MP3_P1B_Develop_Classifier

November 1, 2020

1 Assignment 3 Part 1: Developing Your Own Classifier

[1]:
```
import os
import numpy as np
import torch
import torch.nn as nn
import torchvision

from torchvision import transforms
from sklearn.metrics import average_precision_score
from PIL import Image, ImageDraw
import matplotlib.pyplot as plt
from kaggle_submission import output_submission_csv
from classifier import SimpleClassifier, Classifier#, AlexNet
from voc_dataloader import VocDataset, VOC_CLASSES

%matplotlib inline
%load_ext autoreload
%autoreload 2
```

2 Part 1B: Design your own network

In this notebook, your task is to create and train your own model for multi-label classification on VOC Pascal.

2.1 What to do

1. You will make change on network architecture in `classifier.py`.
2. You may also want to change other hyperparameters to assist your training to get a better performances. Hints will be given in the below instructions.

2.2 What to submit

Check the submission template for details what to submit.

[2]:
```
def train_classifier(train_loader, classifier, criterion, optimizer):
    classifier.train()
    loss_ = 0.0
    losses = []
    for i, (images, labels, _) in enumerate(train_loader):
        images, labels = images.to(device), labels.to(device)
        optimizer.zero_grad()
        logits = classifier(images)
        loss = criterion(logits, labels)
        loss.backward()
        optimizer.step()
        losses.append(loss)
    return torch.stack(losses).mean().item()
```

[3]:
```
def test_classifier(test_loader, classifier, criterion, print_ind_classes=True,␣
↪print_total=True):
    classifier.eval()
    losses = []
    with torch.no_grad():
        y_true = np.zeros((0,21))
        y_score = np.zeros((0,21))
        for i, (images, labels, _) in enumerate(test_loader):
            images, labels = images.to(device), labels.to(device)
            logits = classifier(images)
            y_true = np.concatenate((y_true, labels.cpu().numpy()), axis=0)
            y_score = np.concatenate((y_score, logits.cpu().numpy()), axis=0)
            loss = criterion(logits, labels)
            losses.append(loss.item())
        aps = []
        # ignore first class which is background
        for i in range(1, y_true.shape[1]):
            ap = average_precision_score(y_true[:, i], y_score[:, i])
            if print_ind_classes:
                print('-------  Class: {:<12}     AP: {:>8.4f}  -------'.
↪format(VOC_CLASSES[i], ap))
            aps.append(ap)

        mAP = np.mean(aps)
        test_loss = np.mean(losses)
        if print_total:
            print('mAP: {0:.4f}'.format(mAP))
            print('Avg loss: {}'.format(test_loss))

    return mAP, test_loss, aps
```

[4]:
```
def plot_losses(train, val, test_frequency, num_epochs):
    plt.plot(train, label="train")
```

```
    indices = [i for i in range(num_epochs) if ((i+1)%test_frequency == 0 or i
 ↪==0)]
    plt.plot(indices, val, label="val")
    plt.title("Loss Plot")
    plt.ylabel("Loss")
    plt.xlabel("Epoch")
    plt.legend()
    plt.show()

def plot_mAP(train, val, test_frequency, num_epochs):
    indices = [i for i in range(num_epochs) if ((i+1)%test_frequency == 0 or i
 ↪==0)]
    plt.plot(indices, train, label="train")
    plt.plot(indices, val, label="val")
    plt.title("mAP Plot")
    plt.ylabel("mAP")
    plt.xlabel("Epoch")
    plt.legend()
    plt.show()
```

[5]:
```
def train(classifier, num_epochs, train_loader, val_loader, criterion,
 ↪optimizer, test_frequency=5):
    train_losses = []
    train_mAPs = []
    val_losses = []
    val_mAPs = []

    for epoch in range(1,num_epochs+1):
        print("Starting epoch number " + str(epoch))
        train_loss = train_classifier(train_loader, classifier, criterion,
 ↪optimizer)
        train_losses.append(train_loss)
        print("Loss for Training on Epoch " +str(epoch) + " is "+
 ↪str(train_loss))
        if(epoch%test_frequency==0 or epoch==1):
            mAP_train, _, _ = test_classifier(train_loader, classifier,
 ↪criterion, False, False)
            train_mAPs.append(mAP_train)
            mAP_val, val_loss, _ = test_classifier(val_loader, classifier,
 ↪criterion)
            print('Evaluating classifier')
            print("Mean Precision Score for Testing on Epoch " +str(epoch) + "
 ↪is "+ str(mAP_val))
            val_losses.append(val_loss)
            val_mAPs.append(mAP_val)
```

```
    return classifier, train_losses, val_losses, train_mAPs, val_mAPs
```

