CVPDL: Computer Vision Practice with Deep Learning

Homework #1
Object Detection

Wen-Huang Cheng (鄭文皇)

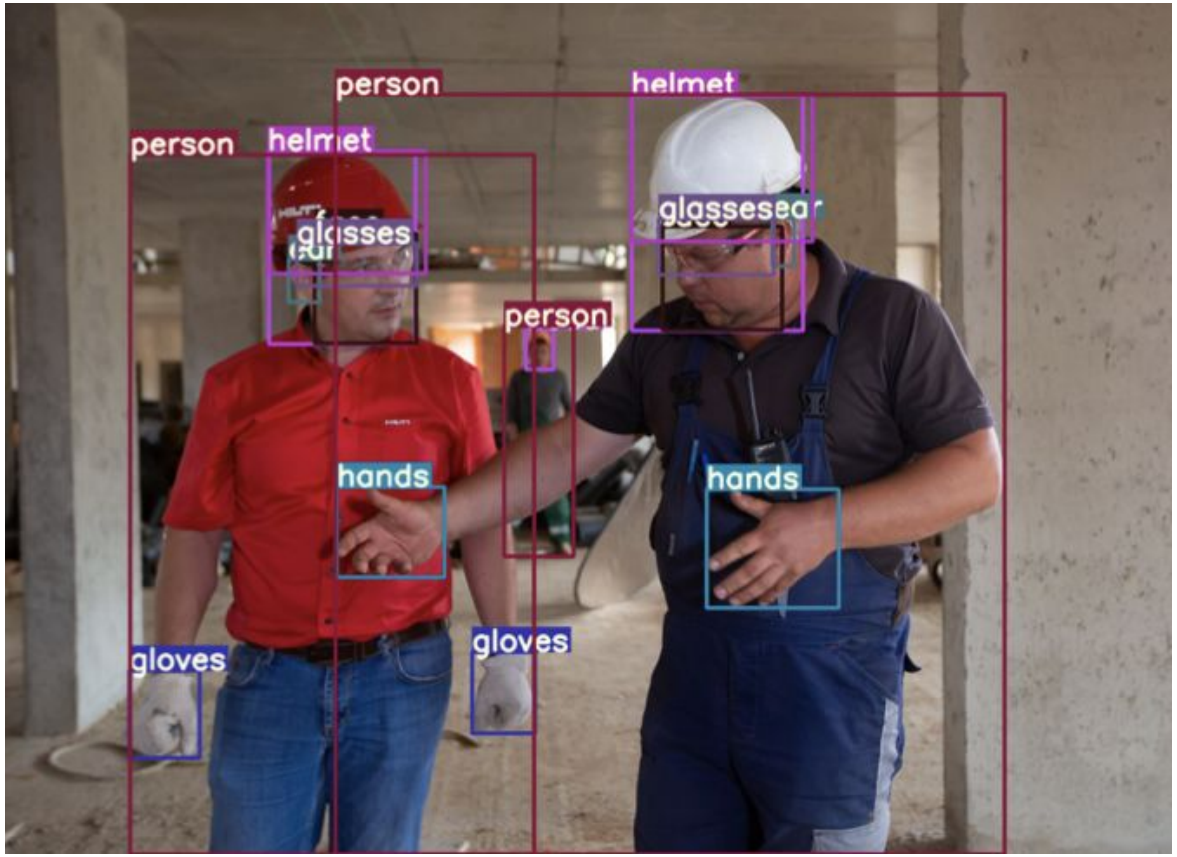
National Taiwan University

wenhuang@csie.ntu.edu.tw

HW1 - Object Detection

TOPIC: Object Detection for Occupational Injury Prevention

- Input: 2D RGB image
- Task: localization and classification
- Output: N x [points, confidence]



Model Constraints

- Model Constraints for this Homework
  - You must use either Transformer-based or Mamba-based model.
  - Failing to do so will result in a deduction of 50 points.
- Within these constraints, any method and pre-trained weights are allowed
- Recommended Model Structure

