MIS 583 Assignment 5: YOLO Object Detection on PASCAL VOC

Before we start, please put your name and SID in following format:
: LASTNAME Firstname, ?00000000 // e.g.) 李晨愷 M114020035

Your Answer:
Hi I'm 游雅淇, B104020012.

Google Colab Setup

Next we need to run a few commands to set up our environment on Google Colab. If you are running this notebook on a local machine you can skip this section.

Run the following cell to mount your Google Drive. Follow the link, sign in to your Google account (the same account you used to store this notebook!) and copy the authorization code into the text box that appears below.

In [ ]:
```
from google.colab import drive
drive.mount('/content/drive', force_remount=True)
```

```
Mounted at /content/drive
```

How to Get Data

請先到共用雲端硬碟將檔案 `VOCdevkit_2007.zip` ，建立捷徑到自己的雲端硬碟中。

> 操作步驟
>
> 1. 點開雲端連結
> 2. 點選右上角「新增雲端硬碟捷徑」
> 3. 點選「我的雲端硬碟」
> 4. 點選「新增捷徑」

完成以上流程會在你的雲端硬碟中建立一個檔案的捷徑，接著我們在colab中取得權限即可使用。

Unzip Data

解壓縮 `VOCdevkit_2007.zip`

- `VOC2007` : 包含了train/val的所有圖片
- `VOC2007test` : 包含了test的所有圖片

其中 `train` 的圖片 3756 張， `val` 的圖片 1255 張， `test` 的圖片 4950 張。

注意: 若有另外設定存放在雲端硬碟中的路徑，請記得本處路徑也須做更動。

Notice: Please put "VOCdevkit_2007" folder under data folder.

In [ ]:
```
!unzip -qq ./drive/MyDrive/DeepLearning/A5/data/VOCdevkit_2007.zip
```

In [ ]:
```
%cd ./drive/MyDrive/A5
```

```
[Errno 2] No such file or directory: './drive/MyDrive/A5'
/content/drive/MyDrive/A5
```

Import package

In [ ]:
```
import os
import random

import cv2
import numpy as np

import csv

import torch
from torch.utils.data import DataLoader
from torchvision import models

from src.resnet_yolo import resnet50
from src.densenet_yolo import densenet121
from yolo_loss import YoloLoss
from src.dataset import VocDetectorDataset
from src.eval_voc import evaluate, test_evaluate
from src.predict import predict_image
from src.config import VOC_CLASSES, COLORS
from kaggle_submission import write_csv

import matplotlib.pyplot as plt
import collections

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

```
The autoreload extension is already loaded. To reload it, use:
  %reload_ext autoreload
```

Initialization

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

In [ ]:
```
# YOLO network hyperparameters
B = 2  # number of bounding box predictions per cell
S = 14  # width/height of network output grid (larger than 7x7 from paper s
```

To implement Yolo we will rely on a pretrained classifier as the backbone for our detection network. PyTorch offers a variety of models which are pretrained on ImageNet in the `torchvision.models` package. In particular, we will use the ResNet50 architecture as a base for our detector. This is different from the base architecture in the Yolo paper and also results in a different output grid size (14x14 instead of 7x7).

Models are typically pretrained on ImageNet since the dataset is very large (> 1 million images) and widely used. The pretrained model provides a very useful weight initialization for our detector, so that the network is able to learn quickly and effectively.

In [ ]:
```
load_network_path = None #'checkpoints/best_detector.pth'
pretrained = True

# use to load a previously trained network
if load_network_path is not None:
    print('Loading saved network from {}'.format(load_network_path))
    net = densenet121(S=S).to(device)
    net.load_state_dict(torch.load(load_network_path))
else:
    print('Load pre-trained model')
    net = densenet121(pretrained=pretrained, S=S).to(device)
```

```
Load pre-trained model
/usr/local/lib/python3.10/dist-packages/torchvision/models/_utils.py:208: U
serWarning: The parameter 'pretrained' is deprecated since 0.13 and may be
removed in the future, please use 'weights' instead.
  warnings.warn(
/usr/local/lib/python3.10/dist-packages/torchvision/models/_utils.py:223: U
serWarning: Arguments other than a weight enum or `None` for 'weights' are
deprecated since 0.13 and may be removed in the future. The current behavio
r is equivalent to passing `weights=DenseNet121_Weights.IMAGENET1K_V1`. You
can also use `weights=DenseNet121_Weights.DEFAULT` to get the most up-to-da
te weights.
  warnings.warn(msg)
```

In [ ]:
```
learning_rate = 0.001
num_epochs = 60
batch_size = 24

# Yolo loss component coefficients (as given in Yolo v1 paper)
lambda_coord = 5
lambda_noobj = 0.5
```

Reading Pascal Data

The train dataset loader also using a variety of data augmentation techniques including random shift, scaling, crop, and flips. Data augmentation is slightly more complicated for detection datasets since the bounding box annotations must be kept consistent throughout the transformations.

Since the output of the detector network we train is an SxSx(B*5+C), we use an encoder to convert the original bounding box coordinates into relative grid bounding box coordinates corresponding to the expected output. We also use a decoder which allows us to convert the opposite direction into image coordinate bounding boxes.

Notice: Please put "VOCdevkit_2007" folder under data folder.

In [ ]:
```
file_root_train = 'data/VOCdevkit_2007/VOC2007/JPEGImages/'
annotation_file_train = 'data/voc2007train.txt'

train_dataset = VocDetectorDataset(root_img_dir=file_root_train,dataset_file
train_loader = DataLoader(train_dataset,batch_size=batch_size,shuffle=True,n
print('Loaded %d train images' % len(train_dataset))
```

```
Initializing dataset
Loaded 3756 train images
```

In [ ]:
```
file_root_val = 'data/VOCdevkit_2007/VOC2007/JPEGImages/'
annotation_file_val = 'data/voc2007val.txt'

val_dataset = VocDetectorDataset(root_img_dir=file_root_val,dataset_file=ann
val_loader = DataLoader(val_dataset,batch_size=batch_size,shuffle=False,num_
print('Loaded %d val images' % len(val_dataset))
```

```
Initializing dataset
Loaded 1255 val images
```

In [ ]:
```
data = train_dataset[0]
```

Set up training tools

In [ ]:
```
criterion = YoloLoss(S, B, lambda_coord, lambda_noobj)
optimizer = torch.optim.SGD(net.parameters(), lr=learning_rate, momentum=0.9
```

In [ ]:
```
# data normalization
```

Train detector

In [ ]:
```
best_val_loss = np.inf
learning_rate = 1e-3
for epoch in range(num_epochs):
    net.train()

    # Update learning rate late in training
    if epoch == 30 or epoch == 40:
        learning_rate /= 10.0

    for param_group in optimizer.param_groups:
        param_group['lr'] = learning_rate

    print('\n\nStarting epoch %d / %d' % (epoch + 1, num_epochs))
    print('Learning Rate for this epoch: {}'.format(learning_rate))

    total_loss = collections.defaultdict(int)

    for i, data in enumerate(train_loader):
        data = (item.to(device) for item in data)
        images, target_boxes, target_cls, has_object_map = data
        pred = net(images)
        loss_dict = criterion(pred, target_boxes, target_cls, has_object_map
        for key in loss_dict:
            total_loss[key] += loss_dict[key].item()

        optimizer.zero_grad()
        loss_dict['total_loss'].backward()
        optimizer.step()

        if (i+1) % 50 == 0:
            outstring = 'Epoch [%d/%d], Iter [%d/%d], Loss: ' % ((epoch+1, n
            outstring += ', '.join( "%s=%.3f" % (key[:-5], val / (i+1)) for 
            print(outstring)

    # evaluate the network on the val data
    if (epoch + 1) % 5 == 0:
```

```
        val_aps = evaluate(net, val_dataset_file=annotation_file_val, img_rc
        print(epoch, val_aps)
    with torch.no_grad():
        val_loss = 0.0
        net.eval()
        for i, data in enumerate(val_loader):
            data = (item.to(device) for item in data)
            images, target_boxes, target_cls, has_object_map = data

            pred = net(images)
            loss_dict = criterion(pred, target_boxes, target_cls, has_objec
            val_loss += loss_dict['total_loss'].item()
        val_loss /= len(val_loader)

    if best_val_loss > val_loss:
        best_val_loss = val_loss
        print('Updating best val loss: %.5f' % best_val_loss)
        torch.save(net.state_dict(),'checkpoints/best_detector.pth')

    if (epoch+1) in [5, 10, 20, 30, 40, 50, 60, 70, 80]:
        torch.save(net.state_dict(),'checkpoints/detector_epoch_%d.pth' % (e

    torch.save(net.state_dict(),'checkpoints/detector.pth')
```

```
Starting epoch 1 / 60
Learning Rate for this epoch: 0.001
Epoch [1/60], Iter [50/157], Loss: total=23.284, reg=0.652, containing_obj=
1.509, no_obj=25.329, cls=5.851
Epoch [1/60], Iter [100/157], Loss: total=15.448, reg=0.566, containing_obj
=1.711, no_obj=13.289, cls=4.264
Epoch [1/60], Iter [150/157], Loss: total=12.426, reg=0.515, containing_obj
=1.730, no_obj=9.159, cls=3.540
Updating best val loss: 6.67858


Starting epoch 2 / 60
Learning Rate for this epoch: 0.001
Epoch [2/60], Iter [50/157], Loss: total=6.055, reg=0.423, containing_obj=
1.637, no_obj=0.978, cls=1.815
Epoch [2/60], Iter [100/157], Loss: total=5.860, reg=0.416, containing_obj=
1.577, no_obj=0.981, cls=1.710
Epoch [2/60], Iter [150/157], Loss: total=5.648, reg=0.406, containing_obj=
1.521, no_obj=0.978, cls=1.610
Updating best val loss: 5.51628


Starting epoch 3 / 60
Learning Rate for this epoch: 0.001
Epoch [3/60], Iter [50/157], Loss: total=4.940, reg=0.372, containing_obj=
1.364, no_obj=0.963, cls=1.235
Epoch [3/60], Iter [100/157], Loss: total=4.970, reg=0.376, containing_obj=
1.363, no_obj=0.946, cls=1.253
Epoch [3/60], Iter [150/157], Loss: total=4.935, reg=0.378, containing_obj=
1.349, no_obj=0.936, cls=1.226
Updating best val loss: 5.00272


Starting epoch 4 / 60
Learning Rate for this epoch: 0.001
Epoch [4/60], Iter [50/157], Loss: total=4.414, reg=0.347, containing_obj=
1.258, no_obj=0.850, cls=0.994
Epoch [4/60], Iter [100/157], Loss: total=4.452, reg=0.354, containing_obj=
1.275, no_obj=0.845, cls=0.986
Epoch [4/60], Iter [150/157], Loss: total=4.423, reg=0.350, containing_obj=
1.268, no_obj=0.855, cls=0.979
Updating best val loss: 4.72996


Starting epoch 5 / 60
Learning Rate for this epoch: 0.001
Epoch [5/60], Iter [50/157], Loss: total=4.063, reg=0.328, containing_obj=
1.173, no_obj=0.856, cls=0.823
Epoch [5/60], Iter [100/157], Loss: total=4.123, reg=0.333, containing_obj=
1.191, no_obj=0.840, cls=0.847
Epoch [5/60], Iter [150/157], Loss: total=4.165, reg=0.340, containing_obj=
1.201, no_obj=0.835, cls=0.846
---Evaluate model on test samples---
```

```
100%|██████████| 1255/1255 [00:58<00:00, 21.48it/s]
```

```
---class aeroplane ap 0.5128774410218597---
---class bicycle ap 0.4080926689793873---
---class bird ap 0.3591520464190196---
---class boat ap 0.14623523566161445---
---class bottle ap 0.2023822459299064---
---class bus ap 0.19506493506493505---
---class car ap 0.4892260661259066---
---class cat ap 0.4284075116376122---
---class chair ap 0.19844370539944864---
---class cow ap 0.12310142623441855---
---class diningtable ap 0.0--- (no predictions for this class)
---class dog ap 0.46758777962266385---
---class horse ap 0.38670109531706004---
---class motorbike ap 0.4152654665198618---
---class person ap 0.3772111220540883---
---class pottedplant ap 0.09845305168941854---
---class sheep ap 0.12333593474557089---
---class sofa ap 0.31616105358795416---
---class train ap 0.5467133637278708---
---class tvmonitor ap 0.5095881915586921---
---map 0.31520001706486445---
4 [0.5128774410218597, 0.4080926689793873, 0.3591520464190196, 0.1462352356
6161445, 0.2023822459299064, 0.19506493506493505, 0.4892260661259066, 0.428
4075116376122, 0.19844370539944864, 0.12310142623441855, 0.0, 0.46758777962
266385, 0.38670109531706004, 0.4152654665198618, 0.3772111220540883, 0.0984
5305168941854, 0.12333593474557089, 0.31616105358795416, 0.546713363727870
8, 0.5095881915586921]
Updating best val loss: 4.53227


Starting epoch 6 / 60
Learning Rate for this epoch: 0.001
Epoch [6/60], Iter [50/157], Loss: total=3.913, reg=0.316, containing_obj=
1.162, no_obj=0.833, cls=0.757
Epoch [6/60], Iter [100/157], Loss: total=3.993, reg=0.326, containing_obj=
1.173, no_obj=0.846, cls=0.768
Epoch [6/60], Iter [150/157], Loss: total=3.938, reg=0.323, containing_obj=
1.153, no_obj=0.845, cls=0.748
Updating best val loss: 4.33410


Starting epoch 7 / 60
Learning Rate for this epoch: 0.001
Epoch [7/60], Iter [50/157], Loss: total=3.736, reg=0.306, containing_obj=
1.167, no_obj=0.782, cls=0.648
Epoch [7/60], Iter [100/157], Loss: total=3.766, reg=0.309, containing_obj=
1.157, no_obj=0.802, cls=0.662
Epoch [7/60], Iter [150/157], Loss: total=3.795, reg=0.311, containing_obj=
1.154, no_obj=0.804, cls=0.686
Updating best val loss: 4.31756


Starting epoch 8 / 60
Learning Rate for this epoch: 0.001
Epoch [8/60], Iter [50/157], Loss: total=3.353, reg=0.274, containing_obj=
1.045, no_obj=0.765, cls=0.558
Epoch [8/60], Iter [100/157], Loss: total=3.574, reg=0.294, containing_obj=
1.108, no_obj=0.785, cls=0.603
Epoch [8/60], Iter [150/157], Loss: total=3.625, reg=0.299, containing_obj=
1.117, no_obj=0.795, cls=0.617
Updating best val loss: 4.16529


Starting epoch 9 / 60
```

```
Learning Rate for this epoch: 0.001
Epoch [9/60], Iter [50/157], Loss: total=3.354, reg=0.281, containing_obj=
1.029, no_obj=0.786, cls=0.529
Epoch [9/60], Iter [100/157], Loss: total=3.379, reg=0.282, containing_obj=
1.044, no_obj=0.781, cls=0.533
Epoch [9/60], Iter [150/157], Loss: total=3.435, reg=0.288, containing_obj=
1.046, no_obj=0.789, cls=0.555


