Configuration Manual

MSc Research Project
Data Analytics

Soma Sekhar Bogisam
Student ID: x23127627@student.ncirl.ie

School of Computing
National College of Ireland

Supervisor: Christian Horn

National College of Ireland
Project Submission Sheet
School of Computing

Student Name:	Soma Sekhar Bogisam
Student ID:	x23127627@student.ncirl.ie
Programme:	Data Analytics
Year:	2024
Module:	MSc Research Project
Supervisor:	Christian Horn
Submission Due Date:	12/08/2024
Project Title:	Configuration Manual
Word Count:	387
Page Count:	7

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.

ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature:	Soma Sekhar Bogisam
Date:	12th August 2024

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST:

Attach a completed copy of this sheet to each project (including multiple copies).	☐
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	☐
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	☐

Assignments that are submitted to the Programme Coordinator office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

Configuration Manual

Soma Sekhar Bogisam
x23127627@student.ncirl.ie

1 Overview

This configuration manual contains of step-by-step instructions, hardware configuration and a list of essential software's to successfully reproduce the research project 'Underwater Plastic Detection Using YOLO V8 AND V10'.

2 System Specifications

This section lists out the the hardware and software setup on which research project was carried out.

2.1 Hardware Specifications

- Processor: 12th Gen Intel(R) Core(TM) i7-1255U 1.70 GHz
- RAM: 32 GB
- Storage: 1TB SSD

2.2 Software Specifications

- Operating System: Windows 11 Home, Version - 23H2
- Environment: Anaconda3 2023.09-0
- Anaconda Navigator: 2.6.0
- Python: 3.12.4

3 Environmental Setup

The research project was executed on Jupyter notebook in Anaconda environment. lick on start and select the Anaconda navigator and launch the Jupyter notebook.

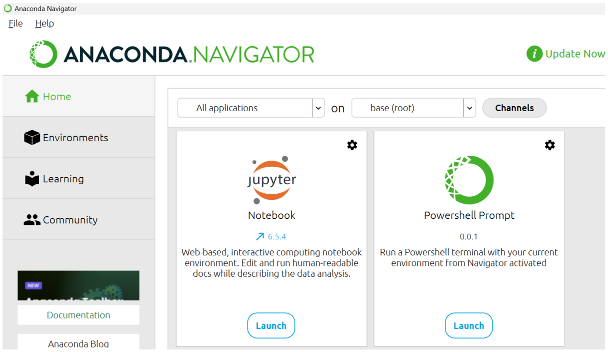

Figure 1: Launch Jupyter Notebook

Figure 2: Jupyter Notebook

4 Data Collection

Images containing small and medium-sized plastic debris were selected from roboflow website, ensuring a diverse representation of underwater conditions, debris types, and sizes.

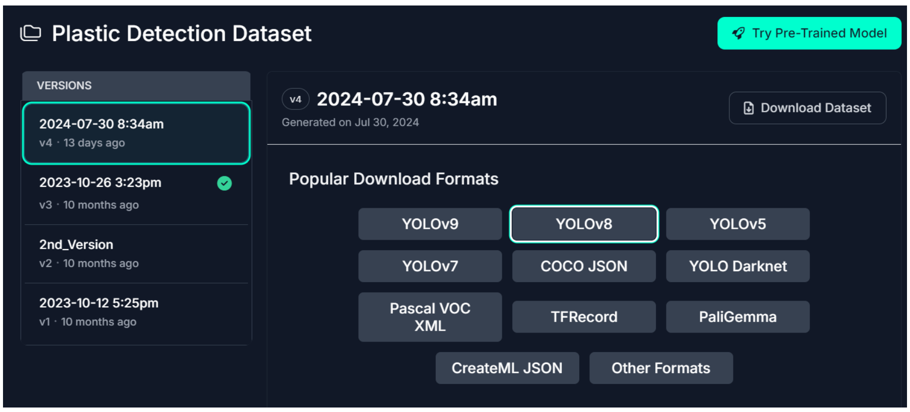

Figure 3: Deep Sea Plastic Images Dataset

5 Data Splitting

1200 images were selected to reduce redundancy out of total images and split the data into train(70%), test(10%) and valid(20%).

