# Lab 9

**Aim:** To implement VGG CNN Architecture

```python
import torch
import torch.nn as nn
import torch.nn.functional as F
import torch.optim as optim
import torch.autograd
import torch.utils.data
import torchvision
import torchvision.datasets as dataset
import torchvision.transforms as T
import matplotlib.pyplot as plt
import numpy as np

from tqdm import tqdm
from torch.utils.data import Dataset
from torch.utils.data import DataLoader

RANDOM_SEED = 42
torch.manual_seed(RANDOM_SEED)
torch.cuda.manual_seed(RANDOM_SEED)
torch.cuda.manual_seed_all(RANDOM_SEED)
torch.backends.cudnn.deterministic = True
torch.backends.cudnn.benchmark = False
```

## Dataset

```python
train_dataset = dataset.CIFAR100(root="./CIFAR100/train", train=True,
transform=None, download=True)
```

```
Downloading https://www.cs.toronto.edu/~kriz/cifar-100-python.tar.gz
to ./CIFAR100/train/cifar-100-python.tar.gz
```

```
100%|██████████| 169001437/169001437 [00:04<00:00, 40635766.53it/s]
```

```
Extracting ./CIFAR100/train/cifar-100-python.tar.gz to
./CIFAR100/train
```

```python
x = np.concatenate([np.asarray(train_dataset[i][0]) for i in
range(len(train_dataset))])
mean = np.mean(x, axis=(0, 1))/255
std = np.std(x, axis=(0,1))/255

mean = mean.tolist()
std = std.tolist()
```

```
print("mean:", mean)
print("std:", std)
```

```
mean: [0.5070751592371323, 0.48654887331495095, 0.4409178433670343]
std: [0.2673342858794185, 0.2564384629212061, 0.2761504713263903]
```

```
transform = T.Compose([T.ToTensor(),
                       T.Normalize(mean, std, inplace=True)])
train_dataset = dataset.CIFAR100(root="./CIFAR100/train", train=True,
transform=transform, download=True)
valid_dataset = dataset.CIFAR100(root="./CIFAR100/val", train=True,
transform=transform, download=True)
test_dataset = dataset.CIFAR100(root="./CIFAR100/test", train=False,
transform=transform, download=True)
```

```
Files already downloaded and verified
Downloading https://www.cs.toronto.edu/~kriz/cifar-100-python.tar.gz
to ./CIFAR100/val/cifar-100-python.tar.gz
```

```
100%|██████████| 169001437/169001437 [00:04<00:00, 41396572.41it/s]
```

```
Extracting ./CIFAR100/val/cifar-100-python.tar.gz to ./CIFAR100/val
Downloading https://www.cs.toronto.edu/~kriz/cifar-100-python.tar.gz
to ./CIFAR100/test/cifar-100-python.tar.gz
```

```
100%|██████████| 169001437/169001437 [00:03<00:00, 43570322.81it/s]
```

```
Extracting ./CIFAR100/test/cifar-100-python.tar.gz to ./CIFAR100/test
```

```
train_loader = DataLoader(train_dataset, batch_size=256, shuffle=True,
num_workers=8)
valid_loader = DataLoader(valid_dataset, batch_size=256,
shuffle=False, num_workers=8)
test_loader = DataLoader(test_dataset, batch_size=256, shuffle=False,
num_workers=8)
```

```
/usr/local/lib/python3.10/dist-packages/torch/utils/data/
dataloader.py:558: UserWarning: This DataLoader will create 8 worker
processes in total. Our suggested max number of worker in current
system is 2, which is smaller than what this DataLoader is going to
create. Please be aware that excessive worker creation might get
DataLoader running slow or even freeze, lower the worker number to
avoid potential slowness/freeze if necessary.
