Tutorial

Dr. Rebeka Sultana
Specially appointed assistant professor
Tokyo University of Agriculture and Technology
Email: sultana@go.tuat.ac.jp

Women in Science Japan

July 26th, 2025, Tokyo

Purpose

- Train an object detection model on Google Colab platform from Google Drive

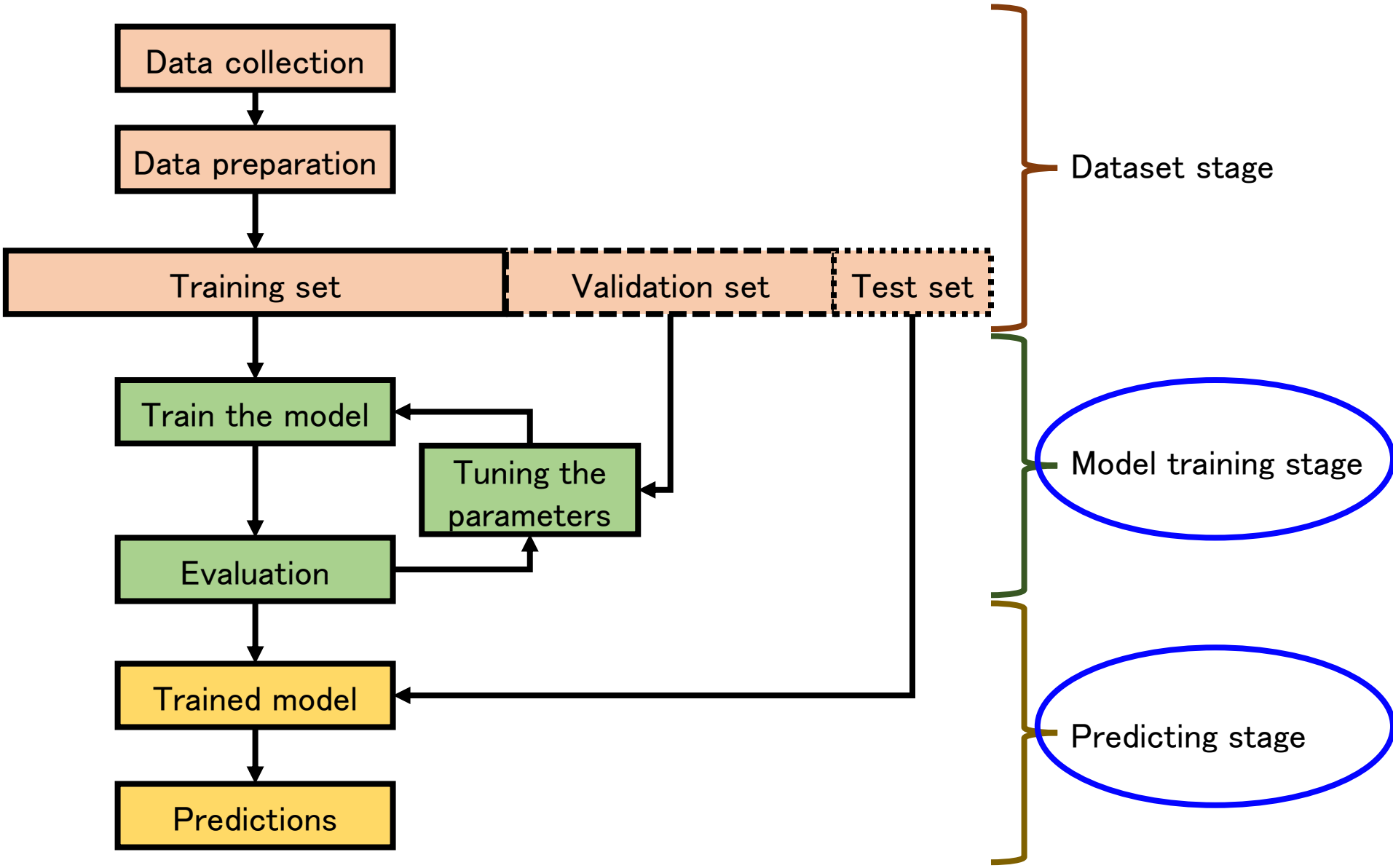
Google Drive

Store datasets and source code

Google Colab

Platform to train model using GPU

Training a deep learning model



Model training stage

- Environment settings
  - Google Drive: Programs and dataset
  - Google Colab: Training the model

Google Drive

Store datasets and source code

Google Colab

Platform to train model using GPU

- Example program and dataset for today's tutorial can be downloaded from following link:

https://drive.google.com/drive/folders/1Wl3fshsYq7VkpBYa3-jbrx3Uif9nB2HD?usp=sharing

