ASSIGNMENT

Name: Niharika Amritkar

Contact: a.niharika146@gmail.com

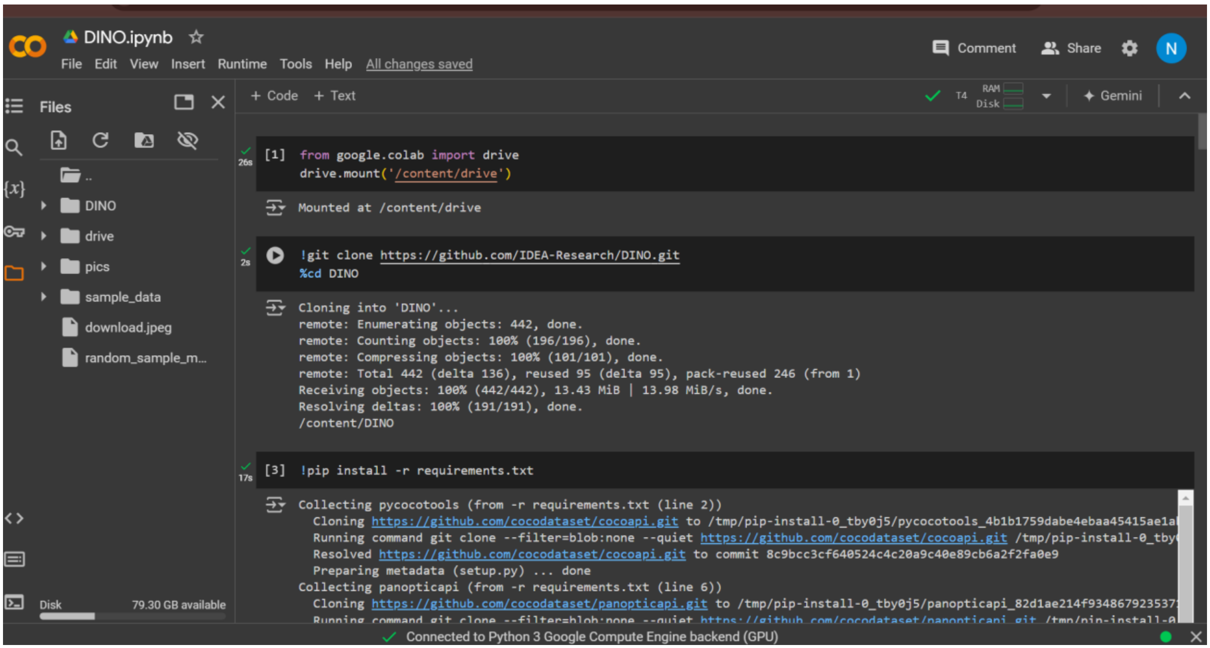
Github Repository link: NiharikaAmritkar/DINO-4scale-model (github.com)

PROBLEM STATEMENT: DINO object detection using pre-trained DINO-4scale model with the ResNet-50 (R50) backbone

METHODOLOGY:

1. Cloning the Github Repository given with the Assignment
2. Running the Inference model given in the repository
3. Downloading the given data

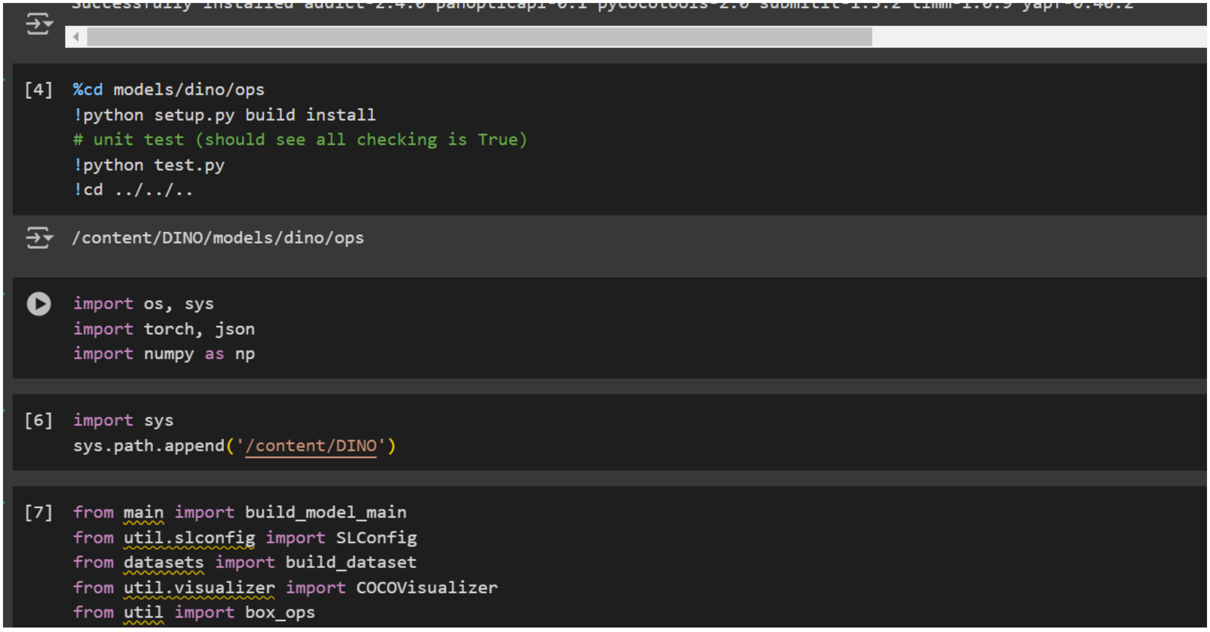
RESULTS:



```
[1] from google.colab import drive
    drive.mount('/content/drive')

Mounted at /content/drive
```

```
!git clone https://github.com/IDEA-Research/DINO.git
%cd DINO

Cloning into 'DINO'...
remote: Enumerating objects: 442, done.
remote: Counting objects: 100% (196/196), done.
remote: Compressing objects: 100% (101/101), done.
remote: Total 442 (delta 136), reused 95 (delta 95), pack-reused 246 (from 1)
Receiving objects: 100% (442/442), 13.43 MiB | 13.98 MiB/s, done.
Resolving deltas: 100% (191/191), done.
/content/DINO
```

```
[3] !pip install -r requirements.txt

Collecting pycocotools (from -r requirements.txt (line 2))
  Cloning https://github.com/cocodataset/cocoapi.git to /tmp/pip-install-0_tby0j5/pycocotools_4b1b1759dabe4ebaa45415ae1a
  Running command git clone --filter=blob:none --quiet https://github.com/cocodataset/cocoapi.git /tmp/pip-install-0_tby
  Resolved https://github.com/cocodataset/cocoapi.git to commit 8c9bcc3cf640524c4c20a9c40e89cb6a2f2fa0e9
  Preparing metadata (setup.py) ... done
Collecting panopticapi (from -r requirements.txt (line 6))
  Cloning https://github.com/cocodataset/panopticapi.git to /tmp/pip-install-0_tby0j5/panopticapi_82d1ae214f934867923537
  Running command git clone --filter=blob:none --quiet https://github.com/cocodataset/panopticapi.git /tmp/pip-install-0
```