6 Installation of Libraries

The libraries necessary are installed using pip command and imported are utilized for successful execution of code.

Jupyter Train_Yolo_Models Last Checkpoint: 5 days ago

File Edit View Run Kernel Settings Help Trusted

Code JupyterLab Python 3 (ipykernel)

```
[10]: pip install ultralytics

Requirement already satisfied: ultralytics in c:\users\dell\anaconda3\envs\yolo_v10\lib\site-packages (8.1.34)
Requirement already satisfied: matplotlib>=3.3.0 in c:\users\dell\anaconda3\envs\yolo_v10\lib\site-packages (from ultralytics) (3.9.0)
Requirement already satisfied: opencv-python>=4.6.0 in c:\users\dell\anaconda3\envs\yolo_v10\lib\site-packages (from ultralytics) (4.10.0.84)
Requirement already satisfied: pillow>=7.1.2 in c:\users\dell\anaconda3\envs\yolo_v10\lib\site-packages (from ultralytics) (10.3.0)
Requirement already satisfied: pyyaml>=5.3.1 in c:\users\dell\anaconda3\envs\yolo_v10\lib\site-packages (from ultralytics) (6.0.1)
Requirement already satisfied: requests>=2.23.0 in c:\users\dell\anaconda3\envs\yolo_v10\lib\site-packages (from ultralytics) (2.32.2)
Requirement already satisfied: scipy>=1.4.1 in c:\users\dell\anaconda3\envs\yolo_v10\lib\site-packages (from ultralytics) (1.13.1)
Requirement already satisfied: torch>=1.8.0 in c:\users\dell\anaconda3\envs\yolo_v10\lib\site-packages (from ultralytics) (2.3.1)
Requirement already satisfied: torchvision>=0.9.0 in c:\users\dell\anaconda3\envs\yolo_v10\lib\site-packages (from ultralytics) (0.18.1)
Requirement already satisfied: tqdm>=4.64.0 in c:\users\dell\anaconda3\envs\yolo_v10\lib\site-packages (from ultralytics) (4.66.4)
Requirement already satisfied: psutil in c:\users\dell\anaconda3\envs\yolo_v10\lib\site-packages (from ultralytics) (5.9.0)
Requirement already satisfied: py-cpuinfo in c:\users\dell\anaconda3\envs\yolo_v10\lib\site-packages (from ultralytics) (9.0.0)
Requirement already satisfied: thop>=0.1.1 in c:\users\dell\anaconda3\envs\yolo_v10\lib\site-packages (from ultralytics) (0.1.1.post2209072238)
Requirement already satisfied: pandas>=1.1.4 in c:\users\dell\anaconda3\envs\yolo_v10\lib\site-packages (from ultralytics) (2.2.2)
Requirement already satisfied: seaborn>=0.11.0 in c:\users\dell\anaconda3\envs\yolo_v10\lib\site-packages (from ultralytics) (0.13.2)
Requirement already satisfied: contourpy>=1.0.1 in c:\users\dell\anaconda3\envs\yolo_v10\lib\site-packages (from matplotlib>=3.3.0->ultralytics) (1.2.1)
Requirement already satisfied: cycler>=0.10 in c:\users\dell\anaconda3\envs\yolo_v10\lib\site-packages (from matplotlib>=3.3.0->ultralytics) (0.12.1)
Requirement already satisfied: fonttools>=4.22.0 in c:\users\dell\anaconda3\envs\yolo_v10\lib\site-packages (from matplotlib>=3.3.0->ultralytics) (4.53.
0)
Requirement already satisfied: kiwisolver>=1.3.1 in c:\users\dell\anaconda3\envs\yolo_v10\lib\site-packages (from matplotlib>=3.3.0->ultralytics) (1.4.
5)
Requirement already satisfied: numpy>=1.23 in c:\users\dell\anaconda3\envs\yolo_v10\lib\site-packages (from matplotlib>=3.3.0->ultralytics) (1.26.4)
Requirement already satisfied: packaging>=20.0 in c:\users\dell\anaconda3\envs\yolo_v10\lib\site-packages (from matplotlib>=3.3.0->ultralytics) (23.2)
Requirement already satisfied: pyparsing>=2.3.1 in c:\users\dell\anaconda3\envs\yolo_v10\lib\site-packages (from matplotlib>=3.3.0->ultralytics) (3.1.2)
Requirement already satisfied: python-dateutil>=2.7 in c:\users\dell\anaconda3\envs\yolo_v10\lib\site-packages (from matplotlib>=3.3.0->ultralytics) (2.
9.0.post0)
Requirement already satisfied: pytz>=2020.1 in c:\users\dell\anaconda3\envs\yolo_v10\lib\site-packages (from pandas>=1.1.4->ultralytics) (2024.1)
Requirement already satisfied: tzdata>=2022.7 in c:\users\dell\anaconda3\envs\yolo_v10\lib\site-packages (from pandas>=1.1.4->ultralytics) (2024.1)
Requirement already satisfied: charset-normalizer<4,>=2 in c:\users\dell\anaconda3\envs\yolo_v10\lib\site-packages (from requests>=2.23.0->ultralytics)
(2.0.4)
```