  warnings.warn(_create_warning_msg(
```

```
images, labels = next(iter(train_loader))
```

```
print(images.shape)
print(labels.shape)
print(labels)
```

```
torch.Size([256, 3, 32, 32])
torch.Size([256])
tensor([27, 10, 73, 51, 53, 82, 21, 44, 47,  5, 94, 57, 17, 94, 92,
91, 53, 76,
        45, 46, 52, 30, 30, 85, 39, 70,  2,  6, 30, 58, 36, 67, 61,
38, 28, 10,
        61, 46, 43, 61, 27, 22, 86, 11, 43, 87, 78, 47, 97,  2, 94,
16, 79, 92,
        84,  8, 91, 35, 61, 55, 75, 42, 60, 49, 41, 96, 46, 97, 37,
34, 58,  4,
        89, 51, 92, 62, 66, 40, 95, 26, 31, 86, 33, 82, 58, 14, 68,
26, 96, 23,
        87, 82, 15, 65, 26,  3,  1, 92, 60, 32, 16, 44, 20, 86, 36,
56, 99, 61,
        72, 46, 91, 84, 89, 30, 37,  0, 69, 68, 58, 11, 46, 41, 98,
46, 15,  2,
        21, 37, 59, 79, 88, 71, 65, 70, 60, 21, 62, 92, 40, 29,  6,
48, 40, 66,
        90, 98, 44, 64, 46, 13,  6,  6, 18, 80, 33, 76, 27, 45, 11,
61, 59, 96,
        60, 25, 80, 66, 48, 87, 24,  1, 35, 56,  2, 30,  3,  7, 29,
47, 81, 23,
         2, 91, 12,  3, 40, 15, 70, 68, 25,  3, 31, 73, 10, 87, 85,
45, 74, 58,
         7, 56, 20, 84, 93, 34, 15, 45,  4, 82, 61, 42, 10, 82, 91,
46, 58, 99,
        74,  6, 43, 99, 21, 23, 37, 89, 35, 17, 32, 16, 88, 21, 14,
38, 28, 27,
        79, 91, 90, 99, 23, 15, 35, 98, 65, 12, 61, 21, 56, 20, 80,
28, 35, 41,
        31, 87,  7, 68])
```

# Model

```python
def conv_block(in_channels, out_channels):
    return nn.Sequential(nn.Conv2d(in_channels=in_channels,
out_channels=out_channels, kernel_size=3, stride=1, padding=1),
                         nn.BatchNorm2d(out_channels),
                         nn.ReLU())

class VGG19(nn.Module):
    def __init__(self, in_channels, out_channels):
        super().__init__()
        self.in_channels = in_channels
        self.out_channels = out_channels

        self.conv_block1 =
nn.Sequential(nn.Conv2d(in_channels=self.in_channels, out_channels=64,
kernel_size=3, stride=1, padding=1),
                                         nn.BatchNorm2d(64),
                                         nn.ReLU(),
                                         conv_block(64, 64),
                                         nn.MaxPool2d(kernel_size=2,
stride=2))

        self.conv_block2 = nn.Sequential(nn.Conv2d(in_channels=64,
out_channels=128, kernel_size=3, stride=1, padding=1),
                                         nn.BatchNorm2d(128),
                                         nn.ReLU(),
                                         conv_block(128, 128),
                                         nn.MaxPool2d(kernel_size=2,
stride=2))

        self.conv_block3 = nn.Sequential(nn.Conv2d(in_channels=128,
out_channels=256, kernel_size=3, stride=1, padding=1),
                                         nn.BatchNorm2d(256),
                                         nn.ReLU(),
                                         *[conv_block(256, 256) for _
in range(3)],
                                         nn.MaxPool2d(kernel_size=2,
stride=2))

        self.conv_block4 = nn.Sequential(nn.Conv2d(in_channels=256,
out_channels=512, kernel_size=3, stride=1, padding=1),
                                         nn.BatchNorm2d(512),
                                         nn.ReLU(),
                                         *[conv_block(512, 512) for _
in range(3)],
                                         nn.MaxPool2d(kernel_size=2,
stride=2))

        self.conv_block5 = nn.Sequential(*[conv_block(512, 512) for _
in range(4)],
                                         nn.MaxPool2d(kernel_size=2,
stride=2))

        self.avg_pool = nn.AdaptiveAvgPool2d(output_size=(7,7))

        self.linear1 = nn.Sequential(nn.Linear(in_features=7*7*512,
out_features=4096, bias=True),
                                     nn.Dropout(0.5),
                                     nn.ReLU())
        self.linear2 = nn.Sequential(nn.Linear(in_features=4096,
out_features=4096, bias=True),
                                     nn.Dropout(0.5),
                                     nn.ReLU())
        self.linear3 = nn.Linear(in_features=4096,
out_features=self.out_channels, bias=True)
```

```
    def forward(self, x):
        x = self.conv_block1(x)
        x = self.conv_block2(x)
        x = self.conv_block3(x)
        x = self.conv_block4(x)
        x = self.conv_block5(x)
        x = self.avg_pool(x)
        x = self.linear1(x.view(x.shape[0], -1))
        x = self.linear2(x)
        x = self.linear3(x)
        return x
```

## Train

```
model = VGG19(in_channels=3, out_channels=100)
optimizer = optim.AdamW(model.parameters(), lr=0.0001,
weight_decay=0.005)
criterion = nn.CrossEntropyLoss()
total_epochs = 50
device = 'cuda' if torch.cuda.is_available() else 'cpu'
model = model.to(device)

def train(model, optimizer, train_loader, val_loader, criterion,
total_epochs):
    print("Training Begin!")