3 Developing Your Own Model

3.0.1 Goal

To meet the benchmark for this assignment you will need to improve the network. Note you should have noticed pretrained Alenxt performs really well, but training Alexnet from scratch performs much worse. We hope you can design a better architecture over both the simple classifier and AlexNet to train from scratch.

3.0.2 How to start

You may take inspiration from other published architectures and architectures discussed in lecture. However, you are NOT allowed to use predefined models (e.g. models from torchvision) or use pretrained weights. Training must be done from scratch with your own custom model.

Some hints There are a variety of different approaches you should try to improve performance from the simple classifier:

- Network architecture changes
  - Number of layers: try adding layers to make your network deeper
  - Batch normalization: adding batch norm between layers will likely give you a significant performance increase
  - Residual connections: as you increase the depth of your network, you will find that having residual connections like those in ResNet architectures will be helpful
- Optimizer: Instead of plain SGD, you may want to add a learning rate schedule, add momentum, or use one of the other optimizers you have learned about like Adam. Check the `torch.optim` package for other optimizers
- Data augmentation: You should use the `torchvision.transforms` module to try adding random resized crops and horizontal flips of the input data. Check `transforms.RandomResizedCrop` and `transforms.RandomHorizontalFlip` for this. Feel free to apply more transforms for data augmentation which can lead to better performance.
- Epochs: Once you have found a generally good hyperparameter setting try training for more epochs
- Loss function: You might want to add weighting to the `MultiLabelSoftMarginLoss` for classes that are less well represented or experiment with a different loss function

Note We will soon be providing some initial expectations of mAP values as a function of epoch so you can get an early idea whether your implementation works without waiting a long time for training to converge.

3.0.3 What to submit

Submit your best model to Kaggle and save all plots for the writeup.

[6]:
```
device = torch.device("cuda:0" if torch.cuda.is_available() else "cpu")
print(device)

normalize = transforms.Normalize(mean=[0.4914, 0.4822, 0.4465],
                                 std= [0.2023, 0.1994, 0.2010])

train_transform = transforms.Compose([
            transforms.RandomRotation(25),
            transforms.Resize(227),
            transforms.CenterCrop(227),
            transforms.RandomHorizontalFlip(),
            transforms.ToTensor(),
            normalize
        ])

test_transform = transforms.Compose([
            transforms.Resize(227),
            transforms.CenterCrop(227),
            transforms.ToTensor(),
            normalize,
        ])

ds_train = VocDataset('VOCdevkit_2007/VOC2007/','train',train_transform)
ds_val = VocDataset('VOCdevkit_2007/VOC2007/','val',test_transform)
ds_test = VocDataset('VOCdevkit_2007/VOC2007test/','test', test_transform)
```

```
cuda:0
```

[7]:
```
num_epochs = 50
test_frequency = 5
batch_size = 64

train_loader = torch.utils.data.DataLoader(dataset=ds_train,
                                           batch_size=batch_size,
                                           shuffle=True,
                                           num_workers=1)

val_loader = torch.utils.data.DataLoader(dataset=ds_val,
                                           batch_size=batch_size,
                                           shuffle=True,
                                           num_workers=1)

test_loader = torch.utils.data.DataLoader(dataset=ds_test,
                                           batch_size=batch_size,
                                           shuffle=False,
                                           num_workers=1)
```

