Ajmal_Assignment3

November 10, 2025

[1]:
```
import tensorflow as tf
import tensorflow_datasets as tfds
import matplotlib.pylab as plt
import numpy as np
import cv2
```

[7]:
```
#a) Load the oxford_iiit_pet dataset
import os

# Disable TFDS progress bars
os.environ['TFDS_TQDM'] = '0'
tfds.disable_progress_bar()

dataset, info = tfds.load('oxford_iiit_pet:4.0.0', with_info=True)
print(info)
```

```
Downloading and preparing dataset Unknown size (download: Unknown size,
generated: Unknown size, total: Unknown size) to
/Users/ajmaljalal/tensorflow_datasets/oxford_iiit_pet/4.0.0…
Corrupt JPEG data: premature end of data segment
Corrupt JPEG data: 240 extraneous bytes before marker 0xd9
Dataset oxford_iiit_pet downloaded and prepared to
/Users/ajmaljalal/tensorflow_datasets/oxford_iiit_pet/4.0.0. Subsequent calls
will reuse this data.
tfds.core.DatasetInfo(
    name='oxford_iiit_pet',
    full_name='oxford_iiit_pet/4.0.0',
    description="""
    The Oxford-IIIT pet dataset is a 37 category pet image dataset with roughly
200
    images for each class. The images have large variations in scale, pose and
    lighting. All images have an associated ground truth annotation of breed and
    species. Additionally, head bounding boxes are provided for the training
split,
    allowing using this dataset for simple object detection tasks. In the test
```

```
    split, the bounding boxes are empty.
    """,
    homepage='http://www.robots.ox.ac.uk/~vgg/data/pets/',
    data_dir='/Users/ajmaljalal/tensorflow_datasets/oxford_iiit_pet/4.0.0',
    file_format=tfrecord,
    download_size=773.52 MiB,
    dataset_size=773.68 MiB,
    features=FeaturesDict({
        'file_name': Text(shape=(), dtype=string),
        'head_bbox': BBoxFeature(shape=(4,), dtype=float32),
        'image': Image(shape=(None, None, 3), dtype=uint8),
        'label': ClassLabel(shape=(), dtype=int64, num_classes=37),
        'segmentation_mask': Image(shape=(None, None, 1), dtype=uint8),
        'species': ClassLabel(shape=(), dtype=int64, num_classes=2),
    }),
    supervised_keys=('image', 'label'),
    disable_shuffling=False,
    nondeterministic_order=False,
    splits={
        'test': <SplitInfo num_examples=3669, num_shards=4>,
        'train': <SplitInfo num_examples=3680, num_shards=4>,
    },
    citation="""@InProceedings{parkhi12a,
      author       = "Parkhi, O. M. and Vedaldi, A. and Zisserman, A. and
Jawahar, C.~V.",
      title        = "Cats and Dogs",
      booktitle    = "IEEE Conference on Computer Vision and Pattern
Recognition",
      year         = "2012",
    }""",
)
```

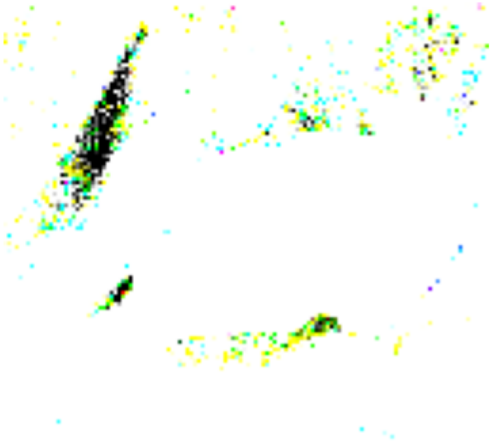
[8]:
```
def read_and_preprocess(data):
  input_image = tf.image.resize(data['image'], (128, 128))     #Resize the␣
 ↪data['image'] to 128x128
  input_mask = tf.image.resize(data['segmentation_mask'], (128, 128))    ␣
 ↪#Resize the data['segmentation_mask'] to 128x128

  input_image = tf.image.convert_image_dtype(input_image, tf.float32) # [0,1]
  input_mask -= 1 # {1,2,3} to {0,1,2}

  return input_image, input_mask
```

[9]:
```
train = dataset['train'].map(read_and_preprocess, num_parallel_calls=tf.data.
 ↪AUTOTUNE)
test = dataset['test'].map(read_and_preprocess)
```

[10]:
```
# b) Create the segmentation mask
# Show some images from dataset and their segmented version

# Take first 3 samples from the train dataset
fig, axes = plt.subplots(3, 2, figsize=(10, 12))

for i, (image, mask) in enumerate(train.take(3)):
    # Display original image
    axes[i, 0].imshow(image)
    axes[i, 0].set_title(f'Sample {i+1} - Original Image')
    axes[i, 0].axis('off')

    # Display segmentation mask
    axes[i, 1].imshow(mask[:, :, 0], cmap='gray')
    axes[i, 1].set_title(f'Sample {i+1} - Segmentation Mask')
    axes[i, 1].axis('off')

plt.tight_layout()
plt.show()
```

```
WARNING:matplotlib.image:Clipping input data to the valid range for imshow with
RGB data ([0..1] for floats or [0..255] for integers). Got range
[0.0..201.92188].
WARNING:matplotlib.image:Clipping input data to the valid range for imshow with
RGB data ([0..1] for floats or [0..255] for integers). Got range [0.0..255.0].
WARNING:matplotlib.image:Clipping input data to the valid range for imshow with
RGB data ([0..1] for floats or [0..255] for integers). Got range
[0.0..251.49176].
2025-11-10 19:38:54.276193: W tensorflow/core/framework/local_rendezvous.cc:404]
Local rendezvous is aborting with status: OUT_OF_RANGE: End of sequence
```