    print()
    best_accuracy = 0

    for epoch in range(total_epochs):
        model.train()

        for step, (images, labels) in enumerate(train_loader):
            images, labels = images.to(device, non_blocking=True),
labels.to(device, non_blocking=True)

            optimizer.zero_grad(set_to_none=True)
            outputs = model(images)
            loss = criterion(outputs, labels)
            loss.backward()
            optimizer.step()

            if (step+1) % 30 == 0:
              print('Epoch: [{}/{}] | Step: [{}/{}] | Loss:
{:.4f}'.format(epoch+1, total_epochs, step+1, len(train_loader),
loss.item()))

        with torch.no_grad():
            print("Validating...")
            model.eval()
            total = 0
            correct = 0
            for (images, labels) in val_loader:
                images, labels = images.to(device, non_blocking=True),
labels.to(device, non_blocking=True)

                outputs = model(images)
                _, predicted = torch.max(outputs, dim=1)
                total += labels.size(0)
                correct += (predicted == labels).sum().item()

            acc = (correct/total)*100
            if acc > best_accuracy:
                best_accuracy = acc
                torch.save(model.state_dict(), "./model_cbam2.pt")
            print(f"Current Accuracy: {acc:.3f}%")
            print(f"Best Accuracy: {best_accuracy:.3f}%")
            print()

    print('Train Finished!')

train(model=model, optimizer=optimizer, train_loader=train_loader,
val_loader=valid_loader, criterion=criterion,
total_epochs=total_epochs)
```

```
Training Begin!

Epoch: [1/50] | Step: [30/196] | Loss: 4.5930
Epoch: [1/50] | Step: [60/196] | Loss: 4.5581
Epoch: [1/50] | Step: [90/196] | Loss: 4.3560
Epoch: [1/50] | Step: [120/196] | Loss: 4.3529
Epoch: [1/50] | Step: [150/196] | Loss: 4.2601
Epoch: [1/50] | Step: [180/196] | Loss: 4.2955
Validating...
Current Accuracy: 3.092%
Best Accuracy: 3.092%

Epoch: [2/50] | Step: [30/196] | Loss: 4.1185
Epoch: [2/50] | Step: [60/196] | Loss: 4.1673
Epoch: [2/50] | Step: [90/196] | Loss: 3.9969
Epoch: [2/50] | Step: [120/196] | Loss: 3.9640
Epoch: [2/50] | Step: [150/196] | Loss: 3.8409
Epoch: [2/50] | Step: [180/196] | Loss: 3.7740
Validating...
Current Accuracy: 7.658%
Best Accuracy: 7.658%

Epoch: [3/50] | Step: [30/196] | Loss: 3.7485
Epoch: [3/50] | Step: [60/196] | Loss: 3.7498
Epoch: [3/50] | Step: [90/196] | Loss: 3.4678
```

```
Epoch: [3/50] | Step: [120/196] | Loss: 3.5256
Epoch: [3/50] | Step: [150/196] | Loss: 3.4212
Epoch: [3/50] | Step: [180/196] | Loss: 3.5045
Validating...
Current Accuracy: 11.292%
Best Accuracy: 11.292%

Epoch: [4/50] | Step: [30/196] | Loss: 3.4148
Epoch: [4/50] | Step: [60/196] | Loss: 3.4541
Epoch: [4/50] | Step: [90/196] | Loss: 3.3348
Epoch: [4/50] | Step: [120/196] | Loss: 3.3435
Epoch: [4/50] | Step: [150/196] | Loss: 3.2696
Epoch: [4/50] | Step: [180/196] | Loss: 3.2548
Validating...
Current Accuracy: 17.736%
Best Accuracy: 17.736%

Epoch: [5/50] | Step: [30/196] | Loss: 3.0753
Epoch: [5/50] | Step: [60/196] | Loss: 2.9494
Epoch: [5/50] | Step: [90/196] | Loss: 3.1305
Epoch: [5/50] | Step: [120/196] | Loss: 3.1097
Epoch: [5/50] | Step: [150/196] | Loss: 3.0452
Epoch: [5/50] | Step: [180/196] | Loss: 3.0318
Validating...
Current Accuracy: 23.816%
Best Accuracy: 23.816%

Epoch: [6/50] | Step: [30/196] | Loss: 2.5738
Epoch: [6/50] | Step: [60/196] | Loss: 2.9318
Epoch: [6/50] | Step: [90/196] | Loss: 2.6363
Epoch: [6/50] | Step: [120/196] | Loss: 2.9299
Epoch: [6/50] | Step: [150/196] | Loss: 2.5706
Epoch: [6/50] | Step: [180/196] | Loss: 2.6794
Validating...
