YOLOv5 object detection and Unsupervised Domain Adaptation tutorial

Step 1: Sourece model

Training (You can refer to ConfMix website)

- I train Source-YOLOv5 model locally with Cityscapes datasets, just follow the steps below (use cmd command line), in this stage will only use the org_train and org_valid to train the source model
- Quickstart: You can use "hw3_train.sh" to directly start traing, or you can also following the detail below

```
bash hw3_train.sh $1 \ # path to org training image
                  $2 \ # path to org validation image
                  $3 \ # path to fog training image
                  $4 \ # path to fog validation image
                  $5 \ # path to org training images' label json file
                  $6 \ # path to org validation images' label json file
                  $7   # path to fog validation images' label json file
```

1. Environment setting (python 3.10 / pytorch 1.13.1)

```
pip install -r requirements.txt
```

2. Data preparation

```
# ConfMix file is already exist, if you miss the file, you can use git clone to
get a new one
git clone https://github.com/giuliomattolin/ConfMix.git
# you can also use "JSON_to_txt.py" if you need to convert json file into txt file
(input_dir should cotain both train.coco.json and val.coco.json)
python3 JSON_to_txt.py --input_dir path/to/train&valid/JSON/file --
output_dir_train path/to/train/txt/labels --output_dir_val
path/to/valid/txt/labels
# you can use "hw3_download.sh" to directly download the five difference pretrain
weight I get
bash hw3_download.sh
```

3. Place your datasets under the folder you want to use (datasets should to put like this), the labels should turn into yolov5 format(txt file)

```
Root___
      |__org___images___train
      |      |        |_valid
      |      |
      |      |_labels___train
      |               |_valid
      |__fog___images___train
             |        |_valid
             |        |_public_test
             |
             |_labels___train(place pseudo-label)
                      |_valid
```

4. Modify the path setting in the Cityscapes2Foggy.yaml(I rename it to hw3_source.yaml) under the data folder or you can use the "prepare_datasets_yaml.py" to generate the hw3_source.yaml

```
# Generate hw3_source.yaml for training source model
python3 prepare_datasets_yaml.py --train path/to/org/images/train --val
path/to/org/images/valid --test path/to/org/images/valid --uda
path/to/fog/images/train --output_dir ConfMix/data/hw3_source.yaml
```

5. The pretrain weight will be downloaded automatically when you start to train the model
6. **If you use the git clone to get a new ConfMix, there has some bug you need to fix by modify the line 234'indices.append...' to 'indices.append((b, a, gj.clamp(0, gain[3] - 1).to(torch.int64), gi.clamp(0, gain[2] - 1).to(torch.int64)))' in ConfMix/utils/loss.py**

```
cd ConfMix
# Run main.py to train the model(you can modify the hyperparameters to whatever
you want)
python3 train.py --name cityscapes --batch 8 --imgsz 640 --epochs 170 --data
data/hw3_source.yaml --weights yolov5x.pt --hyp data/hyps/hyp.scratch-cos.yaml --
image-weights --optimizer AdamW
```

Inference

- You can follow the steps just like hw3_inference.sh (you can inference both org and fog validation sets to compare the different)
- The source model weight obtain by Step 1 can be directly downloaded by hw3_download.sh
- Quickstart: You can use "hw3_inference.sh" to directly start inference

```
bash hw3_inference.sh $1 \ # testing images path (e.g. input/test_dir/)
                  $2 \ # path of output json file (e.g. output/pred.json)
                  $3   # the wight you want to use for inference (e.g. 0-4)
# 0 = source weight, 1 = 33% UDA weight, 2 = 66% UDA weight, 3 = 100% UDA weight,
4 = best UDA weight
```

If you use the git clone to get a new ConfMix, there has some bug you need to fix by replace the 'non_max_suppression' funtion in ConfMix/utils/general.py by newer version of the yolov5's 'non_max_suppression' funtion to perform inference(YOLOv5 link)

Pseudo-Labelling

- Before going to the Step 2, we need to use the best.pt which obtain from the step 1 to inference foggy-train datasets, generating Pseudo-Label to perform the 'Unsupervised Domain Adaptation'
- You need to create a yaml file first (like coco128.yaml format) to detect the specific datasets (The yaml file should cotain the correct class numbers)

```
python3 detect.py --weights path/to/best.pt/from/Step1 --source
path/to/fog/images/train --data path/to/yaml/file --save-txt
# Then move the output txt file to the root/fog/labels/train to create Pseudo-
Label
```

Visualization: Draw the bounding boxes on image

Just using the detect.py (you can modify the path and weight)

```
python3 detect.py --weights path/to/best.pt/from/Step1 --source path/to/test_image
--data path/to/yaml/file
```

Step 2: Unsupervised Domain Adaptation model

Training (You can refer to ConfMix website)

- I train Adapt-YOLOv5 model locally to perform Cityscapes to Foggy, just follow the steps below (use cmd command line), in this stage will only use the fog_train and fog_valid to train the source model
- Quickstart: You can use "hw3_train.sh" to directly start traing, or you can also following the detail below

```
bash hw3_train.sh $1 \ # path to org training image
                   $2 \ # path to org validation image
                   $3 \ # path to fog training image
                   $4 \ # path to fog validation image
                   $5 \ # path to org training images' label json file
                   $6 \ # path to org validation images' label json file
                   $7   # path to fog validation images' label json file
```

---

1. Use the best.pt model obtain by Step 1 to perform adaption (Cityscapes to Foggy), the weight obtain by Step 1 can be directly downloaded by hw3_download.sh
2. Modify the path setting in the Cityscapes2Foggy.yaml(I rename it to hw3_UDA.yaml) under the data folder or you can use the "prepare_datasets_yaml.py" to generate the hw3_UDA.yaml

```
# Generate hw3_UDA.yaml for training UDA model
python3 prepare_datasets_yaml.py --train path/to/org/images/train --val
path/to/fog/images/valid --test path/to/fog/images/valid --uda
path/to/fog/images/train --output_dir ConfMix/data/hw3_UDA.yaml
```

3. Start training UDA model

```
# run uda_train.py to adapt the model
python3 uda_train.py --name cityscapes2foggy --batch 4 --epochs 151 --data
data/hw3_UDA.yaml --weight runs/train/cityscapes/weights/best.pt  --imgsz 640 --
save-period 50
```

Inference

- You can follow the steps just like hw3_inference.sh (you can inference both org and fog validation sets to compare the different)
- The UDA weight obtain by Step 2 can be directly downloaded by hw3_download.sh
- Quickstart: You can use "hw3_inference.sh" to directly start inference

```
bash hw3_inference.sh $1 \ # testing images path (e.g. input/test_dir/)
                  $2 \ # path of output json file (e.g. output/pred.json)
                  $3   # the wight you want to use for inference (e.g. 0-4)
# 0 = source weight, 1 = 33% UDA weight, 2 = 66% UDA weight, 3 = 100% UDA weight,
4 = best UDA weight
```

Visualization: Draw the bounding boxes on image

Just using the detect.py (you can modify the path and weight)

```
python3 detect.py --weights path/to/best.pt/from/Step2 --source path/to/test_image
```