Model training stage

❑Model: YOLO v8

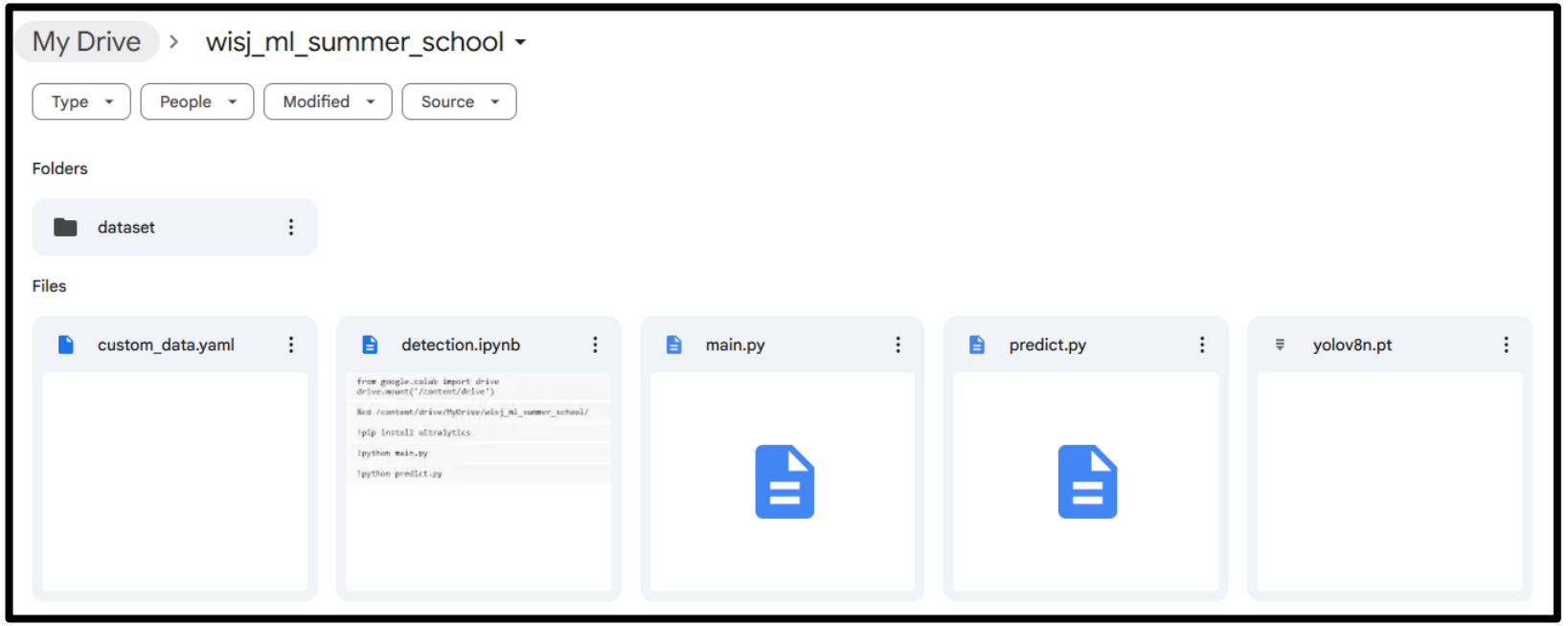
※

※https://yolov8.com/

Model training stage

❑Steps

Put the downloaded folder on your Google Drive

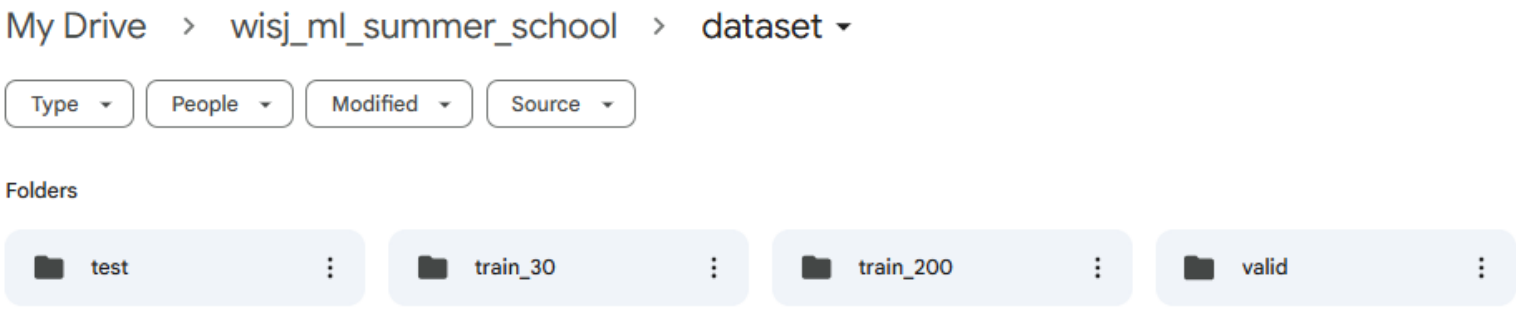


❑ Files:

1. Dataset
2. custom_data.yaml
3. main.py
4. predict.py
5. yolo8n.pt
6. detection.ipynb

Model training stage

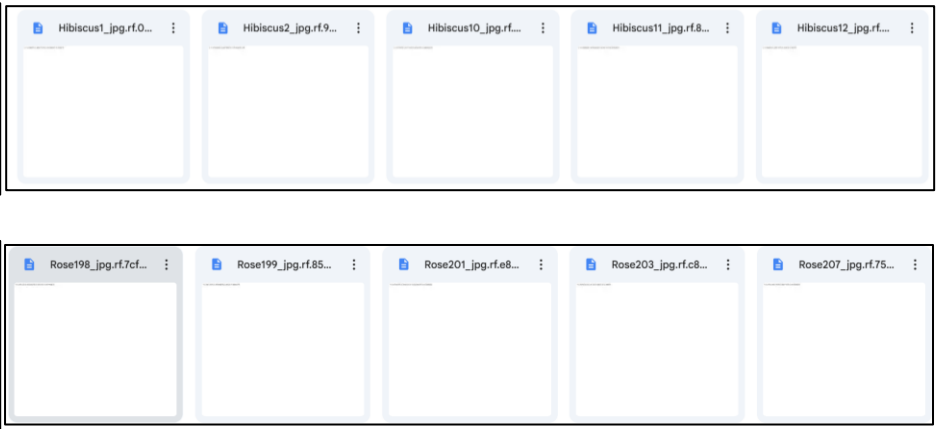
❑Dataset (select only 2 classes for easy understanding)



test: test images and corresponding labels
train_30: 30 train images and corresponding labels
train_200: 200 train images and corresponding labels
valid: validation images and corresponding labels

Images[※] :

Labels[※]:



[※] https://universe.roboflow.com/ananyas/flowers-

Model training stage

❑custom_data.yaml

custom_data.yaml ×

```
1 #train image path
2 train: /content/drive/MyDrive/wisj_ml_summer_school/dataset/train_200/images
3 #valid image path
4 val: /content/drive/MyDrive/wisj_ml_summer_school/dataset/valid/images
5
6 #number of class
7 nc: 2
8 #name of classes
9 names: ['Hibiscus', 'Rose']
```

❑yolo8n.pt

- Pre-trained weight for YOLO version 8 model

Model training stage

❑main.py

custom_data.yaml main.py

```python
from ultralytics import YOLO
from torchsummary import summary
import glob

# Load the model.
model = YOLO('yolov8n.pt')

#epoch
epoch = 500

# Training
results = model.train(
    data='custom_data.yaml',
    imgsz=640, #image size
    epochs=epoch, #epoch
    batch=4, #batch size
    name='epoch ' + str(epoch), #save name
    hsv_h= 0.0, #Hue shift range (color change); Typical Value: 0.015 - 0.1
    hsv_s= 0.0, #Saturation variation; Typical Value: 0.5 - 0.7
    hsv_v= 0.0, #Value (brightness) variation; Typical Value: 0.3 - 0.7
    degrees= 0, #Image rotation (± degrees); Typical Value: 0 - 10
    translate= 0.0, #Random shift along x/y axis;  Typical Value: 0.1 - 0.2
    scale= 0.0, #Resize scale jitter ; Typical Value: 0.5 - 1.5
    shear= 0.0, #Shear transform (tilt); Typical Value: 0 - 2.0
    perspective= 0.0, #Perspective distortion  ; Typical Value: 0 - 0.001
    flipud= 0.0, #Vertical flip probability; Typical Value: 0.0 - 0.5
    fliplr= 0.0, #Horizontal flip probability; Typical Value: 0.0 - 0.5
    mosaic= 0.0, #Enable Mosaic augmentation;  Typical Value: 0.5 - 1.0
    mixup= 0.0, #Enable MixUp (image blending);  Typical Value: 0.0 - 0.5
    copy_paste= 0.0 #Copy-paste objects from other images; Typical Value: 0.0 - 0.5
)
```

