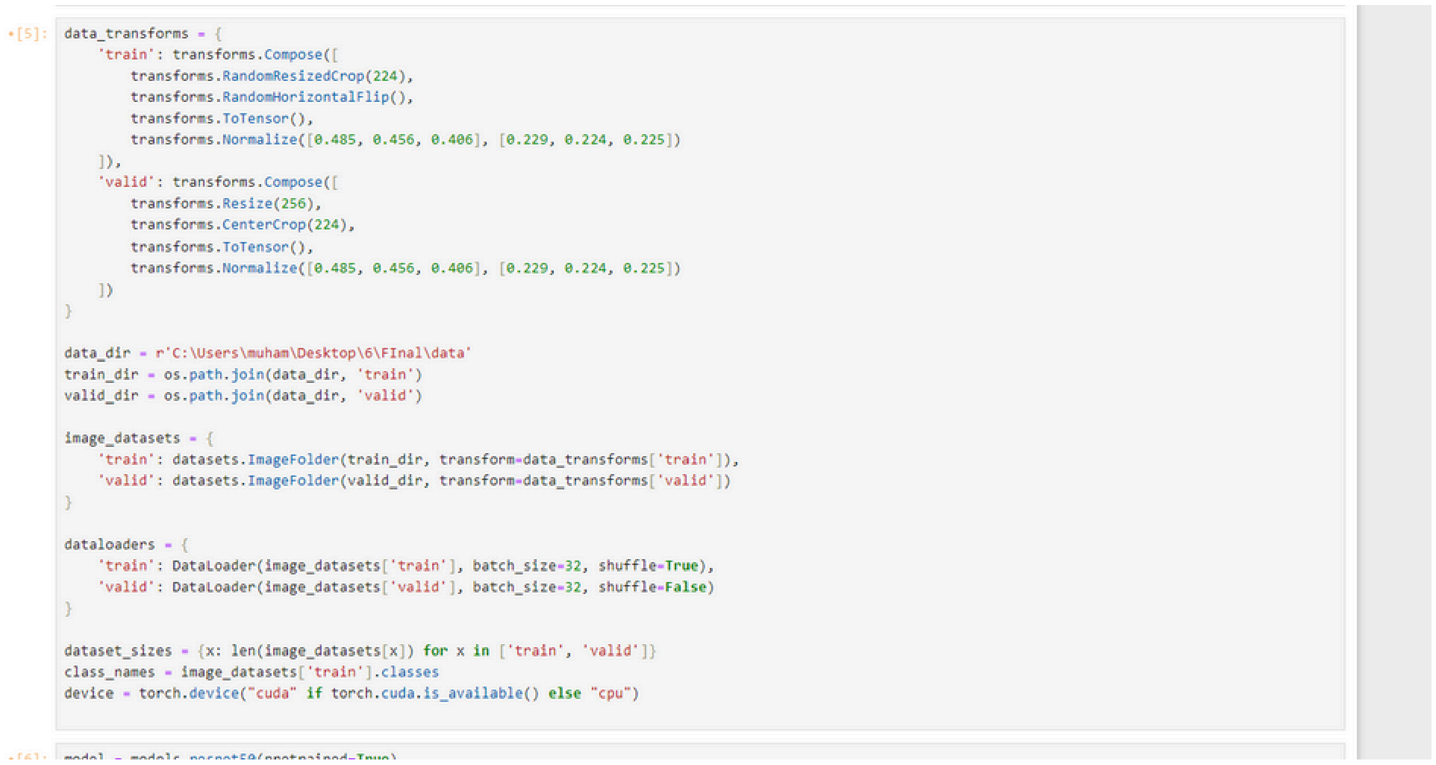
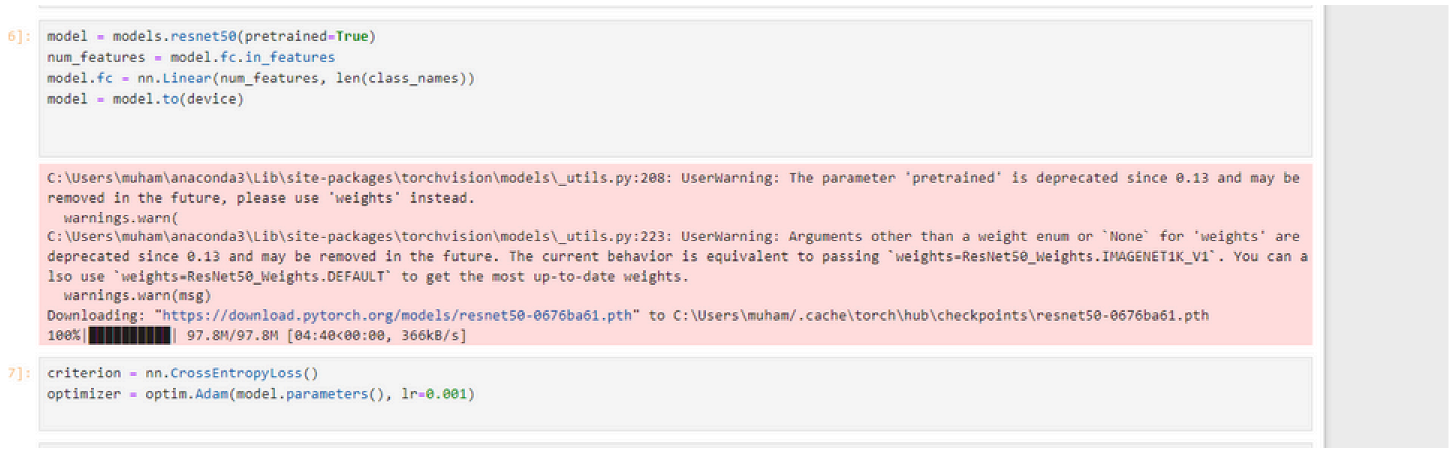


```
•[5]: data_transforms = {
          'train': transforms.Compose([
              transforms.RandomResizedCrop(224),
              transforms.RandomHorizontalFlip(),
              transforms.ToTensor(),
              transforms.Normalize([0.485, 0.456, 0.406], [0.229, 0.224, 0.225])
          ]),
          'valid': transforms.Compose([
              transforms.Resize(256),
              transforms.CenterCrop(224),
              transforms.ToTensor(),
              transforms.Normalize([0.485, 0.456, 0.406], [0.229, 0.224, 0.225])
          ])
      }

      data_dir = r'C:\Users\muham\Desktop\6\FInal\data'
      train_dir = os.path.join(data_dir, 'train')
      valid_dir = os.path.join(data_dir, 'valid')