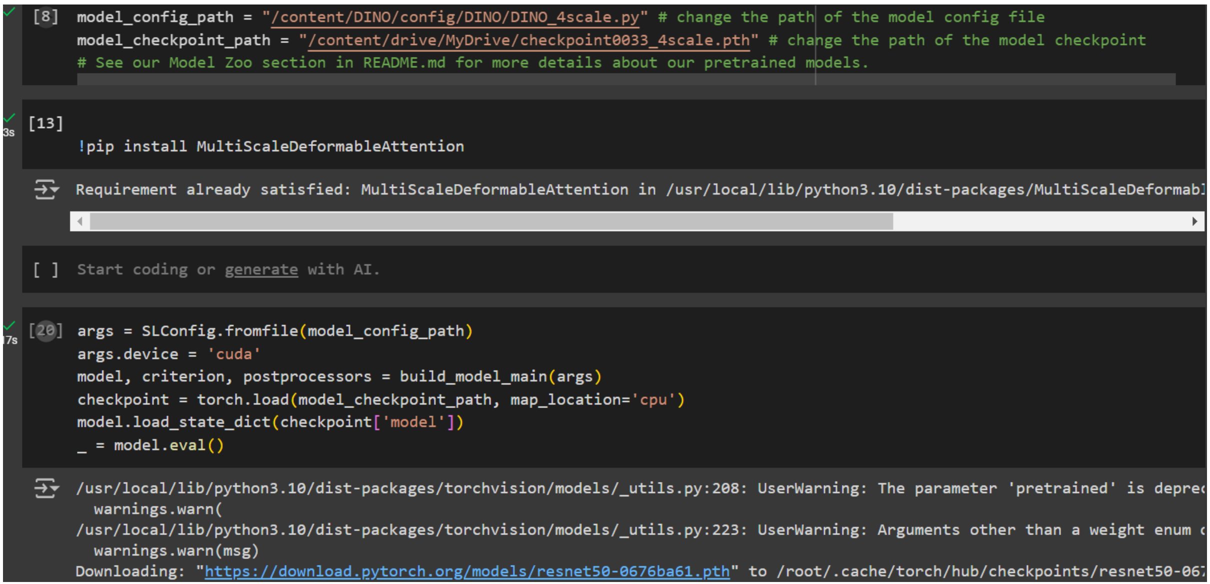
```
Successfully installed addict-2.4.0 panopticapi-0.1 pycocotools-2.0 submitit-1.5.2 timm-1.0.9 yapf-0.40.2
```

```
[4] %cd models/dino/ops
    !python setup.py build install
    # unit test (should see all checking is True)
    !python test.py
    !cd ../../..

/content/DINO/models/dino/ops
```