Import packages

Load initial weights

How many times train

Model training

Augmentation parameters

Get image paths

```python
# get image files
image_files = glob.glob("datasets/valid/images/" + "*")

# Load the best weight of the model
model = YOLO('./runs/detect/epoch_' + str(epoch) + '/weights/best.pt')

#predict the detection results of valid images
for i in range(len(image_files)):
    model.predict(image_files[i], save=True)

## Evaluate the model's performance on the validation set
results = model.val()
print(results)
```

Load best model

Prediction and save results

Evaluation score

Predicting stage

❑predict.py

predict.py

```python
from ultralytics import YOLO
import glob

# get image files
image_files = glob.glob("dataset/test/images/" + "*")

# Load the model.
model = YOLO('./runs/detect/epoch_500/weights/best.pt')

#predict and save the results
for i in range(len(image_files)):
    model.predict(image_files[i], save=True)
```

Import packages

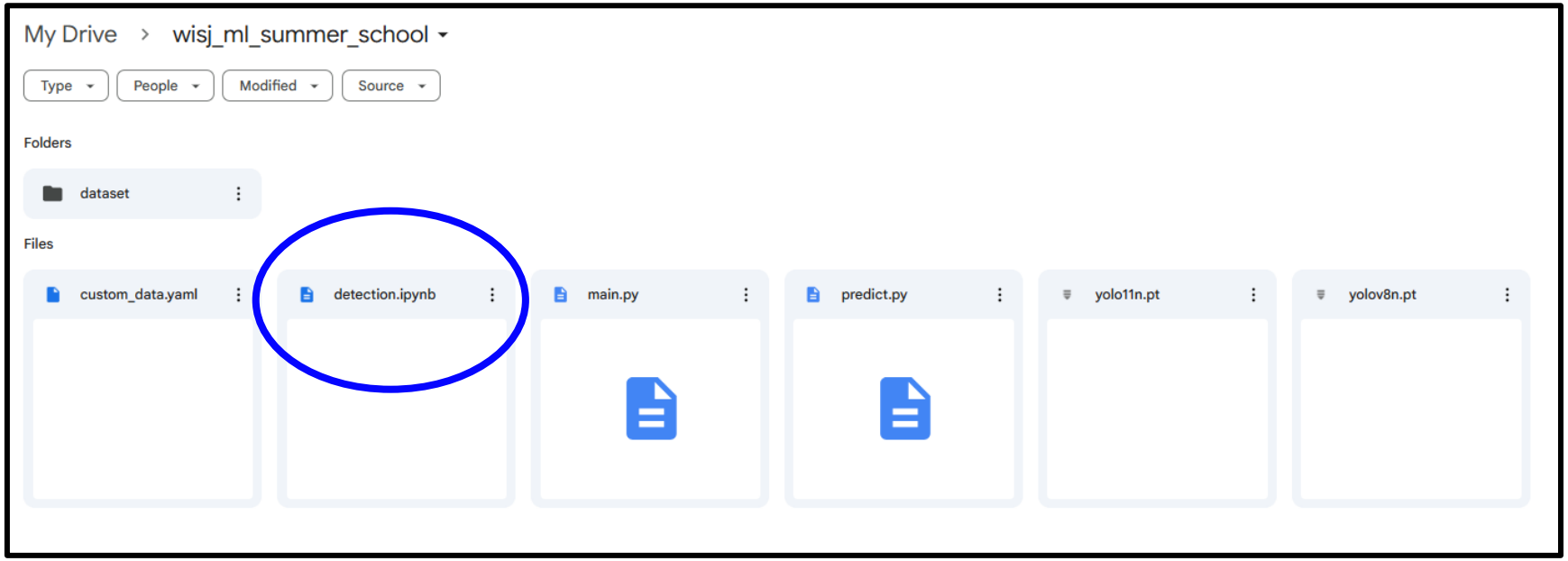
Get image paths

Load best model

Prediction and save results

Training steps

❑Open detection.ipynb from Google Drive folder

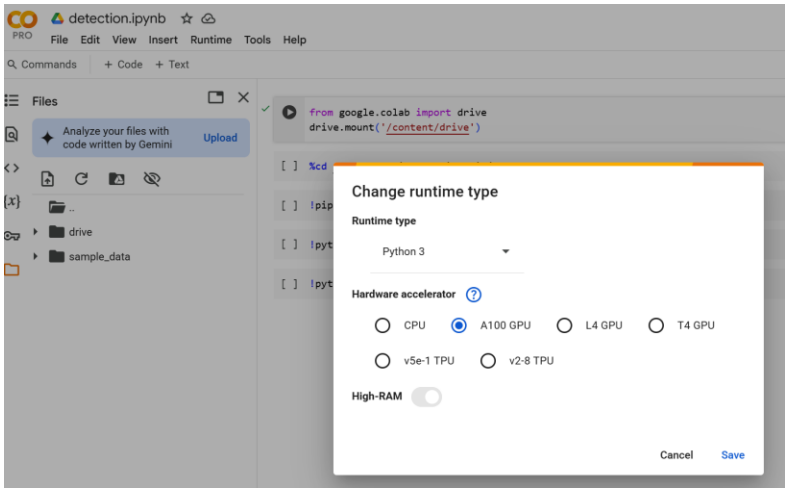


Training steps (Google Colab platform)

❑detection.ipynb

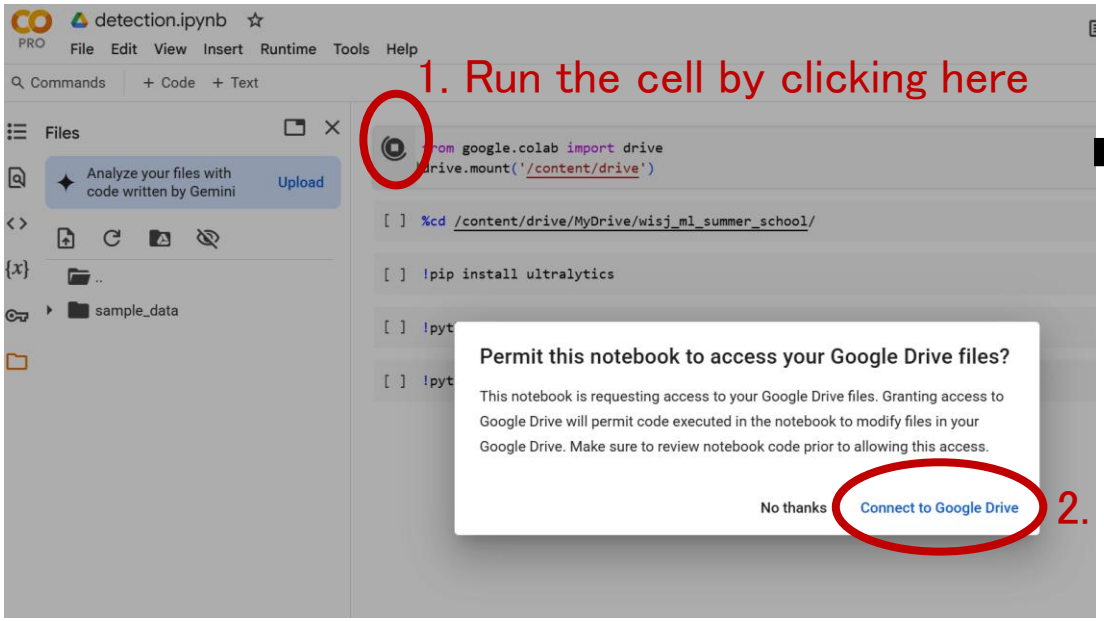
- Change the runtime (CPU/ GPU)

