Homework #3
Domain Adaptation

Computer Vision Practice with Deep Learning
NTU, Spring 23

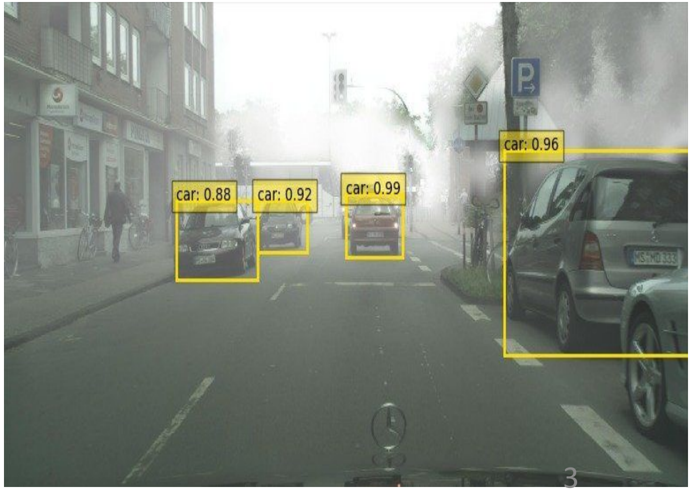
Outline

- Dataset
- Problem
- Evaluation
- Report
- Grading
- Submission
- Rules

Unsupervised Domain Adaptation for Object Detection

- Object detection - localize objects and predict class for each image.
  - Input : RGB image
  - Output : a sequence of bounding boxes with class label and confidence.

- **Unsupervised Domain Adaptation**
  - Adapt a model trained on source dataset A to target dataset B **without using any labels from B**



Dataset

- Source dataset (Clear)
  - **Train / Val : 2575 / 300**
- Target dataset (Foggy)
  - Train / **Val** / Public test
    - 2575 / **300** / 300
- Annotations are in COCO format

Dataset

- download dataset (12 GB)
- Although there is a one-to-one correspondence between the source and target datasets, **attempting to find the corresponding pair is prohibited**.
- Violating any of the rules outlined for this assignment will result in a grade of **zero**.
- If you are uncertain about the legitimacy of the usage, email the TAs for clarification.

Dataset

- The source dataset contains both the input data (e.g., images) and their corresponding ground truth (GT) labels. On the other hand, in the target dataset, only the validation set contains input data and corresponding GT labels, while the other partitions contain only input data.
- Because our task is UDA, it is **prohibited to use the labels for the target dataset** except for validation.

Problem - 1

- Source model (trained on **Source dataset**)
    - Either a **CNN**-based model or a **Transformer**-based model is acceptable
    - Please train a model on your own and the pretrained weight you are only allowed to use is pretrained on **COCO and ImageNet** dataset.

Problem - 2

- Adapt to **target dataset**
  - You are free to reference any paper or develop your self
  - Survey two domain adaptation papers (report Q3/Q4)
    - The reference papers should be cited in the report
  - Implement **one** of the methods you surveyed
    - You are free to combine other tricks
  - The usable data include the **source images, the source labels, and the target images.** Attempting to use **extra data** (e.g., validation data) or to **find the corresponding labels for target images** is **prohibited**.
  - Notes: Since there are various DA methods, problem2 **may not necessarily** require loading the trained source model from problem 1, but can also train from scratch or load pretrained weights from COCO or ImageNet.

Evaluation

- Evaluation metric
  - We'll use the metric taught in class - Average Precision
  - Please refer to the course slides or this intro
  - The performance will be evaluated by this function
- Ranking & Baseline
  - mAP at IoU = 50

Grading

- Baseline(40%) **(Target dataset) (mAP@50)**
  - Public baseline(20%): 300 public testing data
    - simple baseline (7%): 0.28
    - median baseline (7%): 0.32
    - strong baseline (6%): 0.35
  - Private baseline(20%: 7% / 7% / 6%): private testing data
    - Three different baseline levels will be announced after the deadline.
- Ranking(20%) **(Target dataset) (mAP@50)**
  - Public(5%): public testing set
  - Private(15%): private testing set
- Report(40%)

Report

1. (5%) Report the performance of your **trained source model** on the **source validation set**
    a. at least with mAP@[50:5:95], mAP@50, mAP@75
    b. Please plot the mAP curve. (The requirements for the mAP curve are specified on page 15)
    c. The model used should be specified and cited
        - e.g., DETR - https://arxiv.org/abs/2005.12872

2. (5%) Report the performance of your **trained source model** on the **target validation set (w/o any adaptations)**