Figure 4: Installation Of Libraries

Importing Libraries

```python
import os
from glob import glob
from ultralytics import YOLO
import cv2
import numpy as np
import matplotlib.pyplot as plt
from PIL import Image
import tensorflow as tf
from tensorboard.backend.event_processing.event_accumulator import EventAccumulator
```

Figure 5: Importing Libraries

7 Path Configuration

Configure the data.yaml file according to the train,test and valid paths. The below image shows the configured yaml file and also the code to display the number of images,label files and annotations in train, test and valid paths.

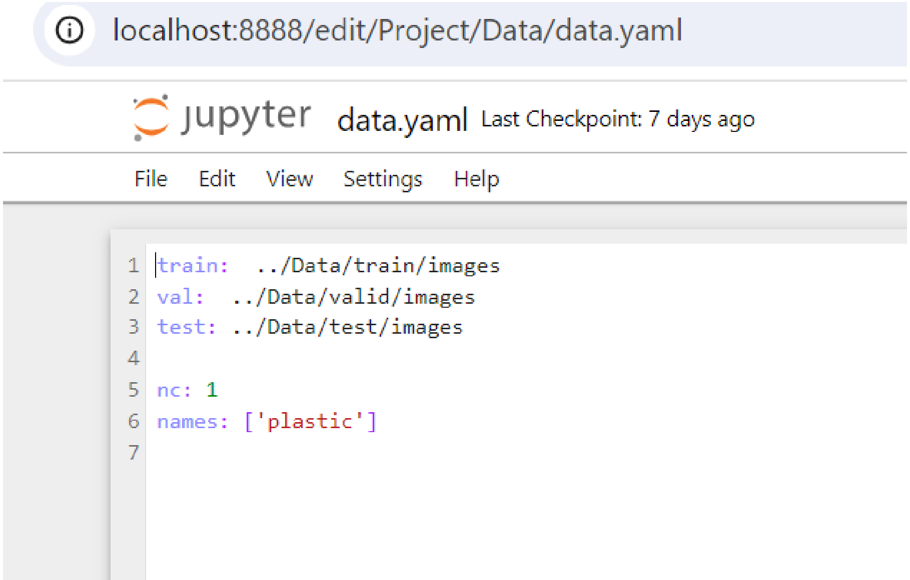

Figure 6: Configuration of yaml file

```
[1]: import os
from glob import glob

def count_images_and_annotations(image_dir, label_dir):
    image_extensions = ['*.jpg', '*.jpeg', '*.png', '*.bmp']
    image_files = []
    for ext in image_extensions:
        image_files.extend(glob(os.path.join(image_dir, '**', ext), recursive=True))

    label_files = glob(os.path.join(label_dir, '**', '*.txt'), recursive=True)

    annotation_count = 0
    for label_file in label_files:
        with open(label_file, 'r') as f:
            annotation_count += len(f.readlines())

    return len(image_files), len(label_files), annotation_count

# Paths

base_path = '../Data/'
train_images = os.path.join(base_path, 'train/images')
train_labels = os.path.join(base_path, 'train/labels')
test_images = os.path.join(base_path, 'test/images')
test_labels = os.path.join(base_path, 'test/labels')
valid_images = os.path.join(base_path, 'valid/images')
valid_labels = os.path.join(base_path, 'valid/labels')


# Count for training set
train_images_count, train_labels_count, train_annotations_count = count_images_and_annotations(train_images, train_labels)

# Count for test set
test_images_count, test_labels_count, test_annotations_count = count_images_and_annotations(test_images, test_labels)

# Count for valid set
valid_images_count, valid_labels_count, valid_annotations_count = count_images_and_annotations(valid_images, valid_labels)

print(f"Training set: {train_images_count} images, {train_labels_count} label files, {train_annotations_count} annotations")
print(f"Test set: {test_images_count} images, {test_labels_count} label files, {test_annotations_count} annotations")
print(f"Valid set: {valid_images_count} images, {valid_labels_count} label files, {valid_annotations_count} annotations")