      image_datasets = {
          'train': datasets.ImageFolder(train_dir, transform=data_transforms['train']),
          'valid': datasets.ImageFolder(valid_dir, transform=data_transforms['valid'])
      }

      dataloaders = {
          'train': DataLoader(image_datasets['train'], batch_size=32, shuffle=True),
          'valid': DataLoader(image_datasets['valid'], batch_size=32, shuffle=False)
      }

      dataset_sizes = {x: len(image_datasets[x]) for x in ['train', 'valid']}
      class_names = image_datasets['train'].classes
      device = torch.device("cuda" if torch.cuda.is_available() else "cpu")

•[6]: model = models.resnet50(pretrained=True)
```

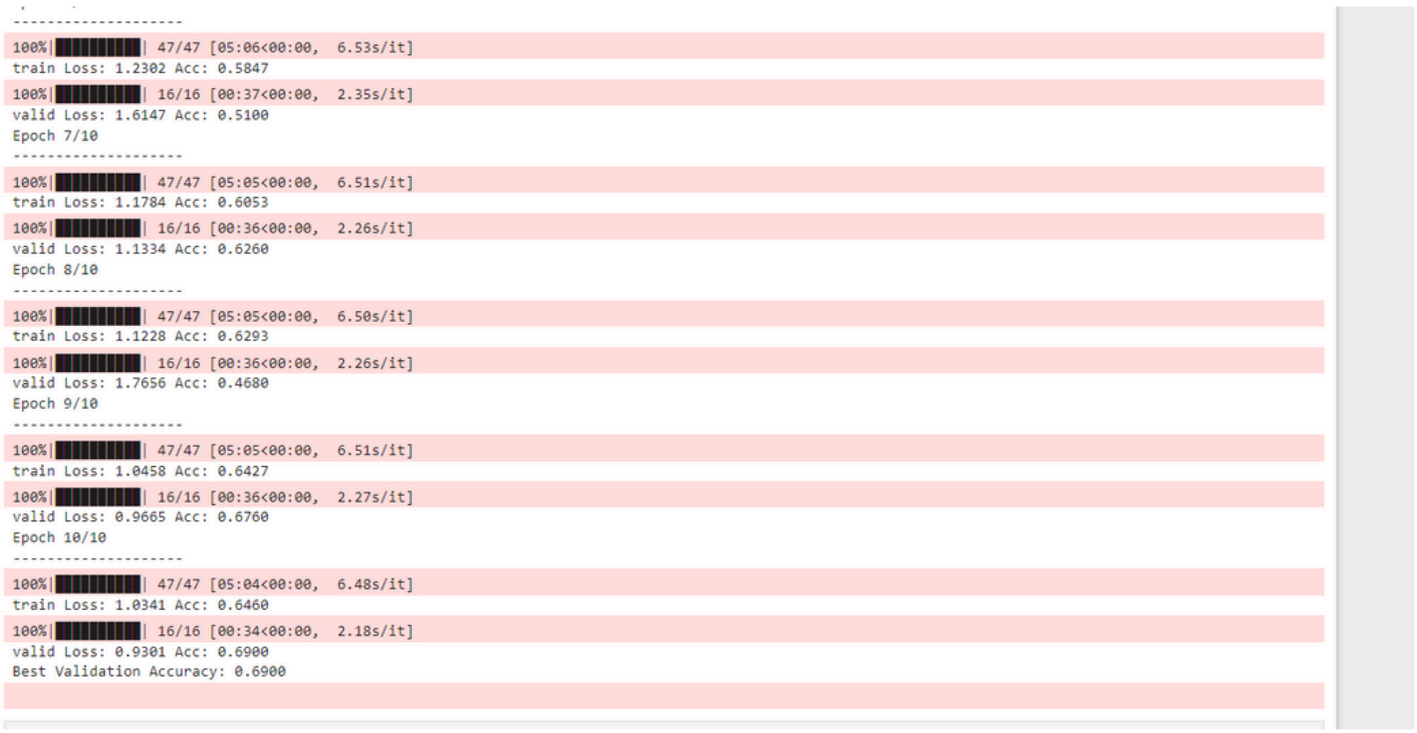
Simple augmentations for train and valid data like crop flip and resize and normalize the pictures



```
6]: model = models.resnet50(pretrained=True)
    num_features = model.fc.in_features
    model.fc = nn.Linear(num_features, len(class_names))
    model = model.to(device)

    C:\Users\muham\anaconda3\Lib\site-packages\torchvision\models\_utils.py:208: UserWarning: The parameter 'pretrained' is deprecated since 0.13 and may be
    removed in the future, please use 'weights' instead.
      warnings.warn(
    C:\Users\muham\anaconda3\Lib\site-packages\torchvision\models\_utils.py:223: UserWarning: Arguments other than a weight enum or `None` for 'weights' are
    deprecated since 0.13 and may be removed in the future. The current behavior is equivalent to passing `weights=ResNet50_Weights.IMAGENET1K_V1`. You can a
    lso use `weights=ResNet50_Weights.DEFAULT` to get the most up-to-date weights.
