HW2 Object Detection

Selected Topics in Visual Recognition using Deep Learning

GitHub link

https://github.com/samuelyutt/Selected-Topics-in-Visual-Recognition-using-Deep-Learning-course/tree/hw2/hw2-ObjectDetection

Utilities and reference

- Pretrained weights, configuration files, and datasets: https://drive.google.com/drive/folders/1eSRBqjNiv8d8feY4nC_fiUk8M_Mji9-R?usp=sharing
- YOLOv5: https://github.com/ultralytics/yolov5
- SVHN: https://github.com/chia56028/Street-View-House-Numbers-Detection

Introduction

YOLO is short for you only look once. It is an object detection technic and was first published in 2015 by Joseph Redmon. The latest YOLOv5 was revealed only a short time after YOLOv4 was released. There are 4 official YOLOv5 models, which are YOLOv5s, YOLOv5m, YOLOv5l, and YOLOv5x. YOLOv5s is the smallest model among all and is beneficial in edge learning.

Methodology

Prepare.py

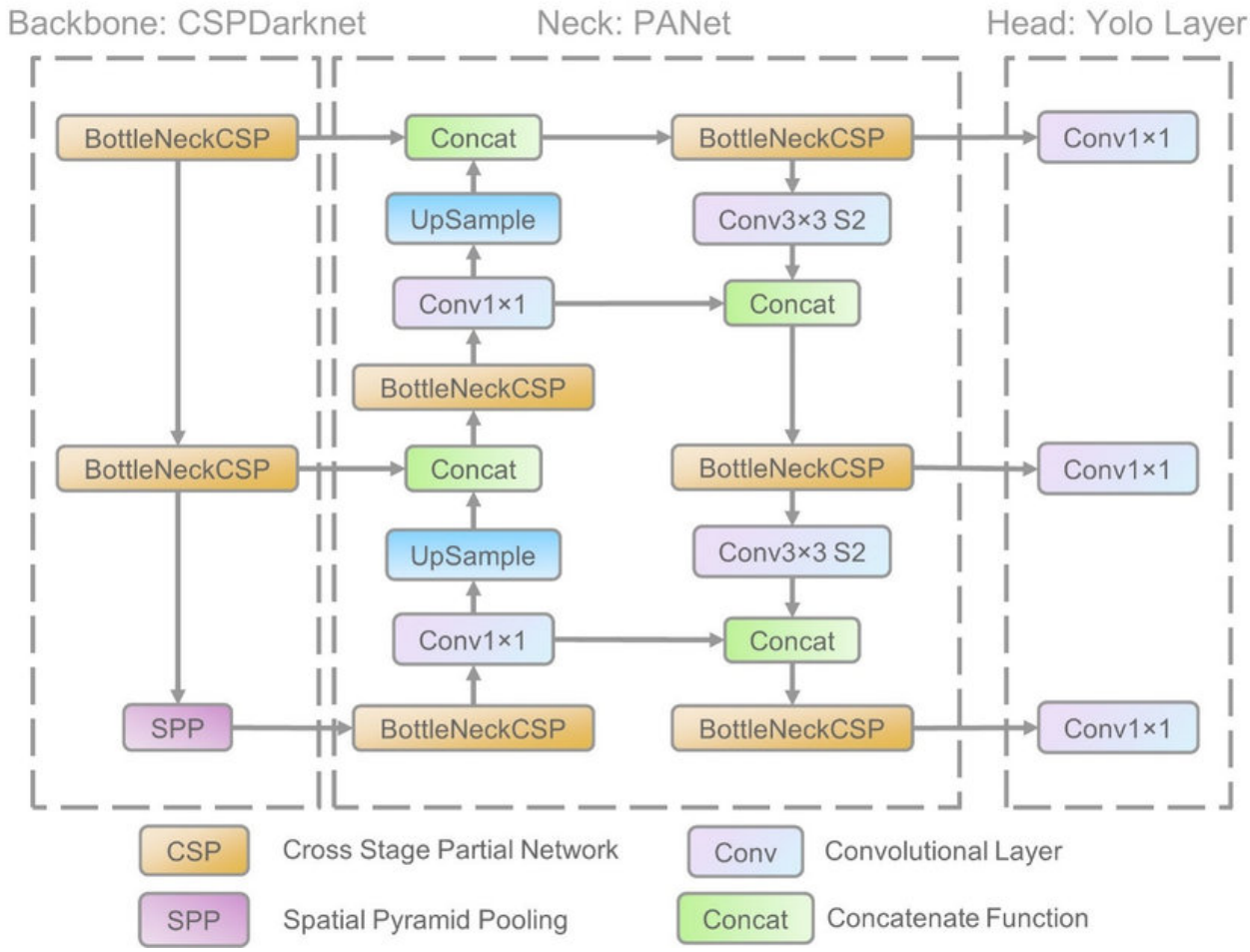
Prepere.py in src/ assumes that the train images is stored in datasets/svhn/origin/train/. After executing prepare.py, train images will be separated into 2 parts. Also, the mat file will be parsed and transform into txt label files.