Current Accuracy: 30.328%
Best Accuracy: 30.328%

Epoch: [7/50] | Step: [30/196] | Loss: 2.5473
Epoch: [7/50] | Step: [60/196] | Loss: 2.5591
Epoch: [7/50] | Step: [90/196] | Loss: 2.4478
Epoch: [7/50] | Step: [120/196] | Loss: 2.4556
Epoch: [7/50] | Step: [150/196] | Loss: 2.5954
Epoch: [7/50] | Step: [180/196] | Loss: 2.4507
Validating...
Current Accuracy: 33.846%
Best Accuracy: 33.846%

Epoch: [8/50] | Step: [30/196] | Loss: 2.2367
Epoch: [8/50] | Step: [60/196] | Loss: 2.2316
Epoch: [8/50] | Step: [90/196] | Loss: 2.1627
```

```
Epoch: [8/50] | Step: [120/196] | Loss: 2.3989
Epoch: [8/50] | Step: [150/196] | Loss: 2.3586
Epoch: [8/50] | Step: [180/196] | Loss: 2.4116
Validating...
Current Accuracy: 40.336%
Best Accuracy: 40.336%

Epoch: [9/50] | Step: [30/196] | Loss: 2.0321
Epoch: [9/50] | Step: [60/196] | Loss: 1.9519
Epoch: [9/50] | Step: [90/196] | Loss: 1.9390
Epoch: [9/50] | Step: [120/196] | Loss: 2.0770
Epoch: [9/50] | Step: [150/196] | Loss: 1.8549
Epoch: [9/50] | Step: [180/196] | Loss: 2.0955
Validating...
Current Accuracy: 47.420%
Best Accuracy: 47.420%

Epoch: [10/50] | Step: [30/196] | Loss: 1.6944
Epoch: [10/50] | Step: [60/196] | Loss: 1.8360
Epoch: [10/50] | Step: [90/196] | Loss: 1.8761
Epoch: [10/50] | Step: [120/196] | Loss: 1.8523
Epoch: [10/50] | Step: [150/196] | Loss: 1.8681
Epoch: [10/50] | Step: [180/196] | Loss: 1.8336
Validating...
Current Accuracy: 49.252%
Best Accuracy: 49.252%

Epoch: [11/50] | Step: [30/196] | Loss: 1.6616
Epoch: [11/50] | Step: [60/196] | Loss: 1.4129
Epoch: [11/50] | Step: [90/196] | Loss: 1.5194
Epoch: [11/50] | Step: [120/196] | Loss: 1.5640
Epoch: [11/50] | Step: [150/196] | Loss: 1.7214
Epoch: [11/50] | Step: [180/196] | Loss: 1.5595
Validating...
Current Accuracy: 54.294%
Best Accuracy: 54.294%

Epoch: [12/50] | Step: [30/196] | Loss: 1.3071
Epoch: [12/50] | Step: [60/196] | Loss: 1.3492
Epoch: [12/50] | Step: [90/196] | Loss: 1.4383
Epoch: [12/50] | Step: [120/196] | Loss: 1.3648
Epoch: [12/50] | Step: [150/196] | Loss: 1.6284
Epoch: [12/50] | Step: [180/196] | Loss: 1.3499
Validating...
Current Accuracy: 56.308%
Best Accuracy: 56.308%

Epoch: [13/50] | Step: [30/196] | Loss: 1.1345
Epoch: [13/50] | Step: [60/196] | Loss: 1.1936
Epoch: [13/50] | Step: [90/196] | Loss: 1.3609
```

```
Epoch: [13/50] | Step: [120/196] | Loss: 1.2013
Epoch: [13/50] | Step: [150/196] | Loss: 1.4292
Epoch: [13/50] | Step: [180/196] | Loss: 1.4351
Validating...
Current Accuracy: 63.114%
Best Accuracy: 63.114%

Epoch: [14/50] | Step: [30/196] | Loss: 0.9927
Epoch: [14/50] | Step: [60/196] | Loss: 1.0132
Epoch: [14/50] | Step: [90/196] | Loss: 1.1422
Epoch: [14/50] | Step: [120/196] | Loss: 1.2032
Epoch: [14/50] | Step: [150/196] | Loss: 1.0971
Epoch: [14/50] | Step: [180/196] | Loss: 1.2577
Validating...
Current Accuracy: 62.154%
Best Accuracy: 63.114%

Epoch: [15/50] | Step: [30/196] | Loss: 0.9990
Epoch: [15/50] | Step: [60/196] | Loss: 0.8528
Epoch: [15/50] | Step: [90/196] | Loss: 0.8496
Epoch: [15/50] | Step: [120/196] | Loss: 1.0253
Epoch: [15/50] | Step: [150/196] | Loss: 0.9815
Epoch: [15/50] | Step: [180/196] | Loss: 0.8002
Validating...