[8]:
```
# TODO: Run your own classifier here
classifier = Classifier().to(device)

criterion = nn.MultiLabelSoftMarginLoss()
#optimizer = torch.optim.SGD(classifier.parameters(), lr=0.1e-4, momentum=0.5)
optimizer = torch.optim.Adam(classifier.parameters(), lr=1e-4)

classifier, train_losses, val_losses, train_mAPs, val_mAPs = train(classifier,␣
 ↪num_epochs, train_loader, val_loader, criterion, optimizer, test_frequency)
```

```
Starting epoch number 1
Loss for Training on Epoch 1 is 0.3516332805156708
-------  Class: aeroplane        AP:   0.2897  -------
-------  Class: bicycle          AP:   0.0940  -------
-------  Class: bird             AP:   0.0851  -------
-------  Class: boat             AP:   0.0899  -------
-------  Class: bottle           AP:   0.0338  -------
-------  Class: bus              AP:   0.0683  -------
-------  Class: car              AP:   0.2366  -------
-------  Class: cat              AP:   0.0790  -------
-------  Class: chair            AP:   0.1371  -------
-------  Class: cow              AP:   0.0291  -------
-------  Class: diningtable      AP:   0.0596  -------
-------  Class: dog              AP:   0.0893  -------
-------  Class: horse            AP:   0.1033  -------
-------  Class: motorbike        AP:   0.1520  -------
-------  Class: person           AP:   0.5271  -------
-------  Class: pottedplant      AP:   0.0839  -------
-------  Class: sheep            AP:   0.0441  -------
-------  Class: sofa             AP:   0.1102  -------
-------  Class: train            AP:   0.1438  -------
-------  Class: tvmonitor        AP:   0.0522  -------
mAP: 0.1254
Avg loss: 0.2449375007301569
Evaluating classifier
Mean Precision Score for Testing on Epoch 1 is 0.12540310819880637
Starting epoch number 2
Loss for Training on Epoch 2 is 0.284593790769577
Starting epoch number 3
Loss for Training on Epoch 3 is 0.27392902970314026
Starting epoch number 4
Loss for Training on Epoch 4 is 0.2684877812862396
Starting epoch number 5
Loss for Training on Epoch 5 is 0.2631917893886566
-------  Class: aeroplane        AP:   0.4437  -------
-------  Class: bicycle          AP:   0.0907  -------
-------  Class: bird             AP:   0.1620  -------
```

```
-------  Class: boat             AP:   0.2077  -------
-------  Class: bottle           AP:   0.1071  -------
-------  Class: bus              AP:   0.0876  -------
-------  Class: car              AP:   0.4726  -------
-------  Class: cat              AP:   0.2113  -------
-------  Class: chair            AP:   0.2974  -------
-------  Class: cow              AP:   0.0364  -------
-------  Class: diningtable      AP:   0.2508  -------
-------  Class: dog              AP:   0.2080  -------
-------  Class: horse            AP:   0.2758  -------
-------  Class: motorbike        AP:   0.1889  -------
-------  Class: person           AP:   0.6190  -------
-------  Class: pottedplant      AP:   0.0871  -------
-------  Class: sheep            AP:   0.0424  -------
-------  Class: sofa             AP:   0.1746  -------
-------  Class: train            AP:   0.1509  -------
-------  Class: tvmonitor        AP:   0.0870  -------
mAP: 0.2100
Avg loss: 0.2144844215363264
Evaluating classifier
Mean Precision Score for Testing on Epoch 5 is 0.21004784320897185
Starting epoch number 6
Loss for Training on Epoch 6 is 0.2599688470363617
Starting epoch number 7
Loss for Training on Epoch 7 is 0.2539454996585846
Starting epoch number 8
Loss for Training on Epoch 8 is 0.2503301799297333
Starting epoch number 9
Loss for Training on Epoch 9 is 0.24734877049922943
Starting epoch number 10
Loss for Training on Epoch 10 is 0.24778710305690765
-------  Class: aeroplane        AP:   0.4948  -------
-------  Class: bicycle          AP:   0.2020  -------
-------  Class: bird             AP:   0.1541  -------
-------  Class: boat             AP:   0.1852  -------
-------  Class: bottle           AP:   0.1229  -------
-------  Class: bus              AP:   0.1422  -------
-------  Class: car              AP:   0.5329  -------
-------  Class: cat              AP:   0.2679  -------
-------  Class: chair            AP:   0.3886  -------
-------  Class: cow              AP:   0.0726  -------
-------  Class: diningtable      AP:   0.2661  -------
-------  Class: dog              AP:   0.1979  -------
-------  Class: horse            AP:   0.2026  -------
-------  Class: motorbike        AP:   0.3248  -------
-------  Class: person           AP:   0.6741  -------
-------  Class: pottedplant      AP:   0.1044  -------
-------  Class: sheep            AP:   0.0820  -------