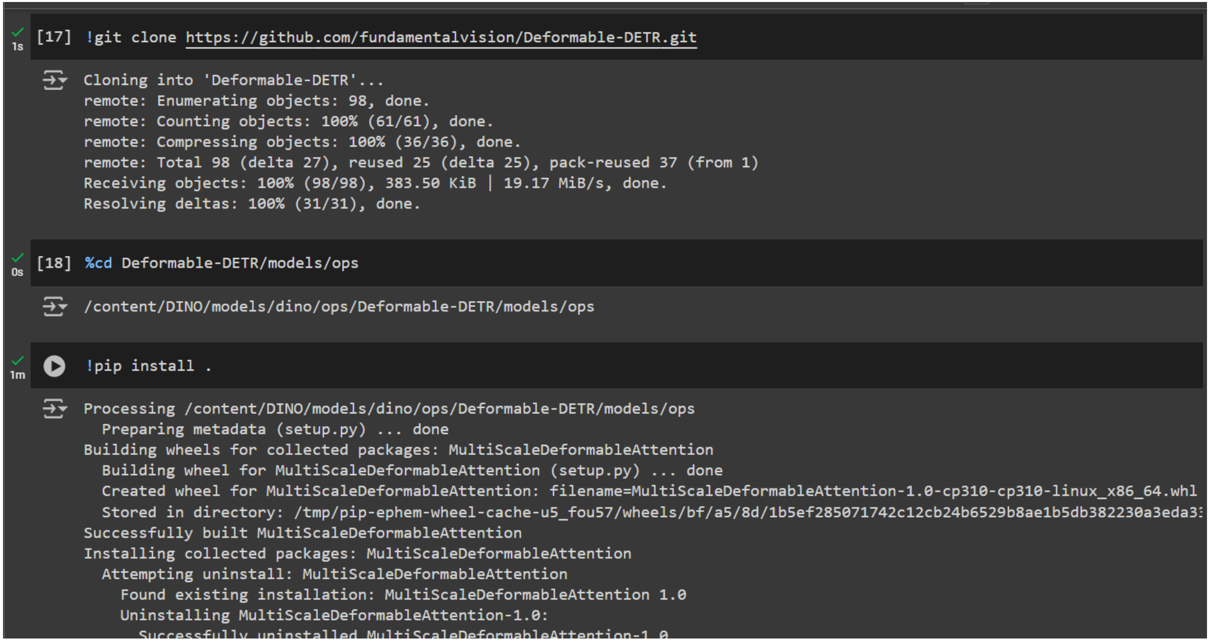
```
import os, sys
import torch, json
import numpy as np
```

```
[6] import sys
    sys.path.append('/content/DINO')
```

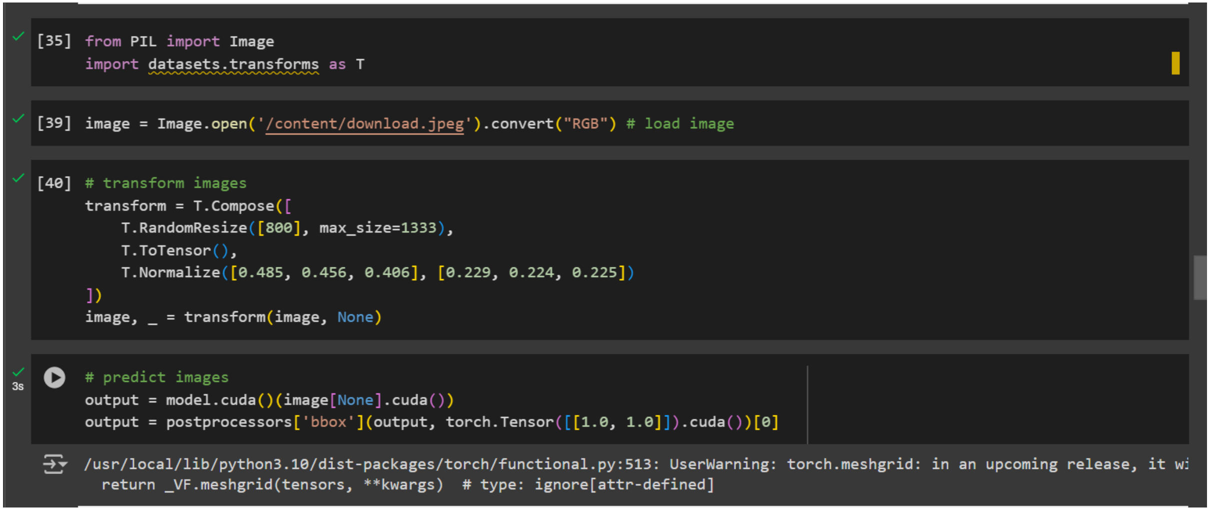
```
[7] from main import build_model_main
    from util.slconfig import SLConfig
    from datasets import build_dataset
    from util.visualizer import COCOVisualizer
    from util import box_ops
```