Current Accuracy: 73.892%
Best Accuracy: 73.892%

Epoch: [16/50] | Step: [30/196] | Loss: 0.8030
Epoch: [16/50] | Step: [60/196] | Loss: 0.7382
Epoch: [16/50] | Step: [90/196] | Loss: 0.8101
Epoch: [16/50] | Step: [120/196] | Loss: 0.9341
Epoch: [16/50] | Step: [150/196] | Loss: 0.8127
Epoch: [16/50] | Step: [180/196] | Loss: 0.9173
Validating...
Current Accuracy: 74.608%
Best Accuracy: 74.608%

Epoch: [17/50] | Step: [30/196] | Loss: 0.8292
Epoch: [17/50] | Step: [60/196] | Loss: 0.7111
Epoch: [17/50] | Step: [90/196] | Loss: 0.8565
Epoch: [17/50] | Step: [120/196] | Loss: 0.7227
Epoch: [17/50] | Step: [150/196] | Loss: 0.7066
Epoch: [17/50] | Step: [180/196] | Loss: 0.7397
Validating...
Current Accuracy: 70.316%
Best Accuracy: 74.608%

Epoch: [18/50] | Step: [30/196] | Loss: 0.7062
Epoch: [18/50] | Step: [60/196] | Loss: 0.6562
Epoch: [18/50] | Step: [90/196] | Loss: 0.6392
```

```
Epoch: [18/50] | Step: [120/196] | Loss: 0.6193
Epoch: [18/50] | Step: [150/196] | Loss: 0.6725
Epoch: [18/50] | Step: [180/196] | Loss: 0.7229
Validating...
Current Accuracy: 74.760%
Best Accuracy: 74.760%

Epoch: [19/50] | Step: [30/196] | Loss: 0.8134
Epoch: [19/50] | Step: [60/196] | Loss: 0.5322
Epoch: [19/50] | Step: [90/196] | Loss: 0.5105
Epoch: [19/50] | Step: [120/196] | Loss: 0.5524
Epoch: [19/50] | Step: [150/196] | Loss: 0.5655
Epoch: [19/50] | Step: [180/196] | Loss: 0.6718
Validating...
Current Accuracy: 82.708%
Best Accuracy: 82.708%

Epoch: [20/50] | Step: [30/196] | Loss: 0.4970
Epoch: [20/50] | Step: [60/196] | Loss: 0.4336
Epoch: [20/50] | Step: [90/196] | Loss: 0.4839
Epoch: [20/50] | Step: [120/196] | Loss: 0.6065
Epoch: [20/50] | Step: [150/196] | Loss: 0.5300
Epoch: [20/50] | Step: [180/196] | Loss: 0.6512
Validating...
Current Accuracy: 86.148%
Best Accuracy: 86.148%

Epoch: [21/50] | Step: [30/196] | Loss: 0.3947
Epoch: [21/50] | Step: [60/196] | Loss: 0.4778
Epoch: [21/50] | Step: [90/196] | Loss: 0.4892
Epoch: [21/50] | Step: [120/196] | Loss: 0.5408
Epoch: [21/50] | Step: [150/196] | Loss: 0.4484
Epoch: [21/50] | Step: [180/196] | Loss: 0.4925
Validating...
Current Accuracy: 86.652%
Best Accuracy: 86.652%

Epoch: [22/50] | Step: [30/196] | Loss: 0.3742
Epoch: [22/50] | Step: [60/196] | Loss: 0.4110
Epoch: [22/50] | Step: [90/196] | Loss: 0.4757
Epoch: [22/50] | Step: [120/196] | Loss: 0.4671
Epoch: [22/50] | Step: [150/196] | Loss: 0.4406
Epoch: [22/50] | Step: [180/196] | Loss: 0.5724
Validating...
Current Accuracy: 89.318%
Best Accuracy: 89.318%

Epoch: [23/50] | Step: [30/196] | Loss: 0.4376
Epoch: [23/50] | Step: [60/196] | Loss: 0.3803
Epoch: [23/50] | Step: [90/196] | Loss: 0.3599
```

```
Epoch: [23/50] | Step: [120/196] | Loss: 0.6157
Epoch: [23/50] | Step: [150/196] | Loss: 0.3926
Epoch: [23/50] | Step: [180/196] | Loss: 0.4734
Validating...
Current Accuracy: 90.442%
Best Accuracy: 90.442%

Epoch: [24/50] | Step: [30/196] | Loss: 0.5321
Epoch: [24/50] | Step: [60/196] | Loss: 0.2890
Epoch: [24/50] | Step: [90/196] | Loss: 0.3901
Epoch: [24/50] | Step: [120/196] | Loss: 0.4156
Epoch: [24/50] | Step: [150/196] | Loss: 0.5370
Epoch: [24/50] | Step: [180/196] | Loss: 0.3055
Validating...
