```
from models import resnet15v2
from train import get_dataloaders, evaluate
import torch
import torch.nn as nn
from torchsummary import summary

print(torch.cuda.is_available())
print(torch.cuda.device_count())
```

```
True
1
```

```
device = torch.device("cuda:0" if torch.cuda.is_available() else
"cpu")
print(device)
```

```
cuda:0
```

Adam Optimizer with 50 Epochs

```
# Load the model
model = resnet15v2()
model.load_state_dict(torch.load('expt_50_adam/best.pth',
map_location=device))
model = model.to(device)
```

Model Architecture

```
summary(model, (2, 32, 32))
```

```
----------------------------------------------------------------
        Layer (type)               Output Shape         Param #
================================================================
            Conv2d-1           [-1, 64, 32, 32]           1,152
       BatchNorm2d-2           [-1, 64, 32, 32]             128
              ReLU-3           [-1, 64, 32, 32]               0
            Conv2d-4           [-1, 64, 32, 32]          36,864
       BatchNorm2d-5           [-1, 64, 32, 32]             128
              ReLU-6           [-1, 64, 32, 32]               0
          Identity-7           [-1, 64, 32, 32]               0
            Conv2d-8           [-1, 64, 32, 32]          36,864
       BatchNorm2d-9           [-1, 64, 32, 32]             128
             ReLU-10           [-1, 64, 32, 32]               0
           Conv2d-11           [-1, 64, 32, 32]          36,864
      BatchNorm2d-12           [-1, 64, 32, 32]             128
             ReLU-13           [-1, 64, 32, 32]               0
       BasicBlock-14           [-1, 64, 32, 32]               0
           Conv2d-15           [-1, 64, 32, 32]          36,864
      BatchNorm2d-16           [-1, 64, 32, 32]             128
```

```
              ReLU-17           [-1, 64, 32, 32]               0
            Conv2d-18           [-1, 64, 32, 32]          36,864
       BatchNorm2d-19           [-1, 64, 32, 32]             128
              ReLU-20           [-1, 64, 32, 32]               0
        BasicBlock-21           [-1, 64, 32, 32]               0
            Conv2d-22          [-1, 128, 16, 16]          73,728
       BatchNorm2d-23          [-1, 128, 16, 16]             256
              ReLU-24          [-1, 128, 16, 16]               0
            Conv2d-25          [-1, 128, 16, 16]         147,456
       BatchNorm2d-26          [-1, 128, 16, 16]             256
            Conv2d-27          [-1, 128, 16, 16]           8,192
       BatchNorm2d-28          [-1, 128, 16, 16]             256
              ReLU-29          [-1, 128, 16, 16]               0
        BasicBlock-30          [-1, 128, 16, 16]               0
            Conv2d-31          [-1, 128, 16, 16]         147,456
       BatchNorm2d-32          [-1, 128, 16, 16]             256
              ReLU-33          [-1, 128, 16, 16]               0
            Conv2d-34          [-1, 128, 16, 16]         147,456
       BatchNorm2d-35          [-1, 128, 16, 16]             256
              ReLU-36          [-1, 128, 16, 16]               0
        BasicBlock-37          [-1, 128, 16, 16]               0
            Conv2d-38            [-1, 256, 8, 8]         294,912
       BatchNorm2d-39            [-1, 256, 8, 8]             512
              ReLU-40            [-1, 256, 8, 8]               0
            Conv2d-41            [-1, 256, 8, 8]         589,824
       BatchNorm2d-42            [-1, 256, 8, 8]             512
            Conv2d-43            [-1, 256, 8, 8]          32,768
       BatchNorm2d-44            [-1, 256, 8, 8]             512
              ReLU-45            [-1, 256, 8, 8]               0
        BasicBlock-46            [-1, 256, 8, 8]               0
            Conv2d-47            [-1, 256, 8, 8]         589,824
       BatchNorm2d-48            [-1, 256, 8, 8]             512
              ReLU-49            [-1, 256, 8, 8]               0
            Conv2d-50            [-1, 256, 8, 8]         589,824
       BatchNorm2d-51            [-1, 256, 8, 8]             512
              ReLU-52            [-1, 256, 8, 8]               0
        BasicBlock-53            [-1, 256, 8, 8]               0
          Identity-54            [-1, 256, 8, 8]               0
AdaptiveAvgPool2d-55            [-1, 256, 1, 1]               0
            Linear-56                  [-1, 2]             514
================================================================
Total params: 2,812,034
Trainable params: 2,812,034
Non-trainable params: 0
----------------------------------------------------------------
Input size (MB): 0.01
Forward/backward pass size (MB): 16.63
Params size (MB): 10.73
```

```
Estimated Total Size (MB): 27.36
----------------------------------------------------------------

train_loader, test_loader = get_dataloaders(batch_size=512)
criterion = nn.BCEWithLogitsLoss()
```

