUR RESNET18 MODEL AND TAKING INSIGHTS ON WHICH LAYER TO REMOVE TO GET OUR BEST FIT RESNET 15 MODEL

Model accuracy

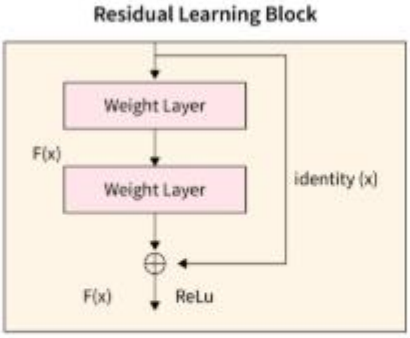


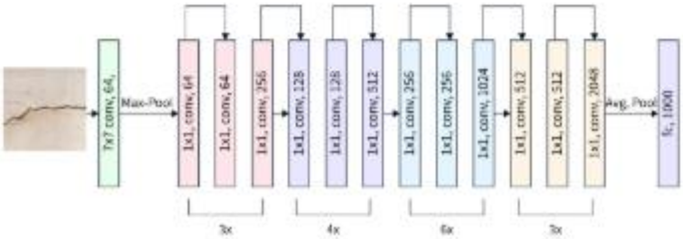
```
Prediction size: torch.Size([10000, 1])
Ground truth size: torch.Size([10000, 1, 1])
Total Samples: 10000
Test Accuracy: 0.7001
```

Resnet 15 MODEL 1 architecture

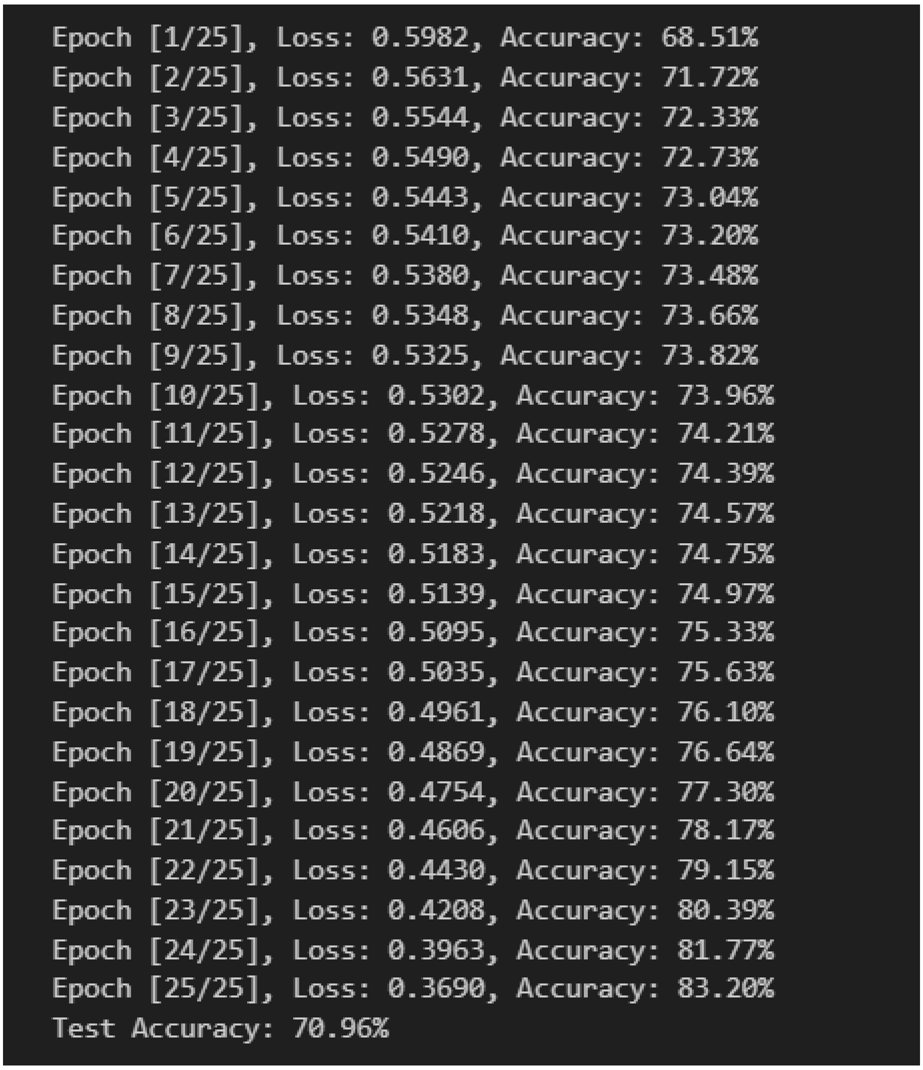


```
ResNet15(
  (conv1): Conv2d(2, 64, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False)
  (bn1): BatchNorm2d(64, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
  (layer1): Sequential(
    (0): Block1(
      (conv1): Conv2d(64, 64, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False)
      (bn1): BatchNorm2d(64, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
      (conv2): Conv2d(64, 64, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False)
      (bn2): BatchNorm2d(64, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
      (shortcut): Sequential()
    )
    (1): Block2(
      (conv1): Conv2d(64, 64, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False)
      (bn1): BatchNorm2d(64, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
      (shortcut): Sequential()
    )
  )
  (layer2): Sequential(
    (0): Block1(
      (conv1): Conv2d(64, 128, kernel_size=(3, 3), stride=(2, 2), padding=(1, 1), bias=False)
      (bn1): BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
      (conv2): Conv2d(128, 128, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False)
      (bn2): BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
      (shortcut): Sequential(
        (0): Conv2d(64, 128, kernel_size=(1, 1), stride=(2, 2), bias=False)