Starting epoch 10 / 60
Learning Rate for this epoch: 0.001
Epoch [10/60], Iter [50/157], Loss: total=3.435, reg=0.292, containing_obj=
1.036, no_obj=0.840, cls=0.521
Epoch [10/60], Iter [100/157], Loss: total=3.411, reg=0.287, containing_obj
=1.045, no_obj=0.814, cls=0.526
Epoch [10/60], Iter [150/157], Loss: total=3.401, reg=0.286, containing_obj
=1.045, no_obj=0.798, cls=0.525
---Evaluate model on test samples---
```

```
100%|██████████| 1255/1255 [00:53<00:00, 23.55it/s]
```

```
---class aeroplane ap 0.5117104167634492---
---class bicycle ap 0.4980647210609129---
---class bird ap 0.5042727868691758---
---class boat ap 0.2892333445962617---
---class bottle ap 0.17754853700339224---
---class bus ap 0.41616254246683065---
---class car ap 0.6012487227559637---
---class cat ap 0.5790325932580195---
---class chair ap 0.2709576660209653---
---class cow ap 0.31085989601911446---
---class diningtable ap 0.1571410786536837---
---class dog ap 0.45423074727987073---
---class horse ap 0.5916310443076743---
---class motorbike ap 0.49200443073480005---
---class person ap 0.4719044011513074---
---class pottedplant ap 0.19203491605232606---
---class sheep ap 0.35983788486766405---
---class sofa ap 0.36259328127591345---
---class train ap 0.667016656307068---
---class tvmonitor ap 0.5282259543334494---
---map 0.4217855810888921---
9 [0.5117104167634492, 0.4980647210609129, 0.5042727868691758, 0.2892333445
962617, 0.17754853700339224, 0.41616254246683065, 0.6012487227559637, 0.579
0325932580195, 0.2709576660209653, 0.31085989601911446, 0.1571410786536837,
0.45423074727987073, 0.5916310443076743, 0.49200443073480005, 0.47190440115
13074, 0.19203491605232606, 0.35983788486766405, 0.36259328127591345, 0.667
016656307068, 0.5282259543334494]
Updating best val loss: 4.15944


Starting epoch 11 / 60
Learning Rate for this epoch: 0.001
Epoch [11/60], Iter [50/157], Loss: total=3.357, reg=0.286, containing_obj=
1.023, no_obj=0.814, cls=0.495
Epoch [11/60], Iter [100/157], Loss: total=3.324, reg=0.282, containing_obj
=1.019, no_obj=0.799, cls=0.496
Epoch [11/60], Iter [150/157], Loss: total=3.321, reg=0.282, containing_obj
=1.018, no_obj=0.794, cls=0.497
Updating best val loss: 4.09678


Starting epoch 12 / 60
Learning Rate for this epoch: 0.001
Epoch [12/60], Iter [50/157], Loss: total=3.191, reg=0.271, containing_obj=
0.997, no_obj=0.782, cls=0.448
Epoch [12/60], Iter [100/157], Loss: total=3.182, reg=0.269, containing_obj
=0.998, no_obj=0.761, cls=0.457
Epoch [12/60], Iter [150/157], Loss: total=3.220, reg=0.271, containing_obj
=1.004, no_obj=0.774, cls=0.473
Updating best val loss: 3.99937


Starting epoch 13 / 60
Learning Rate for this epoch: 0.001
Epoch [13/60], Iter [50/157], Loss: total=3.060, reg=0.258, containing_obj=
0.964, no_obj=0.770, cls=0.420
Epoch [13/60], Iter [100/157], Loss: total=3.061, reg=0.258, containing_obj
=0.958, no_obj=0.788, cls=0.417
Epoch [13/60], Iter [150/157], Loss: total=3.070, reg=0.259, containing_obj
=0.972, no_obj=0.776, cls=0.417


Starting epoch 14 / 60
Learning Rate for this epoch: 0.001
```

```
Epoch [14/60], Iter [50/157], Loss: total=3.089, reg=0.268, containing_obj=
0.957, no_obj=0.784, cls=0.399
Epoch [14/60], Iter [100/157], Loss: total=3.049, reg=0.263, containing_obj
=0.957, no_obj=0.766, cls=0.395
Epoch [14/60], Iter [150/157], Loss: total=3.023, reg=0.257, containing_obj
=0.957, no_obj=0.765, cls=0.396
Updating best val loss: 3.88821


Starting epoch 15 / 60
Learning Rate for this epoch: 0.001
Epoch [15/60], Iter [50/157], Loss: total=2.877, reg=0.249, containing_obj=
0.887, no_obj=0.782, cls=0.356
Epoch [15/60], Iter [100/157], Loss: total=2.927, reg=0.254, containing_obj
=0.907, no_obj=0.785, cls=0.358
Epoch [15/60], Iter [150/157], Loss: total=2.943, reg=0.254, containing_obj
=0.910, no_obj=0.781, cls=0.371
---Evaluate model on test samples---
```

```
100%|██████████| 1255/1255 [00:54<00:00, 23.10it/s]
```

```
---class aeroplane ap 0.6284671251340119---
---class bicycle ap 0.5969974880742045---
---class bird ap 0.4228659841927902---
---class boat ap 0.2487633505400162---
---class bottle ap 0.2733061446403895---
---class bus ap 0.49335326538103735---
---class car ap 0.641302560205872---
---class cat ap 0.7059173770855257---
---class chair ap 0.30890833363036674---
---class cow ap 0.48072676890874616---
---class diningtable ap 0.2195648073696854---
---class dog ap 0.5402492654819987---
---class horse ap 0.5187337783692301---
---class motorbike ap 0.4764289566992827---
---class person ap 0.5185821625434274---
---class pottedplant ap 0.19795422542672084---
---class sheep ap 0.2823588897208816---
---class sofa ap 0.44679944934241805---
---class train ap 0.708707190020227---
---class tvmonitor ap 0.5750144091675318---
---map 0.46425007659671824---
14 [0.6284671251340119, 0.5969974880742045, 0.4228659841927902, 0.248763350
5400162, 0.2733061446403895, 0.49335326538103735, 0.641302560205872, 0.7059
173770855257, 0.30890833363036674, 0.48072676890874616, 0.2195648073696854,
0.5402492654819987, 0.5187337783692301, 0.4764289566992827, 0.5185821625434
274, 0.19795422542672084, 0.2823588897208816, 0.44679944934241805, 0.708707
190020227, 0.5750144091675318]
Updating best val loss: 3.84921


Starting epoch 16 / 60
Learning Rate for this epoch: 0.001
Epoch [16/60], Iter [50/157], Loss: total=2.864, reg=0.244, containing_obj=
0.878, no_obj=0.752, cls=0.390
Epoch [16/60], Iter [100/157], Loss: total=2.905, reg=0.247, containing_obj
=0.908, no_obj=0.756, cls=0.382
Epoch [16/60], Iter [150/157], Loss: total=2.918, reg=0.250, containing_obj
=0.912, no_obj=0.760, cls=0.373


Starting epoch 17 / 60
Learning Rate for this epoch: 0.001
Epoch [17/60], Iter [50/157], Loss: total=2.864, reg=0.249, containing_obj=
0.891, no_obj=0.775, cls=0.341
Epoch [17/60], Iter [100/157], Loss: total=2.876, reg=0.251, containing_obj
=0.882, no_obj=0.788, cls=0.347
Epoch [17/60], Iter [150/157], Loss: total=2.837, reg=0.246, containing_obj
=0.877, no_obj=0.776, cls=0.342


Starting epoch 18 / 60
Learning Rate for this epoch: 0.001
Epoch [18/60], Iter [50/157], Loss: total=2.814, reg=0.244, containing_obj=
0.890, no_obj=0.744, cls=0.333
Epoch [18/60], Iter [100/157], Loss: total=2.826, reg=0.244, containing_obj
=0.898, no_obj=0.753, cls=0.331
Epoch [18/60], Iter [150/157], Loss: total=2.808, reg=0.240, containing_obj
=0.887, no_obj=0.755, cls=0.342
Updating best val loss: 3.83189


Starting epoch 19 / 60
Learning Rate for this epoch: 0.001
Epoch [19/60], Iter [50/157], Loss: total=2.602, reg=0.220, containing_obj=
```

```
0.821, no_obj=0.759, cls=0.300
Epoch [19/60], Iter [100/157], Loss: total=2.676, reg=0.227, containing_obj
=0.854, no_obj=0.757, cls=0.307
Epoch [19/60], Iter [150/157], Loss: total=2.701, reg=0.230, containing_obj
=0.855, no_obj=0.756, cls=0.316


Starting epoch 20 / 60
Learning Rate for this epoch: 0.001
Epoch [20/60], Iter [50/157], Loss: total=2.640, reg=0.231, containing_obj=
0.836, no_obj=0.728, cls=0.285
Epoch [20/60], Iter [100/157], Loss: total=2.664, reg=0.233, containing_obj
=0.837, no_obj=0.741, cls=0.292
Epoch [20/60], Iter [150/157], Loss: total=2.678, reg=0.233, containing_obj
=0.846, no_obj=0.743, cls=0.298
---Evaluate model on test samples---
```

```
100%|██████████| 1255/1255 [00:54<00:00, 22.86it/s]
```

```
---class aeroplane ap 0.5567289778568962---
---class bicycle ap 0.5965211766735979---
---class bird ap 0.49615716511938074---
---class boat ap 0.28964822427301506---
---class bottle ap 0.25131081805156824---
---class bus ap 0.5165638597249916---
---class car ap 0.6456949209177731---
---class cat ap 0.681195657278251---
---class chair ap 0.3164439773856242---
---class cow ap 0.5048663153659372---
---class diningtable ap 0.33361115812189535---
---class dog ap 0.5726212548596231---
---class horse ap 0.6330027775791869---
---class motorbike ap 0.5746623093659794---
---class person ap 0.4784480102910313---
---class pottedplant ap 0.18515028107540848---
---class sheep ap 0.37224171059664546---
---class sofa ap 0.3834791306561304---
---class train ap 0.6594695062021972---
---class tvmonitor ap 0.6712240022787418---
---map 0.48595206168369376---
19 [0.5567289778568962, 0.5965211766735979, 0.49615716511938074, 0.28964822
427301506, 0.25131081805156824, 0.5165638597249916, 0.6456949209177731, 0.6
81195657278251, 0.3164439773856242, 0.5048663153659372, 0.3336111581218953
5, 0.5726212548596231, 0.6330027775791869, 0.5746623093659794, 0.4784480102
910313, 0.18515028107540848, 0.37224171059664546, 0.3834791306561304, 0.659
4695062021972, 0.6712240022787418]


Starting epoch 21 / 60
Learning Rate for this epoch: 0.001
Epoch [21/60], Iter [50/157], Loss: total=2.593, reg=0.222, containing_obj=
0.811, no_obj=0.734, cls=0.306
Epoch [21/60], Iter [100/157], Loss: total=2.633, reg=0.229, containing_obj
=0.819, no_obj=0.745, cls=0.295
Epoch [21/60], Iter [150/157], Loss: total=2.628, reg=0.227, containing_obj
=0.826, no_obj=0.746, cls=0.294


Starting epoch 22 / 60
Learning Rate for this epoch: 0.001
Epoch [22/60], Iter [50/157], Loss: total=2.672, reg=0.226, containing_obj=
0.855, no_obj=0.787, cls=0.291
Epoch [22/60], Iter [100/157], Loss: total=2.576, reg=0.218, containing_obj
=0.823, no_obj=0.756, cls=0.283
Epoch [22/60], Iter [150/157], Loss: total=2.561, reg=0.219, containing_obj
=0.812, no_obj=0.745, cls=0.279
Updating best val loss: 3.80138


Starting epoch 23 / 60
Learning Rate for this epoch: 0.001
Epoch [23/60], Iter [50/157], Loss: total=2.501, reg=0.212, containing_obj=
0.802, no_obj=0.719, cls=0.277
Epoch [23/60], Iter [100/157], Loss: total=2.467, reg=0.212, containing_obj
=0.794, no_obj=0.707, cls=0.262
Epoch [23/60], Iter [150/157], Loss: total=2.525, reg=0.215, containing_obj
=0.814, no_obj=0.723, cls=0.273


Starting epoch 24 / 60
Learning Rate for this epoch: 0.001
Epoch [24/60], Iter [50/157], Loss: total=2.496, reg=0.217, containing_obj=
0.792, no_obj=0.733, cls=0.251
```

```
Epoch [24/60], Iter [100/157], Loss: total=2.478, reg=0.214, containing_obj
=0.785, no_obj=0.740, cls=0.251
Epoch [24/60], Iter [150/157], Loss: total=2.479, reg=0.216, containing_obj
=0.779, no_obj=0.740, cls=0.252


Starting epoch 25 / 60
Learning Rate for this epoch: 0.001
Epoch [25/60], Iter [50/157], Loss: total=2.543, reg=0.224, containing_obj=
0.810, no_obj=0.721, cls=0.253
Epoch [25/60], Iter [100/157], Loss: total=2.525, reg=0.222, containing_obj
=0.795, no_obj=0.725, cls=0.259
Epoch [25/60], Iter [150/157], Loss: total=2.486, reg=0.217, containing_obj
=0.779, no_obj=0.731, cls=0.255
---Evaluate model on test samples---
100%|██████████| 1255/1255 [00:53<00:00, 23.30it/s]
```

```
---class aeroplane ap 0.5095303375452055---
---class bicycle ap 0.5636230268172828---
---class bird ap 0.4868602388369571---
---class boat ap 0.2557331174949898---
---class bottle ap 0.25031323081927626---
---class bus ap 0.43905409953157853---
---class car ap 0.5902147126052246---
---class cat ap 0.6138029832201106---
---class chair ap 0.30726784833897863---
---class cow ap 0.4888058132364483---
---class diningtable ap 0.2462738319881177---
---class dog ap 0.5798287063764304---
---class horse ap 0.5628228256651349---
---class motorbike ap 0.4903721358657399---
---class person ap 0.48483245619229187---
---class pottedplant ap 0.19852983374359182---
---class sheep ap 0.30695684433316023---
---class sofa ap 0.41033692023992163---
---class train ap 0.6903693063580909---
---class tvmonitor ap 0.5636455061022175---
---map 0.45195868876553746---
24 [0.5095303375452055, 0.5636230268172828, 0.4868602388369571, 0.255733117
4949898, 0.25031323081927626, 0.43905409953157853, 0.5902147126052246, 0.61
38029832201106, 0.30726784833897863, 0.4888058132364483, 0.246273831988117
7, 0.5798287063764304, 0.5628228256651349, 0.4903721358657399, 0.4848324561
9229187, 0.19852983374359182, 0.30695684433316023, 0.41033692023992163, 0.6
903693063580909, 0.5636455061022175]