Sample 1 - Original Image



Sample 1 - Segmentation Mask

Sample 2 - Original Image

Sample 2 - Segmentation Mask

Sample 3 - Original Image

Sample 3 - Segmentation Mask

Part 2- Annotation

[ ]:
```
# Install mrcnn
%pip install -q scikit-image
%pip install -q git+https://github.com/matterport/Mask_RCNN.git
```

```
  DEPRECATION: Building 'mask-rcnn' using the legacy setup.py bdist_wheel
mechanism, which will be removed in a future version. pip 25.3 will enforce this
behaviour change. A possible replacement is to use the standardized build
interface by setting the `--use-pep517` option, (possibly combined with `--no-
build-isolation`), or adding a `pyproject.toml` file to the source tree of
'mask-rcnn'. Discussion can be found at
https://github.com/pypa/pip/issues/6334


[notice] A new release of pip is
available: 25.1.1 -> 25.3
[notice] To update, run:
pip install --upgrade pip
```

[12]:
```
# Get data from here https://github.com/experiencor/raccoon_dataset
!git clone https://github.com/experiencor/raccoon_dataset
```

```
Cloning into 'raccoon_dataset'…
remote: Enumerating objects: 646, done.
remote: Counting objects: 100% (646/646), done.
remote: Compressing objects: 100% (232/232), done.
remote: Total 646 (delta 413), reused 643 (delta 412), pack-reused 0 (from 0)
Receiving objects: 100% (646/646), 48.00 MiB | 8.42 MiB/s, done.
Resolving deltas: 100% (413/413), done.
```

[13]:
```
# Look into data
# Plot some samples here

import os
from PIL import Image

# Get list of images
image_dir = 'raccoon_dataset/images/'
image_files = [f for f in os.listdir(image_dir) if f.endswith('.jpg')][:6]

# Display 6 sample images
fig, axes = plt.subplots(2, 3, figsize=(15, 10))
axes = axes.flatten()

for i, img_file in enumerate(image_files):
```

```
    img_path = os.path.join(image_dir, img_file)
    img = Image.open(img_path)
    axes[i].imshow(img)
    axes[i].set_title(f'Sample {i+1}: {img_file}')
    axes[i].axis('off')

plt.tight_layout()
plt.show()
```

[15]:
```
%pip install -q scikit-image
%pip install -q git+https://github.com/matterport/Mask_RCNN.git
```

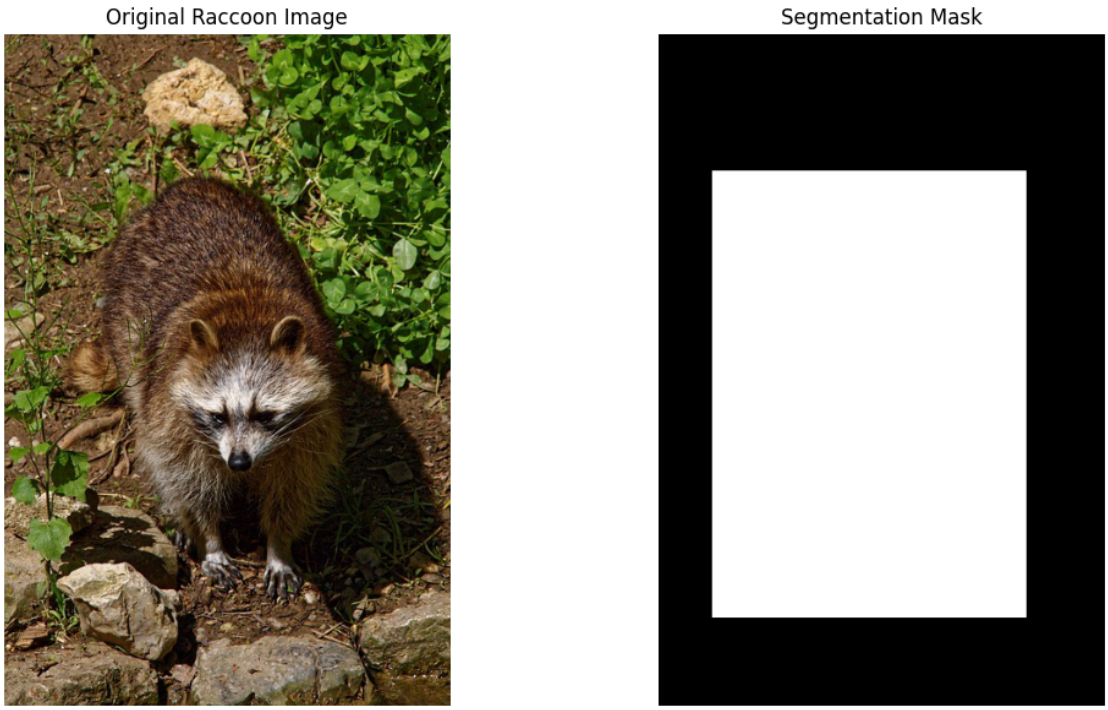
```
[notice] A new release of pip is
available: 25.1.1 -> 25.3
[notice] To update, run:
pip install --upgrade pip
Note: you may need to restart the kernel to use updated packages.

[notice] A new release of pip is
available: 25.1.1 -> 25.3
[notice] To update, run:
pip install --upgrade pip
Note: you may need to restart the kernel to use updated packages.
```