```

```
-------  Class: sofa             AP:   0.2482  -------
-------  Class: train            AP:   0.3213  -------
-------  Class: tvmonitor        AP:   0.1624  -------
mAP: 0.2574
Avg loss: 0.20119226947426797
Evaluating classifier
Mean Precision Score for Testing on Epoch 10 is 0.25736016168683173
Starting epoch number 11
Loss for Training on Epoch 11 is 0.24154697358608246
Starting epoch number 12
Loss for Training on Epoch 12 is 0.24258458614349365
Starting epoch number 13
Loss for Training on Epoch 13 is 0.244707390666008
Starting epoch number 14
Loss for Training on Epoch 14 is 0.23779945075511932
Starting epoch number 15
Loss for Training on Epoch 15 is 0.2384723722934723
-------  Class: aeroplane        AP:   0.5629  -------
-------  Class: bicycle          AP:   0.2453  -------
-------  Class: bird             AP:   0.2279  -------
-------  Class: boat             AP:   0.1971  -------
-------  Class: bottle           AP:   0.1098  -------
-------  Class: bus              AP:   0.1660  -------
-------  Class: car              AP:   0.5499  -------
-------  Class: cat              AP:   0.3119  -------
-------  Class: chair            AP:   0.3811  -------
-------  Class: cow              AP:   0.1071  -------
-------  Class: diningtable      AP:   0.2602  -------
-------  Class: dog              AP:   0.2110  -------
-------  Class: horse            AP:   0.3447  -------
-------  Class: motorbike        AP:   0.2171  -------
-------  Class: person           AP:   0.6975  -------
-------  Class: pottedplant      AP:   0.1203  -------
-------  Class: sheep            AP:   0.1370  -------
-------  Class: sofa             AP:   0.2431  -------
-------  Class: train            AP:   0.3310  -------
-------  Class: tvmonitor        AP:   0.1774  -------
mAP: 0.2799
Avg loss: 0.19579387344419957
Evaluating classifier
Mean Precision Score for Testing on Epoch 15 is 0.2799146817552735
Starting epoch number 16
Loss for Training on Epoch 16 is 0.23590178787708282
Starting epoch number 17
Loss for Training on Epoch 17 is 0.23453259468078613
Starting epoch number 18
Loss for Training on Epoch 18 is 0.2261173576116562
Starting epoch number 19
```

```
Loss for Training on Epoch 19 is 0.22477440536022186
Starting epoch number 20
Loss for Training on Epoch 20 is 0.2262583076953888
-------  Class: aeroplane        AP:   0.5577  -------
-------  Class: bicycle          AP:   0.3394  -------
-------  Class: bird             AP:   0.2398  -------
-------  Class: boat             AP:   0.2293  -------
-------  Class: bottle           AP:   0.1219  -------
-------  Class: bus              AP:   0.1464  -------
-------  Class: car              AP:   0.5808  -------
-------  Class: cat              AP:   0.3432  -------
-------  Class: chair            AP:   0.3667  -------
-------  Class: cow              AP:   0.1319  -------
-------  Class: diningtable      AP:   0.2396  -------
-------  Class: dog              AP:   0.2516  -------
-------  Class: horse            AP:   0.4216  -------
-------  Class: motorbike        AP:   0.3561  -------
-------  Class: person           AP:   0.7262  -------
-------  Class: pottedplant      AP:   0.1219  -------
-------  Class: sheep            AP:   0.1264  -------
-------  Class: sofa             AP:   0.2317  -------
-------  Class: train            AP:   0.4747  -------
-------  Class: tvmonitor        AP:   0.1633  -------
mAP: 0.3085
Avg loss: 0.18719805963337421
Evaluating classifier
Mean Precision Score for Testing on Epoch 20 is 0.3085051379163099
Starting epoch number 21
Loss for Training on Epoch 21 is 0.22343240678310394
Starting epoch number 22
Loss for Training on Epoch 22 is 0.22134056687355042
Starting epoch number 23
Loss for Training on Epoch 23 is 0.2190103381872177
Starting epoch number 24
Loss for Training on Epoch 24 is 0.2157878875732422
Starting epoch number 25
Loss for Training on Epoch 25 is 0.2178138792514801
-------  Class: aeroplane        AP:   0.6049  -------
-------  Class: bicycle          AP:   0.3060  -------
-------  Class: bird             AP:   0.2136  -------
-------  Class: boat             AP:   0.2834  -------
-------  Class: bottle           AP:   0.1205  -------
-------  Class: bus              AP:   0.1826  -------
-------  Class: car              AP:   0.5855  -------
-------  Class: cat              AP:   0.3172  -------
-------  Class: chair            AP:   0.3879  -------
-------  Class: cow              AP:   0.1662  -------
-------  Class: diningtable      AP:   0.2830  -------
```