Data pre-process

In the input part of YOLOv5, Mosaic method is used by applying random scale, random crop, and random arrangement. The configuration of data pre-processing options such as horizontal and vertical flip are written in file hyp.scratch.yaml or hyp.finetune.yaml. You can specify which configuration file to use during training.

Model architecture

There are mainly 4 parts in the YOLOv5 network structure: input, backbone, neck, and prediction.



Backbone

In the backbone part of YOLOv5 network structure, Focus and CSP structure are used.

Neck

CSP2 structures are used in order to increase the ability of feature integration.

Prediction

CIOU_Loss is used as the loss function for the bounding box. Also, weighted NMS increased the ability of detecting hidden objects.

Hyperparameters

Hyperparameters such as lr0, lrf, and momentum are written in hyp.scratch.yaml or hyp.finetune.yaml. The former is used as default when not training from pertained weights. Otherwise, hyp.finetune.yaml is used.

In the training during this homework, I set both flipud and fliplr to 0.0. Therefore, no random flip is applied.

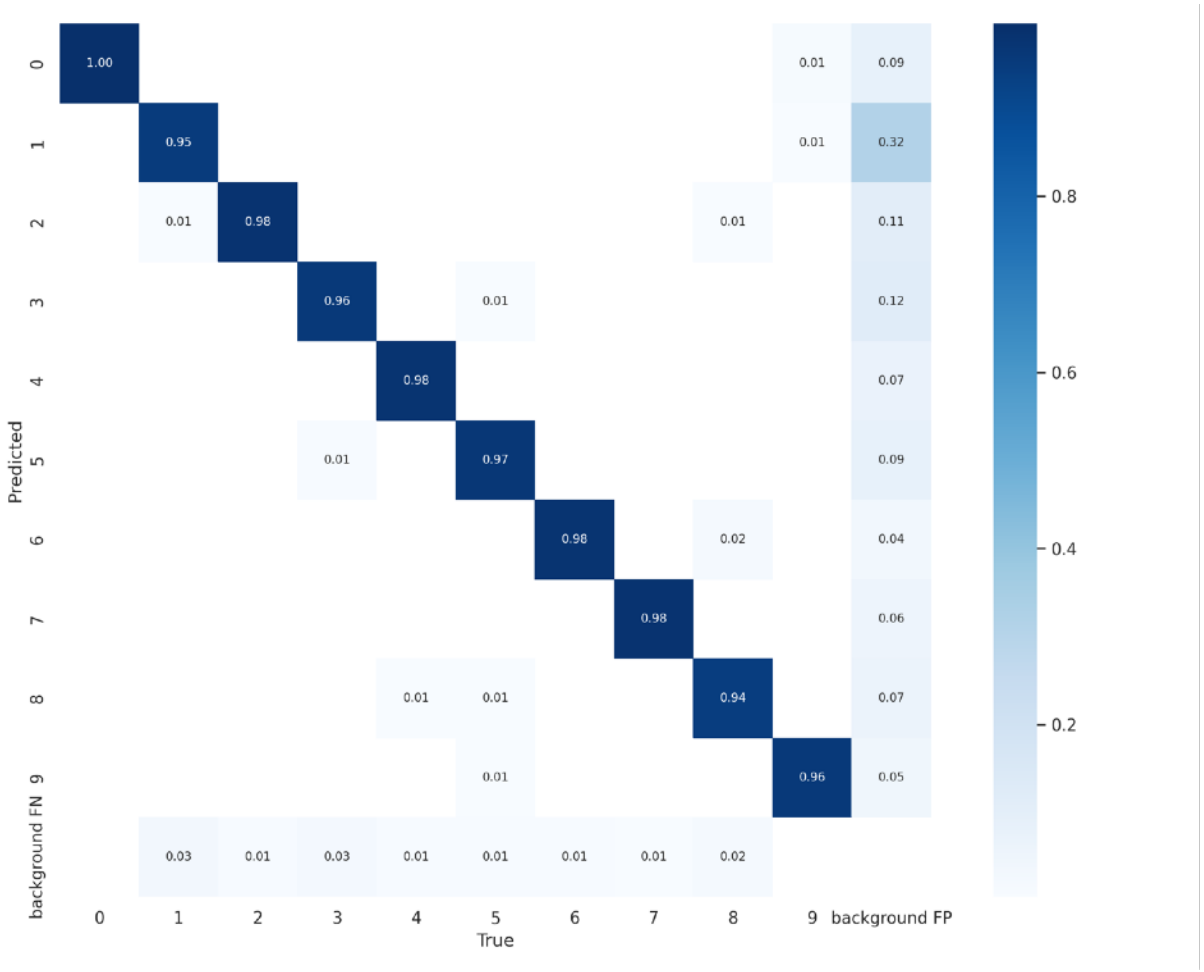
Dataset path, number of classes, and class names are specified in svhn.yaml.