Starting epoch 26 / 60
Learning Rate for this epoch: 0.001
Epoch [26/60], Iter [50/157], Loss: total=2.427, reg=0.216, containing_obj=
0.767, no_obj=0.713, cls=0.225
Epoch [26/60], Iter [100/157], Loss: total=2.393, reg=0.212, containing_obj
=0.738, no_obj=0.714, cls=0.240
Epoch [26/60], Iter [150/157], Loss: total=2.428, reg=0.213, containing_obj
=0.764, no_obj=0.709, cls=0.247


Starting epoch 27 / 60
Learning Rate for this epoch: 0.001
Epoch [27/60], Iter [50/157], Loss: total=2.418, reg=0.212, containing_obj=
0.756, no_obj=0.734, cls=0.236
Epoch [27/60], Iter [100/157], Loss: total=2.413, reg=0.212, containing_obj
=0.749, no_obj=0.734, cls=0.236
Epoch [27/60], Iter [150/157], Loss: total=2.421, reg=0.212, containing_obj
=0.758, no_obj=0.730, cls=0.237
Updating best val loss: 3.78541


Starting epoch 28 / 60
Learning Rate for this epoch: 0.001
Epoch [28/60], Iter [50/157], Loss: total=2.252, reg=0.197, containing_obj=
0.700, no_obj=0.711, cls=0.212
Epoch [28/60], Iter [100/157], Loss: total=2.323, reg=0.203, containing_obj
=0.723, no_obj=0.709, cls=0.228
Epoch [28/60], Iter [150/157], Loss: total=2.335, reg=0.202, containing_obj
=0.733, no_obj=0.708, cls=0.237


Starting epoch 29 / 60
Learning Rate for this epoch: 0.001
Epoch [29/60], Iter [50/157], Loss: total=2.228, reg=0.198, containing_obj=
0.687, no_obj=0.707, cls=0.198
```

```
Epoch [29/60], Iter [100/157], Loss: total=2.269, reg=0.200, containing_obj
=0.698, no_obj=0.710, cls=0.216
Epoch [29/60], Iter [150/157], Loss: total=2.305, reg=0.201, containing_obj
=0.730, no_obj=0.701, cls=0.217


Starting epoch 30 / 60
Learning Rate for this epoch: 0.001
Epoch [30/60], Iter [50/157], Loss: total=2.292, reg=0.201, containing_obj=
0.733, no_obj=0.707, cls=0.198
Epoch [30/60], Iter [100/157], Loss: total=2.299, reg=0.202, containing_obj
=0.722, no_obj=0.725, cls=0.206
Epoch [30/60], Iter [150/157], Loss: total=2.308, reg=0.203, containing_obj
=0.728, no_obj=0.720, cls=0.207
---Evaluate model on test samples---
100%|██████████| 1255/1255 [00:54<00:00, 22.85it/s]
```

```
---class aeroplane ap 0.6177079387884008---
---class bicycle ap 0.5379795867432878---
---class bird ap 0.46769690909452794---
---class boat ap 0.3011821529571555---
---class bottle ap 0.16234119100641375---
---class bus ap 0.47799867421228814---
---class car ap 0.5934834294259874---
---class cat ap 0.6137310110142278---
---class chair ap 0.32715288188207126---
---class cow ap 0.4244127257773749---
---class diningtable ap 0.2430143393911509---
---class dog ap 0.5553825523010245---
---class horse ap 0.5397364128997498---
---class motorbike ap 0.3955670249565824---
---class person ap 0.42082889381186483---
---class pottedplant ap 0.14798643441824103---
---class sheep ap 0.2501305041035306---
---class sofa ap 0.4501045822944506---
---class train ap 0.6859155228309466---
---class tvmonitor ap 0.5844261135102953---
---map 0.43983894407097857---
29 [0.6177079387884008, 0.5379795867432878, 0.46769690909452794, 0.30118215
29571555, 0.16234119100641375, 0.47799867421228814, 0.5934834294259874, 0.6
137310110142278, 0.32715288188207126, 0.4244127257773749, 0.243014339391150
9, 0.5553825523010245, 0.5397364128997498, 0.3955670249565824, 0.4208288938
1186483, 0.14798643441824103, 0.2501305041035306, 0.4501045822944506, 0.685
9155228309466, 0.5844261135102953]


Starting epoch 31 / 60
Learning Rate for this epoch: 0.0001
Epoch [31/60], Iter [50/157], Loss: total=2.220, reg=0.200, containing_obj=
0.694, no_obj=0.679, cls=0.186
Epoch [31/60], Iter [100/157], Loss: total=2.150, reg=0.191, containing_obj
=0.669, no_obj=0.687, cls=0.184
Epoch [31/60], Iter [150/157], Loss: total=2.139, reg=0.191, containing_obj
=0.664, no_obj=0.686, cls=0.180
Updating best val loss: 3.72051


Starting epoch 32 / 60
Learning Rate for this epoch: 0.0001
Epoch [32/60], Iter [50/157], Loss: total=2.087, reg=0.178, containing_obj=
0.657, no_obj=0.684, cls=0.197
Epoch [32/60], Iter [100/157], Loss: total=2.082, reg=0.181, containing_obj
=0.660, no_obj=0.680, cls=0.177
Epoch [32/60], Iter [150/157], Loss: total=2.057, reg=0.180, containing_obj
=0.648, no_obj=0.679, cls=0.169
Updating best val loss: 3.70326


Starting epoch 33 / 60
Learning Rate for this epoch: 0.0001
Epoch [33/60], Iter [50/157], Loss: total=1.880, reg=0.163, containing_obj=
0.586, no_obj=0.681, cls=0.140
Epoch [33/60], Iter [100/157], Loss: total=1.946, reg=0.169, containing_obj
=0.614, no_obj=0.672, cls=0.152
Epoch [33/60], Iter [150/157], Loss: total=1.976, reg=0.171, containing_obj
=0.631, no_obj=0.671, cls=0.156


Starting epoch 34 / 60
Learning Rate for this epoch: 0.0001
Epoch [34/60], Iter [50/157], Loss: total=2.038, reg=0.184, containing_obj=
```

```
0.632, no_obj=0.664, cls=0.155
Epoch [34/60], Iter [100/157], Loss: total=1.982, reg=0.177, containing_obj
=0.608, no_obj=0.672, cls=0.152
Epoch [34/60], Iter [150/157], Loss: total=2.004, reg=0.179, containing_obj
=0.619, no_obj=0.671, cls=0.157
Updating best val loss: 3.69046


Starting epoch 35 / 60
Learning Rate for this epoch: 0.0001
Epoch [35/60], Iter [50/157], Loss: total=2.007, reg=0.175, containing_obj=
0.634, no_obj=0.662, cls=0.169
Epoch [35/60], Iter [100/157], Loss: total=2.001, reg=0.176, containing_obj
=0.634, no_obj=0.659, cls=0.155
Epoch [35/60], Iter [150/157], Loss: total=1.992, reg=0.175, containing_obj
=0.628, no_obj=0.673, cls=0.154
---Evaluate model on test samples---
```

```
100%|██████████| 1255/1255 [00:53<00:00, 23.33it/s]
```

```
---class aeroplane ap 0.5735311007734852---
---class bicycle ap 0.6210962880908787---
---class bird ap 0.5269692712287606---
---class boat ap 0.3458178432929257---
---class bottle ap 0.36489510912293355---
---class bus ap 0.5369657594161664---
---class car ap 0.6645655508709984---
---class cat ap 0.6992976093495711---
---class chair ap 0.4114211078109351---
---class cow ap 0.5709124250592041---
---class diningtable ap 0.3085704930706092---
---class dog ap 0.6133990462071444---
---class horse ap 0.6014821606144455---
---class motorbike ap 0.5392622162839699---
---class person ap 0.5323956861024063---
---class pottedplant ap 0.19597123388387344---
---class sheep ap 0.39345884292032496---
---class sofa ap 0.46830367303292814---
---class train ap 0.7588700558123438---
---class tvmonitor ap 0.6488596219999982---
---map 0.5188022547471951---
34 [0.5735311007734852, 0.6210962880908787, 0.5269692712287606, 0.345817843
2929257, 0.36489510912293355, 0.5369657594161664, 0.6645655508709984, 0.699
2976093495711, 0.4114211078109351, 0.5709124250592041, 0.3085704930706092,
0.6133990462071444, 0.6014821606144455, 0.5392622162839699, 0.5323956861024
063, 0.19597123388387344, 0.39345884292032496, 0.46830367303292814, 0.75887
00558123438, 0.6488596219999982]
Updating best val loss: 3.67718


Starting epoch 36 / 60
Learning Rate for this epoch: 0.0001
Epoch [36/60], Iter [50/157], Loss: total=2.056, reg=0.182, containing_obj=
0.655, no_obj=0.668, cls=0.157
Epoch [36/60], Iter [100/157], Loss: total=1.935, reg=0.170, containing_obj
=0.608, no_obj=0.669, cls=0.144
Epoch [36/60], Iter [150/157], Loss: total=1.952, reg=0.170, containing_obj
=0.619, no_obj=0.669, cls=0.149


Starting epoch 37 / 60
Learning Rate for this epoch: 0.0001
Epoch [37/60], Iter [50/157], Loss: total=1.982, reg=0.173, containing_obj=
0.631, no_obj=0.642, cls=0.166
Epoch [37/60], Iter [100/157], Loss: total=1.958, reg=0.172, containing_obj
=0.623, no_obj=0.644, cls=0.155
Epoch [37/60], Iter [150/157], Loss: total=1.936, reg=0.170, containing_obj
=0.612, no_obj=0.648, cls=0.151


Starting epoch 38 / 60
Learning Rate for this epoch: 0.0001
Epoch [38/60], Iter [50/157], Loss: total=1.874, reg=0.167, containing_obj=
0.577, no_obj=0.662, cls=0.133
Epoch [38/60], Iter [100/157], Loss: total=1.932, reg=0.170, containing_obj
=0.611, no_obj=0.653, cls=0.142
Epoch [38/60], Iter [150/157], Loss: total=1.918, reg=0.169, containing_obj
=0.604, no_obj=0.659, cls=0.141


Starting epoch 39 / 60
Learning Rate for this epoch: 0.0001
Epoch [39/60], Iter [50/157], Loss: total=1.890, reg=0.168, containing_obj=
0.587, no_obj=0.624, cls=0.149
```

```
Epoch [39/60], Iter [100/157], Loss: total=1.889, reg=0.165, containing_obj
=0.598, no_obj=0.638, cls=0.147
Epoch [39/60], Iter [150/157], Loss: total=1.909, reg=0.167, containing_obj
=0.606, no_obj=0.645, cls=0.144


Starting epoch 40 / 60
Learning Rate for this epoch: 0.0001
Epoch [40/60], Iter [50/157], Loss: total=1.953, reg=0.170, containing_obj=
0.624, no_obj=0.662, cls=0.147
Epoch [40/60], Iter [100/157], Loss: total=1.921, reg=0.169, containing_obj
=0.606, no_obj=0.656, cls=0.141
Epoch [40/60], Iter [150/157], Loss: total=1.936, reg=0.169, containing_obj
=0.619, no_obj=0.654, cls=0.144
---Evaluate model on test samples---
100%|██████████| 1255/1255 [00:52<00:00, 23.77it/s]
```

```
---class aeroplane ap 0.6444464550004724---
---class bicycle ap 0.5831834467893946---
---class bird ap 0.5259217475487383---
---class boat ap 0.32998443027696894---
---class bottle ap 0.3412765027544556---
---class bus ap 0.5461452307215877---
---class car ap 0.6757382343206753---
---class cat ap 0.7028979822068162---
---class chair ap 0.3828901887518413---
---class cow ap 0.5628379951498034---
---class diningtable ap 0.2897252466657395---
---class dog ap 0.5910943408343462---
---class horse ap 0.6367554135924314---
---class motorbike ap 0.5617524659811451---
---class person ap 0.5515845948577743---
---class pottedplant ap 0.19956645934214837---
---class sheep ap 0.37321959274045835---
---class sofa ap 0.4576882910202789---
---class train ap 0.7276521436479862---
---class tvmonitor ap 0.6411873997723524---
---map 0.5162774080987708---
39 [0.6444464550004724, 0.5831834467893946, 0.5259217475487383, 0.329984430
27696894, 0.3412765027544556, 0.5461452307215877, 0.6757382343206753, 0.702
8979822068162, 0.3828901887518413, 0.5628379951498034, 0.2897252466657395,
0.5910943408343462, 0.6367554135924314, 0.5617524659811451, 0.5515845948577
743, 0.19956645934214837, 0.37321959274045835, 0.4576882910202789, 0.727652
1436479862, 0.6411873997723524]


Starting epoch 41 / 60
Learning Rate for this epoch: 1e-05
Epoch [41/60], Iter [50/157], Loss: total=1.810, reg=0.163, containing_obj=
0.548, no_obj=0.632, cls=0.132
Epoch [41/60], Iter [100/157], Loss: total=1.896, reg=0.169, containing_obj
=0.584, no_obj=0.646, cls=0.145
Epoch [41/60], Iter [150/157], Loss: total=1.906, reg=0.169, containing_obj
=0.586, no_obj=0.655, cls=0.147


Starting epoch 42 / 60
Learning Rate for this epoch: 1e-05
Epoch [42/60], Iter [50/157], Loss: total=1.880, reg=0.166, containing_obj=
0.599, no_obj=0.639, cls=0.132
Epoch [42/60], Iter [100/157], Loss: total=1.821, reg=0.158, containing_obj
=0.567, no_obj=0.660, cls=0.133
Epoch [42/60], Iter [150/157], Loss: total=1.865, reg=0.162, containing_obj
=0.582, no_obj=0.664, cls=0.140


Starting epoch 43 / 60
Learning Rate for this epoch: 1e-05
Epoch [43/60], Iter [50/157], Loss: total=1.905, reg=0.171, containing_obj=
0.593, no_obj=0.643, cls=0.137
Epoch [43/60], Iter [100/157], Loss: total=1.932, reg=0.172, containing_obj
=0.597, no_obj=0.654, cls=0.148
Epoch [43/60], Iter [150/157], Loss: total=1.902, reg=0.169, containing_obj
=0.587, no_obj=0.656, cls=0.144


Starting epoch 44 / 60
Learning Rate for this epoch: 1e-05
Epoch [44/60], Iter [50/157], Loss: total=1.870, reg=0.164, containing_obj=
0.574, no_obj=0.675, cls=0.138
Epoch [44/60], Iter [100/157], Loss: total=1.866, reg=0.164, containing_obj
```

```
=0.576, no_obj=0.668, cls=0.135
Epoch [44/60], Iter [150/157], Loss: total=1.885, reg=0.166, containing_obj
=0.586, no_obj=0.661, cls=0.140


Starting epoch 45 / 60
Learning Rate for this epoch: 1e-05
Epoch [45/60], Iter [50/157], Loss: total=1.855, reg=0.164, containing_obj=
0.571, no_obj=0.636, cls=0.144
Epoch [45/60], Iter [100/157], Loss: total=1.862, reg=0.166, containing_obj
=0.573, no_obj=0.640, cls=0.141
Epoch [45/60], Iter [150/157], Loss: total=1.899, reg=0.167, containing_obj
=0.595, no_obj=0.643, cls=0.145
---Evaluate model on test samples---
```