        (1): BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
      )
    )
    (1): Block2(
      (conv1): Conv2d(128, 128, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False)
      (bn1): BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
      (shortcut): Sequential()
    )
  )
  (layer3): Sequential(
    (0): Block1(
      (conv1): Conv2d(128, 256, kernel_size=(3, 3), stride=(2, 2), padding=(1, 1), bias=False)
      (bn1): BatchNorm2d(256, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
      (conv2): Conv2d(256, 256, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False)
      (bn2): BatchNorm2d(256, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
      (shortcut): Sequential(
        (0): Conv2d(128, 256, kernel_size=(1, 1), stride=(2, 2), bias=False)
        (1): BatchNorm2d(256, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
      )
    )
    (1): Block1(
      (conv1): Conv2d(256, 256, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False)
      (bn1): BatchNorm2d(256, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
      (conv2): Conv2d(256, 256, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False)
      (bn2): BatchNorm2d(256, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
      (shortcut): Sequential()
    )
  )
  (layer4): Sequential(
    (0): Block1(
      (conv1): Conv2d(256, 512, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False)
      (bn1): BatchNorm2d(512, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
      (conv2): Conv2d(512, 512, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False)
      (bn2): BatchNorm2d(512, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
      (shortcut): Sequential(
        (0): Conv2d(256, 512, kernel_size=(1, 1), stride=(1, 1), bias=False)
        (1): BatchNorm2d(512, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
      )
    )
    (1): Block2(
      (conv1): Conv2d(512, 512, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False)
      (bn1): BatchNorm2d(512, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
      (shortcut): Sequential()
    )
  )
  (avgpool): AdaptiveAvgPool2d(output_size=(1, 1))
  (fc): Linear(in_features=512, out_features=2, bias=True)
)
```



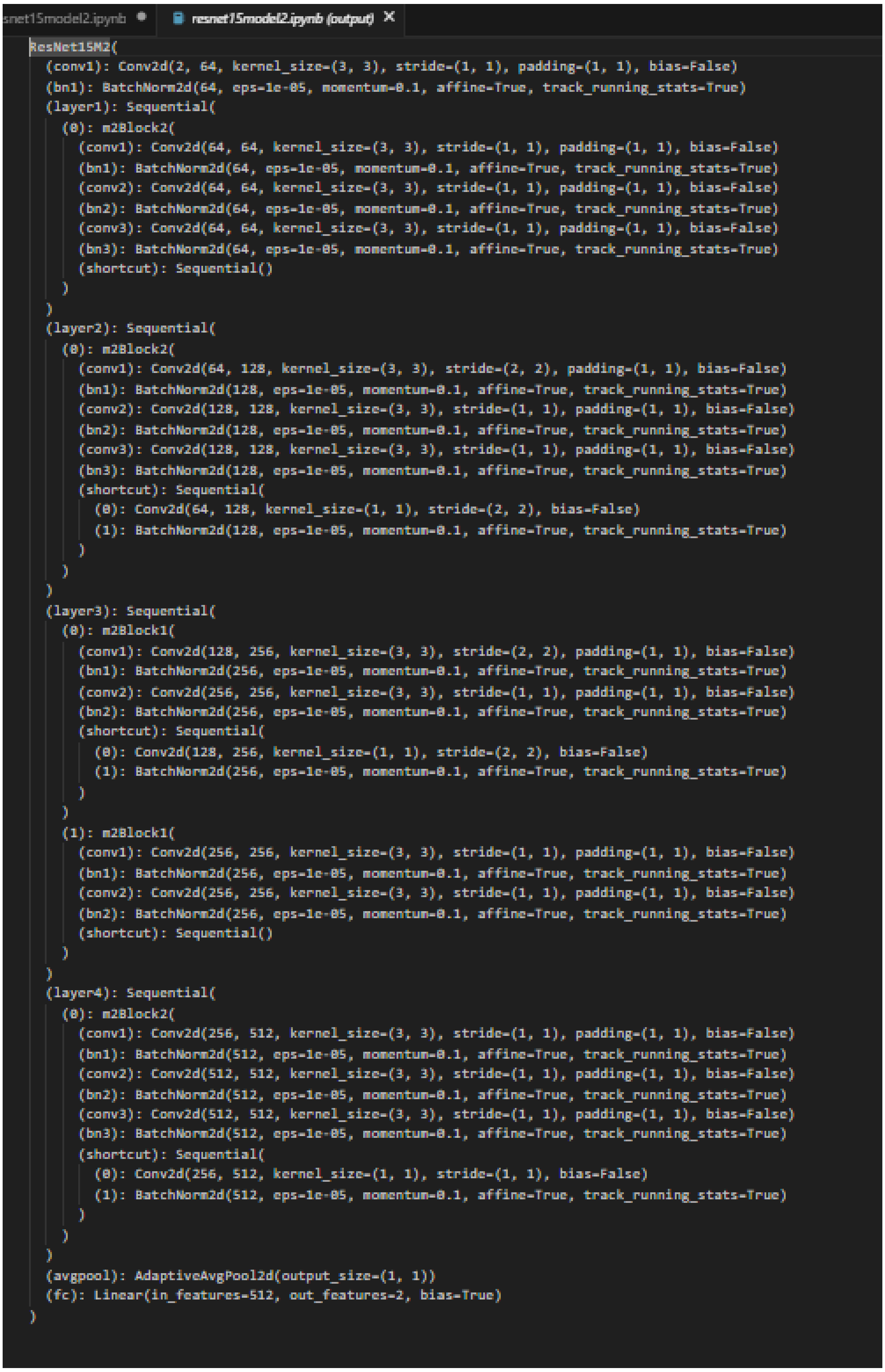


Training and test accuracy of our model1

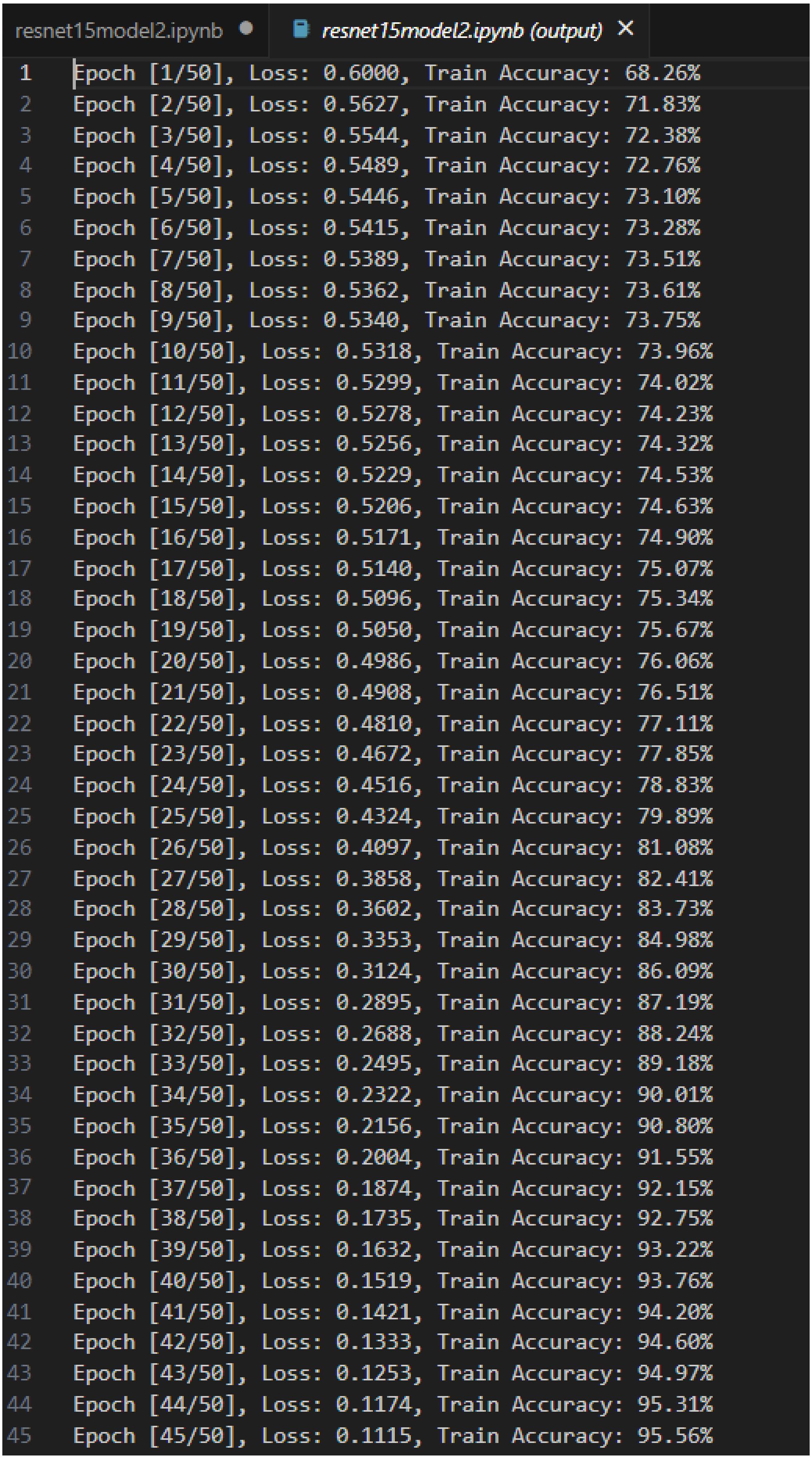


```
Epoch [1/25], Loss: 0.5982, Accuracy: 68.51%
Epoch [2/25], Loss: 0.5631, Accuracy: 71.72%
Epoch [3/25], Loss: 0.5544, Accuracy: 72.33%
Epoch [4/25], Loss: 0.5490, Accuracy: 72.73%
Epoch [5/25], Loss: 0.5443, Accuracy: 73.04%
Epoch [6/25], Loss: 0.5410, Accuracy: 73.20%
Epoch [7/25], Loss: 0.5380, Accuracy: 73.48%
Epoch [8/25], Loss: 0.5348, Accuracy: 73.66%
Epoch [9/25], Loss: 0.5325, Accuracy: 73.82%
Epoch [10/25], Loss: 0.5302, Accuracy: 73.96%
Epoch [11/25], Loss: 0.5278, Accuracy: 74.21%
Epoch [12/25], Loss: 0.5246, Accuracy: 74.39%
Epoch [13/25], Loss: 0.5218, Accuracy: 74.57%
Epoch [14/25], Loss: 0.5183, Accuracy: 74.75%
Epoch [15/25], Loss: 0.5139, Accuracy: 74.97%
Epoch [16/25], Loss: 0.5095, Accuracy: 75.33%
Epoch [17/25], Loss: 0.5035, Accuracy: 75.63%
Epoch [18/25], Loss: 0.4961, Accuracy: 76.10%
Epoch [19/25], Loss: 0.4869, Accuracy: 76.64%
Epoch [20/25], Loss: 0.4754, Accuracy: 77.30%
Epoch [21/25], Loss: 0.4606, Accuracy: 78.17%
Epoch [22/25], Loss: 0.4430, Accuracy: 79.15%