A. Transformer based method

- DETR, deformable DETR

B. Mamba based method

- Vision-Mamba, Mamba-YOLO

Dataset Description

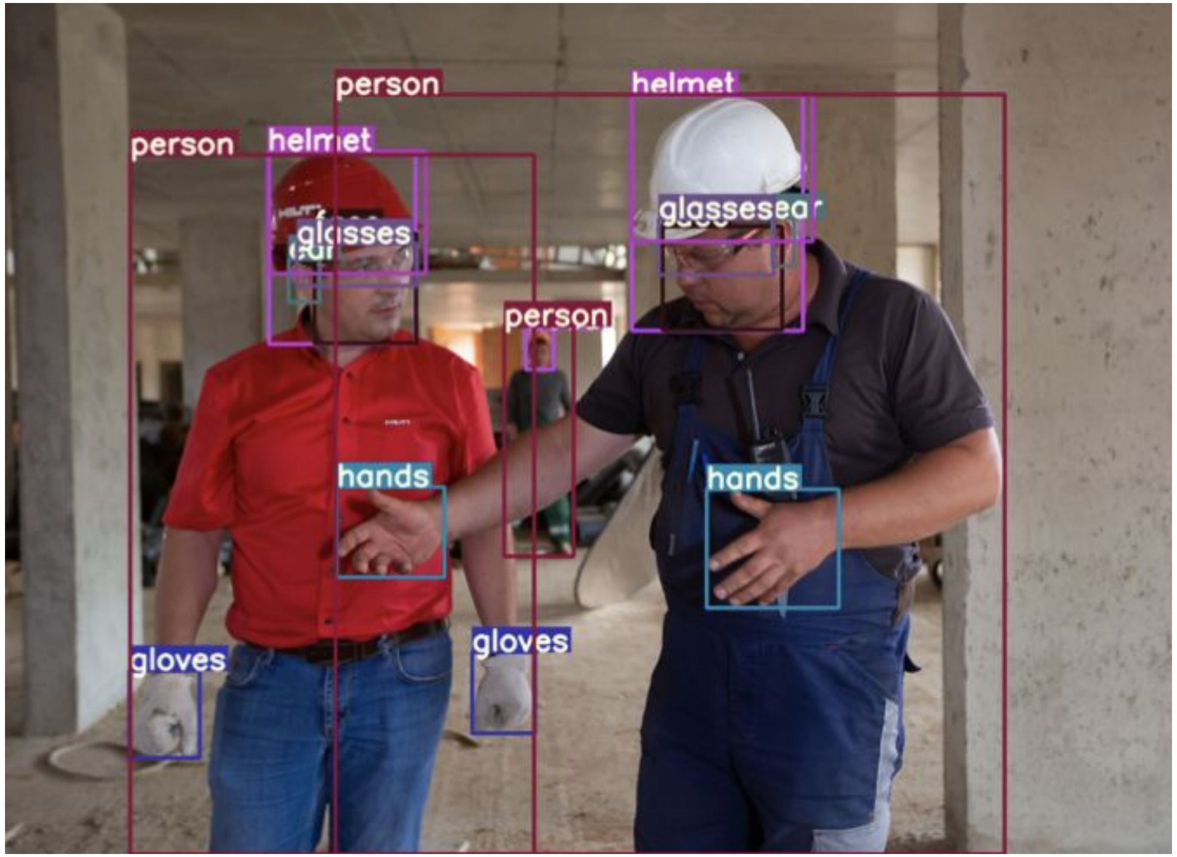
- Download Link: link
- Training Set: 4319 images
- Validation Set: 2160 images
  - DO NOT use the validation set for training
- Testing Set: 1620 images
  - DO NOT try to find the ground truth

- Violating the rules on this page will result in a score of zero.
- If you are uncertain about the legitimacy of the usage, email the TAs for clarification

Dataset Description

The dataset contains a total of 17 categories.

- 0 Person
- 1 Ear
- 2 Earmuffs
- 3 Face
- 4 Face-guard
- 5 Face-mask-medical
- 6 Foot
- 7 Tools
- 8 Glasses
- 9 Gloves
- 10 Helmet
- 11 Hands
- 12 Head
- 13 Medical-suit
- 14 Shoes
- 15 Safety-suit
- 16 Safety-vest



Grading Policies

- Baseline (validation set) (40%)
  - Simple baseline (20%) : 0.35 mAP
  - Strong baseline (20%) : 0.45 mAP

- Performance ranking (testing set) (30%)
  - Linear grading

- Report (30%)

Evaluation Metric

- 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

- mAP is used for all evaluation
  - i.e., AP at IoU = [50:5:95]

Report

1) Draw the architecture of your object detector
    - In brief and clear
    - It would be fine to copy the figure from the paper
2) Implement details
    - e.g.: augmentation, loss function, parameter settings
3) Table of your performance for validation set (mAP, AP$_{50}$, AP$_{75}$
4) Visualization and discussion
    - Demonstrate the detection results, discussion for the long tail effect, etc.

Submission Rules

- Deadline
  - 2024/10/16 (Wed.) 23:59
- Upload filename and format
  - hw1_<student-id>.zip (e.g. hw1_D12345678.zip)
- Submit to NTU cool
  - Do not upload the dataset (or 10 points will be deducted)

Submission Rules

- Your submission should be a zipped file with the following structure:
  - hw1_<student-id>.zip

|-- hw1_<student-id> (Should contain this folder, not separate files)

|-------- hw1_<student-id>.pdf (Your report)

|-------- valid_<student-id>.json (Your prediction file of validation set)

|-------- test_<student-id>.json (Your prediction file of test set)