Current Accuracy: 91.768%
Best Accuracy: 91.768%

Epoch: [25/50] | Step: [30/196] | Loss: 0.2887
Epoch: [25/50] | Step: [60/196] | Loss: 0.4579
Epoch: [25/50] | Step: [90/196] | Loss: 0.3514
Epoch: [25/50] | Step: [120/196] | Loss: 0.4069
Epoch: [25/50] | Step: [150/196] | Loss: 0.3859
Epoch: [25/50] | Step: [180/196] | Loss: 0.3585
Validating...
Current Accuracy: 89.626%
Best Accuracy: 91.768%

Epoch: [26/50] | Step: [30/196] | Loss: 0.3417
Epoch: [26/50] | Step: [60/196] | Loss: 0.2992
Epoch: [26/50] | Step: [90/196] | Loss: 0.2768
Epoch: [26/50] | Step: [120/196] | Loss: 0.2982
Epoch: [26/50] | Step: [150/196] | Loss: 0.5168
Epoch: [26/50] | Step: [180/196] | Loss: 0.3348
Validating...
Current Accuracy: 92.560%
Best Accuracy: 92.560%

Epoch: [27/50] | Step: [30/196] | Loss: 0.2318
Epoch: [27/50] | Step: [60/196] | Loss: 0.3046
Epoch: [27/50] | Step: [90/196] | Loss: 0.4271
Epoch: [27/50] | Step: [120/196] | Loss: 0.3822
Epoch: [27/50] | Step: [150/196] | Loss: 0.2386
Epoch: [27/50] | Step: [180/196] | Loss: 0.3277
Validating...
Current Accuracy: 89.374%
Best Accuracy: 92.560%

Epoch: [28/50] | Step: [30/196] | Loss: 0.3014
Epoch: [28/50] | Step: [60/196] | Loss: 0.2722
Epoch: [28/50] | Step: [90/196] | Loss: 0.2216
```

```
Epoch: [28/50] | Step: [120/196] | Loss: 0.2102
Epoch: [28/50] | Step: [150/196] | Loss: 0.3206
Epoch: [28/50] | Step: [180/196] | Loss: 0.3580
Validating...
Current Accuracy: 91.962%
Best Accuracy: 92.560%

Epoch: [29/50] | Step: [30/196] | Loss: 0.3100
Epoch: [29/50] | Step: [60/196] | Loss: 0.2512
Epoch: [29/50] | Step: [90/196] | Loss: 0.3254
Epoch: [29/50] | Step: [120/196] | Loss: 0.2671
Epoch: [29/50] | Step: [150/196] | Loss: 0.2881
Epoch: [29/50] | Step: [180/196] | Loss: 0.4774
Validating...
Current Accuracy: 89.032%
Best Accuracy: 92.560%

Epoch: [30/50] | Step: [30/196] | Loss: 0.1822
Epoch: [30/50] | Step: [60/196] | Loss: 0.2318
Epoch: [30/50] | Step: [90/196] | Loss: 0.2616
Epoch: [30/50] | Step: [120/196] | Loss: 0.2370
Epoch: [30/50] | Step: [150/196] | Loss: 0.1879
Epoch: [30/50] | Step: [180/196] | Loss: 0.2839
Validating...
Current Accuracy: 93.590%
Best Accuracy: 93.590%

Epoch: [31/50] | Step: [30/196] | Loss: 0.2704
Epoch: [31/50] | Step: [60/196] | Loss: 0.3580
Epoch: [31/50] | Step: [90/196] | Loss: 0.2806
Epoch: [31/50] | Step: [120/196] | Loss: 0.1895
Epoch: [31/50] | Step: [150/196] | Loss: 0.3381
Epoch: [31/50] | Step: [180/196] | Loss: 0.2374
Validating...
Current Accuracy: 90.160%
Best Accuracy: 93.590%

Epoch: [32/50] | Step: [30/196] | Loss: 0.2674
Epoch: [32/50] | Step: [60/196] | Loss: 0.1780
Epoch: [32/50] | Step: [90/196] | Loss: 0.2894
Epoch: [32/50] | Step: [120/196] | Loss: 0.2333
Epoch: [32/50] | Step: [150/196] | Loss: 0.3325
Epoch: [32/50] | Step: [180/196] | Loss: 0.1819
Validating...
Current Accuracy: 92.238%
Best Accuracy: 93.590%

Epoch: [33/50] | Step: [30/196] | Loss: 0.1583
Epoch: [33/50] | Step: [60/196] | Loss: 0.3318
Epoch: [33/50] | Step: [90/196] | Loss: 0.2081
```

```
Epoch: [33/50] | Step: [120/196] | Loss: 0.2041
Epoch: [33/50] | Step: [150/196] | Loss: 0.2378
Epoch: [33/50] | Step: [180/196] | Loss: 0.1705
Validating...