[16]:
```
from os import listdir
from xml.etree import ElementTree
from numpy import zeros
from numpy import asarray
from mrcnn.utils import Dataset

# class that defines and loads the raccoon dataset
class RaccoonDataset(Dataset):
        # load the dataset definitions
        def load_dataset(self, dataset_dir, is_train=True):
                # define one class
                self.add_class("raccoon_dataset", 1, "raccoon")
                # define data locations
                images_dir = dataset_dir + '/images/'
                annotations_dir = dataset_dir + '/annotations/raccoon-'
                # find all images
                for filename in listdir(images_dir):
                        # extract image id
                        image_id = filename[8:-4]
                        # skip bad images
                        if image_id in ['00090']:
                                continue
                        # skip all images after 150 if we are building the␣
 ↪train set
                        if is_train and int(image_id) >= 150:
                                continue
                        # skip all images before 150 if we are building the␣
 ↪test/val set
                        if not is_train and int(image_id) < 150:
                                continue
                        img_path = images_dir + filename
                        ann_path = annotations_dir + image_id + '.xml'
                        # add to dataset
                        self.add_image('dataset', image_id=image_id,␣
 ↪path=img_path, annotation=ann_path)

        # extract bounding boxes from an annotation file
        def extract_boxes(self, filename):
                # load and parse the file
                tree = ElementTree.parse(filename)
                # get the root of the document
                root = tree.getroot()
                # extract each bounding box
                boxes = list()
                for box in root.findall('.//bndbox'):
                        xmin = int(box.find('xmin').text)
                        ymin = int(box.find('ymin').text)
```

```
                        xmax = int(box.find('xmax').text)
                        ymax = int(box.find('ymax').text)
                        coors = [xmin, ymin, xmax, ymax]
                        boxes.append(coors)
                # extract image dimensions
                width = int(root.find('.//size/width').text)
                height = int(root.find('.//size/height').text)
                return boxes, width, height

        # load the masks for an image
        def load_mask(self, image_id):
                # get details of image
                info = self.image_info[image_id]
                # define box file location
                path = info['annotation']
                # load XML
                #path = '/content/raccoon_dataset/annotations/
↪raccoon-'+image_id   #Added by me
                boxes, w, h = self.extract_boxes(path)
                # create one array for all masks, each on a different channel
                masks = zeros([h, w, len(boxes)], dtype='uint8')
                # create masks
                class_ids = list()
                for i in range(len(boxes)):
                        box = boxes[i]
                        row_s, row_e = box[1], box[3]
                        col_s, col_e = box[0], box[2]
                        masks[row_s:row_e, col_s:col_e, i] = 1
                        class_ids.append(self.class_names.index('raccoon'))
                return masks, asarray(class_ids, dtype='int32')

        # load an image reference
        def image_reference(self, image_id):
                info = self.image_info[image_id]
                return info['path']

# train set
train_set = RaccoonDataset()
train_set.load_dataset('raccoon_dataset', is_train=True)
train_set.prepare()
print('Train: %d' % len(train_set.image_ids))

# test/val set
test_set = RaccoonDataset()
test_set.load_dataset('raccoon_dataset', is_train=False)
test_set.prepare()
print('Test: %d' % len(test_set.image_ids))
```

```
Train: 149
Test: 51
```

[17]:
```python
# load an image
# Use the function above to create the image and its mask

# Load a sample image (first image from train set)
image_id = 0
image = train_set.load_image(image_id)
mask, class_ids = train_set.load_mask(image_id)

# Display the image and its mask
fig, axes = plt.subplots(1, 2, figsize=(12, 6))

# Display original image
axes[0].imshow(image)
axes[0].set_title('Original Raccoon Image')
axes[0].axis('off')

# Display mask (if multiple masks, combine them)
if mask.shape[2] > 0:
    combined_mask = np.sum(mask, axis=2)
    axes[1].imshow(combined_mask, cmap='gray')
    axes[1].set_title('Segmentation Mask')
else:
    axes[1].imshow(np.zeros_like(image[:,:,0]), cmap='gray')
    axes[1].set_title('No Mask Available')
axes[1].axis('off')

plt.tight_layout()
plt.show()

print(f"Image shape: {image.shape}")
print(f"Mask shape: {mask.shape}")
print(f"Number of raccoons detected: {mask.shape[2]}")
```



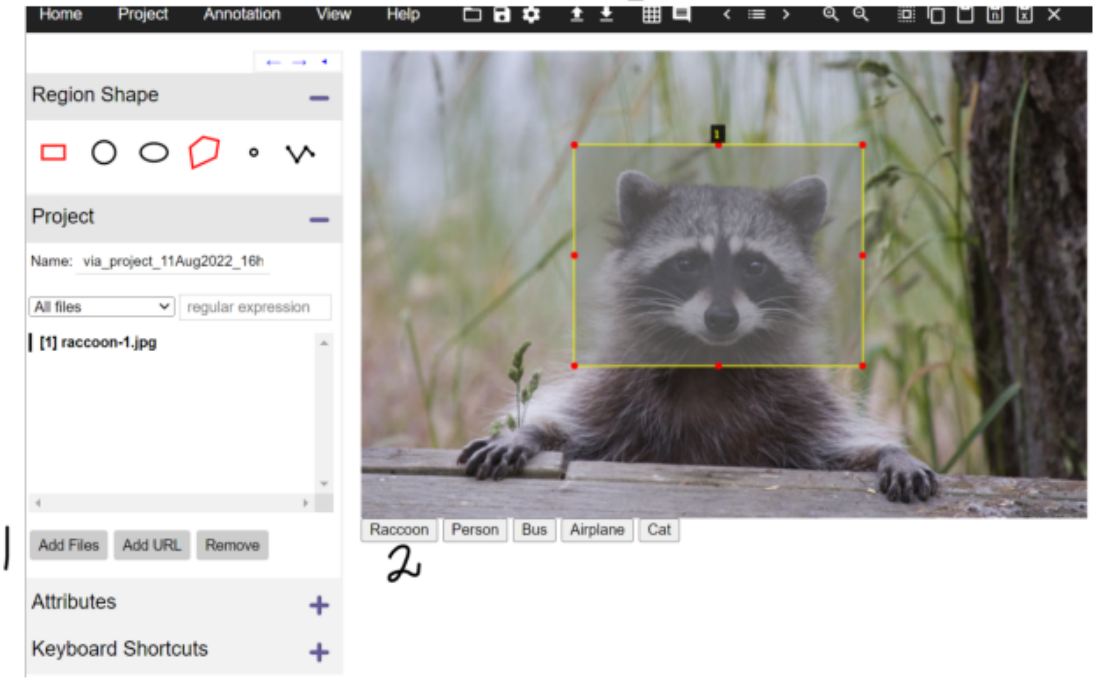

```
Image shape: (640, 426, 3)
Mask shape: (640, 426, 1)
Number of raccoons detected: 1
```

