Computer Vision
Homework 4: Handling Domain Shift

Due on Monday, December 4, 2023

Abstract

We always want a generalized model that can quickly adapt to other target domains. But in practice, the gap between the source and target domains makes it difficult for the model to adapt to various scenarios. So through this assignment, you can understand the domain shift problem between the source domain and the target domain, try to find or propose some solutions, and test and discuss them. Since homework3 has learned the classification problem, it is suggested that you can discuss the difference between object detection and classification problems in this assignment as well.

Model

YOLOv7: We selected recent publications with faster architectures and better object detection performance to reduce computational resource constraints. In this assignment, we uniformly qualify the model named yolov7.pt as the pre-training weight. You **MUST** use this as the initial weights for all questions below. (Run train.py can automatically download the pre-trained weight into the folder) Moreover, Readme.md at the YOLOv7 official GitHub can find additional details about environment installation, training commands, and test commands.

Dataset

Citycam: Many surveillance cameras generate this dataset. In our case, we have chosen 170, 173, 398, 410, 495, and 511 cameras and have converted the labels to YOLO format as our dataset. Here we split different datasets for different questions, such as Q1, Q2, and Q3. Furthermore, download the dataset on google drive to **{your_path}/yolov7/data/** and generate a YAML file from **{your_path}/yolov7/data/preprocess.py** to train or test your model.

Download and Installation

You can download source codes, datasets, and pre-trained weight from here. If you don't have a computer with GPU, you can run your code in colab and skip the Environment step.

Environment

Please use the following command to install the required python libraries or you can see the YOLOv7 official GitHub. For the GPU version of PyTorch, please refer to its official website for downloading.

```
conda create -n yolov7 python=3.7
conda activate yolov7
cd yolov7
pip install -r requirements.txt
```

Pre-process

You need to download the CityCam dataset and put it in {your_path}/yolov7/data.

```
python data/pre_process.py --data_folder './data/CityCam' --ques 'Q1'
```

Training

```
python train.py --project runs/train/Q1 --name refine
```

Testing

trained_path: The path of model weight. Ex: –weights ./runs/train/Q1/refine/weights/best.pt

```
python test.py --verbose --show_div_cams --task test --weights {trained_path}
```

Problems

You need to run the pre-processing command before training for all questions below. Then test and discuss in your report. Write down your insights and results of all problems in the report. The file name should be **report.pdf**.

[TODO]

There are two TODO functions in {your_path}/yolov7/data/pre_process.py, and you can run the pre-process command to generate a training YAML file for each question after you finish the TODO functions.
1. split_train_val_path: You need to split all image paths to train and validate in Q1, Q2, and Q3.
2. select_images: You need to finish this function to choose better images in Q2 and Q3.

Q1

First, we need to learn how to split the data and observe the impact of the train/validation split on the model. Find the **split_train_val_path** function in data/preprocess.py and write the code to generate the training YAML file using the Q1 dataset. You can add any trick with Q1 datasets and discuss the results for each camera in the test set. The discussion of performance and dataset analysis is the focus of scoring. Hint: You can analyze FP or FN with some error cases in the test set. (Note! You need to use the performance, figure, or any result you generated to justify your discussion, not just use text descriptions.) (20%)

- Only can use the 200 labeled data in Q1 datasets.

Q2

Assuming we have more abundant data today and we need, within the constraints of resources, to filter and select useful data to address Domain Shift. Propose a solution to sample the same amount of data as the Q1 dataset from the Q2 dataset, which has more data, and finish the **select_images** function in data/preprocess.py to generate the training YAML file. Furthermore, you can add any trick to train it, then discuss the different performance in each camera compared with Q1. The performance improvement compared to the initial Q1 result and the discussion of your proposed approach are the focus of the scoring. (Note! You need to use the performance, figure, or any result you generated to justify your discussion, not just use text descriptions.) (35%)

- Only can sample 200 labeled data in Q2 datasets.
- Is your proposed approach better than random samples? If not, explain why.

Q3

In addition to the labeled data in our lives, there is also a vast amount of unlabeled information, which can be used to assist in our training and reduce the cost of manual labeling. In the final question, besides utilizing the dataset selected in Q2, you can incorporate another 1,200 unlabeled images. In the following rules, you can try any method to improve performance. Hint: freeze backbone, positive weight, focal loss (ref: **hyp.scratch.p5.yaml + line 533 in train.py**), or pseudo label, among other options. Regarding more strategies related to training and pseudo-labeling, you can refer to the parameters passed in **train.py** and **test.py** (via **parser.add_argument**). Even if the performance is not a significant improvement, you can still get some scores if you try many methods and discuss them in the report. (45%)

- Only can select 200 labeled data in Q2 datasets.
- Other human-labeled data cannot be used.
- While you can employ any strategy in the Q3 dataset, manually labeling 1200 images by yourself is not allowed.

Supplementary Explanation

As the functions `split_train_val_path` and `select_image` may have different designs under various experimental settings, feel free to explain the strategies and include screenshots of the code in the report discussion.

Submission

Please compress all source codes and report in a single zip file (excluding datasets and model weights), and upload it to the eeclass system. The name should be **HW4_YOURID.zip**.

Note

To meet the eeclase upload size requirement, please try removing all weight (`*.pt`) files as follows:

- All files in the `weight/` folder that are contained within the `yolov7/runs/train` directory or its subdirectories.
- `yolov7/traced_model.pt, yolov7/yolov7.pt`

If it still doesn't meet the requirements, try deleting the temporary cache files generated during runtime, such as `__pycache__`.

Contacts

If you find anything wrong or have questions about this homework, please contact me via email (Ting-Ying Lin, tintin890327@gmail.com).

Acknowledgements. This homework is modified from https://github.com/WongKinYiu/yolov7.

Reference

[1] Chien-Yao Wang, Alexey Bochkovskiy, and Hong-Yuan Mark Liao. "YOLOv7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors". In: arXiv preprint arXiv:2207.02696 (2022).

[2] Shanghang Zhang, Guanhang Wu, João P. Costeira, and José MF Moura, Understanding Traffic Density from Large-Scale Web Camera Data, In IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2017, pp. 4264-4273.