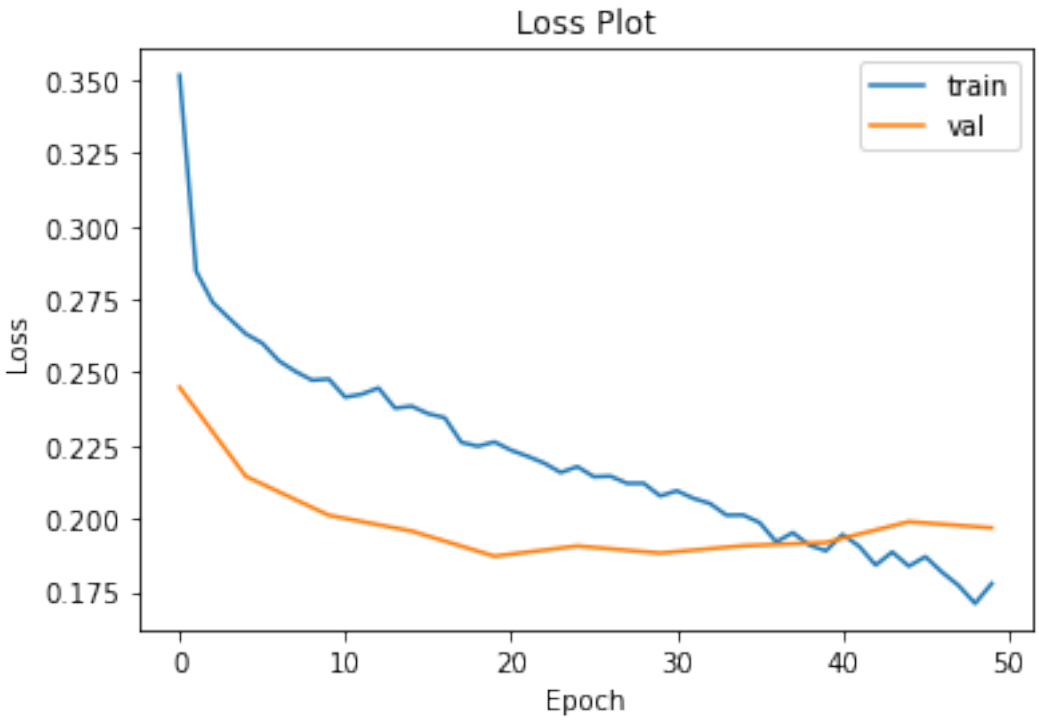
```
-------  Class: dog              AP:   0.2280  -------
-------  Class: horse            AP:   0.4348  -------
-------  Class: motorbike        AP:   0.2478  -------
-------  Class: person           AP:   0.7176  -------
-------  Class: pottedplant      AP:   0.1438  -------
-------  Class: sheep            AP:   0.1120  -------
-------  Class: sofa             AP:   0.2510  -------
-------  Class: train            AP:   0.4078  -------
-------  Class: tvmonitor        AP:   0.2588  -------
mAP: 0.3126
Avg loss: 0.19069521352648736
Evaluating classifier
Mean Precision Score for Testing on Epoch 25 is 0.31261091319397244
Starting epoch number 26
Loss for Training on Epoch 26 is 0.21434524655342102
Starting epoch number 27
Loss for Training on Epoch 27 is 0.21455322206020355
Starting epoch number 28
Loss for Training on Epoch 28 is 0.212093785405159
Starting epoch number 29
Loss for Training on Epoch 29 is 0.21211843192577362
Starting epoch number 30
Loss for Training on Epoch 30 is 0.20776839554309845
-------  Class: aeroplane        AP:   0.5892  -------
-------  Class: bicycle          AP:   0.2861  -------
-------  Class: bird             AP:   0.2362  -------
-------  Class: boat             AP:   0.2823  -------
-------  Class: bottle           AP:   0.1596  -------
-------  Class: bus              AP:   0.1931  -------
-------  Class: car              AP:   0.5494  -------
-------  Class: cat              AP:   0.3413  -------
-------  Class: chair            AP:   0.4217  -------
-------  Class: cow              AP:   0.1642  -------
-------  Class: diningtable      AP:   0.2915  -------
-------  Class: dog              AP:   0.2884  -------
-------  Class: horse            AP:   0.4431  -------
-------  Class: motorbike        AP:   0.3673  -------
-------  Class: person           AP:   0.7420  -------
-------  Class: pottedplant      AP:   0.1856  -------
-------  Class: sheep            AP:   0.1149  -------
-------  Class: sofa             AP:   0.2342  -------
-------  Class: train            AP:   0.5028  -------
-------  Class: tvmonitor        AP:   0.2306  -------
mAP: 0.3312
Avg loss: 0.188265909999609
Evaluating classifier
Mean Precision Score for Testing on Epoch 30 is 0.33117582387926137
Starting epoch number 31
```