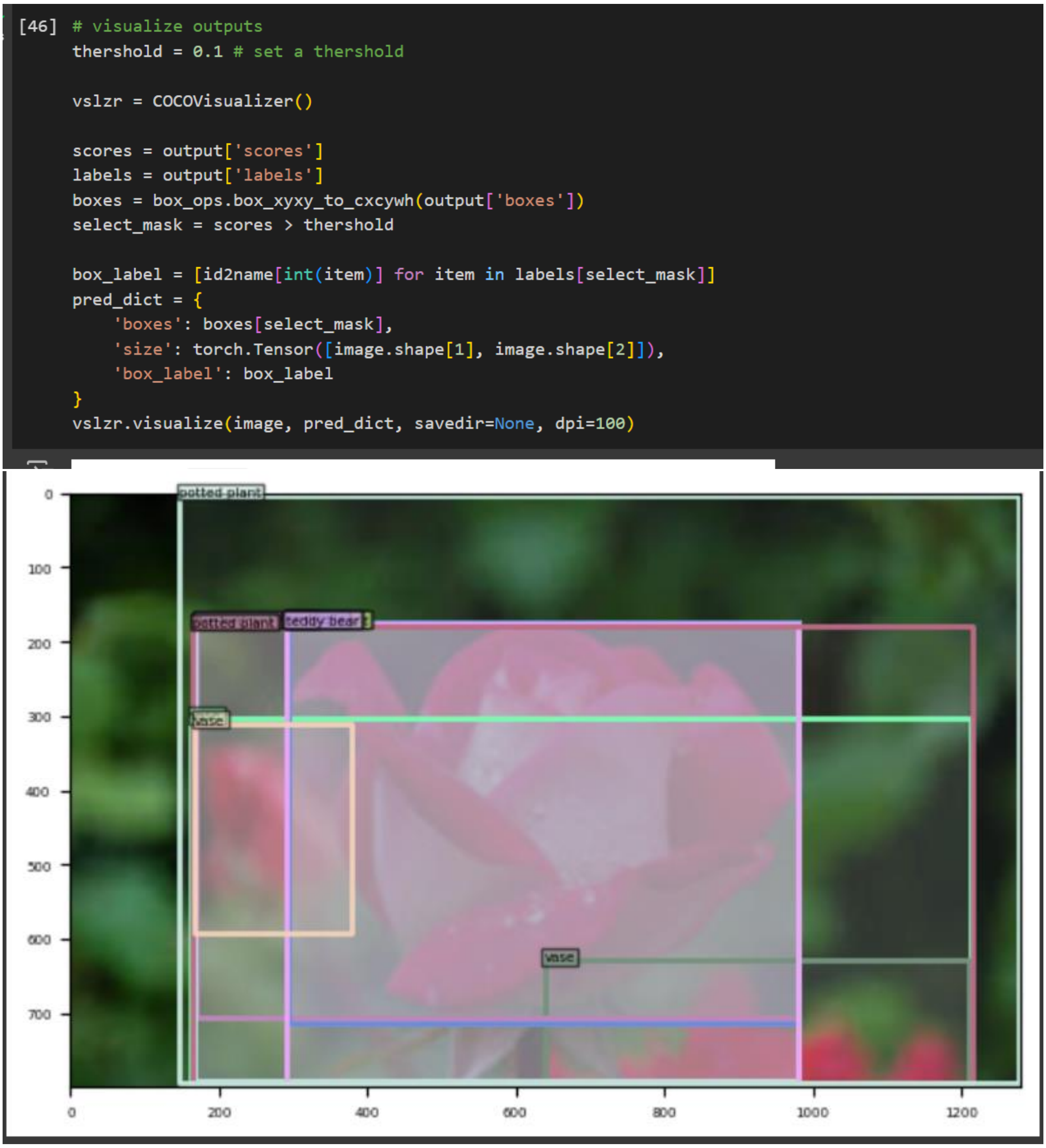
```
[8] model_config_path = "/content/DINO/config/DINO/DINO_4scale.py" # change the path of the model config file
    model_checkpoint_path = "/content/drive/MyDrive/checkpoint0033_4scale.pth" # change the path of the model checkpoint
    # See our Model Zoo section in README.md for more details about our pretrained models.

3s [13]
    !pip install MultiScaleDeformableAttention

    Requirement already satisfied: MultiScaleDeformableAttention in /usr/local/lib/python3.10/dist-packages/MultiScaleDeformabl

[ ] Start coding or generate with AI.

7s [20] args = SLConfig.fromfile(model_config_path)
    args.device = 'cuda'
    model, criterion, postprocessors = build_model_main(args)
    checkpoint = torch.load(model_checkpoint_path, map_location='cpu')
    model.load_state_dict(checkpoint['model'])
    _ = model.eval()

    /usr/local/lib/python3.10/dist-packages/torchvision/models/_utils.py:208: UserWarning: The parameter 'pretrained' is depre
      warnings.warn(
    /usr/local/lib/python3.10/dist-packages/torchvision/models/_utils.py:223: UserWarning: Arguments other than a weight enum o
      warnings.warn(msg)
    Downloading: "https://download.pytorch.org/models/resnet50-0676ba61.pth" to /root/.cache/torch/hub/checkpoints/resnet50-067
```