Part 3- YOLO

You can see how YOLO has been improved in the last few versions on COCO:

[21]:
```
# a- Create annotation
# You can upload the file using ![title](filename.jpeg)
# import image module

from PIL import Image as PILImage
import matplotlib.pyplot as plt

# Display the sample raccoon image
img = PILImage.open('Raccon_sample.jpg')
plt.figure(figsize=(8, 6))
plt.imshow(img)
plt.title('Sample Image: Raccon_sample.jpg')
plt.axis('off')
plt.show()
```

Sample Image: Raccon_sample.jpg



[ ]:
```
#b- Download required tools
import os
HOME = os.getcwd()
!pip install -U ultralytics
```

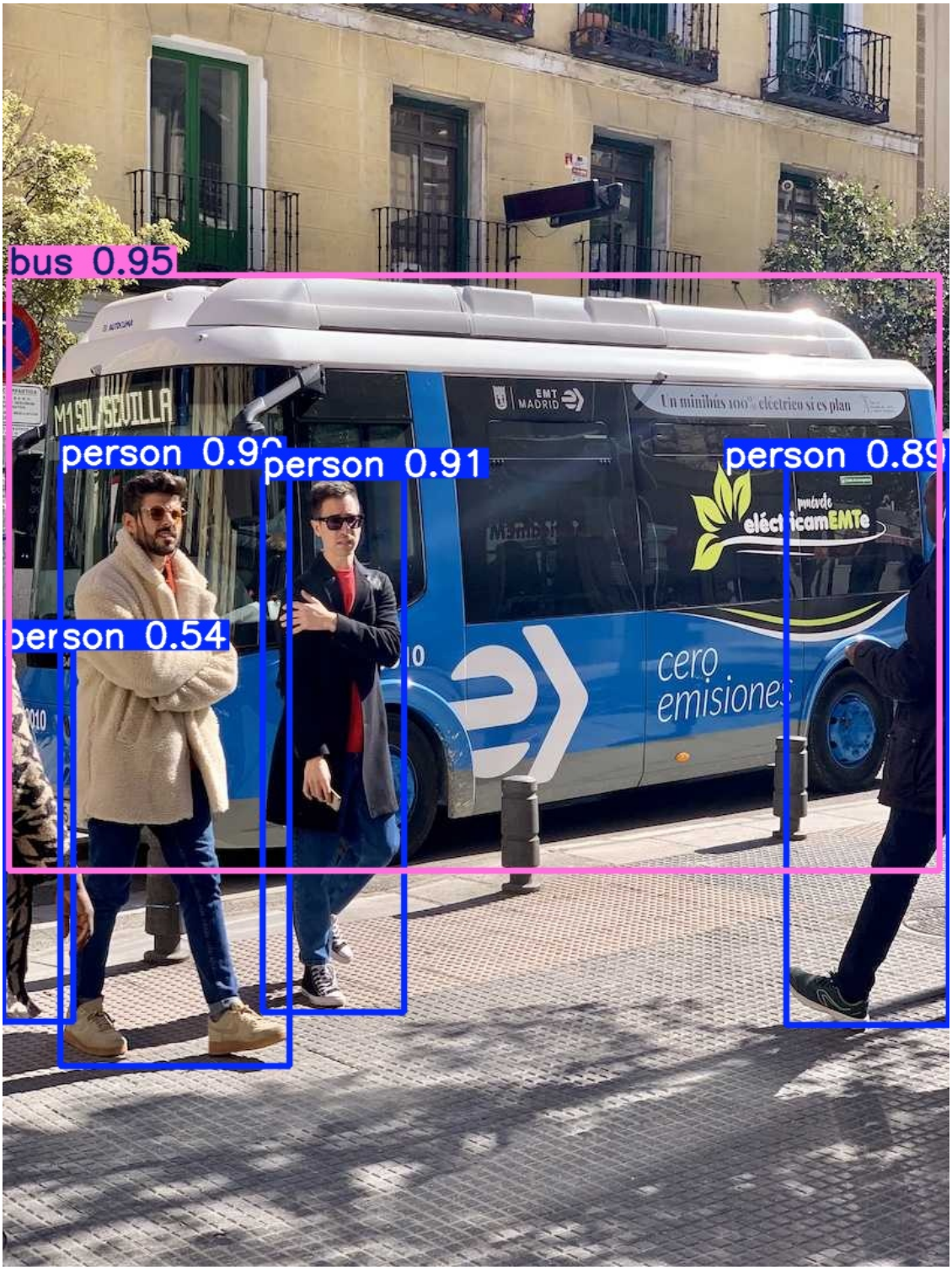
[24]:
```
#c- Test the model out of box
import os
HOME = os.getcwd()

from ultralytics import YOLO
model = YOLO(f'{HOME}/weights/yolov10n.pt') # Load the yolov10n model
results = model.predict(source="https://ultralytics.com/images/bus.jpg", conf=0.
 ↪25, save=True)
print(f"Results saved to: {results[0].save_dir}")
```

```
Downloading
https://github.com/ultralytics/assets/releases/download/v8.3.0/yolov10n.pt to
'/Users/ajmaljalal/Desktop/ML & AI/Master
Program/Semester_four/AAI_521/weights/yolov10n.pt': 100%        5.6MB
2.6MB/s 2.1s.1s<0.0s2.6s

WARNING  Download failure, retrying 1/3
https://ultralytics.com/images/bus.jpg… HTTP Error 403: Forbidden
```

```
################################################################################
100.0%#=#=#
##=#=#

image 1/1 /Users/ajmaljalal/Desktop/ML & AI/Master
Program/Semester_four/AAI_521/bus.jpg: 640x480 4 persons, 1 bus, 107.0ms
Speed: 5.7ms preprocess, 107.0ms inference, 0.3ms postprocess per image at shape
(1, 3, 640, 480)
Results saved to /Users/ajmaljalal/Desktop/ML & AI/Master

Program/Semester_four/AAI_521/runs/detect/predict
Results saved to: /Users/ajmaljalal/Desktop/ML & AI/Master
Program/Semester_four/AAI_521/runs/detect/predict
Speed: 5.7ms preprocess, 107.0ms inference, 0.3ms postprocess per image at shape
(1, 3, 640, 480)
Results saved to /Users/ajmaljalal/Desktop/ML & AI/Master

Program/Semester_four/AAI_521/runs/detect/predict
Results saved to: /Users/ajmaljalal/Desktop/ML & AI/Master
Program/Semester_four/AAI_521/runs/detect/predict
```