Training set: 843 images, 843 label files, 985 annotations
Test set: 118 images, 118 label files, 118 annotations
Valid set: 236 images, 236 label files, 271 annotations
```

Figure 7: Total Instances of data

8 Implementation

To train the Yolo models for both V8 (s,l) and V10 (s,l), the parameters given are batch size : 16, image size: 640*640 pixels, epochs = 75, Optimizer = Adam, Initial learning rate = 0.001 and device = 'cpu'.

8.1 Training & Validation of Yolo Models

```
•[2]: from ultralytics import YOLO
      import os

      def train_yolov8(model_size, data_yaml, epochs, imgsz, batch_size, save_dir):
          # Load the model
          model = YOLO(f'yolov8{model_size}.pt')

          # Train the model
          results = model.train(
              data=data_yaml,
              epochs=epochs,
              imgsz=imgsz,
              batch=batch_size,
              project=save_dir,
              name=f'yolov8{model_size}_underwater_plastic',
              pretrained=True,
              optimizer='Adam',
              lr0=0.001,
              patience=50,
              save=True,
              save_period=10,
              device='cpu'  #Use '0' for GPU
          )

          # Validate the model
          results = model.val()

          print(f"Training and validation complete for YOLOv8{model_size}")

      # Parameters
      base_path = '../Data/'
      data_yaml = os.path.join(base_path, 'data.yaml')
      epochs = 1 #75 for full evaluation
      imgsz = 640
      batch_size = 16 #16
      save_dir = '../Output/'
```

Train YOLOv8s

```
[3]: train_yolov8('s', data_yaml, epochs, imgsz, batch_size, save_dir)

     Starting training for 1 epochs...
```

Figure 8: Training of Yolo Model

```
Validating ..\Output\yolov8s_underwater_plastic\weights\best.pt...
Ultralytics YOLOv8.1.34 🚀 Python-3.12.4 torch-2.3.1+cpu CPU (12th Gen Intel Core(TM) i7-1255U)
Model summary (fused): 168 layers, 11125971 parameters, 0 gradients, 28.4 GFLOPs
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100%|██████████| 8/8 [03:31<00:00, 26.48s/it]
                   all        236        271      0.583      0.384      0.424      0.226
Speed: 5.5ms preprocess, 871.8ms inference, 0.0ms loss, 4.7ms postprocess per image
Results saved to ..\Output\yolov8s_underwater_plastic
Ultralytics YOLOv8.1.34 🚀 Python-3.12.4 torch-2.3.1+cpu CPU (12th Gen Intel Core(TM) i7-1255U)
Model summary (fused): 168 layers, 11125971 parameters, 0 gradients, 28.4 GFLOPs
```