2. A pop-up window will appear and select the hardware accelerator. Then, click save.



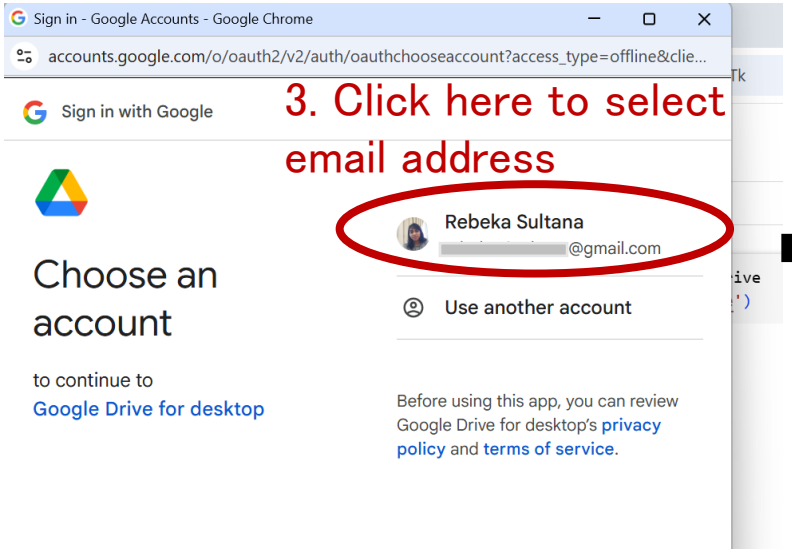
Training steps (Google Colab platform)

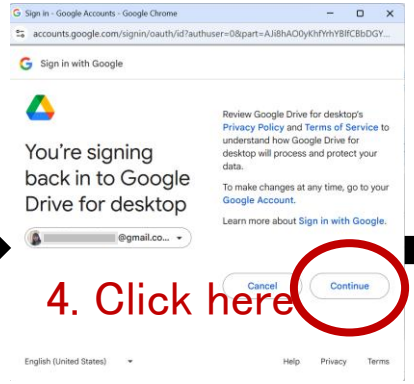
- detection.ipynb
  - Connect to Google Drive

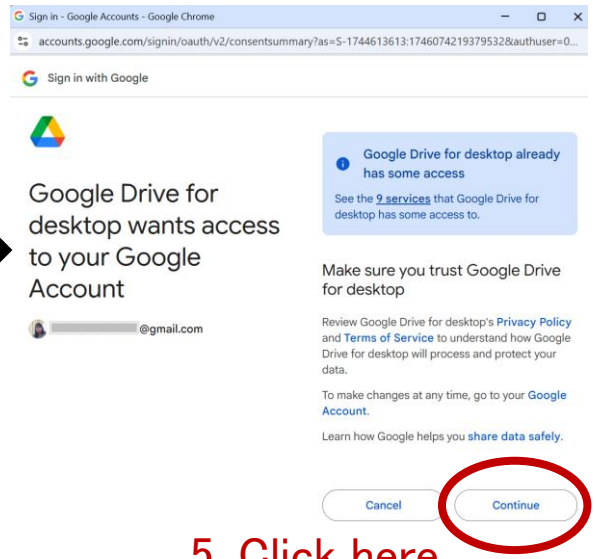


Training steps (Google Colab platform)

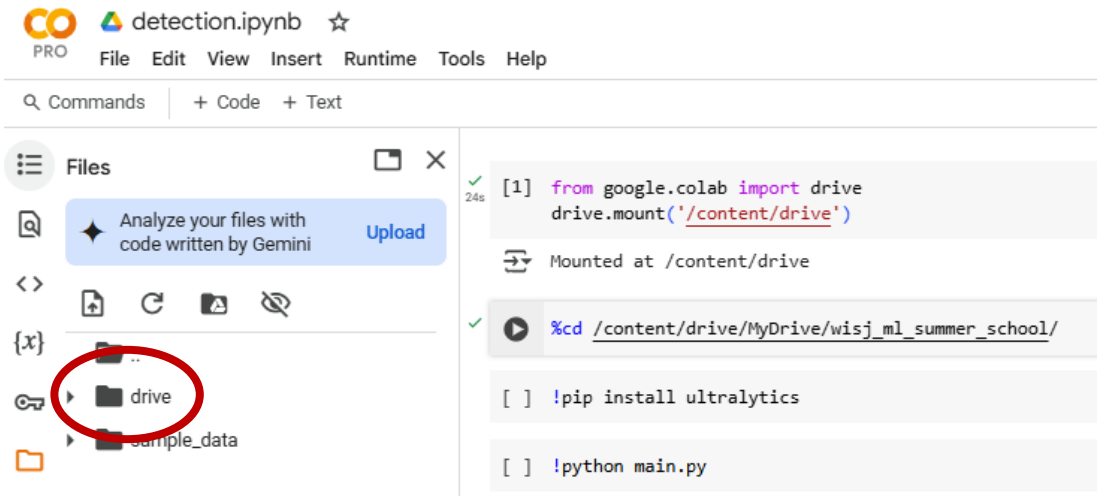
- detection.ipynb
  - Connect to Google Drive