```
100%|██████████| 1255/1255 [00:53<00:00, 23.27it/s]
```

```
---class aeroplane ap 0.6423985603308207---
---class bicycle ap 0.5994162156318386---
---class bird ap 0.510893228544583---
---class boat ap 0.3547740564463797---
---class bottle ap 0.3662133504290982---
---class bus ap 0.5573304440323539---
---class car ap 0.6864546902074757---
---class cat ap 0.6935888604375255---
---class chair ap 0.3813829180557521---
---class cow ap 0.5771469556179596---
---class diningtable ap 0.30328886561979995---
---class dog ap 0.5886321666048049---
---class horse ap 0.6493583074908449---
---class motorbike ap 0.5389043242876863---
---class person ap 0.5505798661495862---
---class pottedplant ap 0.2040611578395672---
---class sheep ap 0.36273830124636575---
---class sofa ap 0.45887889422148365---
---class train ap 0.7446696497785146---
---class tvmonitor ap 0.6616024111588747---
---map 0.5216156612065657---
44 [0.6423985603308207, 0.5994162156318386, 0.510893228544583, 0.3547740564
463797, 0.3662133504290982, 0.5573304440323539, 0.6864546902074757, 0.69358
88604375255, 0.3813829180557521, 0.5771469556179596, 0.30328886561979995,
0.5886321666048049, 0.6493583074908449, 0.5389043242876863, 0.5505798661495
862, 0.2040611578395672, 0.36273830124636575, 0.45887889422148365, 0.744669
6497785146, 0.6616024111588747]


Starting epoch 46 / 60
Learning Rate for this epoch: 1e-05
Epoch [46/60], Iter [50/157], Loss: total=1.882, reg=0.169, containing_obj=
0.576, no_obj=0.671, cls=0.128
Epoch [46/60], Iter [100/157], Loss: total=1.854, reg=0.165, containing_obj
=0.566, no_obj=0.655, cls=0.133
Epoch [46/60], Iter [150/157], Loss: total=1.877, reg=0.167, containing_obj
=0.579, no_obj=0.655, cls=0.135
Updating best val loss: 3.66392


Starting epoch 47 / 60
Learning Rate for this epoch: 1e-05
Epoch [47/60], Iter [50/157], Loss: total=1.869, reg=0.164, containing_obj=
0.593, no_obj=0.657, cls=0.129
Epoch [47/60], Iter [100/157], Loss: total=1.857, reg=0.164, containing_obj
=0.571, no_obj=0.656, cls=0.137
Epoch [47/60], Iter [150/157], Loss: total=1.913, reg=0.168, containing_obj
=0.603, no_obj=0.654, cls=0.144
Updating best val loss: 3.66285


Starting epoch 48 / 60
Learning Rate for this epoch: 1e-05
Epoch [48/60], Iter [50/157], Loss: total=2.014, reg=0.180, containing_obj=
0.643, no_obj=0.627, cls=0.159
Epoch [48/60], Iter [100/157], Loss: total=1.921, reg=0.170, containing_obj
=0.608, no_obj=0.639, cls=0.142
Epoch [48/60], Iter [150/157], Loss: total=1.885, reg=0.167, containing_obj
=0.589, no_obj=0.643, cls=0.139
Updating best val loss: 3.64994


Starting epoch 49 / 60
Learning Rate for this epoch: 1e-05
```

```
Epoch [49/60], Iter [50/157], Loss: total=1.882, reg=0.168, containing_obj=
0.579, no_obj=0.626, cls=0.151
Epoch [49/60], Iter [100/157], Loss: total=1.866, reg=0.165, containing_obj
=0.581, no_obj=0.633, cls=0.145
Epoch [49/60], Iter [150/157], Loss: total=1.889, reg=0.166, containing_obj
=0.592, no_obj=0.643, cls=0.143


Starting epoch 50 / 60
Learning Rate for this epoch: 1e-05
Epoch [50/60], Iter [50/157], Loss: total=1.874, reg=0.168, containing_obj=
0.561, no_obj=0.642, cls=0.153
Epoch [50/60], Iter [100/157], Loss: total=1.943, reg=0.175, containing_obj
=0.594, no_obj=0.648, cls=0.151
Epoch [50/60], Iter [150/157], Loss: total=1.900, reg=0.168, containing_obj
=0.588, no_obj=0.652, cls=0.145
---Evaluate model on test samples---
```

```
100%|██████████| 1255/1255 [00:53<00:00, 23.48it/s]
```

```
---class aeroplane ap 0.6384338474284622---
---class bicycle ap 0.6136327785694097---
---class bird ap 0.5259013196541708---
---class boat ap 0.3263655745083587---
---class bottle ap 0.3650259569367671---
---class bus ap 0.5602035724935875---
---class car ap 0.6868775945441385---
---class cat ap 0.707258812199595---
---class chair ap 0.392217815218111---
---class cow ap 0.5681528308418516---
---class diningtable ap 0.3228357094370166---
---class dog ap 0.6151056262669146---
---class horse ap 0.6326052045566848---
---class motorbike ap 0.5674421802046101---
---class person ap 0.5511173385203563---
---class pottedplant ap 0.20657225054030856---
---class sheep ap 0.3829913763845815---
---class sofa ap 0.46942110099394513---
---class train ap 0.732203683885809---
---class tvmonitor ap 0.661408288710597---
---map 0.5262886430947638---
49 [0.6384338474284622, 0.6136327785694097, 0.5259013196541708, 0.326365574
5083587, 0.3650259569367671, 0.5602035724935875, 0.6868775945441385, 0.7072
58812199595, 0.392217815218111, 0.5681528308418516, 0.3228357094370166, 0.6
151056262669146, 0.6326052045566848, 0.5674421802046101, 0.551117338520356
3, 0.20657225054030856, 0.3829913763845815, 0.46942110099394513, 0.73220368
3885809, 0.661408288710597]


Starting epoch 51 / 60
Learning Rate for this epoch: 1e-05
Epoch [51/60], Iter [50/157], Loss: total=1.807, reg=0.154, containing_obj=
0.575, no_obj=0.651, cls=0.137
Epoch [51/60], Iter [100/157], Loss: total=1.858, reg=0.162, containing_obj
=0.583, no_obj=0.651, cls=0.138
Epoch [51/60], Iter [150/157], Loss: total=1.865, reg=0.162, containing_obj
=0.585, no_obj=0.648, cls=0.145


Starting epoch 52 / 60
Learning Rate for this epoch: 1e-05
Epoch [52/60], Iter [50/157], Loss: total=1.925, reg=0.169, containing_obj=
0.608, no_obj=0.644, cls=0.149
Epoch [52/60], Iter [100/157], Loss: total=1.874, reg=0.165, containing_obj
=0.584, no_obj=0.654, cls=0.139
Epoch [52/60], Iter [150/157], Loss: total=1.848, reg=0.162, containing_obj
=0.574, no_obj=0.656, cls=0.136


Starting epoch 53 / 60
Learning Rate for this epoch: 1e-05
Epoch [53/60], Iter [50/157], Loss: total=1.865, reg=0.164, containing_obj=
0.582, no_obj=0.649, cls=0.137
Epoch [53/60], Iter [100/157], Loss: total=1.867, reg=0.163, containing_obj
=0.584, no_obj=0.650, cls=0.141
Epoch [53/60], Iter [150/157], Loss: total=1.874, reg=0.164, containing_obj
=0.587, no_obj=0.650, cls=0.144


Starting epoch 54 / 60
Learning Rate for this epoch: 1e-05
Epoch [54/60], Iter [50/157], Loss: total=1.818, reg=0.159, containing_obj=
0.555, no_obj=0.670, cls=0.135
Epoch [54/60], Iter [100/157], Loss: total=1.839, reg=0.159, containing_obj
```

```
=0.571, no_obj=0.662, cls=0.142
Epoch [54/60], Iter [150/157], Loss: total=1.861, reg=0.162, containing_obj
=0.587, no_obj=0.655, cls=0.137


Starting epoch 55 / 60
Learning Rate for this epoch: 1e-05
Epoch [55/60], Iter [50/157], Loss: total=1.798, reg=0.156, containing_obj=
0.551, no_obj=0.654, cls=0.140
Epoch [55/60], Iter [100/157], Loss: total=1.849, reg=0.161, containing_obj
=0.580, no_obj=0.647, cls=0.140
Epoch [55/60], Iter [150/157], Loss: total=1.854, reg=0.164, containing_obj
=0.574, no_obj=0.650, cls=0.135
---Evaluate model on test samples---
100%|██████████| 1255/1255 [00:54<00:00, 22.93it/s]
---class aeroplane ap 0.6335184556932523---
---class bicycle ap 0.5917583435812003---
---class bird ap 0.5234884068276374---
---class boat ap 0.3494766127912661---
---class bottle ap 0.35724440816858893---
---class bus ap 0.5255734594308085---
---class car ap 0.6811668767126993---
---class cat ap 0.7246887851801376---
---class chair ap 0.3931015015042486---
---class cow ap 0.5742979496715412---
---class diningtable ap 0.3277197587933954---
---class dog ap 0.5878976421260815---
---class horse ap 0.676276914334822---
---class motorbike ap 0.5410341101907261---
---class person ap 0.5537199911590934---
---class pottedplant ap 0.19078454041899562---
---class sheep ap 0.3585872832407281---
---class sofa ap 0.4392835229624705---
---class train ap 0.7161043768417188---
---class tvmonitor ap 0.6511845295439105---
---map 0.5198453734586661---
54 [0.6335184556932523, 0.5917583435812003, 0.5234884068276374, 0.349476612
7912661, 0.35724440816858893, 0.5255734594308085, 0.6811668767126993, 0.724
6887851801376, 0.3931015015042486, 0.5742979496715412, 0.3277197587933954,
0.5878976421260815, 0.676276914334822, 0.5410341101907261, 0.55371999115909
34, 0.19078454041899562, 0.3585872832407281, 0.4392835229624705, 0.71610437
68417188, 0.6511845295439105]


Starting epoch 56 / 60
Learning Rate for this epoch: 1e-05
Epoch [56/60], Iter [50/157], Loss: total=1.964, reg=0.172, containing_obj=
0.639, no_obj=0.640, cls=0.143
Epoch [56/60], Iter [100/157], Loss: total=1.860, reg=0.162, containing_obj
=0.592, no_obj=0.652, cls=0.132
Epoch [56/60], Iter [150/157], Loss: total=1.849, reg=0.161, containing_obj
=0.581, no_obj=0.656, cls=0.136


Starting epoch 57 / 60
Learning Rate for this epoch: 1e-05
Epoch [57/60], Iter [50/157], Loss: total=1.817, reg=0.160, containing_obj=
0.546, no_obj=0.646, cls=0.146
Epoch [57/60], Iter [100/157], Loss: total=1.895, reg=0.167, containing_obj
=0.581, no_obj=0.649, cls=0.155
```

View example predictions

In [ ]:
```
net.eval()

# select random image from val set
image_name = random.choice(val_dataset.fnames)
image = cv2.imread(os.path.join(file_root_val, image_name))
image = cv2.cvtColor(image, cv2.COLOR_BGR2RGB)

print('predicting...')
result = predict_image(net, image_name, root_img_directory=file_root_val)
for left_up, right_bottom, class_name, _, prob in result:
    color = COLORS[VOC_CLASSES.index(class_name)]
    cv2.rectangle(image, left_up, right_bottom, color, 2)
    label = class_name + str(round(prob, 2))
    text_size, baseline = cv2.getTextSize(label, cv2.FONT_HERSHEY_SIMPLEX, (
    p1 = (left_up[0], left_up[1] - text_size[1])
    cv2.rectangle(image, (p1[0] - 2 // 2, p1[1] - 2 - baseline), (p1[0] + te
                  color, -1)
    cv2.putText(image, label, (p1[0], p1[1] + baseline), cv2.FONT_HERSHEY_S

plt.figure(figsize = (15,15))
plt.imshow(image)
```

Kaggle submission (85%)

Predict Result

Predict the results based on testing set. Upload to Kaggle.

How to upload

1. Click the folder icon in the left hand side of Colab.
2. Right click "result.csv". Select "Download"
3. To kaggle. Click "Submit Predictions"
4. Upload the result.csv
5. System will automaticlaly calculate the accuracy of 50% dataset and publish this result to leaderboard.

預測 `test` 並將結果上傳至Kaggle。**連結**

執行完畢此區的程式碼後，會將 `test` 預測完的結果存下來。

上傳流程

1. 點選左側選單最下方的資料夾圖示
2. 右鍵「result.csv」
3. 點選「Download」
4. 至連結網頁點選「Submit Predictions」
5. 將剛剛下載的檔案上傳
6. 系統會計算並公布其中50%資料的正確率

In [ ]:
```
root_test = 'data/VOCdevkit_2007 2/VOC2007test/JPEGImages/'
file_test = 'data/voc2007test.txt'
```

By using the test_evaluate function, you will obtain predictions for each image.

In [ ]:
```
preds_submission = test_evaluate(net, test_dataset_file=file_test, img_root=
```

The write_csv function will use preds_submission to write into a CSV file called 'result.csv'.

In [ ]:
```
write_csv(preds_submission)
```

In [ ]:
```
from google.colab import drive
drive.mount('/content/drive')
```

Report (15%)

In your report, please include:

a. A brief discussion on your implementation.

b. Report the best train and validation accuracy in all of your experiments and discuss any strategies or tricks you've employed.

c. Report the results for extra credits and also provide a discussion, if any.

Extra Credit (15%)

• Pick a fun video like **this one**, run your detector on it (a subset of frames would be OK), and produce a video showing your results.

• Try to replace the provided pre-trained network with a different one and train with the YOLO loss on top to attempt to get better accuracy.

• Or any other methods that you try to improve the performance.

MIS 583 Assignment 5: YOLO Object Detection on PASCAL VOC

Before we start, please put your name and SID in following format:
: LASTNAME Firstname, ?00000000 // e.g.) 李晨愷 M114020035

Your Answer:
Hi I'm 游雅淇, B104020012.

Google Colab Setup

Next we need to run a few commands to set up our environment on Google Colab. If you are running this notebook on a local machine you can skip this section.

Run the following cell to mount your Google Drive. Follow the link, sign in to your Google account (the same account you used to store this notebook!) and copy the authorization code into the text box that appears below.

In [ ]:
```
from google.colab import drive
drive.mount('/content/drive', force_remount=True)
```

```
Mounted at /content/drive
```

How to Get Data

請先到共用雲端硬碟將檔案 `VOCdevkit_2007.zip` ，建立捷徑到自己的雲端硬碟中。

> 操作步驟
>
> 1. 點開雲端連結
> 2. 點選右上角「新增雲端硬碟捷徑」
> 3. 點選「我的雲端硬碟」
> 4. 點選「新增捷徑」

完成以上流程會在你的雲端硬碟中建立一個檔案的捷徑，接著我們在colab中取得權限即可使用。

Unzip Data

解壓縮 `VOCdevkit_2007.zip`

- `VOC2007` : 包含了train/val的所有圖片
- `VOC2007test` : 包含了test的所有圖片

其中 `train` 的圖片 3756 張， `val` 的圖片 1255 張， `test` 的圖片 4950 張。

注意: 若有另外設定存放在雲端硬碟中的路徑，請記得本處路徑也須做更動。

Notice: Please put "VOCdevkit_2007" folder under data folder.

In [ ]:
```
!unzip -qq ./drive/MyDrive/DeepLearning/A5/data/VOCdevkit_2007.zip
```

In [ ]:
```
%cd ./drive/MyDrive/DeepLearning/A5
```

```
/content/drive/MyDrive/DeepLearning/A5
```

Import package

In [ ]:
```
import os
import random

import cv2
import numpy as np

import csv

import torch
from torch.utils.data import DataLoader
from torchvision import models

from src.resnet_yolo import resnet50
from src.densenet_yolo import densenet121
from yolo_loss import YoloLoss
from src.dataset import VocDetectorDataset
from src.eval_voc import evaluate, test_evaluate
from src.predict import predict_image
from src.config import VOC_CLASSES, COLORS
from kaggle_submission import write_csv

import matplotlib.pyplot as plt
import collections

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

Initialization

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

In [ ]:
```
# YOLO network hyperparameters
B = 2  # number of bounding box predictions per cell
S = 14  # width/height of network output grid (larger than 7x7 from paper si
```

To implement Yolo we will rely on a pretrained classifier as the backbone for our detection network. PyTorch offers a variety of models which are pretrained on ImageNet in the `torchvision.models` package. In particular, we will use the ResNet50 architecture as a base for our detector. This is different from the base architecture in the Yolo paper and also results in a different output grid size (14x14 instead of 7x7).