[25]:
```
from IPython.display import Image
import os
HOME = os.getcwd()
Image(filename=f'{HOME}/runs/detect/predict/bus.jpg') # Display the prediction␣
 ↪result
```

[25]:



[27]:
```
#d- Apply model here
import os
```

```
HOME = os.getcwd()
!yolo task=detect mode=train model="{HOME}/weights/yolov10n.pt" data=coco128.
 ↪yaml epochs=10 batch=32 imgsz=640
```

```
Ultralytics 8.3.227  Python-3.12.5 torch-2.9.0 CPU (Apple M1 Max)
engine/trainer: agnostic_nms=False, amp=True, augment=False,
auto_augment=randaugment, batch=32, bgr=0.0, box=7.5, cache=False, cfg=None,
classes=None, close_mosaic=10, cls=0.5, compile=False, conf=None,
copy_paste=0.0, copy_paste_mode=flip, cos_lr=False, cutmix=0.0,
data=coco128.yaml, degrees=0.0, deterministic=True, device=cpu, dfl=1.5,
dnn=False, dropout=0.0, dynamic=False, embed=None, epochs=10, erasing=0.4,
exist_ok=False, fliplr=0.5, flipud=0.0, format=torchscript, fraction=1.0,
freeze=None, half=False, hsv_h=0.015, hsv_s=0.7, hsv_v=0.4, imgsz=640,
int8=False, iou=0.7, keras=False, kobj=1.0, line_width=None, lr0=0.01, lrf=0.01,
mask_ratio=4, max_det=300, mixup=0.0, mode=train,
model=/Users/ajmaljalal/Desktop/ML & AI/Master
Program/Semester_four/AAI_521/weights/yolov10n.pt, momentum=0.937, mosaic=1.0,
multi_scale=False, name=train, nbs=64, nms=False, opset=None, optimize=False,
optimizer=auto, overlap_mask=True, patience=100, perspective=0.0, plots=True,
pose=12.0, pretrained=True, profile=False, project=None, rect=False,
resume=False, retina_masks=False, save=True, save_conf=False, save_crop=False,
save_dir=/Users/ajmaljalal/Desktop/ML & AI/Master
Program/Semester_four/AAI_521/runs/detect/train, save_frames=False,
save_json=False, save_period=-1, save_txt=False, scale=0.5, seed=0, shear=0.0,
show=False, show_boxes=True, show_conf=True, show_labels=True, simplify=True,
single_cls=False, source=None, split=val, stream_buffer=False, task=detect,
time=None, tracker=botsort.yaml, translate=0.1, val=True, verbose=True,
vid_stride=1, visualize=False, warmup_bias_lr=0.1, warmup_epochs=3.0,
warmup_momentum=0.8, weight_decay=0.0005, workers=8, workspace=None

WARNING  Dataset 'coco128.yaml' images not found, missing path
'/Users/ajmaljalal/Desktop/ML & AI/Master
Program/Semester_four/AAI_521/datasets/coco128/images/train2017'
Downloading https://ultralytics.com/assets/coco128.zip to
'/Users/ajmaljalal/Desktop/ML & AI/Master
Program/Semester_four/AAI_521/datasets/coco128.zip': 100%        6.7MB
3.0MB/s 2.3s.2s<0.1s2.7s
Unzipping /Users/ajmaljalal/Desktop/ML & AI/Master
Program/Semester_four/AAI_521/datasets/coco128.zip to
/Users/ajmaljalal/Desktop/ML & AI/Master
Program/Semester_four/AAI_521/datasets/coco128…: 100%        263/263
4.8Kfiles/s 0.1s
Dataset download success  (2.7s), saved to /Users/ajmaljalal/Desktop/ML &
AI/Master Program/Semester_four/AAI_521/datasets


                   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]
  2                  -1  1      7360  ultralytics.nn.modules.block.C2f
[32, 32, 1, True]
  3                  -1  1     18560  ultralytics.nn.modules.conv.Conv
[32, 64, 3, 2]
  4                  -1  2     49664  ultralytics.nn.modules.block.C2f
[64, 64, 2, True]
  5                  -1  1      9856  ultralytics.nn.modules.block.SCDown
[64, 128, 3, 2]
  6                  -1  2    197632  ultralytics.nn.modules.block.C2f
[128, 128, 2, True]
  7                  -1  1     36096  ultralytics.nn.modules.block.SCDown
[128, 256, 3, 2]
  8                  -1  1    460288  ultralytics.nn.modules.block.C2f
[256, 256, 1, True]
  9                  -1  1    164608  ultralytics.nn.modules.block.SPPF
[256, 256, 5]
 10                  -1  1    249728  ultralytics.nn.modules.block.PSA
[256, 256]
 11                  -1  1         0  torch.nn.modules.upsampling.Upsample
[None, 2, 'nearest']
 12             [-1, 6]  1         0  ultralytics.nn.modules.conv.Concat
[1]
 13                  -1  1    148224  ultralytics.nn.modules.block.C2f
[384, 128, 1]
 14                  -1  1         0  torch.nn.modules.upsampling.Upsample
[None, 2, 'nearest']
 15             [-1, 4]  1         0  ultralytics.nn.modules.conv.Concat
[1]
 16                  -1  1     37248  ultralytics.nn.modules.block.C2f
[192, 64, 1]
 17                  -1  1     36992  ultralytics.nn.modules.conv.Conv
[64, 64, 3, 2]
 18            [-1, 13]  1         0  ultralytics.nn.modules.conv.Concat
[1]
 19                  -1  1    123648  ultralytics.nn.modules.block.C2f
[192, 128, 1]
 20                  -1  1     18048  ultralytics.nn.modules.block.SCDown
[128, 128, 3, 2]
 21            [-1, 10]  1         0  ultralytics.nn.modules.conv.Concat
[1]
 22                  -1  1    282624  ultralytics.nn.modules.block.C2fCIB
[384, 256, 1, True, True]
 23        [16, 19, 22]  1    929808  ultralytics.nn.modules.head.v10Detect
```

```
[80, [64, 128, 256]]
YOLOv10n summary: 223 layers, 2,775,520 parameters, 2,775,504 gradients, 8.7
GFLOPs

Transferred 595/595 items from pretrained weights
Freezing layer 'model.23.dfl.conv.weight'
train: Fast image access  (ping: 0.0±0.0 ms, read: 1460.0±607.3
MB/s, size: 46.0 KB)
train: Scanning /Users/ajmaljalal/Desktop/ML & AI/Master
Program/Semester_four/AAI_521/datasets/coco128/labels/train2017… 126 images, 2
backgrounds, 0 corrupt: 100%        128/128 2.3Kit/s 0.1s
train: New cache created: /Users/ajmaljalal/Desktop/ML & AI/Master
Program/Semester_four/AAI_521/datasets/coco128/labels/train2017.cache
val: Fast image access  (ping: 0.0±0.0 ms, read: 1698.4±777.5
MB/s, size: 56.4 KB)
val: Scanning /Users/ajmaljalal/Desktop/ML & AI/Master
Program/Semester_four/AAI_521/datasets/coco128/labels/train2017.cache… 126
images, 2 backgrounds, 0 corrupt: 100%        128/128 5.2Mit/s 0.0s0s
Plotting labels to /Users/ajmaljalal/Desktop/ML & AI/Master
Program/Semester_four/AAI_521/runs/detect/train/labels.jpg…
