# Lab 3: Finetuning

### Assignment 1 (2p)

# Load Data  
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])  
 ]),  
 'val': transforms.Compose([  
 transforms.Resize(256),  
 transforms.CenterCrop(224),  
 transforms.ToTensor(),  
 transforms.Normalize([0.485, 0.456, 0.406], [0.229, 0.224, 0.225])  
 ]),  
}  
  
num\_classes = 10  
data\_dir = './data/butterflies/'  
image\_datasets\_train = datasets.ImageFolder(os.path.join(data\_dir, 'train'), data\_transforms['train'])  
image\_datasets\_val = datasets.ImageFolder(os.path.join(data\_dir, 'val'), data\_transforms['val'])  
  
select\_datasets\_train = torch.utils.data.SubsetRandomSampler(list(range(len(image\_datasets\_train))))  
select\_datasets\_val = torch.utils.data.SubsetRandomSampler(list(range(len(image\_datasets\_val))))  
  
dataloaders = {  
 'train': torch.utils.data.DataLoader(image\_datasets\_train, batch\_size=4,  
 shuffle=False, sampler=select\_datasets\_train),  
 'val': torch.utils.data.DataLoader(image\_datasets\_val, batch\_size=4,  
 shuffle=False, sampler=select\_datasets\_val)}  
  
dataset\_sizes = {  
 'train': len(image\_datasets\_train),  
 'val': len(image\_datasets\_val)}  
  
class\_names = image\_datasets\_train.classes

### Assignment 2 (4p)

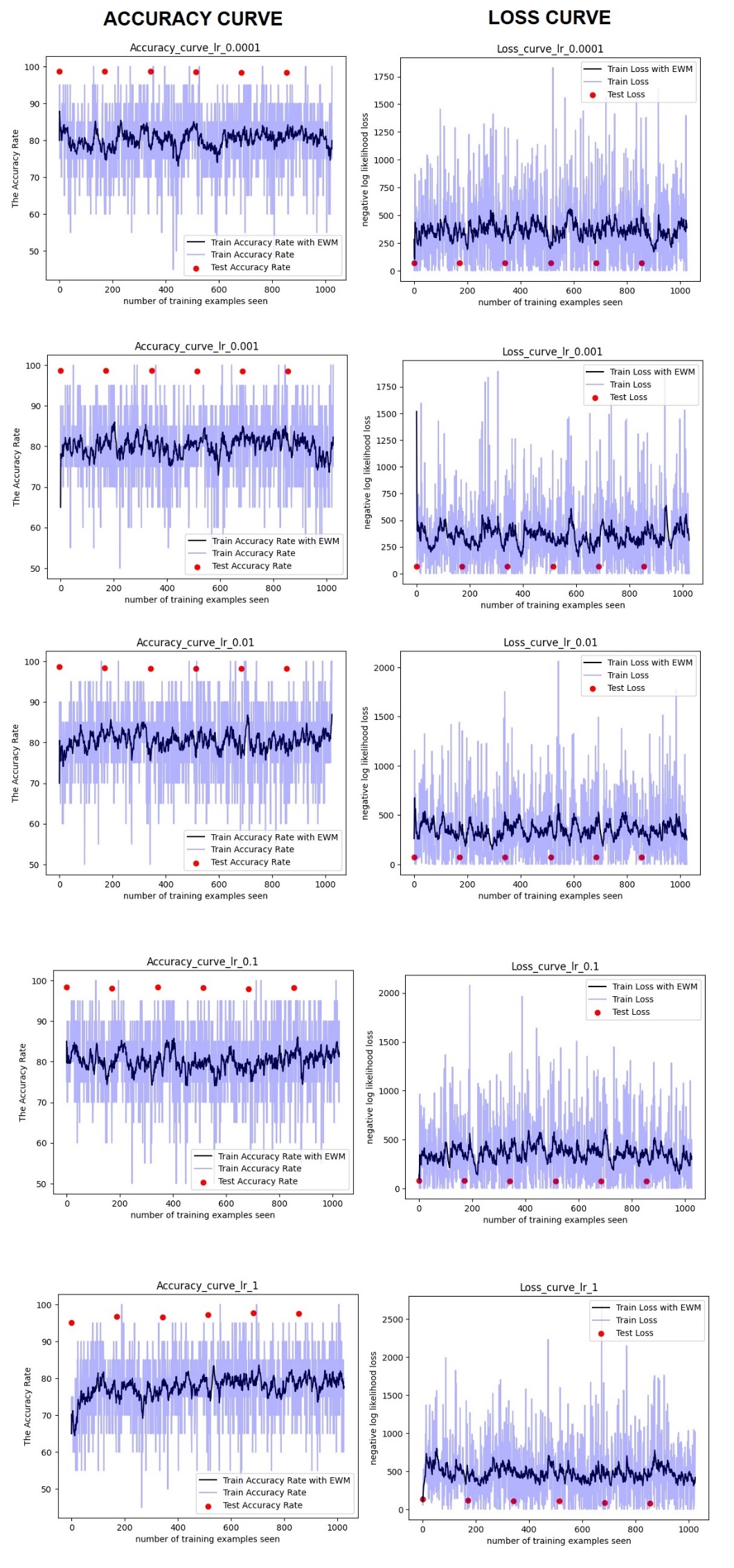
I used the VGG net as the base network, and concatenated a 10 classes fully connected layer as the last layer. torch.nn.Dropout  was used after (after activations) inside the classifier block.

model\_ft.classifier[-1] = nn.Sequential(nn.Linear(in\_features=num\_ftrs, out\_features=num\_classes), nn.Dropout(p=0.2))

optimizer\_ft = optim.SGD(model\_ft.parameters(), lr=0.001, momentum=0.9)  
  
# Decay LR by a factor of 0.1 every 7 epochs  
exp\_lr\_scheduler = lr\_scheduler.StepLR(optimizer\_ft, step\_size=5, gamma=0.1)

Set the initial learning rate as 0.001, and after every 10 epoches, the learning rate will become gamma \* lr = 0.1\*0.001

The learning rate range as [1, 0.1, 0.01, 0.001, 0.0001], I found when it equals 0.001, the accuracy curve and the loss curve performance best.



### Assignment 3 (4p)

After converted the 1000-class fully connected layer to 10-class fully connected layer, I selected the learning rate as 0.001, and other parameters as the same. The result is shown below.