Models are typically pretrained on ImageNet since the dataset is very large (> 1 million images) and widely used. The pretrained model provides a very useful weight initialization for our detector, so that the network is able to learn quickly and effectively.

Resnet50

In [ ]:
```
load_network_path = None #'checkpoints/best_detector.pth'
pretrained = True

# use to load a previously trained network
if load_network_path is not None:
    print('Loading saved network from {}'.format(load_network_path))
    net = resnet50().to(device)
    net.load_state_dict(torch.load(load_network_path))
else:
    print('Load pre-trained model')
    net = resnet50(pretrained=pretrained).to(device)
```

```
Load pre-trained model
/usr/local/lib/python3.10/dist-packages/torchvision/models/_utils.py:208: U
serWarning: The parameter 'pretrained' is deprecated since 0.13 and may be
removed in the future, please use 'weights' instead.
  warnings.warn(
/usr/local/lib/python3.10/dist-packages/torchvision/models/_utils.py:223: U
serWarning: Arguments other than a weight enum or `None` for 'weights' are
deprecated since 0.13 and may be removed in the future. The current behavio
r is equivalent to passing `weights=ResNet50_Weights.IMAGENET1K_V1`. You ca
n also use `weights=ResNet50_Weights.DEFAULT` to get the most up-to-date we
ights.
  warnings.warn(msg)
Downloading: "https://download.pytorch.org/models/resnet50-0676ba61.pth" to
/root/.cache/torch/hub/checkpoints/resnet50-0676ba61.pth
100%|██████████| 97.8M/97.8M [00:00<00:00, 165MB/s]
```

DenseNet121

In [ ]:
```
# Densenet121
load_network_path = None #'checkpoints/best_detector.pth'
pretrained = True

# use to load a previously trained network
if load_network_path is not None:
    print('Loading saved network from {}'.format(load_network_path))
    net = densenet121(S=S).to(device)
    net.load_state_dict(torch.load(load_network_path))
else:
    print('Load pre-trained model')
    net = densenet121(pretrained=pretrained, S=S).to(device)
```

In [ ]:
```
learning_rate = 0.001
num_epochs = 50
batch_size = 24

# Yolo loss component coefficients (as given in Yolo v1 paper)
lambda_coord = 5
lambda_noobj = 0.5
```

Reading Pascal Data

The train dataset loader also using a variety of data augmentation techniques including random shift, scaling, crop, and flips. Data augmentation is slightly more complicated for detection datasets since the bounding box annotations must be kept consistent throughout the transformations.

Since the output of the detector network we train is an SxSx(B*5+C), we use an encoder to convert the original bounding box coordinates into relative grid bounding box coordinates corresponding to the expected output. We also use a decoder which allows us to convert the opposite direction into image coordinate bounding boxes.

Notice: Please put "VOCdevkit_2007" folder under data folder.

In [ ]:
```
file_root_train = 'data/VOCdevkit_2007/VOC2007/JPEGImages/'
annotation_file_train = 'data/voc2007train.txt'

train_dataset = VocDetectorDataset(root_img_dir=file_root_train,dataset_file
train_loader = DataLoader(train_dataset,batch_size=batch_size,shuffle=True,n
print('Loaded %d train images' % len(train_dataset))
```

```
Initializing dataset
Loaded 3756 train images
```

In [ ]:
```
file_root_val = 'data/VOCdevkit_2007/VOC2007/JPEGImages/'
annotation_file_val = 'data/voc2007val.txt'

val_dataset = VocDetectorDataset(root_img_dir=file_root_val,dataset_file=ann
val_loader = DataLoader(val_dataset,batch_size=batch_size,shuffle=False,num_
print('Loaded %d val images' % len(val_dataset))
```

```
Initializing dataset
Loaded 1255 val images
```

In [ ]:
```
data = train_dataset[0]
```

Set up training tools

In [ ]:
```
criterion = YoloLoss(S, B, lambda_coord, lambda_noobj)
optimizer = torch.optim.SGD(net.parameters(), lr=learning_rate, momentum=0.9
```