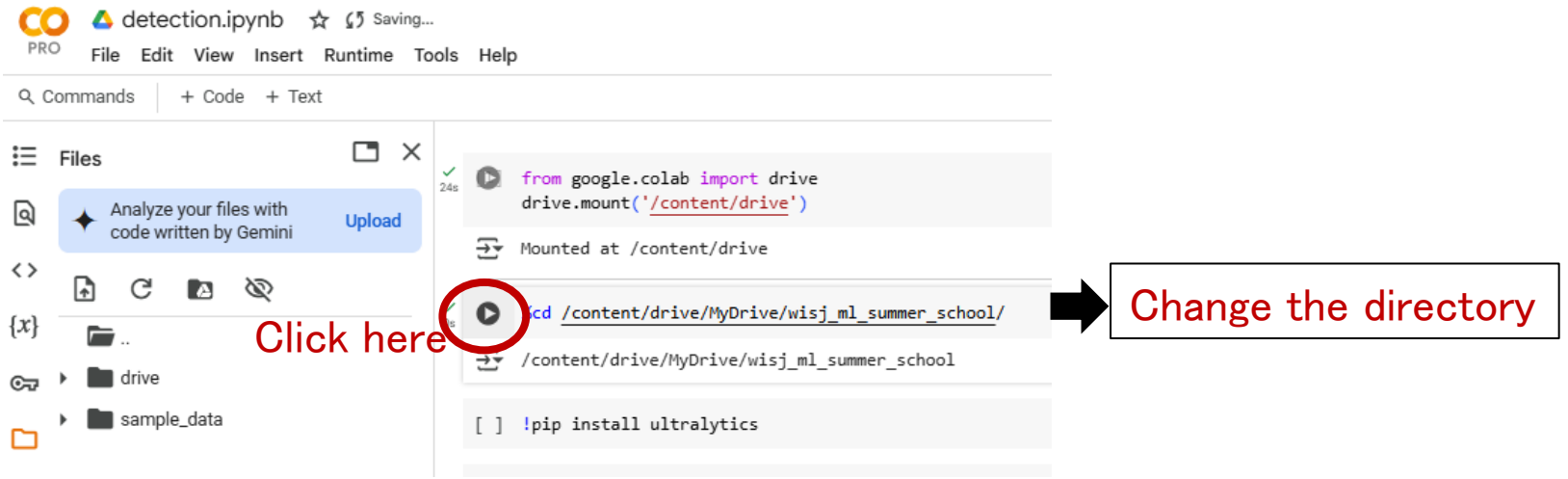


- Google drive will be mounted on google colab



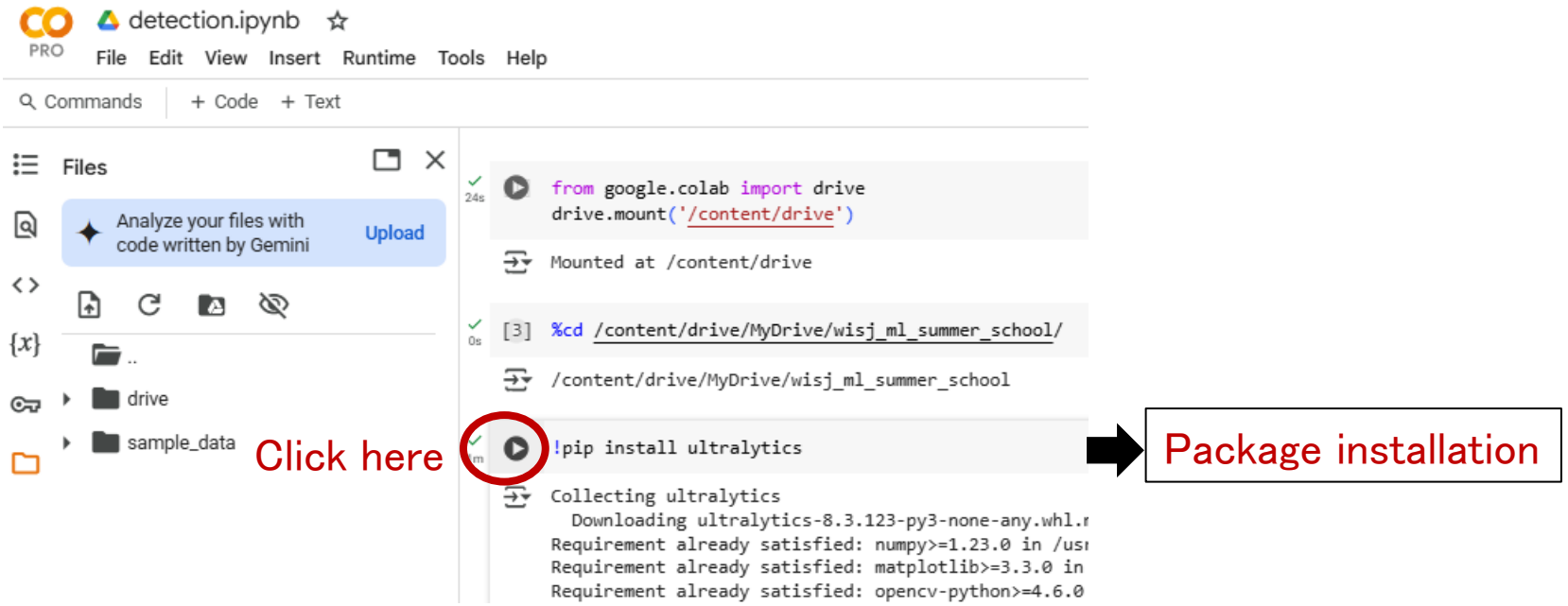
Training steps (Google Colab platform)

- detection.ipynb
  - Change directory



Training steps (Google Colab platform)

- detection.ipynb
  - Install ultralytics package



Training steps (Google Colab platform)

- detection.ipynb
  - Run file to train YOLO model

Click here

```
python main.py
```

Training

```
Creating new Ultralytics Settings v0.0.6 file
View Ultralytics Settings with 'yolo settings' or at '/root/.config/Ultralytics/settings.json'
Update Settings with 'yolo settings key=value', i.e. 'yolo settings runs_dir=path/to/dir'. For help see https://docs.ultralytics.com/quickstart/#ultralyt
Ultralytics 8.3.122 🚀 Python-3.11.12 torch-2.6.0+cu124 CUDA:0 (NVIDIA A100-SXM4-40GB, 40507MiB)
engine/trainer: task=detect, mode=train, model=yolov8n.pt, data=custom_data.yaml, epochs=500, time=None, patience=100, batch=4, imgsz=640, save=True, sav
Downloading https://ultralytics.com/assets/Arial.ttf to '/root/.config/Ultralytics/Arial.ttf'...
100% 755k/755k [00:00<00:00, 15.3MB/s]
Overriding model.yaml nc=80 with nc=2

                   from  n    params  module                                       arguments
  0                  -1  1       464  ultralytics.nn.modules.conv.Conv             [3, 16, 3, 2]
  1                  -1  1      4672  ultralytics.nn.modules.conv.Conv             [16, 32, 3, 2]
```

Start training…

```
Using 8 dataloader workers
Logging results to runs/detect/epoch_500
Starting training for 500 epochs...

      Epoch    GPU_mem   box_loss   cls_loss   dfl_loss  Instances       Size
      1/500     0.598G      0.87      3.078      1.386          3        640: 100% 50/50 [00:04<00:00, 12.23it/s]
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100% 5/5 [00:00<00:00, 12.21it/s]
                   all         40         40      0.835       0.15      0.471      0.325

      Epoch    GPU_mem   box_loss   cls_loss   dfl_loss  Instances       Size
      2/500     0.711G     0.6604      2.107      1.168          3        640: 100% 50/50 [00:03<00:00, 15.45it/s]
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100% 5/5 [00:00<00:00, 15.58it/s]
                   all         40         40      0.771      0.727      0.851      0.628

      Epoch    GPU_mem   box_loss   cls_loss   dfl_loss  Instances       Size
      3/500     0.711G     0.6705      1.888      1.118          3        640: 100% 50/50 [00:03<00:00, 15.57it/s]
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100% 5/5 [00:00<00:00, 17.60it/s]
                   all         40         40       0.56      0.681      0.612       0.38
```