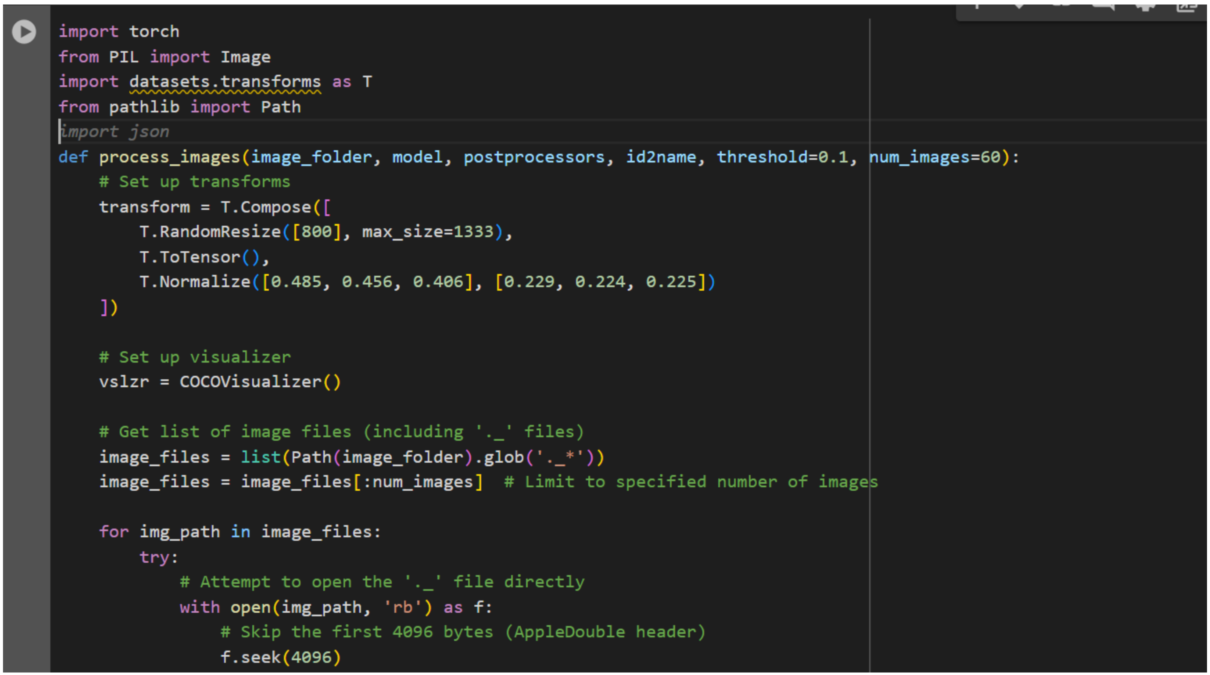
```
1s [17] !git clone https://github.com/fundamentalvision/Deformable-DETR.git

    Cloning into 'Deformable-DETR'...
    remote: Enumerating objects: 98, done.
    remote: Counting objects: 100% (61/61), done.
    remote: Compressing objects: 100% (36/36), done.
    remote: Total 98 (delta 27), reused 25 (delta 25), pack-reused 37 (from 1)
    Receiving objects: 100% (98/98), 383.50 KiB | 19.17 MiB/s, done.
    Resolving deltas: 100% (31/31), done.

0s [18] %cd Deformable-DETR/models/ops

    /content/DINO/models/dino/ops/Deformable-DETR/models/ops

1m  !pip install .

    Processing /content/DINO/models/dino/ops/Deformable-DETR/models/ops
      Preparing metadata (setup.py) ... done
    Building wheels for collected packages: MultiScaleDeformableAttention
      Building wheel for MultiScaleDeformableAttention (setup.py) ... done
      Created wheel for MultiScaleDeformableAttention: filename=MultiScaleDeformableAttention-1.0-cp310-cp310-linux_x86_64.whl
      Stored in directory: /tmp/pip-ephem-wheel-cache-u5_fou57/wheels/bf/a5/8d/1b5ef285071742c12cb24b6529b8ae1b5db382230a3eda3:
    Successfully built MultiScaleDeformableAttention
    Installing collected packages: MultiScaleDeformableAttention
      Attempting uninstall: MultiScaleDeformableAttention
        Found existing installation: MultiScaleDeformableAttention 1.0
        Uninstalling MultiScaleDeformableAttention-1.0:
          Successfully uninstalled MultiScaleDeformableAttention-1.0
```