      warnings.warn(msg)
    Downloading: "https://download.pytorch.org/models/resnet50-0676ba61.pth" to C:\Users\muham/.cache\torch\hub\checkpoints\resnet50-0676ba61.pth
    100%|██████████| 97.8M/97.8M [04:40<00:00, 366kB/s]

7]: criterion = nn.CrossEntropyLoss()
    optimizer = optim.Adam(model.parameters(), lr=0.001)
```

Download PreTrained ResNet and start training

Base PreTrained ResNet

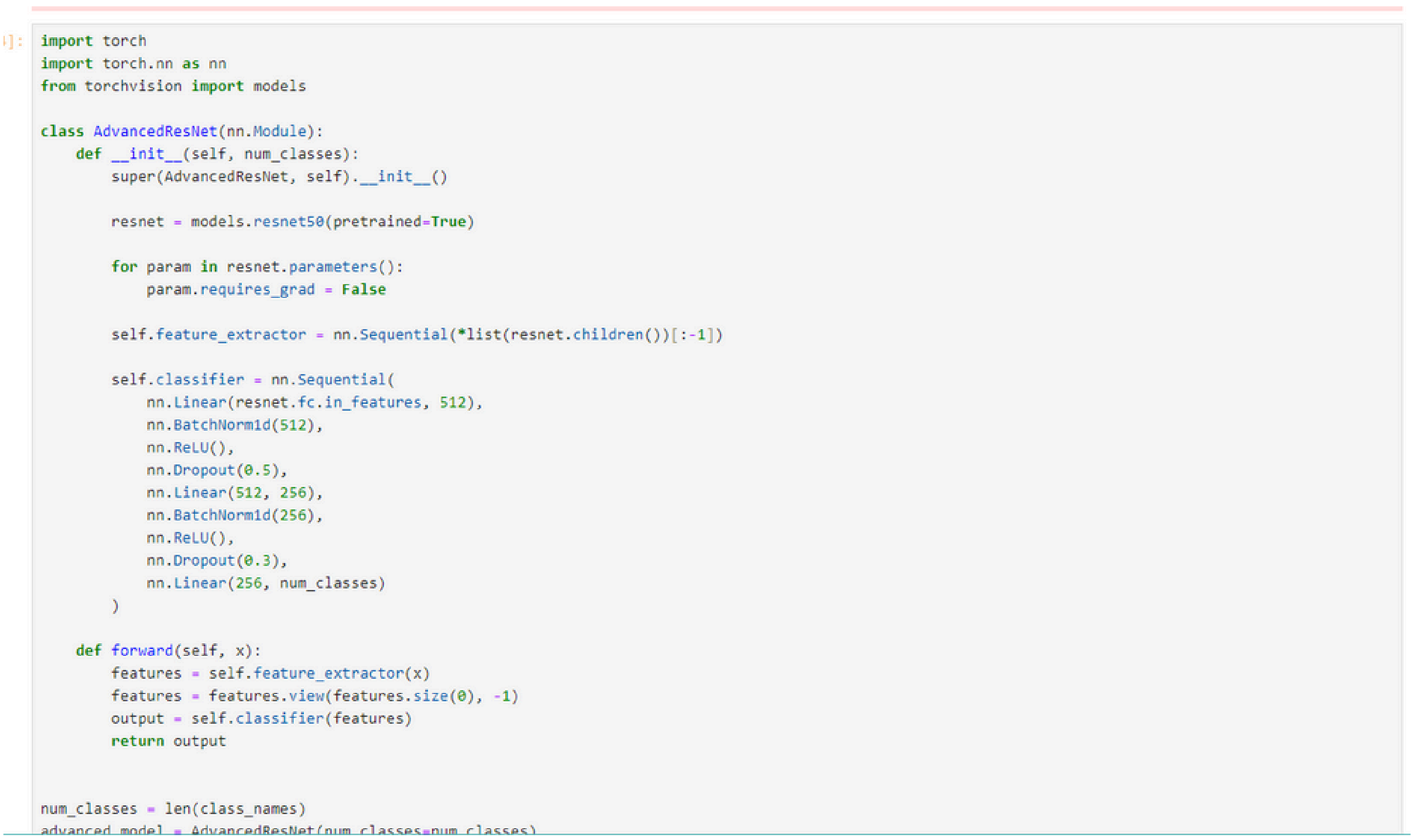


```
--------------------
100%|██████████| 47/47 [05:06<00:00,  6.53s/it]
train Loss: 1.2302 Acc: 0.5847
100%|██████████| 16/16 [00:37<00:00,  2.35s/it]
valid Loss: 1.6147 Acc: 0.5100
Epoch 7/10
--------------------
100%|██████████| 47/47 [05:05<00:00,  6.51s/it]
train Loss: 1.1784 Acc: 0.6053
100%|██████████| 16/16 [00:36<00:00,  2.26s/it]
valid Loss: 1.1334 Acc: 0.6260
Epoch 8/10
--------------------
100%|██████████| 47/47 [05:05<00:00,  6.50s/it]
train Loss: 1.1228 Acc: 0.6293
100%|██████████| 16/16 [00:36<00:00,  2.26s/it]
valid Loss: 1.7656 Acc: 0.4680
Epoch 9/10
--------------------
