```
class Model(nn.Module):
    def __init__(self):
        super(Model, self).__init__()
        self.linear1 = nn.Linear(135, 256)
        self.linear2 = nn.Linear(256, 128)
        self.linear3 = nn.Linear(128, 64)
        self.linear4 = nn.Linear(64, 32)
        self.linear5 = nn.Linear(32, 4)

    def forward(self, x):
        x = x.view(-1, 135)
        x = F.relu(self.linear1(x))
        x = F.relu(self.linear2(x))
        x = F.relu(self.linear3(x))
        x = F.relu(self.linear4(x))
        return self.linear5(x)

model = Model()
```

```
[79] class Model(nn.Module):
         def __init__(self):
             super(Model, self).__init__()
             self.linear1 = nn.Linear(135, 256)
             self.linear2 = nn.Linear(256, 128)
             self.linear3 = nn.Linear(128, 64)
             self.linear4 = nn.Linear(64, 32)
             self.linear5 = nn.Linear(32, 4)

         def forward(self, x):
             x = x.view(-1, 135)
             x = F.relu(self.linear1(x))
             x = F.relu(self.linear2(x))
             x = F.relu(self.linear3(x))
             x = F.relu(self.linear4(x))
             return self.linear5(x)

     model = Model()
```

```
[80] criterion = nn.CrossEntropyLoss()
     optimizer = torch.optim.Adam(model.parameters(), lr=0.001)
     scheduler = torch.optim.lr_scheduler.StepLR(optimizer, step_size=5, gamma=0.5)
```

```
Epoch 48, training accuracy: 85.375%
Finished Training
Finished Testing
[49,    10] loss: 0.036
[49,    20] loss: 0.037
[49,    30] loss: 0.038
[49,    40] loss: 0.033
[49,    50] loss: 0.040
[49,    60] loss: 0.039
Epoch 49, training accuracy: 85.475%
Finished Training
Finished Testing
[50,    10] loss: 0.035
[50,    20] loss: 0.037
[50,    30] loss: 0.037
[50,    40] loss: 0.038
[50,    50] loss: 0.040
[50,    60] loss: 0.038
Epoch 50, training accuracy: 85.400%
Finished Training
Finished Testing
```

```
class Model(nn.Module):
    def __init__(self):
        super(Model, self).__init__()
        self.linear1 = nn.Linear(135, 256)
        self.bn1 = nn.BatchNorm1d(256)
        self.dropout1 = nn.Dropout(0.5)
        self.linear2 = nn.Linear(256, 128)
        self.bn2 = nn.BatchNorm1d(128)
        self.dropout2 = nn.Dropout(0.3)
        self.linear3 = nn.Linear(128, 64)
        self.bn3 = nn.BatchNorm1d(64)
        self.dropout3 = nn.Dropout(0.1)
        self.linear4 = nn.Linear(64, 32)
        self.bn4 = nn.BatchNorm1d(32)
        self.dropout4 = nn.Dropout(0.1)
        self.linear5 = nn.Linear(32, 4)

    def forward(self, x):
        x = x.view(-1, 135)
        x = self.dropout1(F.relu(self.bn1(self.linear1(x))))
        x = self.dropout2(F.relu(self.bn2(self.linear2(x))))
        x = self.dropout3(F.relu(self.bn3(self.linear3(x))))
        x = self.dropout4(F.relu(self.bn4(self.linear4(x))))
        return self.linear5(x)

model = Model()
```

```
[91] class Model(nn.Module):
         def __init__(self):
             super(Model, self).__init__()
             self.linear1 = nn.Linear(135, 256)
             self.bn1 = nn.BatchNorm1d(256)
             self.dropout1 = nn.Dropout(0.5)
             self.linear2 = nn.Linear(256, 128)
             self.bn2 = nn.BatchNorm1d(128)
             self.dropout2 = nn.Dropout(0.3)
             self.linear3 = nn.Linear(128, 64)
             self.bn3 = nn.BatchNorm1d(64)
             self.dropout3 = nn.Dropout(0.1)
             self.linear4 = nn.Linear(64, 32)
             self.bn4 = nn.BatchNorm1d(32)
             self.dropout4 = nn.Dropout(0.1)
             self.linear5 = nn.Linear(32, 4)

         def forward(self, x):
             x = x.view(-1, 135)
             x = self.dropout1(F.relu(self.bn1(self.linear1(x))))
             x = self.dropout2(F.relu(self.bn2(self.linear2(x))))
             x = self.dropout3(F.relu(self.bn3(self.linear3(x))))
             x = self.dropout4(F.relu(self.bn4(self.linear4(x))))
             return self.linear5(x)

model = Model()
```

```
[92] criterion = nn.CrossEntropyLoss()
     optimizer = torch.optim.Adam(model.parameters(), lr=0.001)
     scheduler = torch.optim.lr_scheduler.StepLR(optimizer, step_size=5, gamma=0.5)
```

```
Epoch 48, training accuracy: 91.525%
Finished Training
Finished Testing
[49,    10] loss: 0.021
[49,    20] loss: 0.025
[49,    30] loss: 0.023
[49,    40] loss: 0.024
[49,    50] loss: 0.020
[49,    60] loss: 0.024
Epoch 49, training accuracy: 91.550%
Finished Training
Finished Testing
[50,    10] loss: 0.021
[50,    20] loss: 0.024
[50,    30] loss: 0.024
[50,    40] loss: 0.024
[50,    50] loss: 0.024
[50,    60] loss: 0.019
Epoch 50, training accuracy: 91.500%
Finished Training
Finished Testing
```