optimizer: 'optimizer=auto' found, ignoring 'lr0=0.01' and
'momentum=0.937' and determining best 'optimizer', 'lr0' and 'momentum'
automatically…
optimizer: AdamW(lr=0.000119, momentum=0.9) with parameter groups
95 weight(decay=0.0), 108 weight(decay=0.0005), 107 bias(decay=0.0)
Image sizes 640 train, 640 val
Using 0 dataloader workers
Logging results to /Users/ajmaljalal/Desktop/ML & AI/Master

Program/Semester_four/AAI_521/runs/detect/train
Starting training for 10 epochs…
Closing dataloader mosaic

      Epoch    GPU_mem   box_loss   cls_loss   dfl_loss  Instances       Size
       1/10         0G      2.514      2.664      2.368        189
640: 100%        4/4 0.0it/s 1:2832.6s
                 Class     Images  Instances      Box(P          R      mAP50
mAP50-95): 100%        2/2 0.1it/s 22.1s2.3s
                   all        128        929      0.679      0.534      0.633
0.466

      Epoch    GPU_mem   box_loss   cls_loss   dfl_loss  Instances       Size
       2/10         0G      2.522      2.674      2.392        235
640: 100%        4/4 0.0it/s 1:2733.7s
                 Class     Images  Instances      Box(P          R      mAP50
mAP50-95): 100%        2/2 0.1it/s 21.4s0.4s
                   all        128        929      0.712      0.535      0.642
0.476
```

```
      Epoch    GPU_mem   box_loss   cls_loss   dfl_loss  Instances       Size
       3/10         0G      2.496      2.566      2.321        238
640: 100%        4/4 0.0it/s 1:2632.5s
                 Class     Images  Instances      Box(P          R      mAP50
mAP50-95): 100%        2/2 0.1it/s 22.4s1.8s
                   all        128        929      0.755      0.536      0.649
0.486

      Epoch    GPU_mem   box_loss   cls_loss   dfl_loss  Instances       Size
       4/10         0G      2.438      2.669       2.29        204
640: 100%        4/4 0.0it/s 1:2833.3s
                 Class     Images  Instances      Box(P          R      mAP50
mAP50-95): 100%        2/2 0.1it/s 22.8s2.1s
                   all        128        929      0.694      0.562      0.657
0.495

      Epoch    GPU_mem   box_loss   cls_loss   dfl_loss  Instances       Size
       5/10         0G      2.404      2.421      2.271        187
640: 100%        4/4 0.0it/s 1:2933.3s
                 Class     Images  Instances      Box(P          R      mAP50
mAP50-95): 100%        2/2 0.1it/s 22.6s2.7s
                   all        128        929      0.677      0.607      0.682
0.512

      Epoch    GPU_mem   box_loss   cls_loss   dfl_loss  Instances       Size
       6/10         0G      2.473      2.515      2.336        173
640: 100%        4/4 0.0it/s 1:2933.4s
                 Class     Images  Instances      Box(P          R      mAP50
mAP50-95): 100%        2/2 0.1it/s 21.8s0.9s
                   all        128        929      0.679      0.626      0.692
0.521

      Epoch    GPU_mem   box_loss   cls_loss   dfl_loss  Instances       Size
       7/10         0G      2.298      2.342      2.261        277
640: 100%        4/4 0.0it/s 1:3133.6s
                 Class     Images  Instances      Box(P          R      mAP50
mAP50-95): 100%        2/2 0.1it/s 22.0s1.2s
                   all        128        929      0.665      0.634      0.698
0.525

      Epoch    GPU_mem   box_loss   cls_loss   dfl_loss  Instances       Size
       8/10         0G      2.366      2.347      2.302        235
640: 100%        4/4 0.0it/s 1:2832.9s
                 Class     Images  Instances      Box(P          R      mAP50
mAP50-95): 100%        2/2 0.1it/s 22.0s1.2s
                   all        128        929      0.692       0.63      0.704
0.532
```

```
      Epoch    GPU_mem   box_loss   cls_loss   dfl_loss  Instances       Size
       9/10         0G      2.331      2.264      2.259        259
640: 100%        4/4 0.0it/s 1:2733.1s
                 Class     Images  Instances      Box(P          R      mAP50
mAP50-95): 100%        2/2 0.1it/s 22.7s1.9s
                   all        128        929      0.706      0.628      0.704
0.531

      Epoch    GPU_mem   box_loss   cls_loss   dfl_loss  Instances       Size
      10/10         0G      2.353      2.262      2.276        237
640: 100%        4/4 0.0it/s 1:2732.9s
                 Class     Images  Instances      Box(P          R      mAP50
mAP50-95): 100%        2/2 0.1it/s 22.1s0.8s
                   all        128        929      0.707      0.631      0.709
0.535

10 epochs completed in 0.306 hours.
Optimizer stripped from /Users/ajmaljalal/Desktop/ML & AI/Master
Program/Semester_four/AAI_521/runs/detect/train/weights/last.pt, 5.9MB
Optimizer stripped from /Users/ajmaljalal/Desktop/ML & AI/Master
Program/Semester_four/AAI_521/runs/detect/train/weights/best.pt, 5.9MB

Validating /Users/ajmaljalal/Desktop/ML & AI/Master
Program/Semester_four/AAI_521/runs/detect/train/weights/best.pt…
Ultralytics 8.3.227  Python-3.12.5 torch-2.9.0 CPU (Apple M1 Max)
YOLOv10n summary (fused): 102 layers, 2,299,264 parameters, 0 gradients, 6.7
GFLOPs
                 Class     Images  Instances      Box(P          R      mAP50
mAP50-95): 100%        2/2 0.1it/s 17.4s4.6s
                   all        128        929      0.705      0.632       0.71
0.535
                person         61        254      0.837      0.686      0.787
0.58
               bicycle          3          6      0.945      0.333      0.366
0.241
                   car         12         46      0.623      0.326       0.39
0.223
            motorcycle          4          5      0.747          1      0.995
0.824
              airplane          5          6      0.864          1      0.995
0.928
                   bus          5          7      0.793      0.714      0.761
0.655
                 train          3          3      0.528          1      0.995
0.81
                 truck          5         12      0.662      0.417        0.5
0.304
```