Report

3. (10%) Please provide a introduction to the **two domain adaptation methods** you used.
    a. The reference papers used should be cited.
    b. e.g., A Robust Learning Approach to Domain Adaptive Object Detection - https://arxiv.org/abs/1904.02361
4. (10%) Please compare the two methods and describe their respective **advantages and disadvantages**.
5. (10%) Report the performance of the **adapted model** on the **target validation set**
    a. at least with mAP@[50:5:95], mAP@50, mAP@75
    b. plot the mAP curve.
    c. Magnitude of improvement in mAP@50 (adapted model - source model)

Report (Bonus)

6. (5%) Please use either **UMAP, PCA, or T-SNE** to project the backbone features onto a 2D or 3D space, and include all four features extracted from the following four settings, resulting in a total of 1200 data points. **Color-code each data point** according to which feature it corresponds to in each setting, and provide a **legend** to clarify the color scheme. Finally, please **briefly discuss any interesting findings** from the projection results.
   The four settings are:
   a. source model (prob.1) - inference on clear-val (300 data)
   b. source model (prob.1) - inference on foggy-val (300 data)
   c. adapted model (prob.2) - inference on clear-val (300 data)
   d. adapted model (prob.2) - inference on foggy-val (300 data)

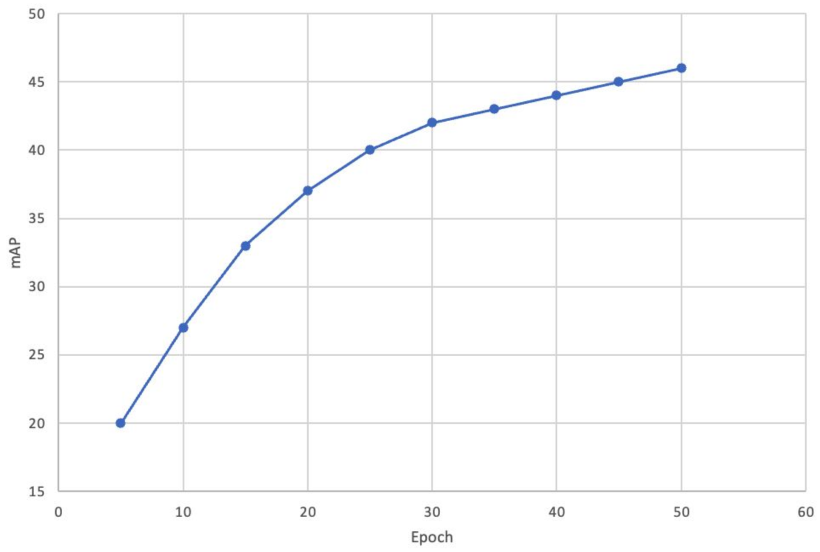
demo video

Report (Bonus)

7. (5%) Please compare the final mAP50 of the adapted model trained from the following two different initial weights. (refer to page.8)
   a. trained source model
   b. random initial or pretrained weights from [COCO, ImageNet] datasets

mAP Curve

1. Plot a graph to log the validation **mAP@50** at different epochs.
2. The validation frequency can be set to any number to ensure that there are at least **10** logged points.



Submission

- Deadline: **2023/5/23 (Tue.) 23:59 (GMT+8)**
- Submit report to Gradescope
- Compress all files except for report, then submit it to NTU Cool.(hw3_<studentID>.zip)
- Your submission should include the following files:
  - hw3_inference.sh
  - hw3_train.sh
  - hw3_download.sh
  - requirements.txt
  - README.md (if needed)
  - train.md
  - your python files
- **DO NOT submit the dataset**

Check your submission

unzip -l hw3_r11912345.zip

```
Name
----
hw3_r11912345/
hw3_r11912345/train.md
hw3_r11912345/hw3_train.sh
hw3_r11912345/hw3_download.sh
hw3_r11912345/requirements.txt
hw3_r11912345/your_python_files/
hw3_r11912345/README.md
hw3_r11912345/hw3_inference.sh
```

Submission

- 注意
  - 請將檔案放於學號資料夾內再進行壓縮
  - 請提供 requirements.txt，助教會幫你安裝
  - 如果無法直接使用pip install -r requirements.txt，請提供README.md
  - hw3_inference.sh 只能有執行 inference code的指令，不要在裡面加 pip install

Model Checkpoint

- Download and preprocess (e.g. unzip, tar zxf, etc.) your model in hw3_download.sh.
    - TAs will run `bash hw3_download.sh` prior to any inference.

- Four model checkpoints need to be submitted.
    - Please choose one adaptation method from the two methods you used.
    - Please submit **a initial checkpoint**, **two intermediate checkpoints** and **a final checkpoint**.
        - The four models should correspond to **0% (source, random initial or pre-trained weight), 33%, 66%, and 100%** of your adaptation process.
        - Ex: If you adapt your source model with 100 epochs, the checkpoints at the 0th, 33rd, 66th, and 100th epochs need to be submitted.

- **DO NOT** delete your model checkpoints before the TAs release your score and before you have ensured that your score is correct.