Current Accuracy: 95.162%
Best Accuracy: 95.162%

Epoch: [34/50] | Step: [30/196] | Loss: 0.2039
Epoch: [34/50] | Step: [60/196] | Loss: 0.2023
Epoch: [34/50] | Step: [90/196] | Loss: 0.2905
Epoch: [34/50] | Step: [120/196] | Loss: 0.2381
Epoch: [34/50] | Step: [150/196] | Loss: 0.1624
Epoch: [34/50] | Step: [180/196] | Loss: 0.2450
Validating...
Current Accuracy: 93.312%
Best Accuracy: 95.162%

Epoch: [35/50] | Step: [30/196] | Loss: 0.1574
Epoch: [35/50] | Step: [60/196] | Loss: 0.2376
Epoch: [35/50] | Step: [90/196] | Loss: 0.1862
Epoch: [35/50] | Step: [120/196] | Loss: 0.3565
Epoch: [35/50] | Step: [150/196] | Loss: 0.2255
Epoch: [35/50] | Step: [180/196] | Loss: 0.1966
Validating...
Current Accuracy: 94.740%
Best Accuracy: 95.162%

Epoch: [36/50] | Step: [30/196] | Loss: 0.2437
Epoch: [36/50] | Step: [60/196] | Loss: 0.1467
Epoch: [36/50] | Step: [90/196] | Loss: 0.1313
Epoch: [36/50] | Step: [120/196] | Loss: 0.1681
Epoch: [36/50] | Step: [150/196] | Loss: 0.1868
Epoch: [36/50] | Step: [180/196] | Loss: 0.3403
Validating...
Current Accuracy: 92.860%
Best Accuracy: 95.162%

Epoch: [37/50] | Step: [30/196] | Loss: 0.1608
Epoch: [37/50] | Step: [60/196] | Loss: 0.1599
Epoch: [37/50] | Step: [90/196] | Loss: 0.2995
Epoch: [37/50] | Step: [120/196] | Loss: 0.2515
Epoch: [37/50] | Step: [150/196] | Loss: 0.1938
Epoch: [37/50] | Step: [180/196] | Loss: 0.1642
Validating...
Current Accuracy: 95.004%
Best Accuracy: 95.162%

Epoch: [38/50] | Step: [30/196] | Loss: 0.3003
Epoch: [38/50] | Step: [60/196] | Loss: 0.1432
Epoch: [38/50] | Step: [90/196] | Loss: 0.1632
```

```
Epoch: [38/50] | Step: [120/196] | Loss: 0.2311
Epoch: [38/50] | Step: [150/196] | Loss: 0.1146
Epoch: [38/50] | Step: [180/196] | Loss: 0.1856
Validating...
Current Accuracy: 94.894%
Best Accuracy: 95.162%

Epoch: [39/50] | Step: [30/196] | Loss: 0.1936
Epoch: [39/50] | Step: [60/196] | Loss: 0.2319
Epoch: [39/50] | Step: [90/196] | Loss: 0.1594
Epoch: [39/50] | Step: [120/196] | Loss: 0.1759
Epoch: [39/50] | Step: [150/196] | Loss: 0.1901
Epoch: [39/50] | Step: [180/196] | Loss: 0.1759
Validating...
Current Accuracy: 94.898%
Best Accuracy: 95.162%

Epoch: [40/50] | Step: [30/196] | Loss: 0.1774
Epoch: [40/50] | Step: [60/196] | Loss: 0.1769
Epoch: [40/50] | Step: [90/196] | Loss: 0.2918
Epoch: [40/50] | Step: [120/196] | Loss: 0.1600
Epoch: [40/50] | Step: [150/196] | Loss: 0.2207
Epoch: [40/50] | Step: [180/196] | Loss: 0.2688
Validating...
Current Accuracy: 93.214%
Best Accuracy: 95.162%

Epoch: [41/50] | Step: [30/196] | Loss: 0.2672
Epoch: [41/50] | Step: [60/196] | Loss: 0.1002
Epoch: [41/50] | Step: [90/196] | Loss: 0.2723
Epoch: [41/50] | Step: [120/196] | Loss: 0.2825
Epoch: [41/50] | Step: [150/196] | Loss: 0.2541
Epoch: [41/50] | Step: [180/196] | Loss: 0.2758
Validating...