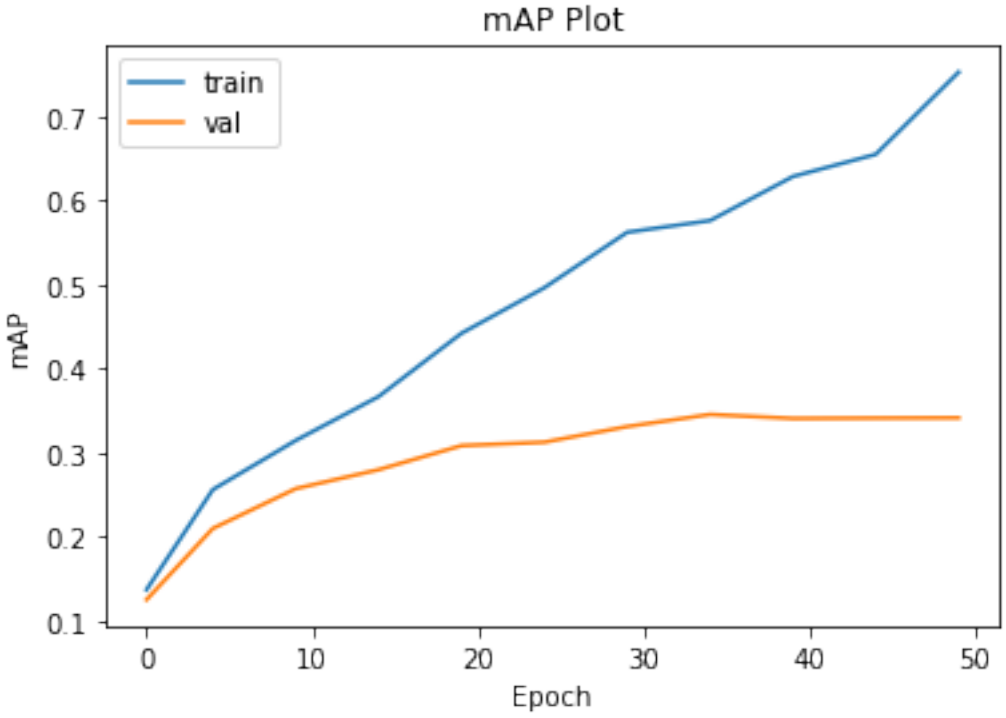
```
Loss for Training on Epoch 31 is 0.20955367386341095
Starting epoch number 32
Loss for Training on Epoch 32 is 0.20696671307086945
Starting epoch number 33
Loss for Training on Epoch 33 is 0.20521247386932373
Starting epoch number 34
Loss for Training on Epoch 34 is 0.20119576156139374
Starting epoch number 35
Loss for Training on Epoch 35 is 0.20127394795417786
-------  Class: aeroplane        AP:   0.6120  -------
-------  Class: bicycle          AP:   0.3742  -------
-------  Class: bird             AP:   0.2325  -------
-------  Class: boat             AP:   0.3378  -------
-------  Class: bottle           AP:   0.1244  -------
-------  Class: bus              AP:   0.2726  -------
-------  Class: car              AP:   0.5862  -------
-------  Class: cat              AP:   0.3401  -------
-------  Class: chair            AP:   0.3970  -------
-------  Class: cow              AP:   0.1612  -------
-------  Class: diningtable      AP:   0.2876  -------
-------  Class: dog              AP:   0.2490  -------
-------  Class: horse            AP:   0.4764  -------
-------  Class: motorbike        AP:   0.4434  -------
-------  Class: person           AP:   0.7566  -------
-------  Class: pottedplant      AP:   0.1494  -------
-------  Class: sheep            AP:   0.1168  -------
-------  Class: sofa             AP:   0.2769  -------
-------  Class: train            AP:   0.4658  -------
-------  Class: tvmonitor        AP:   0.2512  -------
mAP: 0.3456
Avg loss: 0.19076567329466343
Evaluating classifier
Mean Precision Score for Testing on Epoch 35 is 0.3455633205387576
Starting epoch number 36
Loss for Training on Epoch 36 is 0.1986854374408722
Starting epoch number 37
Loss for Training on Epoch 37 is 0.19207696616649628
Starting epoch number 38
Loss for Training on Epoch 38 is 0.19522680342197418
Starting epoch number 39
Loss for Training on Epoch 39 is 0.190956711769104
Starting epoch number 40
Loss for Training on Epoch 40 is 0.18902122974395752
-------  Class: aeroplane        AP:   0.6107  -------
-------  Class: bicycle          AP:   0.3538  -------
-------  Class: bird             AP:   0.1811  -------
-------  Class: boat             AP:   0.3250  -------
-------  Class: bottle           AP:   0.1308  -------
```

```
-------  Class: bus              AP:   0.2356  -------
-------  Class: car              AP:   0.5583  -------
-------  Class: cat              AP:   0.3399  -------
-------  Class: chair            AP:   0.3927  -------
-------  Class: cow              AP:   0.1742  -------
-------  Class: diningtable      AP:   0.2772  -------
-------  Class: dog              AP:   0.2967  -------
-------  Class: horse            AP:   0.4905  -------
-------  Class: motorbike        AP:   0.4491  -------
-------  Class: person           AP:   0.7550  -------
-------  Class: pottedplant      AP:   0.1547  -------
-------  Class: sheep            AP:   0.0898  -------
-------  Class: sofa             AP:   0.2324  -------
-------  Class: train            AP:   0.5429  -------
-------  Class: tvmonitor        AP:   0.2261  -------
mAP: 0.3408
Avg loss: 0.19189773201942445
Evaluating classifier
Mean Precision Score for Testing on Epoch 40 is 0.34083697199875646