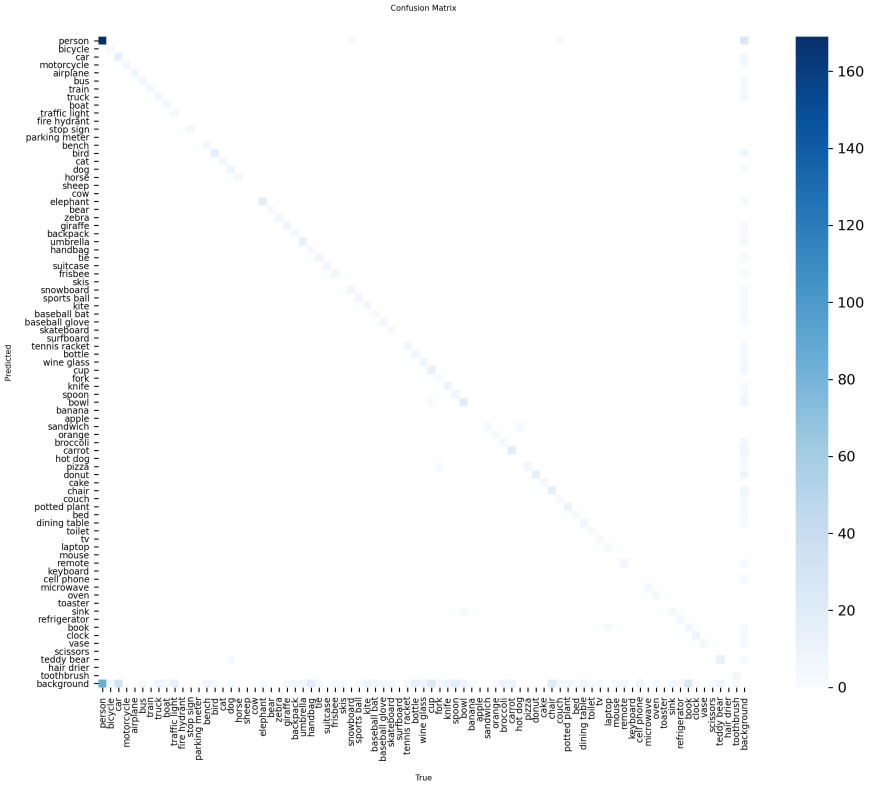
boat	2	6	1	0.541	0.729
0.459					
traffic light	4	14	0.916	0.214	0.263
0.178					
stop sign	2	2	0.753	1	0.995
0.895					
bench	5	9	0.78	0.401	0.63
0.352					
bird	2	16	0.714	0.938	0.936
0.681					
cat	4	4	0.835	1	0.995
0.952					
dog	9	9	0.929	0.889	0.956
0.773					
horse	1	2	0.741	1	0.995
0.702					
elephant	4	17	0.869	0.941	0.947
0.797					
bear	1	1	0.605	1	0.995
0.895					
zebra	2	4	0.845	1	0.995
0.975					
giraffe	4	9	0.886	0.866	0.962
0.747					
backpack	4	6	0.404	0.333	0.353
0.24					
umbrella	4	18	0.756	0.722	0.833
0.532					
handbag	9	19	0.745	0.158	0.25
0.143					
tie	6	7	0.681	0.571	0.693
0.497					
suitcase	2	4	0.801	1	0.995
0.672					
frisbee	5	5	0.722	0.8	0.76
0.687					
skis	1	1	0.738	1	0.995
0.21					
snowboard	2	7	0.802	0.857	0.848
0.615					
sports ball	6	6	0.653	0.667	0.669
0.328					
kite	2	10	0.61	0.3	0.453
0.187					
baseball bat	4	4	0.338	0.396	0.441
0.233					
baseball glove	4	7	0.648	0.571	0.579
0.283					