Script - hw3_inference.sh

- Provide a **script** to test images under the specified directory with your model, and save the results in the specified json file.
- TAs will run your script as shown below:
    - bash hw3_inference.sh $1 $2 $3 (e.g. bash hw3_inference.sh "./input/test_dir/" "./output/pred.json" 3)
        - $1: testing images directory (e.g. input/test_dir/) that contains images in subdirectories
            - You may use `glob.glob('input/test_dir/**/*.png', recursive=True)`
            - The keys in prediction should contain the path of subdirectories
            - e.g. for an image ./input/test_dir/fog/val/3000.png, the key in json should be fog/val/3000.png
        - $2: path of output json file (e.g. output/pred.json)
        - $3: one of the numbers between 0 and 3, specifying which checkpoint to use, e.g., 0 indicates the 0% checkpoint, and 3 indicates the 100% checkpoint.
- The performance of the four checkpoints should be **consistent** with the mAP curve.
- The output json file **must** have the same format as the sample_submission
- This section must be finished in **5 mins**, otherwise would be considered as a failed run.
- Your code should only output indicated files
    - Do not use imshow() or show() or save images in your code

Script - hw3_train.sh

- Please provide a script to reproduce the results of problem 1 and 2.
- If the reported performance and submitted checkpoints cannot be reproduced, your assignment score will be **0**.
- TAs will run your hw3_train.sh following your train.md:
    - You have to write a readme.md to specify any training details
    - train.md MUST contain command and explanation
    - The outputs should at least contain four checkpoints (0%, 33%, 66%, 100%)
    - example

- commamd

```
bash hw3_train.sh $1 $2 $3
```

- explanation
    - $1: training images directory (e.g. `data/train_dir/`)
    - $2: validation images directory (e.g. `data/val_dir/`)
    - $3: path of the best trained model checkpoint (e.g. `best_model.pth`)

Sample JSON Format

- Overview
  - sample_submission.json is provided

    key: “[subdirectories]/****.png”

```
"fog/public_test/2934.png": {
    "boxes": [...
    ],
    "labels": [...
    ],
    "scores": [...
    ]
},
"fog/public_test/3058.png": {...
},
```

Each prediction contains:
labels, *boxes*, and *scores*
Label: class
Box: [x_min, y_min, x_max, y_max]
(according to coordinate
before any transformation)
Score: confidence

```
"labels": [
    8,
    8,
    1,
    1,
    1
],
```

```
"boxes": [
    [
        1180.511962890625,
        420.691162109375,
        1251.191650390625,
        500.2518005371094
    ],
```

```
"scores": [
    0.2499220371246338,
    0.23858648538589478,
    0.23759225010871887,
    0.23512735962867737,
    0.23340454697608948
]
```

Check your JSON file

- check_your_prediction_valid.py is the same as the one in hw1
- python3 check_your_prediction_valid.py [your_prediction_path] [target_path]
    - e.g. python3 **check_your_prediction_valid.py** pred.json data/val.coco.json

Rules – Environment

- Ensure your code can be executed successfully on **Linux Ubuntu**
- Language and framework
  - **python3 (3.10)** is the only language you should use
  - **PyTorch (1.13.1)** is the only allowed **framework**
- Please keep in mind that any attempt to attack the system, including the use of 'sudo', is not allowed and will result in a 0 for this homework.
- We do NOT recommend using distributed inference, and if you choose to do so, you assume all associated risks if your code cannot be executed properly.
- Do NOT hardcode any absolute paths in your program, but instead use input arguments to specify the paths.

Rules – Code

- If your code cannot be executed, you have one chance to make minor modifications to your code.

	Success to fix the code	Fail to fix the code
deduction	80%	0

- Reproducibility
  - you must make sure that the training results can be reproduced with the code you submit to NTU COOL

Rules – Policy

- **Late policy:**

late (hour)	(0, 24]	(24, 48]	(48, 72]	>72
deduction	70%	50%	30%	0

- **Plagiarism**
  - You must document all of your source material in your report. You must also cite any sources (e.g. GitHub repo, paper, website) from which you obtain numbers, ideas, or other material. Use of ChatGPT or other AI composition software is permitted, but you must specify the usage.
  - We encourage you to discuss ideas with your classmates, but please do NOT share your code directly with them.
  - Plagiarism is a serious offense and will not be treated lightly.

Helps

- Free GPU Resources
  - Google Colab
    - Introduction
    - GPUs: K80, or rarely T4
    - Limits
  - Kaggle
    - Introduction
    - GPUs: T4 x 2 or P100
    - Limits: 30 hours per week

Helps

- Mail
  - If you have any questions, contact TAs via this email
    - ntu.cvpdl.ta@gmail.com
  - Please note that emails sent to TAs' personal email addresses will not receive a response.