Starting epoch number 41
Loss for Training on Epoch 41 is 0.19463185966014862
Starting epoch number 42
Loss for Training on Epoch 42 is 0.19058771431446075
Starting epoch number 43
Loss for Training on Epoch 43 is 0.18413951992988586
Starting epoch number 44
Loss for Training on Epoch 44 is 0.18861474096775055
Starting epoch number 45
Loss for Training on Epoch 45 is 0.18376581370830536
-------  Class: aeroplane        AP:   0.5499  -------
-------  Class: bicycle          AP:   0.4002  -------
-------  Class: bird             AP:   0.2147  -------
-------  Class: boat             AP:   0.3703  -------
-------  Class: bottle           AP:   0.1244  -------
-------  Class: bus              AP:   0.2368  -------
-------  Class: car              AP:   0.5655  -------
-------  Class: cat              AP:   0.3434  -------
-------  Class: chair            AP:   0.4133  -------
-------  Class: cow              AP:   0.1471  -------
-------  Class: diningtable      AP:   0.2872  -------
-------  Class: dog              AP:   0.2614  -------
-------  Class: horse            AP:   0.4559  -------
-------  Class: motorbike        AP:   0.4070  -------
-------  Class: person           AP:   0.7613  -------
-------  Class: pottedplant      AP:   0.1644  -------
-------  Class: sheep            AP:   0.1298  -------
-------  Class: sofa             AP:   0.2945  -------
-------  Class: train            AP:   0.4808  -------
```

```
-------  Class: tvmonitor        AP:   0.2168  -------
mAP: 0.3412
Avg loss: 0.1989801198244095
Evaluating classifier
Mean Precision Score for Testing on Epoch 45 is 0.34124561176452267
Starting epoch number 46
Loss for Training on Epoch 46 is 0.18702268600463867
Starting epoch number 47
Loss for Training on Epoch 47 is 0.18175064027309418
Starting epoch number 48
Loss for Training on Epoch 48 is 0.17719262838363647
Starting epoch number 49
Loss for Training on Epoch 49 is 0.17102868854999542
Starting epoch number 50
Loss for Training on Epoch 50 is 0.17784133553504944
-------  Class: aeroplane        AP:   0.6294  -------
-------  Class: bicycle          AP:   0.3448  -------
-------  Class: bird             AP:   0.2178  -------
-------  Class: boat             AP:   0.3566  -------
-------  Class: bottle           AP:   0.1303  -------
-------  Class: bus              AP:   0.2669  -------
-------  Class: car              AP:   0.5934  -------
-------  Class: cat              AP:   0.3014  -------
-------  Class: chair            AP:   0.4208  -------
-------  Class: cow              AP:   0.1459  -------
-------  Class: diningtable      AP:   0.2896  -------
-------  Class: dog              AP:   0.2498  -------
-------  Class: horse            AP:   0.4929  -------
-------  Class: motorbike        AP:   0.3879  -------
-------  Class: person           AP:   0.7467  -------
-------  Class: pottedplant      AP:   0.1636  -------
-------  Class: sheep            AP:   0.1212  -------
-------  Class: sofa             AP:   0.2260  -------
-------  Class: train            AP:   0.5030  -------
-------  Class: tvmonitor        AP:   0.2434  -------
mAP: 0.3416
Avg loss: 0.1968972124159336
Evaluating classifier
Mean Precision Score for Testing on Epoch 50 is 0.34156199890125183
```