skateboard	3	5	0.62	0.4	0.571	0.458
tennis racket	5	7	0.605	0.571	0.655	0.41
bottle	6	18	0.626	0.278	0.564	0.354
wine glass	5	16	0.469	0.375	0.56	0.333
cup	10	36	0.68	0.417	0.546	0.371
fork	6	6	0.553	0.167	0.352	0.251
knife	7	16	0.81	0.5	0.647	0.483
spoon	5	22	0.746	0.409	0.505	0.329
bowl	9	28	0.682	0.679	0.681	0.569
banana	1	1	1	0	0.0369	0.00369
sandwich	2	2	0.432	1	0.995	0.995
orange	1	4	0.882	0.75	0.912	0.594
broccoli	4	11	0.464	0.273	0.393	0.31
carrot	3	24	0.715	0.733	0.737	0.485
hot dog	1	2	0.661	1	0.995	0.995
pizza	5	5	0.651	1	0.995	0.883
donut	2	14	0.611	1	0.915	0.819
cake	4	4	0.748	1	0.995	0.878
chair	9	35	0.586	0.514	0.533	0.313
couch	5	6	0.523	0.372	0.7	0.567
potted plant	9	14	0.799	0.714	0.804	0.567
bed	3	3	0.737	1	0.913	0.648
dining table	10	13	0.567	0.462	0.568	0.451
toilet	2	2	0.602	0.5	0.606	0.59

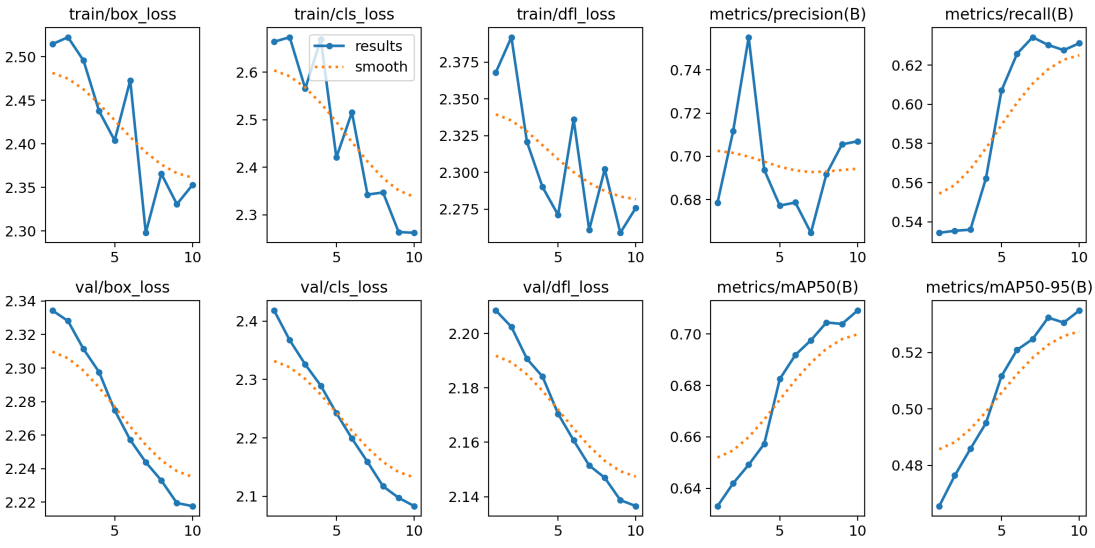
```
                  tv          2          2      0.943          1      0.995
0.896
              laptop          2          3      0.705      0.667      0.913
0.755
               mouse          2          2          1          0      0.551
0.282
              remote          5          8      0.678        0.5      0.584
0.486
          cell phone          5          8          0          0      0.162
0.0929
           microwave          3          3      0.585          1      0.995
0.825
                oven          5          5      0.506        0.4      0.476
0.366
                sink          4          6      0.484      0.333      0.371
0.228
        refrigerator          5          5          1      0.694      0.995
0.817
                book          6         29      0.711      0.207      0.446
0.216
               clock          8          9      0.752      0.889      0.917
0.787
                vase          2          2      0.437          1      0.995
0.898
            scissors          1          1          1          0     0.0524
0.00524
          teddy bear          6         21      0.765      0.619      0.756
0.496
          toothbrush          2          5      0.954        0.8      0.962
0.573
Speed: 0.4ms preprocess, 131.8ms inference, 0.0ms loss, 0.1ms postprocess per
image
Results saved to /Users/ajmaljalal/Desktop/ML & AI/Master
Program/Semester_four/AAI_521/runs/detect/train
  Learn more at https://docs.ultralytics.com/modes/train
```

[28]:
```python
# Show the learning curve and confusion matrix
from IPython.display import Image, display
import os
HOME = os.getcwd()

print("Confusion Matrix:")
display(Image(filename=f'{HOME}/runs/detect/train/confusion_matrix.png'))

print("\nLearning Curves:")
display(Image(filename=f'{HOME}/runs/detect/train/results.png'))
```

Confusion Matrix:



Learning Curves:



1 e- Explain what you see in the learning curves and confusion matrix.

1.1 Analysis of Learning Curves and Confusion Matrix

The learning curves display key training metrics over 10 epochs. The loss curves show how well the model learns, with decreasing values indicating improvement. If training and validation losses remain close together, this suggests the model generalizes well without overfitting. The precision and recall metrics measure prediction accuracy and coverage respectively, with higher values indicating better performance. The mAP (mean Average Precision) scores at different IoU thresholds provide comprehensive measures of object detection accuracy.

The confusion matrix reveals the model's classification performance across different object classes. Values along the diagonal represent correct predictions, while off-diagonal elements show misclassifications and which classes the model confuses with each other. The background class row indicates how well the model distinguishes between objects and empty regions.

Overall, training on COCO128 with just 10 epochs provides a quick demonstration of fine-tuning YOLOv10n. While this limited training shows the process, more epochs would likely improve performance. The confusion matrix is particularly useful for identifying which object classes pose the greatest challenges for detection.