Train detector

In [ ]:
```
best_val_loss = np.inf
learning_rate = 1e-3
for epoch in range(num_epochs):
    net.train()

    # Update learning rate late in training
    if epoch == 30 or epoch == 40 or epoch == 50 :
        learning_rate /= 10.0

    for param_group in optimizer.param_groups:
        param_group['lr'] = learning_rate

    print('\n\nStarting epoch %d / %d' % (epoch + 1, num_epochs))
    print('Learning Rate for this epoch: {}'.format(learning_rate))

    total_loss = collections.defaultdict(int)

    for i, data in enumerate(train_loader):
        data = (item.to(device) for item in data)
```

```
        images, target_boxes, target_cls, has_object_map = data
        pred = net(images)
        loss_dict = criterion(pred, target_boxes, target_cls, has_object_map
        for key in loss_dict:
            total_loss[key] += loss_dict[key].item()

        optimizer.zero_grad()
        loss_dict['total_loss'].backward()
        optimizer.step()

        if (i+1) % 50 == 0:
            outstring = 'Epoch [%d/%d], Iter [%d/%d], Loss: ' % ((epoch+1, n
            outstring += ', '.join( "%s=%.3f" % (key[:-5], val / (i+1)) for
            print(outstring)

    # evaluate the network on the val data
    if (epoch + 1) % 5 == 0:
        val_aps = evaluate(net, val_dataset_file=annotation_file_val, img_ro
        print(epoch, val_aps)
    with torch.no_grad():
        val_loss = 0.0
        net.eval()
        for i, data in enumerate(val_loader):
            data = (item.to(device) for item in data)
            images, target_boxes, target_cls, has_object_map = data

            pred = net(images)
            loss_dict = criterion(pred, target_boxes, target_cls, has_object
            val_loss += loss_dict['total_loss'].item()
        val_loss /= len(val_loader)

    if best_val_loss > val_loss:
        best_val_loss = val_loss
        print('Updating best val loss: %.5f' % best_val_loss)
        torch.save(net.state_dict(),'checkpoints/best_detector.pth')

    if (epoch+1) in [5, 10, 20, 30, 40, 50, 60, 70, 80, 90, 100]:
        torch.save(net.state_dict(),'checkpoints/detector_epoch_%d.pth' % (e

    torch.save(net.state_dict(),'checkpoints/detector.pth')
```

```
Starting epoch 1 / 50
Learning Rate for this epoch: 0.001
Epoch [1/50], Iter [50/157], Loss: total=24.103, reg=0.758, containing_obj=
1.414, no_obj=25.251, cls=6.271
Epoch [1/50], Iter [100/157], Loss: total=16.279, reg=0.631, containing_obj
=1.722, no_obj=13.118, cls=4.844
Epoch [1/50], Iter [150/157], Loss: total=13.335, reg=0.576, containing_obj
=1.826, no_obj=8.995, cls=4.133
Updating best val loss: 7.04412


Starting epoch 2 / 50
Learning Rate for this epoch: 0.001
Epoch [2/50], Iter [50/157], Loss: total=6.573, reg=0.428, containing_obj=
1.887, no_obj=0.681, cls=2.204
Epoch [2/50], Iter [100/157], Loss: total=6.342, reg=0.420, containing_obj=
1.799, no_obj=0.725, cls=2.078
Epoch [2/50], Iter [150/157], Loss: total=6.245, reg=0.419, containing_obj=
1.751, no_obj=0.761, cls=2.016
Updating best val loss: 6.16320


Starting epoch 3 / 50
Learning Rate for this epoch: 0.001
Epoch [3/50], Iter [50/157], Loss: total=5.470, reg=0.389, containing_obj=
1.523, no_obj=0.811, cls=1.595
Epoch [3/50], Iter [100/157], Loss: total=5.573, reg=0.401, containing_obj=
1.570, no_obj=0.801, cls=1.599
Epoch [3/50], Iter [150/157], Loss: total=5.488, reg=0.396, containing_obj=
1.543, no_obj=0.808, cls=1.564
Updating best val loss: 5.70232


Starting epoch 4 / 50
Learning Rate for this epoch: 0.001
Epoch [4/50], Iter [50/157], Loss: total=5.366, reg=0.391, containing_obj=
1.528, no_obj=0.841, cls=1.465
Epoch [4/50], Iter [100/157], Loss: total=5.109, reg=0.378, containing_obj=
1.467, no_obj=0.811, cls=1.348
Epoch [4/50], Iter [150/157], Loss: total=5.031, reg=0.373, containing_obj=
1.454, no_obj=0.795, cls=1.315
Updating best val loss: 5.27348


Starting epoch 5 / 50
Learning Rate for this epoch: 0.001
Epoch [5/50], Iter [50/157], Loss: total=4.898, reg=0.375, containing_obj=
1.408, no_obj=0.790, cls=1.219
Epoch [5/50], Iter [100/157], Loss: total=4.730, reg=0.361, containing_obj=
1.393, no_obj=0.767, cls=1.148
Epoch [5/50], Iter [150/157], Loss: total=4.690, reg=0.361, containing_obj=
1.385, no_obj=0.764, cls=1.119
---Evaluate model on test samples---
```

```
100%|██████████| 1255/1255 [00:50<00:00, 24.82it/s]
```

```
---class aeroplane ap 0.23404448973733707---
---class bicycle ap 0.2674225959214281---
---class bird ap 0.3158070274510974---
---class boat ap 0.08444207196280162---
---class bottle ap 0.023050210899984915---
---class bus ap 0.07844155844155844---
---class car ap 0.33017389990423207---
---class cat ap 0.32077132121363017---
---class chair ap 0.15445498904886262---
---class cow ap 0.0608330288637194---
---class diningtable ap 0.031746031746031744---
---class dog ap 0.13054914156015418---
---class horse ap 0.31551656224285335---
---class motorbike ap 0.050271739130434784---
---class person ap 0.320592429697816---
---class pottedplant ap 0.0400545531069177---
---class sheep ap 0.12137958935623835---
---class sofa ap 0.041666666666666664---
---class train ap 0.30149593114709394---
---class tvmonitor ap 0.2850767792967337---
---map 0.17538953086977963---
4 [0.23404448973733707, 0.2674225959214281, 0.3158070274510974, 0.084442071
96280162, 0.023050210899984915, 0.07844155844155844, 0.33017389990423207,
0.32077132121363017, 0.15445498904886262, 0.0608330288637194, 0.03174603174
6031744, 0.13054914156015418, 0.31551656224285335, 0.050271739130434784, 0.
320592429697816, 0.0400545531069177, 0.12137958935623835, 0.041666666666666
664, 0.30149593114709394, 0.2850767792967337]
Updating best val loss: 5.04816


Starting epoch 6 / 50
Learning Rate for this epoch: 0.001
Epoch [6/50], Iter [50/157], Loss: total=4.375, reg=0.339, containing_obj=
1.308, no_obj=0.767, cls=0.988
Epoch [6/50], Iter [100/157], Loss: total=4.360, reg=0.337, containing_obj=
1.318, no_obj=0.774, cls=0.971
Epoch [6/50], Iter [150/157], Loss: total=4.365, reg=0.342, containing_obj=
1.314, no_obj=0.775, cls=0.955
Updating best val loss: 4.79899


Starting epoch 7 / 50
Learning Rate for this epoch: 0.001
Epoch [7/50], Iter [50/157], Loss: total=4.086, reg=0.324, containing_obj=
1.239, no_obj=0.784, cls=0.837
Epoch [7/50], Iter [100/157], Loss: total=4.106, reg=0.326, containing_obj=
1.243, no_obj=0.774, cls=0.845
Epoch [7/50], Iter [150/157], Loss: total=4.115, reg=0.329, containing_obj=
1.248, no_obj=0.765, cls=0.839
Updating best val loss: 4.72316


Starting epoch 8 / 50
Learning Rate for this epoch: 0.001
Epoch [8/50], Iter [50/157], Loss: total=4.122, reg=0.329, containing_obj=
1.268, no_obj=0.786, cls=0.814
Epoch [8/50], Iter [100/157], Loss: total=3.999, reg=0.320, containing_obj=
1.224, no_obj=0.779, cls=0.786
Epoch [8/50], Iter [150/157], Loss: total=3.998, reg=0.322, containing_obj=
1.218, no_obj=0.777, cls=0.783
Updating best val loss: 4.62204


Starting epoch 9 / 50
```

```
Learning Rate for this epoch: 0.001
Epoch [9/50], Iter [50/157], Loss: total=3.807, reg=0.313, containing_obj=
1.173, no_obj=0.787, cls=0.673
Epoch [9/50], Iter [100/157], Loss: total=3.846, reg=0.317, containing_obj=
1.185, no_obj=0.791, cls=0.680
Epoch [9/50], Iter [150/157], Loss: total=3.824, reg=0.313, containing_obj=
1.181, no_obj=0.785, cls=0.683
Updating best val loss: 4.43981


Starting epoch 10 / 50
Learning Rate for this epoch: 0.001
Epoch [10/50], Iter [50/157], Loss: total=3.753, reg=0.314, containing_obj=
1.177, no_obj=0.765, cls=0.625
Epoch [10/50], Iter [100/157], Loss: total=3.686, reg=0.305, containing_obj
=1.146, no_obj=0.777, cls=0.627
Epoch [10/50], Iter [150/157], Loss: total=3.695, reg=0.305, containing_obj
=1.154, no_obj=0.772, cls=0.628
---Evaluate model on test samples---
```

```
100%|██████████| 1255/1255 [00:44<00:00, 28.33it/s]
```

```
---class aeroplane ap 0.41599905429399675---
---class bicycle ap 0.3650426306577636---
---class bird ap 0.43968664510398064---
---class boat ap 0.23028502981241755---
---class bottle ap 0.07277936130933406---
---class bus ap 0.36589732992018886---
---class car ap 0.5150837674287456---
---class cat ap 0.6245462882596137---
---class chair ap 0.20866950602831133---
---class cow ap 0.20186145807263822---
---class diningtable ap 0.17838287047196416---
---class dog ap 0.44923107279254954---
---class horse ap 0.46145730771109944---
---class motorbike ap 0.3802583905415713---
---class person ap 0.43237009481686006---
---class pottedplant ap 0.120868787584249---
---class sheep ap 0.18854576502493667---
---class sofa ap 0.31401575662560055---
---class train ap 0.462676251948982---
---class tvmonitor ap 0.4626558406795302---
---map 0.34451566045421667---
9 [0.41599905429399675, 0.3650426306577636, 0.43968664510398064, 0.23028502
981241755, 0.07277936130933406, 0.36589732992018886, 0.5150837674287456, 0.
6245462882596137, 0.20866950602831133, 0.20186145807263822, 0.1783828704719
6416, 0.44923107279254954, 0.46145730771109944, 0.3802583905415713, 0.43237
009481686006, 0.120868787584249, 0.18854576502493667, 0.31401575662560055,
0.462676251948982, 0.4626558406795302]
Updating best val loss: 4.43909


Starting epoch 11 / 50
Learning Rate for this epoch: 0.001
Epoch [11/50], Iter [50/157], Loss: total=3.413, reg=0.281, containing_obj=
1.069, no_obj=0.735, cls=0.571
Epoch [11/50], Iter [100/157], Loss: total=3.528, reg=0.292, containing_obj
=1.102, no_obj=0.765, cls=0.582
Epoch [11/50], Iter [150/157], Loss: total=3.560, reg=0.295, containing_obj
=1.111, no_obj=0.771, cls=0.589
Updating best val loss: 4.36562


Starting epoch 12 / 50
Learning Rate for this epoch: 0.001
Epoch [12/50], Iter [50/157], Loss: total=3.348, reg=0.281, containing_obj=
1.048, no_obj=0.776, cls=0.506
Epoch [12/50], Iter [100/157], Loss: total=3.423, reg=0.286, containing_obj
=1.071, no_obj=0.784, cls=0.531
Epoch [12/50], Iter [150/157], Loss: total=3.432, reg=0.286, containing_obj
=1.073, no_obj=0.780, cls=0.538
Updating best val loss: 4.25210


Starting epoch 13 / 50
Learning Rate for this epoch: 0.001
Epoch [13/50], Iter [50/157], Loss: total=3.374, reg=0.280, containing_obj=
1.089, no_obj=0.724, cls=0.522
Epoch [13/50], Iter [100/157], Loss: total=3.324, reg=0.279, containing_obj
=1.061, no_obj=0.751, cls=0.494
Epoch [13/50], Iter [150/157], Loss: total=3.302, reg=0.277, containing_obj
=1.049, no_obj=0.763, cls=0.489
Updating best val loss: 4.13852


Starting epoch 14 / 50
```

```
Learning Rate for this epoch: 0.001
Epoch [14/50], Iter [50/157], Loss: total=3.192, reg=0.269, containing_obj=
0.996, no_obj=0.765, cls=0.468
Epoch [14/50], Iter [100/157], Loss: total=3.203, reg=0.270, containing_obj
=1.009, no_obj=0.767, cls=0.459
Epoch [14/50], Iter [150/157], Loss: total=3.226, reg=0.272, containing_obj
=1.019, no_obj=0.777, cls=0.461


Starting epoch 15 / 50
Learning Rate for this epoch: 0.001
Epoch [15/50], Iter [50/157], Loss: total=3.094, reg=0.261, containing_obj=
0.988, no_obj=0.757, cls=0.423
Epoch [15/50], Iter [100/157], Loss: total=3.173, reg=0.268, containing_obj
=1.008, no_obj=0.762, cls=0.445
Epoch [15/50], Iter [150/157], Loss: total=3.178, reg=0.270, containing_obj
=1.008, no_obj=0.769, cls=0.437
---Evaluate model on test samples---
```

```
100%|██████████| 1255/1255 [00:43<00:00, 28.92it/s]
```

```
---class aeroplane ap 0.3344076871015168---
---class bicycle ap 0.5425683158298162---
---class bird ap 0.5413258457213874---
---class boat ap 0.1748446140741921---
---class bottle ap 0.08956548758759073---
---class bus ap 0.44675533773595416---
---class car ap 0.5692992441660448---
---class cat ap 0.6372950048910108---
---class chair ap 0.21840869015689787---
---class cow ap 0.43179382275070716---
---class diningtable ap 0.2503938327467739---
---class dog ap 0.49069522054881254---
---class horse ap 0.5463111492013286---
---class motorbike ap 0.4347249289374292---
---class person ap 0.46142858756641214---
---class pottedplant ap 0.11552953546726205---
---class sheep ap 0.266905456607383---
---class sofa ap 0.3241537033666099---
---class train ap 0.5533180439960101---
---class tvmonitor ap 0.48334994517193763---
---map 0.39565372268125387---
14 [0.3344076871015168, 0.5425683158298162, 0.5413258457213874, 0.174844614
0741921, 0.08956548758759073, 0.44675533773595416, 0.5692992441660448, 0.63
72950048910108, 0.21840869015689787, 0.43179382275070716, 0.250393832746773
9, 0.49069522054881254, 0.5463111492013286, 0.4347249289374292, 0.461428587
56641214, 0.11552953546726205, 0.266905456607383, 0.3241537033666099, 0.553
3180439960101, 0.48334994517193763]


Starting epoch 16 / 50
Learning Rate for this epoch: 0.001
Epoch [16/50], Iter [50/157], Loss: total=3.134, reg=0.263, containing_obj=
1.001, no_obj=0.761, cls=0.437
Epoch [16/50], Iter [100/157], Loss: total=3.063, reg=0.260, containing_obj
=0.961, no_obj=0.769, cls=0.417
Epoch [16/50], Iter [150/157], Loss: total=3.090, reg=0.263, containing_obj
=0.973, no_obj=0.768, cls=0.416
Updating best val loss: 4.04159


Starting epoch 17 / 50
Learning Rate for this epoch: 0.001
Epoch [17/50], Iter [50/157], Loss: total=2.996, reg=0.252, containing_obj=
0.954, no_obj=0.767, cls=0.396
Epoch [17/50], Iter [100/157], Loss: total=2.925, reg=0.250, containing_obj
=0.917, no_obj=0.777, cls=0.372
Epoch [17/50], Iter [150/157], Loss: total=2.967, reg=0.252, containing_obj
=0.941, no_obj=0.769, cls=0.383


Starting epoch 18 / 50
Learning Rate for this epoch: 0.001
Epoch [18/50], Iter [50/157], Loss: total=2.873, reg=0.244, containing_obj=
0.926, no_obj=0.763, cls=0.348
Epoch [18/50], Iter [100/157], Loss: total=2.915, reg=0.248, containing_obj
=0.919, no_obj=0.768, cls=0.374
Epoch [18/50], Iter [150/157], Loss: total=2.909, reg=0.247, containing_obj
=0.919, no_obj=0.762, cls=0.375
Updating best val loss: 3.99177


Starting epoch 19 / 50
Learning Rate for this epoch: 0.001
Epoch [19/50], Iter [50/157], Loss: total=2.853, reg=0.244, containing_obj=
```

```
0.905, no_obj=0.750, cls=0.353
Epoch [19/50], Iter [100/157], Loss: total=2.879, reg=0.245, containing_obj
=0.918, no_obj=0.753, cls=0.362
Epoch [19/50], Iter [150/157], Loss: total=2.842, reg=0.241, containing_obj
=0.900, no_obj=0.755, cls=0.361


Starting epoch 20 / 50
Learning Rate for this epoch: 0.001
Epoch [20/50], Iter [50/157], Loss: total=2.865, reg=0.244, containing_obj=
0.890, no_obj=0.811, cls=0.350
Epoch [20/50], Iter [100/157], Loss: total=2.816, reg=0.241, containing_obj
=0.876, no_obj=0.779, cls=0.345
Epoch [20/50], Iter [150/157], Loss: total=2.803, reg=0.240, containing_obj
=0.873, no_obj=0.774, cls=0.342
---Evaluate model on test samples---
100%|██████████| 1255/1255 [00:44<00:00, 28.36it/s]
```

```
---class aeroplane ap 0.5178394111955906---
---class bicycle ap 0.5920482660698859---
---class bird ap 0.5201035804962366---
---class boat ap 0.23536275470413345---
---class bottle ap 0.20375844874492555---
---class bus ap 0.5361999417416985---
---class car ap 0.567787259824586---
---class cat ap 0.6341958822919206---
---class chair ap 0.29711824675561227---
---class cow ap 0.35192256418430234---
---class diningtable ap 0.30101536335799317---
---class dog ap 0.5490534122664519---
---class horse ap 0.5727695541299401---
---class motorbike ap 0.5154927620846118---
---class person ap 0.4224381653945489---
---class pottedplant ap 0.1393448673075399---
---class sheep ap 0.19784476321785938---
---class sofa ap 0.39209188130178696---
---class train ap 0.5548672829813818---
---class tvmonitor ap 0.5139563804401719---
---map 0.4307605394245589---
19 [0.5178394111955906, 0.5920482660698859, 0.5201035804962366, 0.235362754
70413345, 0.20375844874492555, 0.5361999417416985, 0.567787259824586, 0.634
1958822919206, 0.29711824675561227, 0.35192256418430234, 0.3010153633579931
7, 0.5490534122664519, 0.5727695541299401, 0.5154927620846118, 0.4224381653
945489, 0.1393448673075399, 0.19784476321785938, 0.39209188130178696, 0.554
8672829813818, 0.5139563804401719]


Starting epoch 21 / 50
Learning Rate for this epoch: 0.001
Epoch [21/50], Iter [50/157], Loss: total=2.745, reg=0.241, containing_obj=
0.854, no_obj=0.741, cls=0.315
Epoch [21/50], Iter [100/157], Loss: total=2.716, reg=0.236, containing_obj
=0.854, no_obj=0.744, cls=0.313
Epoch [21/50], Iter [150/157], Loss: total=2.735, reg=0.236, containing_obj
=0.852, no_obj=0.754, cls=0.328
Updating best val loss: 3.93851


Starting epoch 22 / 50
Learning Rate for this epoch: 0.001
Epoch [22/50], Iter [50/157], Loss: total=2.718, reg=0.239, containing_obj=
0.835, no_obj=0.740, cls=0.321
Epoch [22/50], Iter [100/157], Loss: total=2.736, reg=0.238, containing_obj
=0.852, no_obj=0.754, cls=0.317
Epoch [22/50], Iter [150/157], Loss: total=2.708, reg=0.234, containing_obj
=0.846, no_obj=0.750, cls=0.318
Updating best val loss: 3.90568


Starting epoch 23 / 50
Learning Rate for this epoch: 0.001
Epoch [23/50], Iter [50/157], Loss: total=2.492, reg=0.216, containing_obj=
0.782, no_obj=0.726, cls=0.266
Epoch [23/50], Iter [100/157], Loss: total=2.551, reg=0.218, containing_obj
=0.809, no_obj=0.733, cls=0.284
Epoch [23/50], Iter [150/157], Loss: total=2.609, reg=0.223, containing_obj
=0.825, no_obj=0.738, cls=0.300


Starting epoch 24 / 50
Learning Rate for this epoch: 0.001
Epoch [24/50], Iter [50/157], Loss: total=2.617, reg=0.221, containing_obj=
```

```
0.829, no_obj=0.765, cls=0.298
Epoch [24/50], Iter [100/157], Loss: total=2.630, reg=0.227, containing_obj
=0.822, no_obj=0.754, cls=0.293
Epoch [24/50], Iter [150/157], Loss: total=2.594, reg=0.225, containing_obj
=0.813, no_obj=0.741, cls=0.283


Starting epoch 25 / 50
Learning Rate for this epoch: 0.001
Epoch [25/50], Iter [50/157], Loss: total=2.545, reg=0.223, containing_obj=
0.790, no_obj=0.726, cls=0.278
Epoch [25/50], Iter [100/157], Loss: total=2.493, reg=0.216, containing_obj
=0.776, no_obj=0.732, cls=0.271
Epoch [25/50], Iter [150/157], Loss: total=2.505, reg=0.218, containing_obj
=0.780, no_obj=0.725, cls=0.275
---Evaluate model on test samples---
100%|██████████| 1255/1255 [00:44<00:00, 28.06it/s]