Evaluation

```
test_acc, test_loss = evaluate(model, criterion, test_loader, device)
print('Test accuracy: ', test_acc*100, '%')
print('Test loss: ', test_loss)

Test accuracy:  50.43473895582329 %
Test loss:  1.015372067187206
```

Adam Optimizer with 20 Epochs

```
model2 = resnet15v2()
model2.load_state_dict(torch.load('expt_20_adam/best.pth',
map_location=device))
model2 = model2.to(device)
```

Evaluation

```
test_acc2, test_loss2 = evaluate(model2, criterion, test_loader,
device)
print('Test accuracy: ', test_acc2*100, '%')
print('Test loss: ', test_loss2)

Test accuracy:  75.47791164658635 %
Test loss:  0.5066263087111783
```

AdamW optimizer with 20 Epochs

```
model3 = resnet15v2()
model3.load_state_dict(torch.load('expt_20_adamw/best.pth',
map_location=device))
model3 = model3.to(device)
```

Evaluation

```
test_acc3, test_loss3 = evaluate(model3, criterion, test_loader,
device)
print('Test accuracy: ', test_acc3*100, '%')
print('Test loss: ', test_loss3)
```

```
Test accuracy:  74.68072289156626 %
Test loss:  0.5188913260885032
```

Plot

```
import matplotlib.pyplot as plt

test_accs = [test_acc * 100, test_acc2 * 100, test_acc3 * 100]
test_losses = [test_loss, test_loss2, test_loss3]
model_names = ['expt_50_adam', 'expt_20_adam', 'expt_20_adamw']

fig, ax1 = plt.subplots(figsize=(8,6))

color_acc = 'tab:blue'
ax1.set_xlabel('Model')
ax1.set_ylabel('Test Accuracy (%)', color=color_acc)
bars = ax1.bar(model_names, test_accs, color=color_acc, alpha=0.6)
ax1.tick_params(axis='y', labelcolor=color_acc)
ax1.set_ylim(0, 100)

for bar in bars:
    height = bar.get_height()
    ax1.annotate(f'{height:.1f}%',
                 xy=(bar.get_x() + bar.get_width() / 2, height),
                 xytext=(0, 3),
                 textcoords="offset points",
                 ha='center', va='bottom')

ax2 = ax1.twinx()
color_loss = 'tab:red'
ax2.set_ylabel('Test Loss', color=color_loss)
ax2.plot(model_names, test_losses, color=color_loss, marker='o',
linestyle='--', linewidth=2)
ax2.tick_params(axis='y', labelcolor=color_loss)

plt.title('Comparison of Test Accuracy and Loss for Three Models')
plt.show()
```


Comparison of Test Accuracy and Loss for Three Models
Test Accuracy (%)
Test Loss
100
80
60
40
20
0
1.0
0.9
0.8
0.7
0.6
0.5
50.4%
75.5%
74.7%
expt_50_adam
expt_20_adam
expt_20_adamw
Model