Current Accuracy: 93.622%
Best Accuracy: 95.162%

Epoch: [42/50] | Step: [30/196] | Loss: 0.2782
Epoch: [42/50] | Step: [60/196] | Loss: 0.1228
Epoch: [42/50] | Step: [90/196] | Loss: 0.1437
Epoch: [42/50] | Step: [120/196] | Loss: 0.1088
Epoch: [42/50] | Step: [150/196] | Loss: 0.1667
Epoch: [42/50] | Step: [180/196] | Loss: 0.1493
Validating...
Current Accuracy: 95.028%
Best Accuracy: 95.162%

Epoch: [43/50] | Step: [30/196] | Loss: 0.1445
Epoch: [43/50] | Step: [60/196] | Loss: 0.1022
Epoch: [43/50] | Step: [90/196] | Loss: 0.1152
```

```
Epoch: [43/50] | Step: [120/196] | Loss: 0.2064
Epoch: [43/50] | Step: [150/196] | Loss: 0.2467
Epoch: [43/50] | Step: [180/196] | Loss: 0.1679
Validating...
Current Accuracy: 95.154%
Best Accuracy: 95.162%

Epoch: [44/50] | Step: [30/196] | Loss: 0.1357
Epoch: [44/50] | Step: [60/196] | Loss: 0.2318
Epoch: [44/50] | Step: [90/196] | Loss: 0.1784
Epoch: [44/50] | Step: [120/196] | Loss: 0.2045
Epoch: [44/50] | Step: [150/196] | Loss: 0.1728
Epoch: [44/50] | Step: [180/196] | Loss: 0.2039
Validating...
Current Accuracy: 94.860%
Best Accuracy: 95.162%

Epoch: [45/50] | Step: [30/196] | Loss: 0.2195
Epoch: [45/50] | Step: [60/196] | Loss: 0.1060
Epoch: [45/50] | Step: [90/196] | Loss: 0.3006
Epoch: [45/50] | Step: [120/196] | Loss: 0.2460
Epoch: [45/50] | Step: [150/196] | Loss: 0.1085
Epoch: [45/50] | Step: [180/196] | Loss: 0.1181
Validating...
Current Accuracy: 94.324%
Best Accuracy: 95.162%

Epoch: [46/50] | Step: [30/196] | Loss: 0.1754
Epoch: [46/50] | Step: [60/196] | Loss: 0.2498
Epoch: [46/50] | Step: [90/196] | Loss: 0.1897
Epoch: [46/50] | Step: [120/196] | Loss: 0.1125
Epoch: [46/50] | Step: [150/196] | Loss: 0.1215
Epoch: [46/50] | Step: [180/196] | Loss: 0.2441
Validating...
Current Accuracy: 95.212%
Best Accuracy: 95.212%

Epoch: [47/50] | Step: [30/196] | Loss: 0.1578
Epoch: [47/50] | Step: [60/196] | Loss: 0.1981
Epoch: [47/50] | Step: [90/196] | Loss: 0.2197
Epoch: [47/50] | Step: [120/196] | Loss: 0.1889
Epoch: [47/50] | Step: [150/196] | Loss: 0.1220
Epoch: [47/50] | Step: [180/196] | Loss: 0.1405
Validating...
Current Accuracy: 95.434%
Best Accuracy: 95.434%

Epoch: [48/50] | Step: [30/196] | Loss: 0.1854
Epoch: [48/50] | Step: [60/196] | Loss: 0.1238
Epoch: [48/50] | Step: [90/196] | Loss: 0.2043
```

```
Epoch: [48/50] | Step: [120/196] | Loss: 0.1682
Epoch: [48/50] | Step: [150/196] | Loss: 0.2698
Epoch: [48/50] | Step: [180/196] | Loss: 0.1339
Validating...
Current Accuracy: 95.898%
Best Accuracy: 95.898%

Epoch: [49/50] | Step: [30/196] | Loss: 0.1912
Epoch: [49/50] | Step: [60/196] | Loss: 0.0875
Epoch: [49/50] | Step: [90/196] | Loss: 0.1059
Epoch: [49/50] | Step: [120/196] | Loss: 0.2092
Epoch: [49/50] | Step: [150/196] | Loss: 0.1142
Epoch: [49/50] | Step: [180/196] | Loss: 0.1259
Validating...
Current Accuracy: 92.330%
Best Accuracy: 95.898%

Epoch: [50/50] | Step: [30/196] | Loss: 0.1686
Epoch: [50/50] | Step: [60/196] | Loss: 0.2331
Epoch: [50/50] | Step: [90/196] | Loss: 0.1349
Epoch: [50/50] | Step: [120/196] | Loss: 0.1994
Epoch: [50/50] | Step: [150/196] | Loss: 0.1960
Epoch: [50/50] | Step: [180/196] | Loss: 0.1879
Validating...
Current Accuracy: 96.390%
Best Accuracy: 96.390%

Train Finished!
```