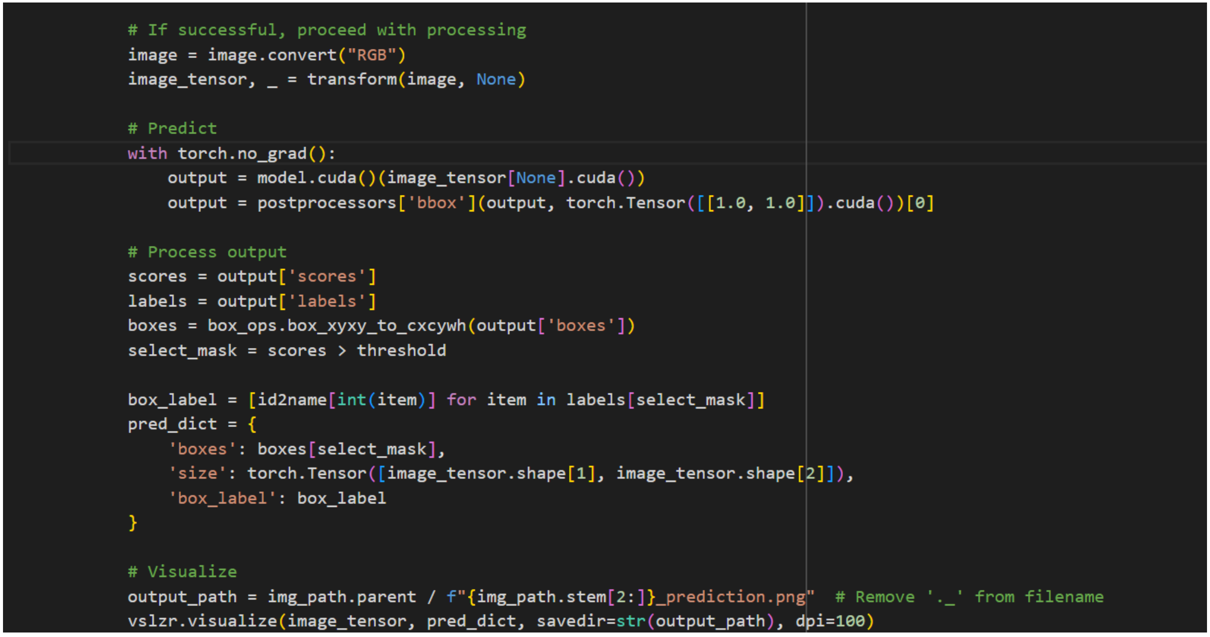
```
[35] from PIL import Image
     import datasets.transforms as T

[39] image = Image.open('/content/download.jpeg').convert("RGB") # load image

[40] # transform images
     transform = T.Compose([
         T.RandomResize([800], max_size=1333),
         T.ToTensor(),
         T.Normalize([0.485, 0.456, 0.406], [0.229, 0.224, 0.225])
     ])
     image, _ = transform(image, None)

3s   # predict images
     output = model.cuda()(image[None].cuda())
     output = postprocessors['bbox'](output, torch.Tensor([[1.0, 1.0]]).cuda())[0]

     /usr/local/lib/python3.10/dist-packages/torch/functional.py:513: UserWarning: torch.meshgrid: in an upcoming release, it wi
       return _VF.meshgrid(tensors, **kwargs)  # type: ignore[attr-defined]
```



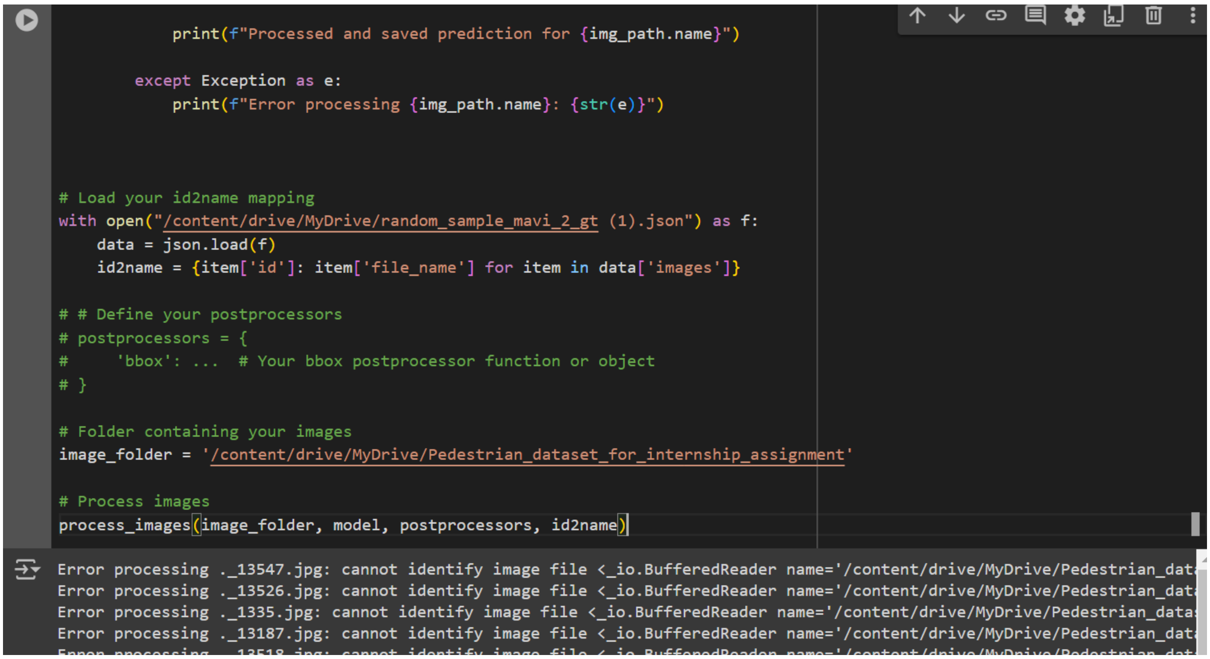
```
[46] # visualize outputs
     thershold = 0.1 # set a thershold

     vslzr = COCOVisualizer()

     scores = output['scores']
     labels = output['labels']
     boxes = box_ops.box_xyxy_to_cxcywh(output['boxes'])
     select_mask = scores > thershold

     box_label = [id2name[int(item)] for item in labels[select_mask]]
     pred_dict = {
         'boxes': boxes[select_mask],
         'size': torch.Tensor([image.shape[1], image.shape[2]]),
         'box_label': box_label
     }
     vslzr.visualize(image, pred_dict, savedir=None, dpi=100)
```



```python
import torch
from PIL import Image
import datasets.transforms as T
from pathlib import Path
import json
def process_images(image_folder, model, postprocessors, id2name, threshold=0.1, num_images=60):
    # Set up transforms
    transform = T.Compose([
        T.RandomResize([800], max_size=1333),
        T.ToTensor(),
        T.Normalize([0.485, 0.456, 0.406], [0.229, 0.224, 0.225])
    ])

    # Set up visualizer
    vslzr = COCOVisualizer()

    # Get list of image files (including '._' files)
    image_files = list(Path(image_folder).glob('._*'))
    image_files = image_files[:num_images]  # Limit to specified number of images

    for img_path in image_files:
        try:
            # Attempt to open the '._' file directly
            with open(img_path, 'rb') as f:
                # Skip the first 4096 bytes (AppleDouble header)
                f.seek(4096)
```