```

```
---class aeroplane ap 0.6618834802952756---
---class bicycle ap 0.5474977979359678---
---class bird ap 0.48342630900207656---
---class boat ap 0.35753590984435524---
---class bottle ap 0.2006726119240646---
---class bus ap 0.43443409407850947---
---class car ap 0.6481704403259481---
---class cat ap 0.6656904993642817---
---class chair ap 0.30660411942819715---
---class cow ap 0.4623193901826518---
---class diningtable ap 0.27730169709091534---
---class dog ap 0.6065831582261425---
---class horse ap 0.5826933866134423---
---class motorbike ap 0.6082032515809171---
---class person ap 0.47263671045605116---
---class pottedplant ap 0.19817411343375152---
---class sheep ap 0.35481881569183216---
---class sofa ap 0.4716556041879993---
---class train ap 0.6732552865099146---
---class tvmonitor ap 0.513116122402945---
---map 0.4763336399287619---
24 [0.6618834802952756, 0.5474977979359678, 0.48342630900207656, 0.35753590
984435524, 0.2006726119240646, 0.43443409407850947, 0.6481704403259481, 0.6
656904993642817, 0.30660411942819715, 0.4623193901826518, 0.277301697090915
34, 0.6065831582261425, 0.5826933866134423, 0.6082032515809171, 0.472636710
45605116, 0.19817411343375152, 0.35481881569183216, 0.4716556041879993, 0.6
732552865099146, 0.513116122402945]


Starting epoch 26 / 50
Learning Rate for this epoch: 0.001
Epoch [26/50], Iter [50/157], Loss: total=2.432, reg=0.212, containing_obj=
0.746, no_obj=0.746, cls=0.253
Epoch [26/50], Iter [100/157], Loss: total=2.478, reg=0.214, containing_obj
=0.772, no_obj=0.736, cls=0.268
Epoch [26/50], Iter [150/157], Loss: total=2.520, reg=0.218, containing_obj
=0.785, no_obj=0.736, cls=0.277


Starting epoch 27 / 50
Learning Rate for this epoch: 0.001
Epoch [27/50], Iter [50/157], Loss: total=2.478, reg=0.212, containing_obj=
0.795, no_obj=0.728, cls=0.257
Epoch [27/50], Iter [100/157], Loss: total=2.441, reg=0.208, containing_obj
=0.778, no_obj=0.724, cls=0.262
Epoch [27/50], Iter [150/157], Loss: total=2.451, reg=0.211, containing_obj
=0.779, no_obj=0.722, cls=0.257


Starting epoch 28 / 50
Learning Rate for this epoch: 0.001
Epoch [28/50], Iter [50/157], Loss: total=2.378, reg=0.206, containing_obj=
0.758, no_obj=0.693, cls=0.243
Epoch [28/50], Iter [100/157], Loss: total=2.369, reg=0.205, containing_obj
=0.759, no_obj=0.704, cls=0.236
Epoch [28/50], Iter [150/157], Loss: total=2.366, reg=0.203, containing_obj
=0.753, no_obj=0.707, cls=0.246


Starting epoch 29 / 50
Learning Rate for this epoch: 0.001
Epoch [29/50], Iter [50/157], Loss: total=2.350, reg=0.201, containing_obj=
0.726, no_obj=0.720, cls=0.259
Epoch [29/50], Iter [100/157], Loss: total=2.323, reg=0.201, containing_obj
```

```
=0.723, no_obj=0.709, cls=0.241
Epoch [29/50], Iter [150/157], Loss: total=2.359, reg=0.205, containing_obj
=0.737, no_obj=0.715, cls=0.240


Starting epoch 30 / 50
Learning Rate for this epoch: 0.001
Epoch [30/50], Iter [50/157], Loss: total=2.300, reg=0.193, containing_obj=
0.726, no_obj=0.740, cls=0.237
Epoch [30/50], Iter [100/157], Loss: total=2.347, reg=0.201, containing_obj
=0.738, no_obj=0.731, cls=0.238
Epoch [30/50], Iter [150/157], Loss: total=2.352, reg=0.201, containing_obj
=0.739, no_obj=0.740, cls=0.236
---Evaluate model on test samples---
```

```
100%|██████████| 1255/1255 [00:44<00:00, 28.15it/s]
```

```
---class aeroplane ap 0.5690340210774798---
---class bicycle ap 0.5190527848003978---
---class bird ap 0.5672158450048697---
---class boat ap 0.3222040025624234---
---class bottle ap 0.24144738633958504---
---class bus ap 0.5241597408320332---
---class car ap 0.6976395280141234---
---class cat ap 0.7012469064932182---
---class chair ap 0.32479549979349936---
---class cow ap 0.44406714178766327---
---class diningtable ap 0.2797851151439072---
---class dog ap 0.6677157348219405---
---class horse ap 0.6273468301531409---
---class motorbike ap 0.44962175727908---
---class person ap 0.4533645350526373---
---class pottedplant ap 0.15738595286061463---
---class sheep ap 0.281355942423871---
---class sofa ap 0.357736699166503---
---class train ap 0.6703515581943431---
---class tvmonitor ap 0.5684723384404881---
---map 0.47119996601209085---
29 [0.5690340210774798, 0.5190527848003978, 0.5672158450048697, 0.322204002
5624234, 0.24144738633958504, 0.5241597408320332, 0.6976395280141234, 0.701
2469064932182, 0.32479549979349936, 0.44406714178766327, 0.279785115143907
2, 0.6677157348219405, 0.6273468301531409, 0.44962175727908, 0.453364535052
6373, 0.15738595286061463, 0.281355942423871, 0.357736699166503, 0.67035155
81943431, 0.5684723384404881]
Updating best val loss: 3.89517


Starting epoch 31 / 50
Learning Rate for this epoch: 0.0001
Epoch [31/50], Iter [50/157], Loss: total=2.259, reg=0.197, containing_obj=
0.719, no_obj=0.680, cls=0.213
Epoch [31/50], Iter [100/157], Loss: total=2.231, reg=0.193, containing_obj
=0.719, no_obj=0.683, cls=0.205
Epoch [31/50], Iter [150/157], Loss: total=2.206, reg=0.190, containing_obj
=0.704, no_obj=0.685, cls=0.208
Updating best val loss: 3.78885


Starting epoch 32 / 50
Learning Rate for this epoch: 0.0001
Epoch [32/50], Iter [50/157], Loss: total=2.152, reg=0.183, containing_obj=
0.694, no_obj=0.675, cls=0.204
Epoch [32/50], Iter [100/157], Loss: total=2.099, reg=0.180, containing_obj
=0.671, no_obj=0.676, cls=0.191
Epoch [32/50], Iter [150/157], Loss: total=2.115, reg=0.183, containing_obj
=0.676, no_obj=0.669, cls=0.192


Starting epoch 33 / 50
Learning Rate for this epoch: 0.0001
Epoch [33/50], Iter [50/157], Loss: total=2.168, reg=0.189, containing_obj=
0.690, no_obj=0.688, cls=0.187
Epoch [33/50], Iter [100/157], Loss: total=2.140, reg=0.186, containing_obj
=0.677, no_obj=0.687, cls=0.190
Epoch [33/50], Iter [150/157], Loss: total=2.097, reg=0.182, containing_obj
=0.665, no_obj=0.680, cls=0.182


Starting epoch 34 / 50
Learning Rate for this epoch: 0.0001
Epoch [34/50], Iter [50/157], Loss: total=2.017, reg=0.171, containing_obj=
```

```
0.644, no_obj=0.681, cls=0.179
Epoch [34/50], Iter [100/157], Loss: total=2.064, reg=0.175, containing_obj
=0.665, no_obj=0.668, cls=0.190
Epoch [34/50], Iter [150/157], Loss: total=2.081, reg=0.178, containing_obj
=0.666, no_obj=0.673, cls=0.188


Starting epoch 35 / 50
Learning Rate for this epoch: 0.0001
Epoch [35/50], Iter [50/157], Loss: total=2.058, reg=0.176, containing_obj=
0.665, no_obj=0.683, cls=0.170
Epoch [35/50], Iter [100/157], Loss: total=2.044, reg=0.176, containing_obj
=0.645, no_obj=0.685, cls=0.175
Epoch [35/50], Iter [150/157], Loss: total=2.018, reg=0.173, containing_obj
=0.640, no_obj=0.680, cls=0.174
---Evaluate model on test samples---
100%|██████████| 1255/1255 [00:44<00:00, 28.39it/s]
```

```
---class aeroplane ap 0.5401731935960894---
---class bicycle ap 0.5954041400939147---
---class bird ap 0.5333325485578122---
---class boat ap 0.3937482947649102---
---class bottle ap 0.23302052145410085---
---class bus ap 0.5003197129834902---
---class car ap 0.706759231526782---
---class cat ap 0.7521723091486834---
---class chair ap 0.3467813809019478---
---class cow ap 0.4601272257689342---
---class diningtable ap 0.33357351593055967---
---class dog ap 0.6557478081154963---
---class horse ap 0.6800002750675859---
---class motorbike ap 0.5222948102817202---
---class person ap 0.4991196878808704---
---class pottedplant ap 0.19954851133647403---
---class sheep ap 0.3334870259996898---
---class sofa ap 0.44599772703153207---
---class train ap 0.781119370750939---
---class tvmonitor ap 0.5851359284523406---
---map 0.5048931609821936---
34 [0.5401731935960894, 0.5954041400939147, 0.5333325485578122, 0.393748294
7649102, 0.23302052145410085, 0.5003197129834902, 0.706759231526782, 0.7521
723091486834, 0.3467813809019478, 0.4601272257689342, 0.33357351593055967,
0.6557478081154963, 0.6800002750675859, 0.5222948102817202, 0.4991196878808
704, 0.19954851133647403, 0.3334870259996898, 0.44599772703153207, 0.781119
370750939, 0.5851359284523406]
Updating best val loss: 3.76522


Starting epoch 36 / 50
Learning Rate for this epoch: 0.0001
Epoch [36/50], Iter [50/157], Loss: total=1.976, reg=0.167, containing_obj=
0.639, no_obj=0.661, cls=0.172
Epoch [36/50], Iter [100/157], Loss: total=1.995, reg=0.171, containing_obj
=0.644, no_obj=0.651, cls=0.172
Epoch [36/50], Iter [150/157], Loss: total=2.022, reg=0.173, containing_obj
=0.658, no_obj=0.645, cls=0.175


Starting epoch 37 / 50
Learning Rate for this epoch: 0.0001
Epoch [37/50], Iter [50/157], Loss: total=2.033, reg=0.176, containing_obj=
0.646, no_obj=0.693, cls=0.160
Epoch [37/50], Iter [100/157], Loss: total=2.028, reg=0.179, containing_obj
=0.630, no_obj=0.668, cls=0.168
Epoch [37/50], Iter [150/157], Loss: total=2.018, reg=0.177, containing_obj
=0.629, no_obj=0.666, cls=0.174


Starting epoch 38 / 50
Learning Rate for this epoch: 0.0001
Epoch [38/50], Iter [50/157], Loss: total=2.041, reg=0.178, containing_obj=
0.644, no_obj=0.675, cls=0.171
Epoch [38/50], Iter [100/157], Loss: total=2.040, reg=0.179, containing_obj
=0.630, no_obj=0.672, cls=0.178
Epoch [38/50], Iter [150/157], Loss: total=2.006, reg=0.176, containing_obj
=0.618, no_obj=0.677, cls=0.168


Starting epoch 39 / 50
Learning Rate for this epoch: 0.0001
Epoch [39/50], Iter [50/157], Loss: total=2.013, reg=0.172, containing_obj=
0.659, no_obj=0.660, cls=0.164
```

```
Epoch [39/50], Iter [100/157], Loss: total=2.026, reg=0.173, containing_obj
=0.648, no_obj=0.677, cls=0.173
Epoch [39/50], Iter [150/157], Loss: total=1.993, reg=0.170, containing_obj
=0.635, no_obj=0.661, cls=0.177


Starting epoch 40 / 50
Learning Rate for this epoch: 0.0001
Epoch [40/50], Iter [50/157], Loss: total=1.936, reg=0.162, containing_obj=
0.629, no_obj=0.647, cls=0.173
Epoch [40/50], Iter [100/157], Loss: total=1.924, reg=0.164, containing_obj
=0.619, no_obj=0.653, cls=0.160
Epoch [40/50], Iter [150/157], Loss: total=1.955, reg=0.167, containing_obj
=0.624, no_obj=0.660, cls=0.165
---Evaluate model on test samples---
```

```
100%|██████████| 1255/1255 [00:46<00:00, 27.16it/s]
```

```
---class aeroplane ap 0.6445665071585093---
---class bicycle ap 0.5807727318309939---
---class bird ap 0.5294340875989305---
---class boat ap 0.38203859577435095---
---class bottle ap 0.20810453609862578---
---class bus ap 0.4679455698474837---
---class car ap 0.6921615741842123---
---class cat ap 0.806354583132037---
---class chair ap 0.37258656515824684---
---class cow ap 0.47365256927985827---
---class diningtable ap 0.3122347649267324---
---class dog ap 0.6736621107023588---
---class horse ap 0.6730531670241466---
---class motorbike ap 0.4971324649895331---
---class person ap 0.508266678823903---
---class pottedplant ap 0.21517028871543967---
---class sheep ap 0.37828634298869174---
---class sofa ap 0.46522404849192556---
---class train ap 0.7540345444515948---
---class tvmonitor ap 0.5843622117553461---
---map 0.5109521971466461---
39 [0.6445665071585093, 0.5807727318309939, 0.5294340875989305, 0.382038595
77435095, 0.20810453609862578, 0.4679455698474837, 0.6921615741842123, 0.80
6354583132037, 0.37258656515824684, 0.47365256927985827, 0.312234764926732
4, 0.6736621107023588, 0.6730531670241466, 0.4971324649895331, 0.5082666788
23903, 0.21517028871543967, 0.37828634298869174, 0.46522404849192556, 0.754
0345444515948, 0.5843622117553461]


Starting epoch 41 / 50
Learning Rate for this epoch: 1e-05
Epoch [41/50], Iter [50/157], Loss: total=1.989, reg=0.176, containing_obj=
0.613, no_obj=0.671, cls=0.163
Epoch [41/50], Iter [100/157], Loss: total=2.002, reg=0.176, containing_obj
=0.625, no_obj=0.667, cls=0.165
Epoch [41/50], Iter [150/157], Loss: total=1.993, reg=0.174, containing_obj
=0.624, no_obj=0.668, cls=0.163


Starting epoch 42 / 50
Learning Rate for this epoch: 1e-05
Epoch [42/50], Iter [50/157], Loss: total=1.926, reg=0.167, containing_obj=
0.594, no_obj=0.658, cls=0.168
Epoch [42/50], Iter [100/157], Loss: total=1.944, reg=0.169, containing_obj
=0.598, no_obj=0.654, cls=0.176
Epoch [42/50], Iter [150/157], Loss: total=1.977, reg=0.173, containing_obj
=0.613, no_obj=0.651, cls=0.172
Updating best val loss: 3.76134


Starting epoch 43 / 50
Learning Rate for this epoch: 1e-05
Epoch [43/50], Iter [50/157], Loss: total=1.978, reg=0.174, containing_obj=
0.610, no_obj=0.668, cls=0.165
Epoch [43/50], Iter [100/157], Loss: total=1.967, reg=0.173, containing_obj
=0.606, no_obj=0.666, cls=0.163
Epoch [43/50], Iter [150/157], Loss: total=1.987, reg=0.173, containing_obj
=0.625, no_obj=0.662, cls=0.166
Updating best val loss: 3.75632


Starting epoch 44 / 50
Learning Rate for this epoch: 1e-05
Epoch [44/50], Iter [50/157], Loss: total=1.873, reg=0.162, containing_obj=
```

```
0.592, no_obj=0.648, cls=0.149
Epoch [44/50], Iter [100/157], Loss: total=1.942, reg=0.164, containing_obj
=0.620, no_obj=0.654, cls=0.174
Epoch [44/50], Iter [150/157], Loss: total=1.943, reg=0.164, containing_obj
=0.623, no_obj=0.660, cls=0.170


Starting epoch 45 / 50
Learning Rate for this epoch: 1e-05
Epoch [45/50], Iter [50/157], Loss: total=2.012, reg=0.173, containing_obj=
0.650, no_obj=0.650, cls=0.173
Epoch [45/50], Iter [100/157], Loss: total=1.955, reg=0.169, containing_obj
=0.617, no_obj=0.657, cls=0.163
Epoch [45/50], Iter [150/157], Loss: total=1.941, reg=0.168, containing_obj
=0.610, no_obj=0.651, cls=0.167
---Evaluate model on test samples---
```

```
100%|██████████| 1255/1255 [00:44<00:00, 28.47it/s]
```

```
---class aeroplane ap 0.6291865457980178---
---class bicycle ap 0.5615867307808892---
---class bird ap 0.5259251846047481---
---class boat ap 0.3669550472014277---
---class bottle ap 0.20827355047135687---
---class bus ap 0.5496316303030001---
---class car ap 0.6770119205188168---
---class cat ap 0.8125604630438231---
---class chair ap 0.35799751740477953---
---class cow ap 0.5125449697672544---
---class diningtable ap 0.3178491265637037---
---class dog ap 0.6692945783281833---
---class horse ap 0.6616610195272735---
---class motorbike ap 0.4932249646445406---
---class person ap 0.5086574985624499---
---class pottedplant ap 0.23186776304809406---
---class sheep ap 0.406924284256662---
---class sofa ap 0.45818712911435067---
---class train ap 0.7282925398935043---
---class tvmonitor ap 0.5879711866561131---
---map 0.5132801825244494---
44 [0.6291865457980178, 0.5615867307808892, 0.5259251846047481, 0.366955047
2014277, 0.20827355047135687, 0.5496316303030001, 0.6770119205188168, 0.812
5604630438231, 0.35799751740477953, 0.5125449697672544, 0.3178491265637037,
0.6692945783281833, 0.6616610195272735, 0.4932249646445406, 0.5086574985624
499, 0.23186776304809406, 0.406924284256662, 0.45818712911435067, 0.7282925
398935043, 0.5879711866561131]
Updating best val loss: 3.75198


Starting epoch 46 / 50
Learning Rate for this epoch: 1e-05
Epoch [46/50], Iter [50/157], Loss: total=1.925, reg=0.168, containing_obj=
0.594, no_obj=0.641, cls=0.173
Epoch [46/50], Iter [100/157], Loss: total=1.951, reg=0.168, containing_obj
=0.607, no_obj=0.649, cls=0.177
Epoch [46/50], Iter [150/157], Loss: total=1.919, reg=0.166, containing_obj
=0.599, no_obj=0.647, cls=0.168


Starting epoch 47 / 50
Learning Rate for this epoch: 1e-05
Epoch [47/50], Iter [50/157], Loss: total=1.997, reg=0.173, containing_obj=
0.636, no_obj=0.644, cls=0.174
Epoch [47/50], Iter [100/157], Loss: total=1.965, reg=0.171, containing_obj
=0.621, no_obj=0.649, cls=0.166
Epoch [47/50], Iter [150/157], Loss: total=1.952, reg=0.170, containing_obj
=0.611, no_obj=0.651, cls=0.166


Starting epoch 48 / 50
Learning Rate for this epoch: 1e-05
Epoch [48/50], Iter [50/157], Loss: total=1.885, reg=0.162, containing_obj=
0.583, no_obj=0.654, cls=0.164
Epoch [48/50], Iter [100/157], Loss: total=1.926, reg=0.165, containing_obj
=0.610, no_obj=0.653, cls=0.165
Epoch [48/50], Iter [150/157], Loss: total=1.922, reg=0.165, containing_obj
=0.608, no_obj=0.650, cls=0.162


Starting epoch 49 / 50
Learning Rate for this epoch: 1e-05
Epoch [49/50], Iter [50/157], Loss: total=2.017, reg=0.171, containing_obj=
0.645, no_obj=0.668, cls=0.181
```

```
Epoch [49/50], Iter [100/157], Loss: total=2.002, reg=0.171, containing_obj
=0.635, no_obj=0.666, cls=0.177
Epoch [49/50], Iter [150/157], Loss: total=1.963, reg=0.168, containing_obj
=0.621, no_obj=0.663, cls=0.169


Starting epoch 50 / 50
Learning Rate for this epoch: 1e-05
Epoch [50/50], Iter [50/157], Loss: total=1.991, reg=0.169, containing_obj=
0.639, no_obj=0.662, cls=0.173
Epoch [50/50], Iter [100/157], Loss: total=1.990, reg=0.172, containing_obj
=0.625, no_obj=0.666, cls=0.171
Epoch [50/50], Iter [150/157], Loss: total=1.988, reg=0.173, containing_obj
=0.622, no_obj=0.669, cls=0.167
---Evaluate model on test samples---
```

```
100%|██████████| 1255/1255 [00:45<00:00, 27.35it/s]
```

```
---class aeroplane ap 0.6022119783511177---
---class bicycle ap 0.5721199352176295---
---class bird ap 0.5272814077186743---
---class boat ap 0.37958515424242983---
---class bottle ap 0.1970295655483863---
---class bus ap 0.5388653729384907---
---class car ap 0.6964292289889491---
---class cat ap 0.7818175308873413---
---class chair ap 0.36967891274796927---
---class cow ap 0.4839731910388159---
---class diningtable ap 0.3087458017418613---
---class dog ap 0.6580636661769885---
---class horse ap 0.6655591059538684---
---class motorbike ap 0.4908202964998829---
---class person ap 0.5032313386719985---
---class pottedplant ap 0.21362909914368527---
---class sheep ap 0.4128880742075391---
---class sofa ap 0.49837665109447216---
---class train ap 0.7531128368339353---
---class tvmonitor ap 0.5869395004225786---
---map 0.5120179324213308---
49 [0.6022119783511177, 0.5721199352176295, 0.5272814077186743, 0.379585154
24242983, 0.1970295655483863, 0.5388653729384907, 0.6964292289889491, 0.781
8175308873413, 0.36967891274796927, 0.4839731910388159, 0.3087458017418613,
0.6580636661769885, 0.6655591059538684, 0.4908202964998829, 0.5032313386719
985, 0.21362909914368527, 0.4128880742075391, 0.49837665109447216, 0.753112
8368339353, 0.5869395004225786]
```