Checkpoint of weights are stored in runs/train/weights/ during each epoch. Meanwhile, a validation test will be performed as well.

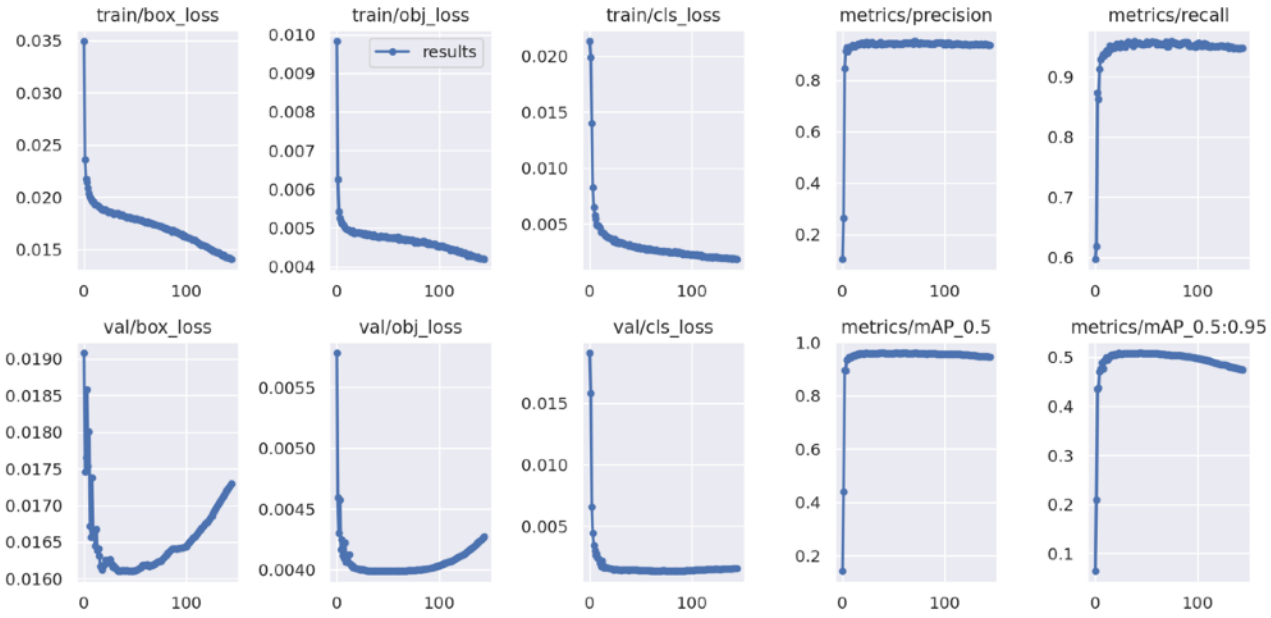
Results

Validation batch examples

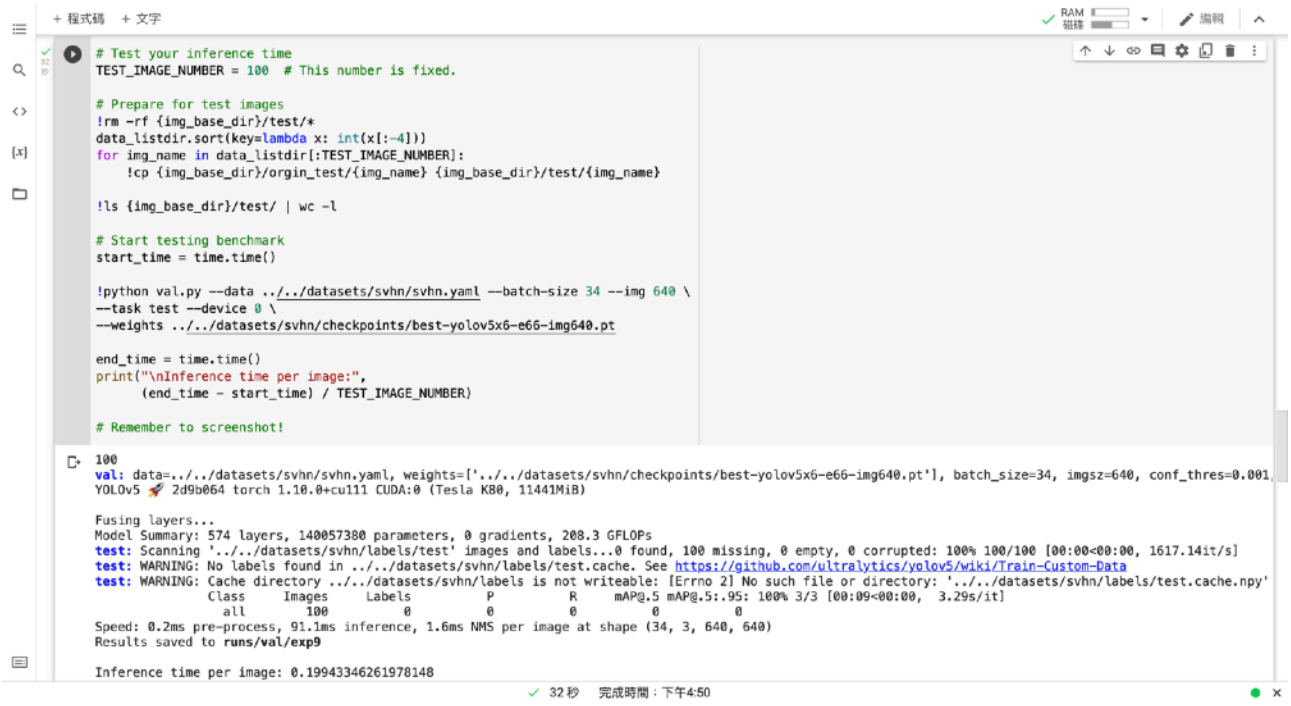
Confusion matrix



Loss



Benchmark



Summary

In this homework, I have learned to train custom datasets using YOLOv5 to perform object detection. I have trained all 4 official models, v5s, v5m, v5l, and v5x. Unsurprisingly, YOLOv5x received the highest score on test set. Since, YOLOv5 is capable of fitting different image size automatically, I set image size to 320 and 640. The former image size is 4 times faster than the latter is in training but also results in less accurate than the latter results in.