Epoch [23/25], Loss: 0.4208, Accuracy: 80.39%
Epoch [24/25], Loss: 0.3963, Accuracy: 81.77%
Epoch [25/25], Loss: 0.3690, Accuracy: 83.20%
Test Accuracy: 70.96%
```

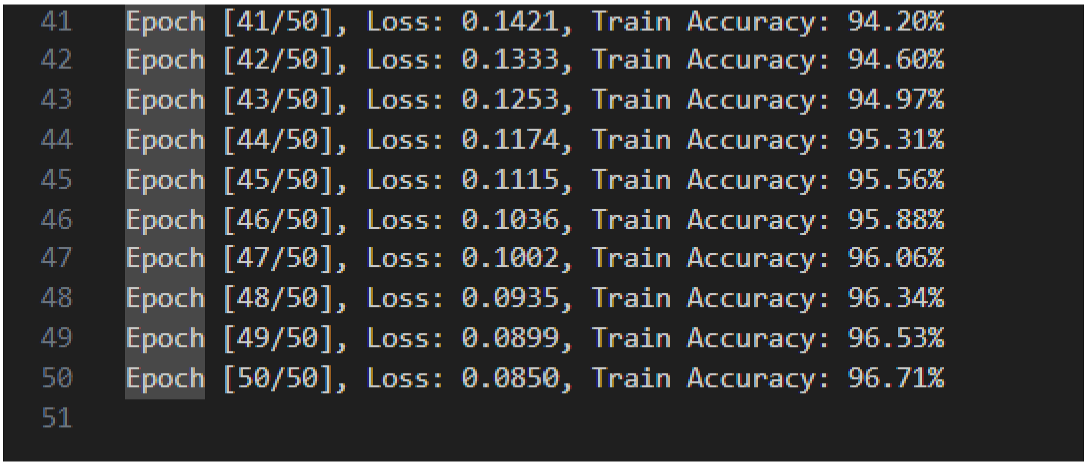
Resnet15 Model2 Architecture



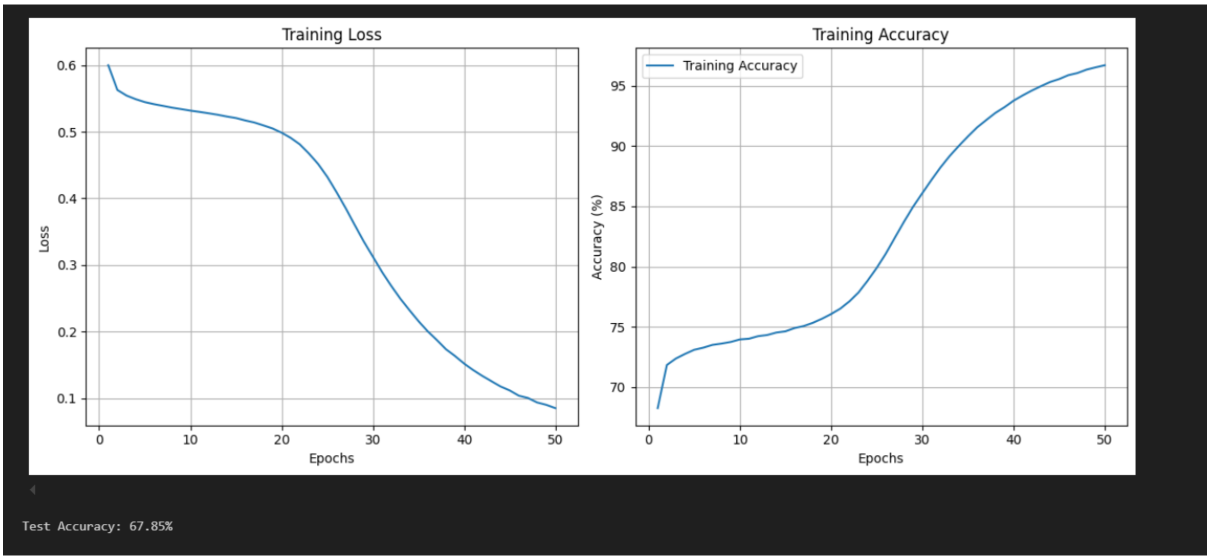
```
ResNet15M2(
  (conv1): Conv2d(2, 64, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False)
  (bn1): BatchNorm2d(64, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
  (layer1): Sequential(
    (0): m2Block2(
      (conv1): Conv2d(64, 64, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False)
      (bn1): BatchNorm2d(64, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
      (conv2): Conv2d(64, 64, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False)
      (bn2): BatchNorm2d(64, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
      (conv3): Conv2d(64, 64, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False)
      (bn3): BatchNorm2d(64, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
      (shortcut): Sequential()
    )
  )
  (layer2): Sequential(
    (0): m2Block2(
      (conv1): Conv2d(64, 128, kernel_size=(3, 3), stride=(2, 2), padding=(1, 1), bias=False)
      (bn1): BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
      (conv2): Conv2d(128, 128, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False)
      (bn2): BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
      (conv3): Conv2d(128, 128, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False)
      (bn3): BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
      (shortcut): Sequential(
        (0): Conv2d(64, 128, kernel_size=(1, 1), stride=(2, 2), bias=False)