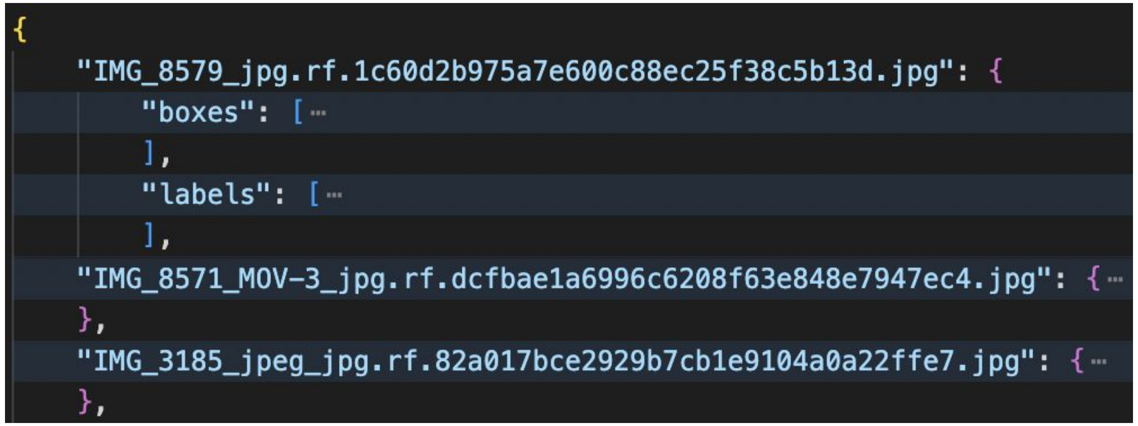
|-------- Codes for training and testing

|-------- README file

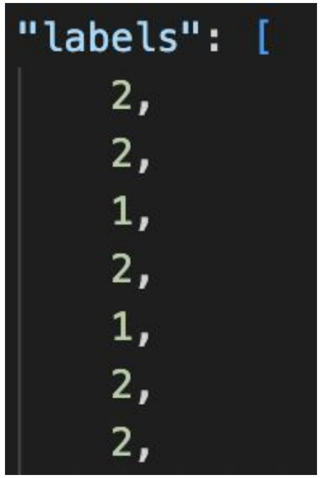
- Your environment details
- How to run your code

Output JSON format

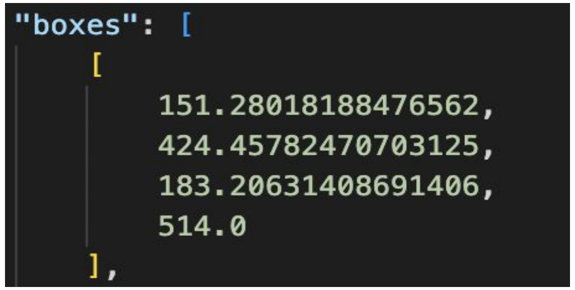
- ➢ sample_submission.json will be provided



```
{
    "IMG_8579_jpg.rf.1c60d2b975a7e600c88ec25f38c5b13d.jpg": {
        "boxes": [⋯
        ],
        "labels": [⋯
        ],
    "IMG_8571_MOV-3_jpg.rf.dcfbae1a6996c6208f63e848e7947ec4.jpg": {⋯
    },
    "IMG_3185_jpeg_jpg.rf.82a017bce2929b7cb1e9104a0a22ffe7.jpg": {⋯
    },
```



```
"labels": [
    2,
    2,
    1,
    2,
    1,
    2,
    2,
```



```
"boxes": [
    [
        151.28018188476562,
        424.45782470703125,
        183.20631408691406,
        514.0
    ],
```

- ➢ Check your performance on the validation set
  - ○ $ python eval.py <`your_prediction.json`> <`valid_target.json`>
  - ○ The coordinate format for training set : Normalized (x_center, y_center, width, hight)
  - ○ The coordinate format for `valid_target.json` and the test set : (x_min, y_min, x_max, y_max)

Homework Guidelines

- **Step 1: Confirm the Model to Use**
  - Determine if the model is Transformer-based or Mamba-based.
  - Use resources like the official GitHub, PyTorch Lightning, or Transformers library.
- **Step 2: Adapt the Model for the Dataset**
  - Check if pre-trained weights are available and identify how to fine-tune the model (e.g., modifying the output layer).
  - Confirm the model's input and output coordinate formats (may need preprocess or post-process).
- **Step 3: Fine-tune the Model on the Dataset**
  - Set up the validation function and ensure model weights are saved.
  - Monitor the training process to ensure loss convergence.

Homework Guidelines

- **Step 4: Output Predictions for Validation and Test Sets**
  - Ensure the output JSON can be evaluated by eval.py for calculating the score.
  - Save files in the specified format: ***valid_<student-id>.json*** and ***test _<student-id>.json***

- **Step 5: Write the Report and Submit**
  - Prepare the report and submit all necessary files according to the provided guidelines.

Useful Resources

- **Train HuggingFace DETR on Custom Dataset**: Colab Notebook
- **Transformers Tutorials**: GitHub Repository
- **Mamba-YOLO**: GitHub Repository
- **DINO**: GitHub Repository

Any Question

cvpdl.ta.2024@gmail.com