[9]:
```
plot_losses(train_losses, val_losses, test_frequency, num_epochs)
plot_mAP(train_mAPs, val_mAPs, test_frequency, num_epochs)
```


Loss Plot
train
val
Loss
Epoch
0.350
0.325
0.300
0.275
0.250
0.225
0.200
0.175
0
10
20
30
40
50


mAP Plot
train
val
mAP
Epoch
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
10
20
30
40
50

[10]:
```
mAP_test, test_loss, test_aps = test_classifier(test_loader, classifier,␣
 ↪criterion)
print(mAP_test)
```

```
-------  Class: aeroplane        AP:   0.6208  -------
-------  Class: bicycle          AP:   0.3731  -------
-------  Class: bird             AP:   0.2344  -------
-------  Class: boat             AP:   0.3791  -------
-------  Class: bottle           AP:   0.1503  -------
-------  Class: bus              AP:   0.1948  -------
-------  Class: car              AP:   0.6192  -------
-------  Class: cat              AP:   0.2888  -------
-------  Class: chair            AP:   0.4048  -------
-------  Class: cow              AP:   0.1532  -------
-------  Class: diningtable      AP:   0.3429  -------
-------  Class: dog              AP:   0.2339  -------
-------  Class: horse            AP:   0.6464  -------
-------  Class: motorbike        AP:   0.4136  -------
-------  Class: person           AP:   0.7652  -------
-------  Class: pottedplant      AP:   0.1833  -------
-------  Class: sheep            AP:   0.1869  -------
-------  Class: sofa             AP:   0.2857  -------
-------  Class: train            AP:   0.4973  -------
-------  Class: tvmonitor        AP:   0.2162  -------
mAP: 0.3595
Avg loss: 0.1903794669570067
0.3594972393845192
```

[11]:
```
torch.save(classifier.state_dict(), './voc_my_best_classifier.pth')
output_submission_csv('my_solution.csv', test_aps)
```