        (1): BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
      )
    )
  )
  (layer3): Sequential(
    (0): m2Block1(
      (conv1): Conv2d(128, 256, kernel_size=(3, 3), stride=(2, 2), padding=(1, 1), bias=False)
      (bn1): BatchNorm2d(256, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
      (conv2): Conv2d(256, 256, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False)
      (bn2): BatchNorm2d(256, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
      (shortcut): Sequential(
        (0): Conv2d(128, 256, kernel_size=(1, 1), stride=(2, 2), bias=False)
        (1): BatchNorm2d(256, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
      )
    )
    (1): m2Block1(
      (conv1): Conv2d(256, 256, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False)
      (bn1): BatchNorm2d(256, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
      (conv2): Conv2d(256, 256, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False)
      (bn2): BatchNorm2d(256, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
      (shortcut): Sequential()
    )
  )
  (layer4): Sequential(
    (0): m2Block2(
      (conv1): Conv2d(256, 512, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False)
      (bn1): BatchNorm2d(512, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
      (conv2): Conv2d(512, 512, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False)
      (bn2): BatchNorm2d(512, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
      (conv3): Conv2d(512, 512, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False)
      (bn3): BatchNorm2d(512, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
      (shortcut): Sequential(
        (0): Conv2d(256, 512, kernel_size=(1, 1), stride=(1, 1), bias=False)
        (1): BatchNorm2d(512, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
      )
    )
  )
  (avgpool): AdaptiveAvgPool2d(output_size=(1, 1))
  (fc): Linear(in_features=512, out_features=2, bias=True)
)
```



```
Epoch [1/50], Loss: 0.6000, Train Accuracy: 68.26%
Epoch [2/50], Loss: 0.5627, Train Accuracy: 71.83%
Epoch [3/50], Loss: 0.5544, Train Accuracy: 72.38%
Epoch [4/50], Loss: 0.5489, Train Accuracy: 72.76%
Epoch [5/50], Loss: 0.5446, Train Accuracy: 73.10%
Epoch [6/50], Loss: 0.5415, Train Accuracy: 73.28%
Epoch [7/50], Loss: 0.5389, Train Accuracy: 73.51%
Epoch [8/50], Loss: 0.5362, Train Accuracy: 73.61%
Epoch [9/50], Loss: 0.5340, Train Accuracy: 73.75%
Epoch [10/50], Loss: 0.5318, Train Accuracy: 73.96%
Epoch [11/50], Loss: 0.5299, Train Accuracy: 74.02%
Epoch [12/50], Loss: 0.5278, Train Accuracy: 74.23%
Epoch [13/50], Loss: 0.5256, Train Accuracy: 74.32%
Epoch [14/50], Loss: 0.5229, Train Accuracy: 74.53%
Epoch [15/50], Loss: 0.5206, Train Accuracy: 74.63%
Epoch [16/50], Loss: 0.5171, Train Accuracy: 74.90%
Epoch [17/50], Loss: 0.5140, Train Accuracy: 75.07%
Epoch [18/50], Loss: 0.5096, Train Accuracy: 75.34%
Epoch [19/50], Loss: 0.5050, Train Accuracy: 75.67%
Epoch [20/50], Loss: 0.4986, Train Accuracy: 76.06%
Epoch [21/50], Loss: 0.4908, Train Accuracy: 76.51%
Epoch [22/50], Loss: 0.4810, Train Accuracy: 77.11%
Epoch [23/50], Loss: 0.4672, Train Accuracy: 77.85%
Epoch [24/50], Loss: 0.4516, Train Accuracy: 78.83%
Epoch [25/50], Loss: 0.4324, Train Accuracy: 79.89%
Epoch [26/50], Loss: 0.4097, Train Accuracy: 81.08%
Epoch [27/50], Loss: 0.3858, Train Accuracy: 82.41%