View example predictions

In [ ]:

```
net.eval()

# select random image from val set
image_name = random.choice(val_dataset.fnames)
image = cv2.imread(os.path.join(file_root_val, image_name))
image = cv2.cvtColor(image, cv2.COLOR_BGR2RGB)

print('predicting...')
result = predict_image(net, image_name, root_img_directory=file_root_val)
for left_up, right_bottom, class_name, _, prob in result:
    color = COLORS[VOC_CLASSES.index(class_name)]
    cv2.rectangle(image, left_up, right_bottom, color, 2)
    label = class_name + str(round(prob, 2))
    text_size, baseline = cv2.getTextSize(label, cv2.FONT_HERSHEY_SIMPLEX, (
    p1 = (left_up[0], left_up[1] - text_size[1])
    cv2.rectangle(image, (p1[0] - 2 // 2, p1[1] - 2 - baseline), (p1[0] + te
```

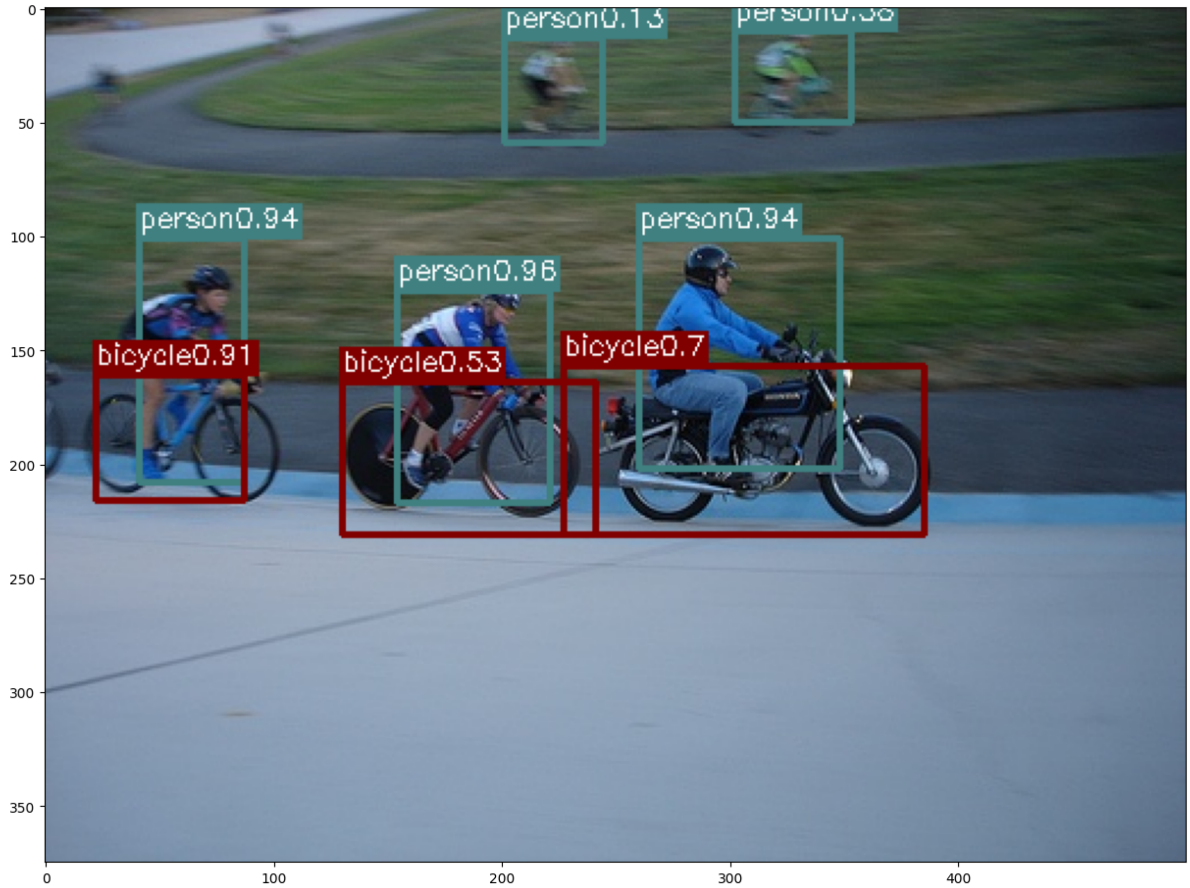
```
                 color, -1)
    cv2.putText(image, label, (p1[0], p1[1] + baseline), cv2.FONT_HERSHEY_SI

plt.figure(figsize = (15,15))
plt.imshow(image)
```

```
predicting...
```

Out[ ]:

```
<matplotlib.image.AxesImage at 0x7d09a7269f60>
```



Kaggle submission (85%)

Predict Result

Predict the results based on testing set. Upload to Kaggle.

How to upload

1. Click the folder icon in the left hand side of Colab.
2. Right click "result.csv". Select "Download"
3. To kaggle. Click "Submit Predictions"
4. Upload the result.csv
5. System will automaticlaly calculate the accuracy of 50% dataset and publish this result to leaderboard.

預測 `test` 並將結果上傳至Kaggle。**連結**

執行完畢此區的程式碼後，會將 `test` 預測完的結果存下來。

上傳流程

1. 點選左側選單最下方的資料夾圖示
2. 右鍵「result.csv」
3. 點選「Download」
4. 至連結網頁點選「Submit Predictions」
5. 將剛剛下載的檔案上傳
6. 系統會計算並公布其中50%資料的正確率

In [ ]:
```
from google.colab import drive
drive.mount('/content/drive', force_remount=True)
```

In [ ]:
```
%cd /content/drive/MyDrive/DeepLearning/A5
```

```
/content/drive/MyDrive/DeepLearning/A5
```

In [ ]:
```
root_test = 'data/VOCdevkit_2007 2/VOC2007test/JPEGImages/'
file_test = 'data/voc2007test.txt'
```

By using the test_evaluate function, you will obtain predictions for each image.

In [ ]:
```
preds_submission = test_evaluate(net, test_dataset_file=file_test, img_root=
```

```
---Evaluate model on test samples---
100%|██████████| 4950/4950 [43:41<00:00,  1.89it/s]
```

The write_csv function will use preds_submission to write into a CSV file called 'result.csv'.

In [ ]:
```
write_csv(preds_submission)
```

Report (15%)

In your report, please include:

a. A brief discussion on your implementation.

b. Report the best train and validation accuracy in all of your experiments and discuss any strategies or tricks you've employed.

c. Report the results for extra credits and also provide a discussion, if any.

Extra Credit (15%)

• Pick a fun video like this one, run your detector on it (a subset of frames would be OK), and produce a video showing your results.

• Try to replace the provided pre-trained network with a different one and train with the YOLO loss on top to attempt to get better accuracy.

• Or any other methods that you try to improve the performance.

Video

Result link: https://youtu.be/PN9ECezV37E

In [ ]:
```
ckpt = torch.load('/content/drive/MyDrive/DeepLearning/A5/checkpoints/best_c
net.load_state_dict(ckpt)
```

In [ ]:
```
video = cv2.VideoCapture("/content/Taylor Swift - We Are Never Ever Getting 
output_dir = "VideoData"

# Create the output directory if it doesn't exist
os.makedirs(output_dir, exist_ok=True)
frame_list = os.path.join(output_dir, 'frame_images_list.txt')
currentframe = 0

with open(frame_list, 'w') as frame_txt_f:
  while True:
    ret, frame = video.read()
    if ret:
        name = os.path.join(output_dir,'frame' + str(currentframe) + '.jpg')
        # print('Creating...' + name)
        cv2.imwrite(name, frame)
        frame_txt_f.write(name + '\n')
        currentframe += 1
    else:
        break

video.release()
cv2.destroyAllWindows()
```

In [ ]:
```
images_dir = "/content/drive/MyDrive/DeepLearning/A5/VideoData"
output_dir = "/content/drive/MyDrive/DeepLearning/A5/ResultData"
output_video_path = "/content/drive/MyDrive/DeepLearning/A5/video_result.mp4
os.makedirs(output_dir, exist_ok=True)
```

In [ ]:
```
image_path = os.path.join('VideoData', 'frame0.jpg')
image = cv2.imread(image_path)
image.shape
```

In [ ]:
```
image_files = [f for f in os.listdir(images_dir)][1:]
fourcc = cv2.VideoWriter_fourcc(*'mp4v')
video_writer = cv2.VideoWriter(output_video_path, fourcc, 20.0, (1280, 720))
```

In [ ]:
```
from tqdm import tqdm
from matplotlib import pyplot as plt
```

In [ ]:
```
net.eval()
for image_file in tqdm(image_files, desc="Processing images"):
    image_path = os.path.join(images_dir, image_file)

    image = cv2.imread(image_path)

    result = predict_image(net, image_file, root_img_directory=f"{images_di
    if result:
        for left_up, right_bottom, class_name, _, prob in result:
            color = COLORS[VOC_CLASSES.index(class_name)]
            cv2.rectangle(image, left_up, right_bottom, color, 2)
            label = class_name + str(round(prob, 2))
```

```
            text_size, baseline = cv2.getTextSize(label, cv2.FONT_HERSHEY_SI
            p1 = (left_up[0], left_up[1] - text_size[1])
            cv2.rectangle(image, (p1[0] - 2 // 2, p1[1] - 2 - baseline), (p1
                          color, -1)
            cv2.putText(image, label, (p1[0], p1[1] + baseline), cv2.FONT_HE

        output_image_path = os.path.join(output_dir, f'pred_{image_file}')
        cv2.imwrite(output_image_path, image)

        video_writer.write(image)

video_writer.release()
```