Finish training…

```
       [          1,          1,          1, ...,          0,          0,          0]]), 'Confidence', 'Recall']]
fitness: np.float64(0.9499767833861056)
keys: ['metrics/precision(B)', 'metrics/recall(B)', 'metrics/mAP50(B)', 'metrics/mAP50-95(B)']
maps: array([    0.92863,     0.96132])
names: {0: 'Hibiscus', 1: 'Rose'}
plot: True
results_dict: {'metrics/precision(B)': np.float64(0.9718476659098572), 'metrics/recall(B)': np.float64(1.0), 'metrics/mAP50(B)': np.float64(0.995), 'metrics/mAP50-95(B)': np.float64(0.9449742037623394), '
save_dir: PosixPath('runs/detect/val')
speed: {'preprocess': 3.245090900009018, 'inference': 8.27100954998059, 'loss': 0.0007450250222973409, 'postprocess': 5.267873400021017}
task: 'detect'
```

Training steps (Google Colab platform)

- detection.ipynb
  - Run file to detect objects in test images by trained YOLO model

Click here

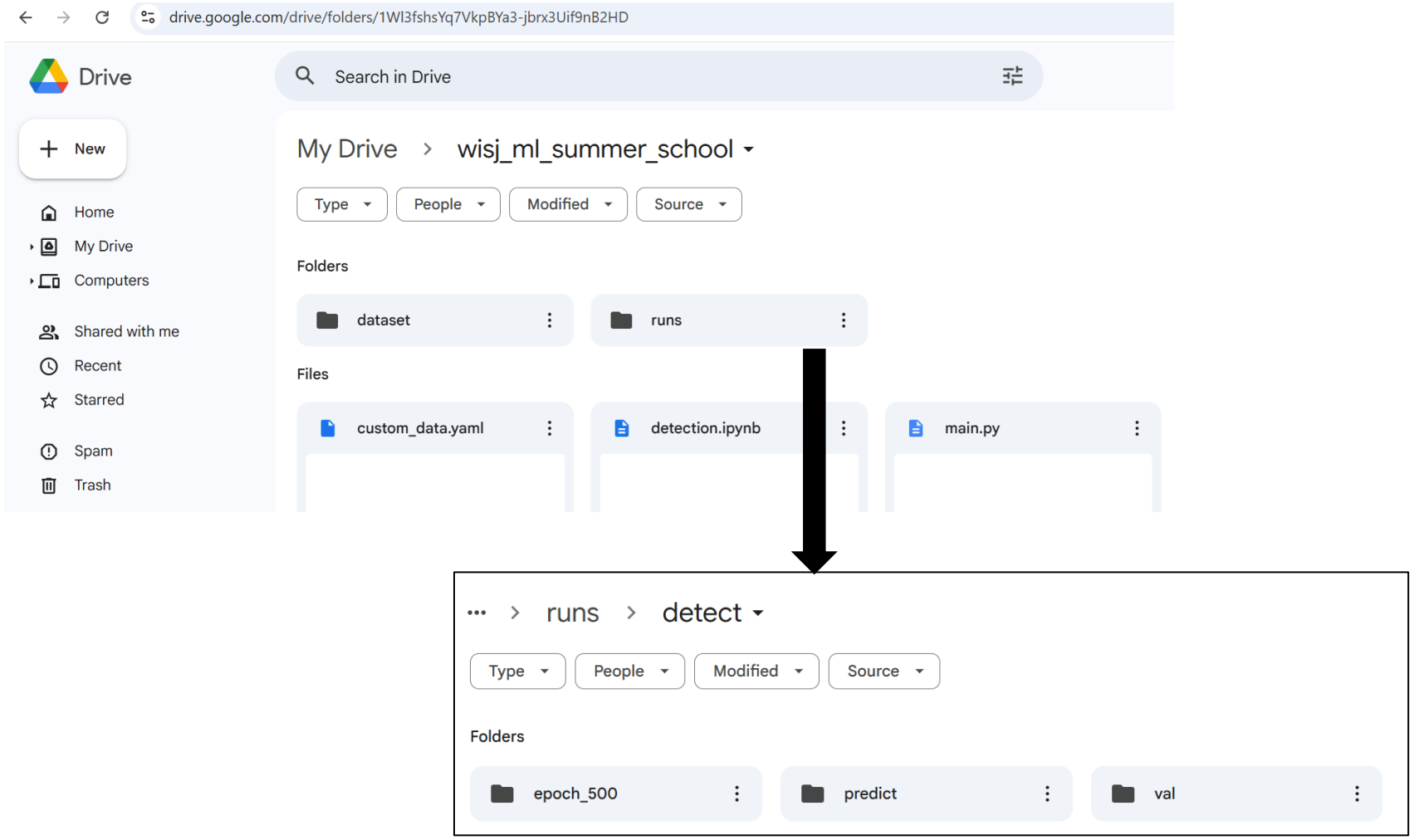
```
!python predict.py
```

Prediction

```
Results saved to runs/detect/predict

image 1/1 /content/drive/MyDrive/wisj_ml_summer_school/dataset/test/images/Rose146_jpg.rf.c8e59acbe63e9e23acb72f77402f3a2e
Speed: 2.1ms preprocess, 8.3ms inference, 2.0ms postprocess per image at shape (1, 3, 640, 640)
Results saved to runs/detect/predict

image 1/1 /content/drive/MyDrive/wisj_ml_summer_school/dataset/test/images/Hibiscus100_jpg.rf.b0ffe5e41f4e6db9387e3dd468bb
Speed: 2.2ms preprocess, 8.3ms inference, 1.8ms postprocess per image at shape (1, 3, 640, 640)
Results saved to runs/detect/predict

image 1/1 /content/drive/MyDrive/wisj_ml_summer_school/dataset/test/images/Rose149_jpg.rf.51b765d2eb7251b328cbc3f74388d2fb
Speed: 2.1ms preprocess, 7.7ms inference, 1.7ms postprocess per image at shape (1, 3, 640, 640)
Results saved to runs/detect/predict

image 1/1 /content/drive/MyDrive/wisj_ml_summer_school/dataset/test/images/Hibiscus199_jpg.rf.cb08fc8df874f8cb6b70fbd0cc6a
Speed: 2.1ms preprocess, 7.8ms inference, 1.7ms postprocess per image at shape (1, 3, 640, 640)
Results saved to runs/detect/predict

image 1/1 /content/drive/MyDrive/wisj_ml_summer_school/dataset/test/images/Hibiscus113_jpg.rf.cf2a6173dffb6ea05da2d23dac79
Speed: 2.1ms preprocess, 7.6ms inference, 1.6ms postprocess per image at shape (1, 3, 640, 640)
Results saved to runs/detect/predict

image 1/1 /content/drive/MyDrive/wisj_ml_summer_school/dataset/test/images/Rose13_jpg.rf.4cafc40ab8a154ac18d699552eccff2f.
Speed: 2.1ms preprocess, 7.8ms inference, 0.8ms postprocess per image at shape (1, 3, 640, 640)
Results saved to runs/detect/predict
```