Epoch [28/50], Loss: 0.3602, Train Accuracy: 83.73%
Epoch [29/50], Loss: 0.3353, Train Accuracy: 84.98%
Epoch [30/50], Loss: 0.3124, Train Accuracy: 86.09%
Epoch [31/50], Loss: 0.2895, Train Accuracy: 87.19%
Epoch [32/50], Loss: 0.2688, Train Accuracy: 88.24%
Epoch [33/50], Loss: 0.2495, Train Accuracy: 89.18%
Epoch [34/50], Loss: 0.2322, Train Accuracy: 90.01%
Epoch [35/50], Loss: 0.2156, Train Accuracy: 90.80%
Epoch [36/50], Loss: 0.2004, Train Accuracy: 91.55%
Epoch [37/50], Loss: 0.1874, Train Accuracy: 92.15%
Epoch [38/50], Loss: 0.1735, Train Accuracy: 92.75%
Epoch [39/50], Loss: 0.1632, Train Accuracy: 93.22%
Epoch [40/50], Loss: 0.1519, Train Accuracy: 93.76%
Epoch [41/50], Loss: 0.1421, Train Accuracy: 94.20%
Epoch [42/50], Loss: 0.1333, Train Accuracy: 94.60%
Epoch [43/50], Loss: 0.1253, Train Accuracy: 94.97%
Epoch [44/50], Loss: 0.1174, Train Accuracy: 95.31%
Epoch [45/50], Loss: 0.1115, Train Accuracy: 95.56%
```



```
Epoch [41/50], Loss: 0.1421, Train Accuracy: 94.20%
Epoch [42/50], Loss: 0.1333, Train Accuracy: 94.60%
Epoch [43/50], Loss: 0.1253, Train Accuracy: 94.97%
Epoch [44/50], Loss: 0.1174, Train Accuracy: 95.31%
Epoch [45/50], Loss: 0.1115, Train Accuracy: 95.56%
Epoch [46/50], Loss: 0.1036, Train Accuracy: 95.88%
Epoch [47/50], Loss: 0.1002, Train Accuracy: 96.06%
Epoch [48/50], Loss: 0.0935, Train Accuracy: 96.34%
Epoch [49/50], Loss: 0.0899, Train Accuracy: 96.53%
Epoch [50/50], Loss: 0.0850, Train Accuracy: 96.71%
```



```
Test Accuracy: 67.85%
```

Final conclusion :

The train accuracy of model 2 is more as the number of epoch are 50 but the test accuracy is just 67.85 so compared to that model 1 perform better than that