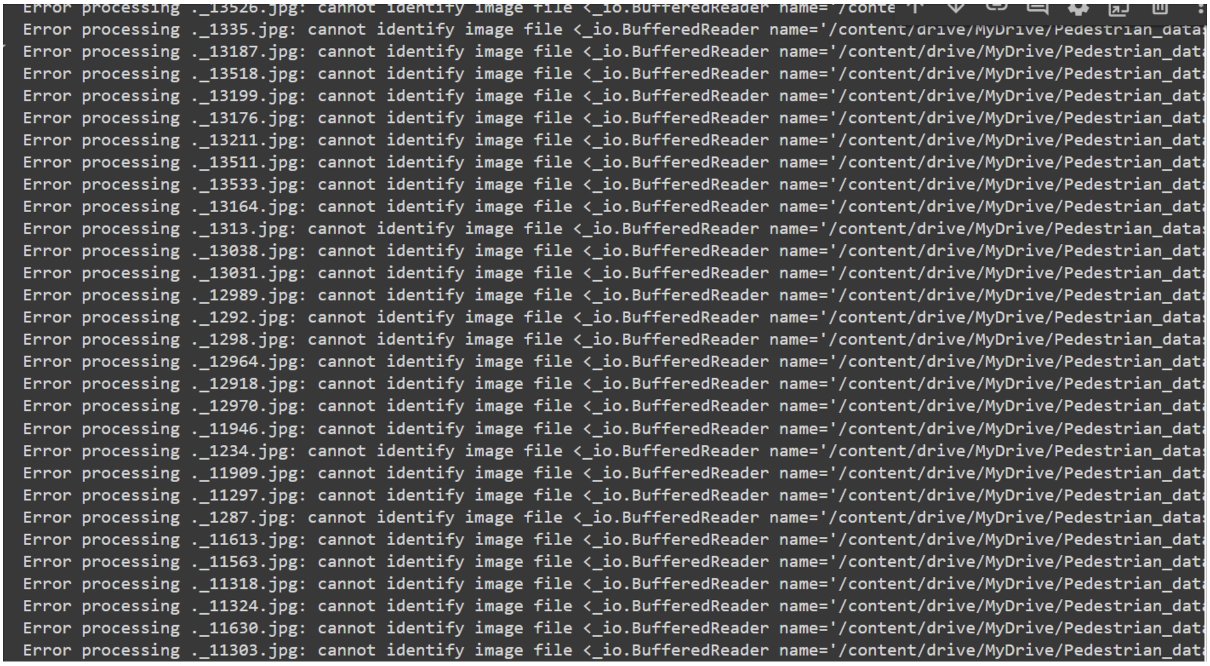


```python
            # If successful, proceed with processing
            image = image.convert("RGB")
            image_tensor, _ = transform(image, None)

            # Predict
            with torch.no_grad():
                output = model.cuda()(image_tensor[None].cuda())
                output = postprocessors['bbox'](output, torch.Tensor([[1.0, 1.0]]).cuda())[0]

            # Process output
            scores = output['scores']
            labels = output['labels']
            boxes = box_ops.box_xyxy_to_cxcywh(output['boxes'])
            select_mask = scores > threshold

            box_label = [id2name[int(item)] for item in labels[select_mask]]
            pred_dict = {
                'boxes': boxes[select_mask],
                'size': torch.Tensor([image_tensor.shape[1], image_tensor.shape[2]]),
                'box_label': box_label
            }

            # Visualize
            output_path = img_path.parent / f"{img_path.stem[2:]}_prediction.png"  # Remove '._' from filename
            vslzr.visualize(image_tensor, pred_dict, savedir=str(output_path), dpi=100)
```



```
            print(f"Processed and saved prediction for {img_path.name}")

        except Exception as e:
            print(f"Error processing {img_path.name}: {str(e)}")



# Load your id2name mapping
with open("/content/drive/MyDrive/random_sample_mavi_2_gt (1).json") as f:
    data = json.load(f)
    id2name = {item['id']: item['file_name'] for item in data['images']}

# # Define your postprocessors
# postprocessors = {
#     'bbox': ...  # Your bbox postprocessor function or object
# }

# Folder containing your images
image_folder = '/content/drive/MyDrive/Pedestrian_dataset_for_internship_assignment'

# Process images
process_images(image_folder, model, postprocessors, id2name)
```

```
Error processing ._13547.jpg: cannot identify image file <_io.BufferedReader name='/content/drive/MyDrive/Pedestrian_dat
Error processing ._13526.jpg: cannot identify image file <_io.BufferedReader name='/content/drive/MyDrive/Pedestrian_dat
Error processing ._1335.jpg: cannot identify image file <_io.BufferedReader name='/content/drive/MyDrive/Pedestrian_data
Error processing ._13187.jpg: cannot identify image file <_io.BufferedReader name='/content/drive/MyDrive/Pedestrian_dat
Error processing ._13518.jpg: cannot identify image file <_io.BufferedReader name='/content/drive/MyDrive/Pedestrian_dat
```



```
Error processing ._13526.jpg: cannot identify image file <_io.BufferedReader name='/conte
Error processing ._1335.jpg: cannot identify image file <_io.BufferedReader name='/content/drive/MyDrive/Pedestrian_data
Error processing ._13187.jpg: cannot identify image file <_io.BufferedReader name='/content/drive/MyDrive/Pedestrian_dat
Error processing ._13518.jpg: cannot identify image file <_io.BufferedReader name='/content/drive/MyDrive/Pedestrian_dat
Error processing ._13199.jpg: cannot identify image file <_io.BufferedReader name='/content/drive/MyDrive/Pedestrian_dat
Error processing ._13176.jpg: cannot identify image file <_io.BufferedReader name='/content/drive/MyDrive/Pedestrian_dat
Error processing ._13211.jpg: cannot identify image file <_io.BufferedReader name='/content/drive/MyDrive/Pedestrian_dat