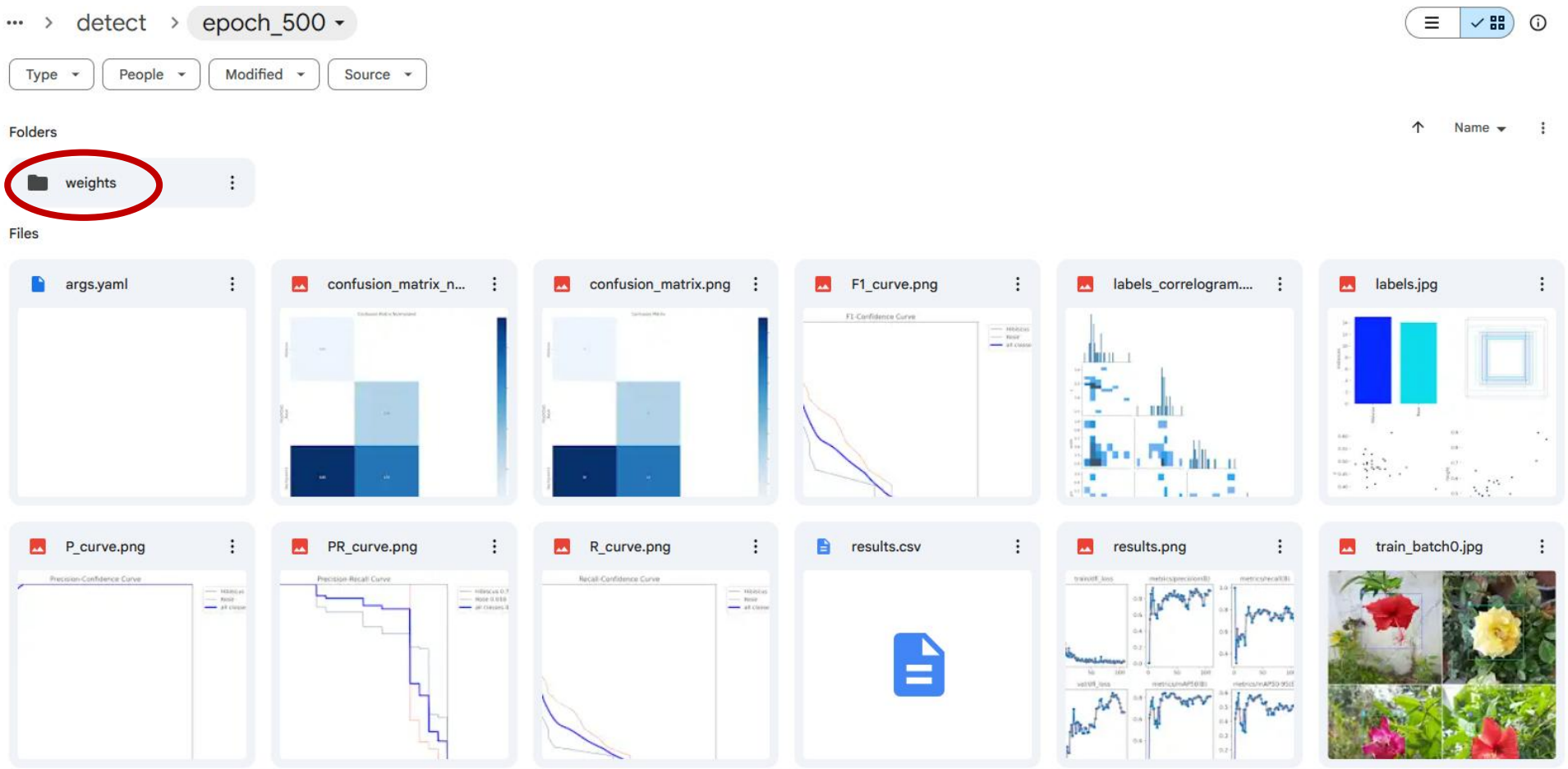
Files on Google Drive

❑Files will be created on Google Drive in runs folder



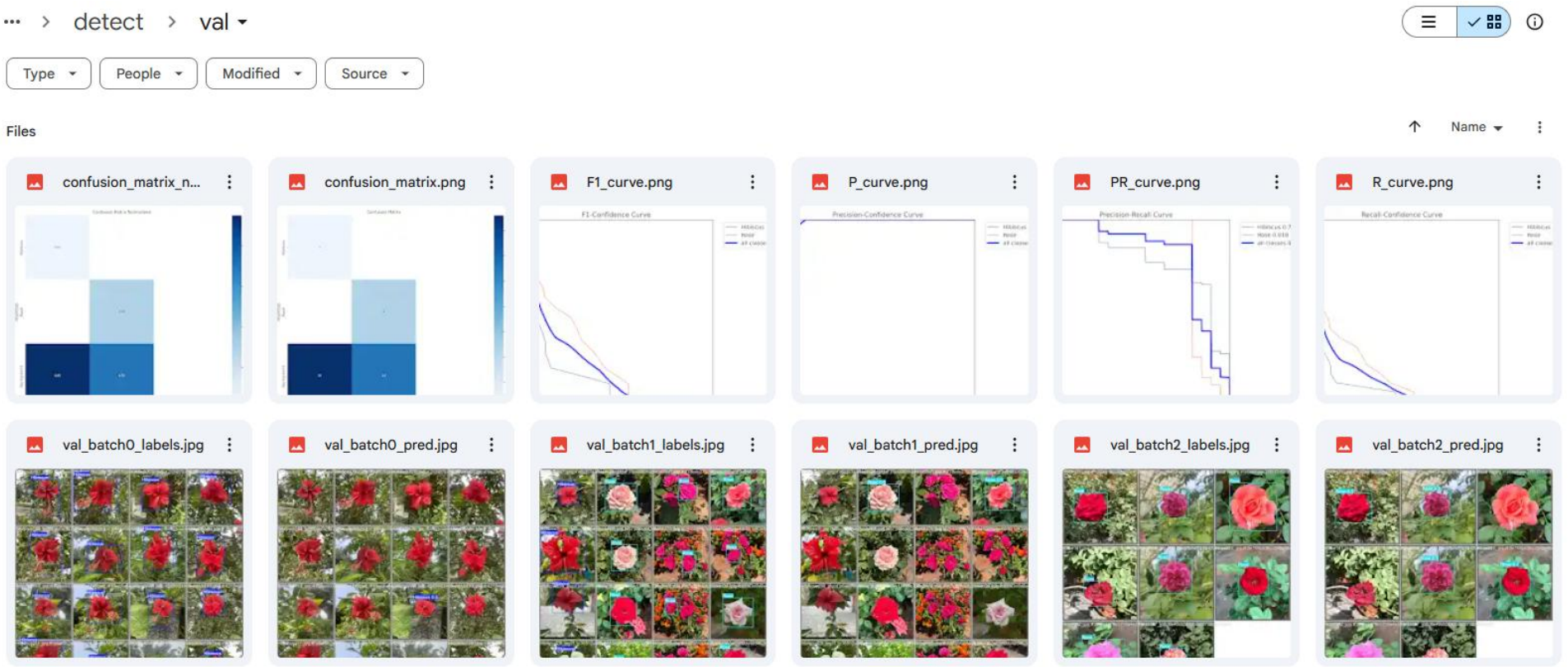
Files on Google Drive

❑Files will be created on Google Drive in runs folder



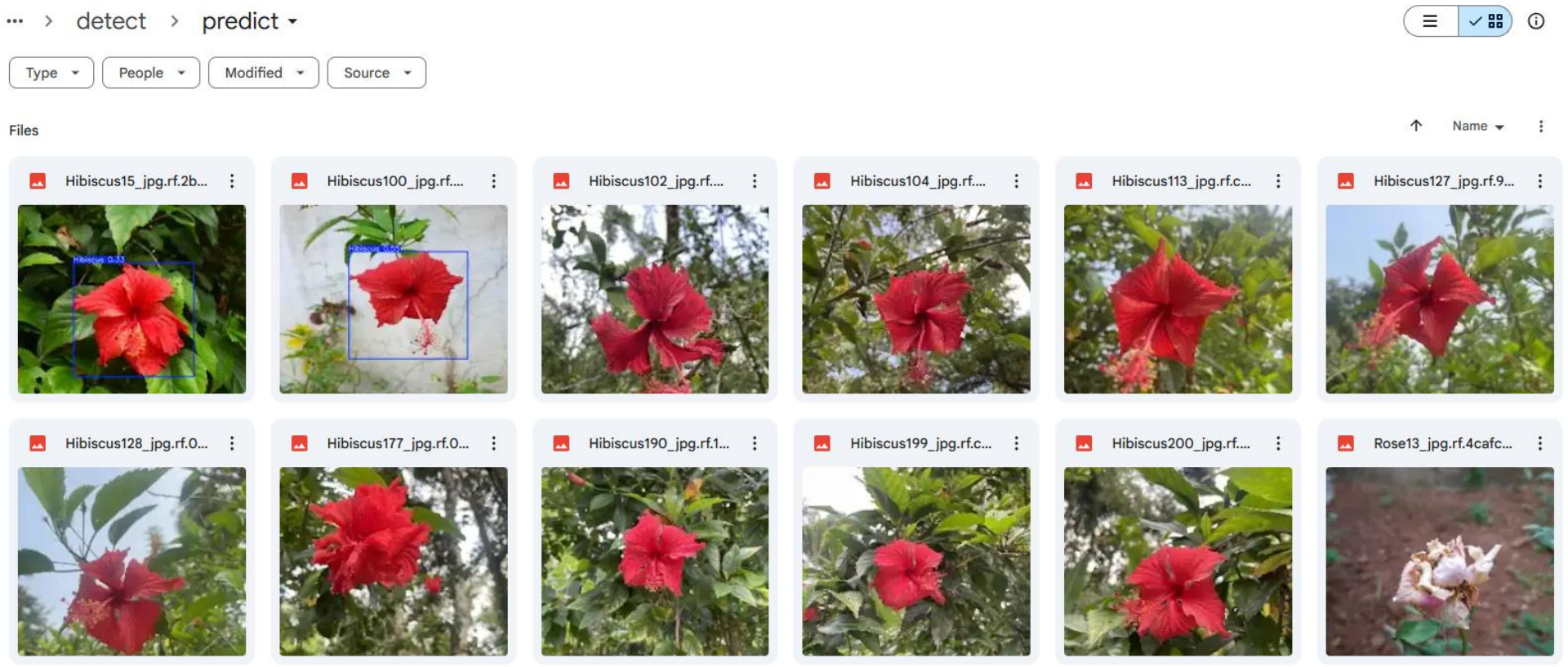
Files on Google Drive

❑Files will be created on Google Drive in runs folder



Files on Google Drive

❑Files will be created on Google Drive in runs folder



Train object detection model

Google Drive

Store datasets and source code

Google Colab

Platform to train model using GPU

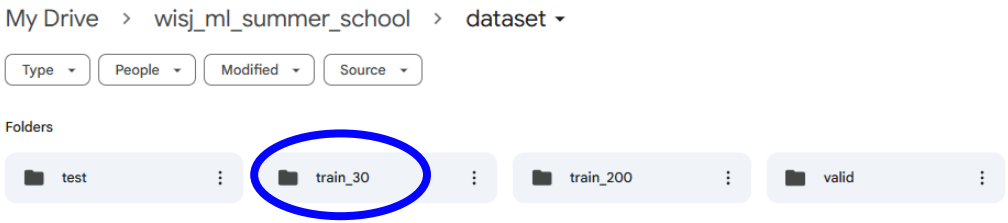
Exercise

❑Purpose

- How augmentation and increasing training dataset improve the detection performance

❑Problem 1:

Train YOLO model by changing the augmentation parameter

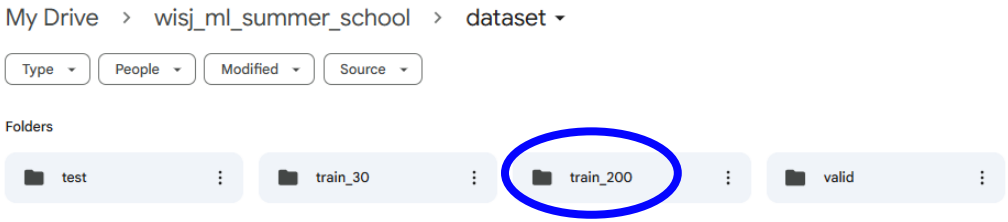


※Parameters value

hsv_h:	0.015
hsv_s:	0.7
hsv_v:	0.4
translate:	0.1
fliplr:	0.5
mosaic:	1.0

❑Problem 2:

- Train YOLO model with original 200 images without augmentation parameter



❑Compare prediction results

- ❑Model: 30 training images + no augmentation (results are already on google drive)
- ❑Mode: 30 training images + with augmentation
- ❑Model: 200 training images + no augmentation

Solution

❑Google Drive link: Files necessary to train the model

https://drive.google.com/drive/folders/1hR4wWiNUTqbW7P7ktMTZA9wWfDC-Zgee?usp=sharing

Note: If you cannot train the model, the results of problem 1 and problem 2 are also uploaded on google drive