100%|██████████| 47/47 [05:05<00:00,  6.51s/it]
train Loss: 1.0458 Acc: 0.6427
100%|██████████| 16/16 [00:36<00:00,  2.27s/it]
valid Loss: 0.9665 Acc: 0.6760
Epoch 10/10
--------------------
100%|██████████| 47/47 [05:04<00:00,  6.48s/it]
train Loss: 1.0341 Acc: 0.6460
100%|██████████| 16/16 [00:34<00:00,  2.18s/it]
valid Loss: 0.9301 Acc: 0.6900
Best Validation Accuracy: 0.6900
```

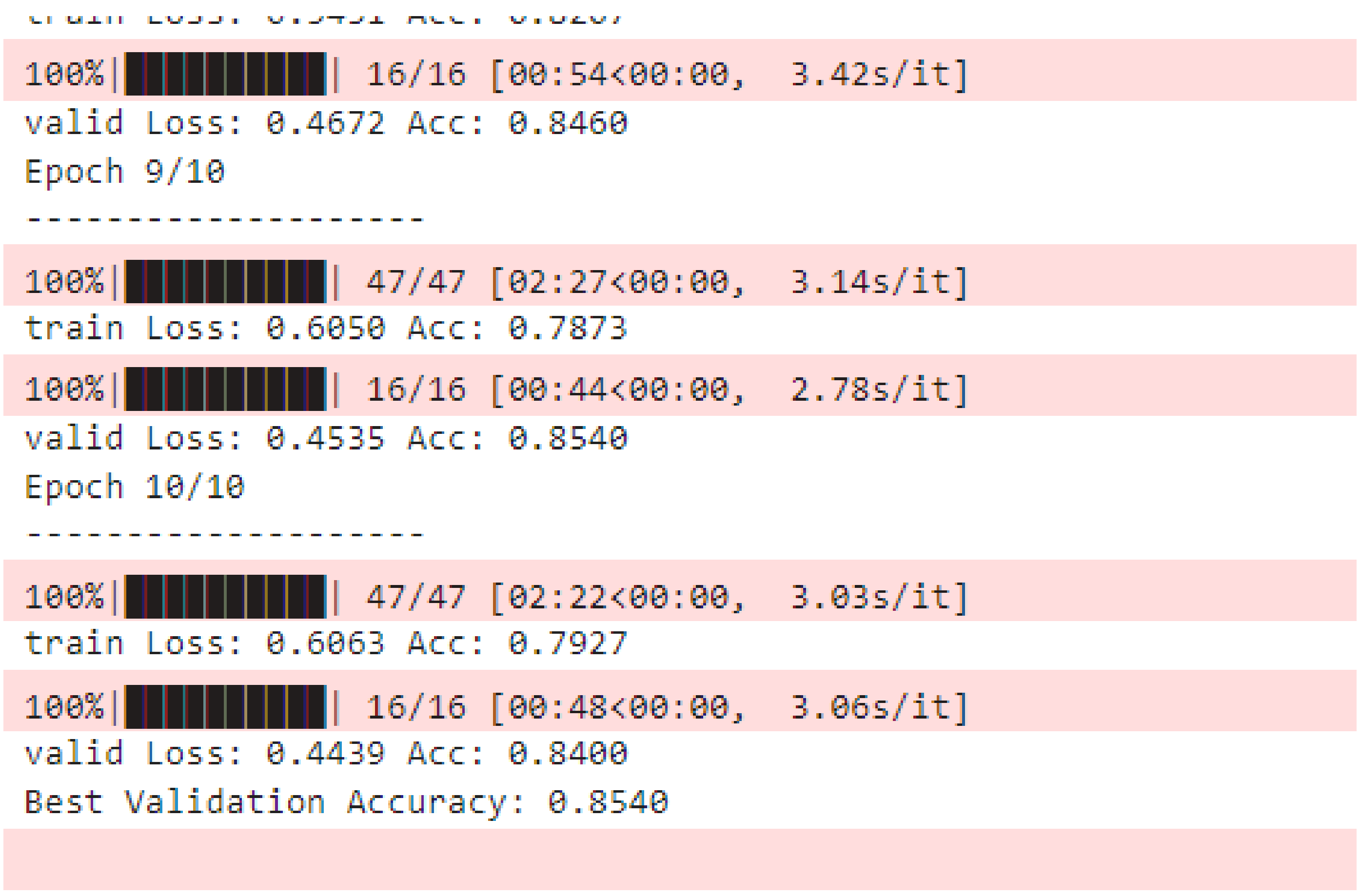
64% on train

69% on Validation

Advansed ResNet



```python
import torch
import torch.nn as nn
from torchvision import models

class AdvancedResNet(nn.Module):
    def __init__(self, num_classes):
        super(AdvancedResNet, self).__init__()

        resnet = models.resnet50(pretrained=True)

        for param in resnet.parameters():
            param.requires_grad = False

        self.feature_extractor = nn.Sequential(*list(resnet.children())[:-1])

        self.classifier = nn.Sequential(
            nn.Linear(resnet.fc.in_features, 512),
            nn.BatchNorm1d(512),
            nn.ReLU(),
            nn.Dropout(0.5),
            nn.Linear(512, 256),
            nn.BatchNorm1d(256),
            nn.ReLU(),
            nn.Dropout(0.3),
            nn.Linear(256, num_classes)
        )

    def forward(self, x):
        features = self.feature_extractor(x)
        features = features.view(features.size(0), -1)
        output = self.classifier(features)
        return output


num_classes = len(class_names)
advanced_model = AdvancedResNet(num_classes=num_classes)
```



```
100%|██████████| 16/16 [00:54<00:00,  3.42s/it]
valid Loss: 0.4672 Acc: 0.8460
Epoch 9/10
--------------------
100%|██████████| 47/47 [02:27<00:00,  3.14s/it]
train Loss: 0.6050 Acc: 0.7873
100%|██████████| 16/16 [00:44<00:00,  2.78s/it]
valid Loss: 0.4535 Acc: 0.8540
Epoch 10/10
--------------------
100%|██████████| 47/47 [02:22<00:00,  3.03s/it]
train Loss: 0.6063 Acc: 0.7927
100%|██████████| 16/16 [00:48<00:00,  3.06s/it]
valid Loss: 0.4439 Acc: 0.8400
Best Validation Accuracy: 0.8540
```

79% on train

85% on Validation data