Error processing ._13511.jpg: cannot identify image file <_io.BufferedReader name='/content/drive/MyDrive/Pedestrian_dat
Error processing ._13533.jpg: cannot identify image file <_io.BufferedReader name='/content/drive/MyDrive/Pedestrian_dat
Error processing ._13164.jpg: cannot identify image file <_io.BufferedReader name='/content/drive/MyDrive/Pedestrian_dat
Error processing ._1313.jpg: cannot identify image file <_io.BufferedReader name='/content/drive/MyDrive/Pedestrian_data
Error processing ._13038.jpg: cannot identify image file <_io.BufferedReader name='/content/drive/MyDrive/Pedestrian_dat
Error processing ._13031.jpg: cannot identify image file <_io.BufferedReader name='/content/drive/MyDrive/Pedestrian_dat
Error processing ._12989.jpg: cannot identify image file <_io.BufferedReader name='/content/drive/MyDrive/Pedestrian_dat
Error processing ._1292.jpg: cannot identify image file <_io.BufferedReader name='/content/drive/MyDrive/Pedestrian_data
Error processing ._1298.jpg: cannot identify image file <_io.BufferedReader name='/content/drive/MyDrive/Pedestrian_data
Error processing ._12964.jpg: cannot identify image file <_io.BufferedReader name='/content/drive/MyDrive/Pedestrian_dat
Error processing ._12918.jpg: cannot identify image file <_io.BufferedReader name='/content/drive/MyDrive/Pedestrian_dat
Error processing ._12970.jpg: cannot identify image file <_io.BufferedReader name='/content/drive/MyDrive/Pedestrian_dat
Error processing ._11946.jpg: cannot identify image file <_io.BufferedReader name='/content/drive/MyDrive/Pedestrian_dat
Error processing ._1234.jpg: cannot identify image file <_io.BufferedReader name='/content/drive/MyDrive/Pedestrian_data
Error processing ._11909.jpg: cannot identify image file <_io.BufferedReader name='/content/drive/MyDrive/Pedestrian_dat
Error processing ._11297.jpg: cannot identify image file <_io.BufferedReader name='/content/drive/MyDrive/Pedestrian_dat
Error processing ._1287.jpg: cannot identify image file <_io.BufferedReader name='/content/drive/MyDrive/Pedestrian_data
Error processing ._11613.jpg: cannot identify image file <_io.BufferedReader name='/content/drive/MyDrive/Pedestrian_dat
Error processing ._11563.jpg: cannot identify image file <_io.BufferedReader name='/content/drive/MyDrive/Pedestrian_dat
Error processing ._11318.jpg: cannot identify image file <_io.BufferedReader name='/content/drive/MyDrive/Pedestrian_dat
Error processing ._11324.jpg: cannot identify image file <_io.BufferedReader name='/content/drive/MyDrive/Pedestrian_dat
Error processing ._11630.jpg: cannot identify image file <_io.BufferedReader name='/content/drive/MyDrive/Pedestrian_dat
Error processing ._11303.jpg: cannot identify image file <_io.BufferedReader name='/content/drive/MyDrive/Pedestrian_dat
```

OBSERVATIONS:

1. Successfully Cloned and run the DINO-4Scale model.
2. Checkpoint used: Checkpoint0033_4scale.pth for running the model.
3. Since the Dataset provided is MacOS hidden files (MetaData), they cannot be easily accessed in Windows. The dataset also includes AppleHeader code for accessing the files.
4. The Pretrained DINO-4scale model with Resnet50 backbone successfully detects the flower image as seen in the above screenshot.