Figure 9: Validation of Yolo Model

8.2 Inference of Yolo Models

Inference performed for Yolo V8(s,l) and Yolo V10(s,l) variants on a single image for comparing detecting capabilities and accuracies.

```
# Specify the paths to your trained models
model_paths = [
    '../Output/yolov8s_underwater_plastic/weights/best.pt',
    '../Output/yolov8l_underwater_plastic/weights/best.pt',
    '../Output/yolov10s_underwater_plastic/weights/best.pt',
   '../Output/yolov10l_underwater_plastic2/weights/best.pt'
]

# Specify the paths to the images you want to perform inference on
image_paths = [
    '../Data/test/images/obj0002_frame0000060_jpg.rf.106f6fe4454c119e125e445f9c0b2bb5.jpg'
    ,'../Data/test/images/obj0007_frame0000016_jpg.rf.1a6b27d9cd6a7cb0607b3550df79fcb8.jpg'
]

inference_images_multiple_models(model_paths, image_paths)
```

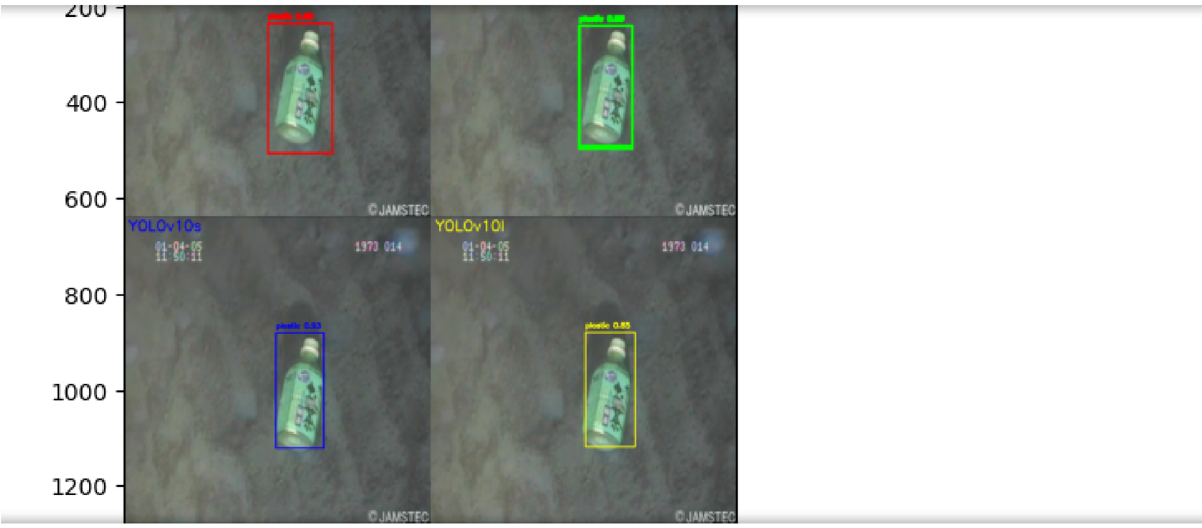

Figure 10: Inference

8.3 Evaluation & Comparison of YOLO Models

The below bar graph depicts the comparison of precision, recall, map50 and map50_95 metrics for all Yolo variants.

YOLO MODELS PERFORMANCE COMPARISON

```python
import matplotlib.pyplot as plt
import numpy as np

# Data for plotting
models = ['YOLOv8-s', 'YOLOv8-l', 'YOLOv10-s', 'YOLOv10-l']
box_p = [0.941,  0.891, 0.931, 0.896]
box_r = [0.664, 0.662, 0.653, 0.636]
map50 = [0.759, 0.733, 0.772, 0.725]
map50_95 = [0.556, 0.526, 0.556, 0.529]

x = np.arange(len(models))  # the label locations
width = 0.2  # the width of the bars

fig, ax = plt.subplots(figsize=(12, 6))
rects1 = ax.bar(x - 1.5*width, box_p, width, label='Box(P)')
rects2 = ax.bar(x - 0.5*width, box_r, width, label='Box(R)')
rects3 = ax.bar(x + 0.5*width, map50, width, label='mAP50')
rects4 = ax.bar(x + 1.5*width, map50_95, width, label='mAP50-95')

# Add some text for labels, title and custom x-axis tick labels, etc.
ax.set_xlabel('Models')
ax.set_ylabel('Scores')
ax.set_title('YOLO Models Performance Comparison')
ax.set_xticks(x)
ax.set_xticklabels(models)
ax.legend()

fig.tight_layout()

# Display plot
plt.show()
```

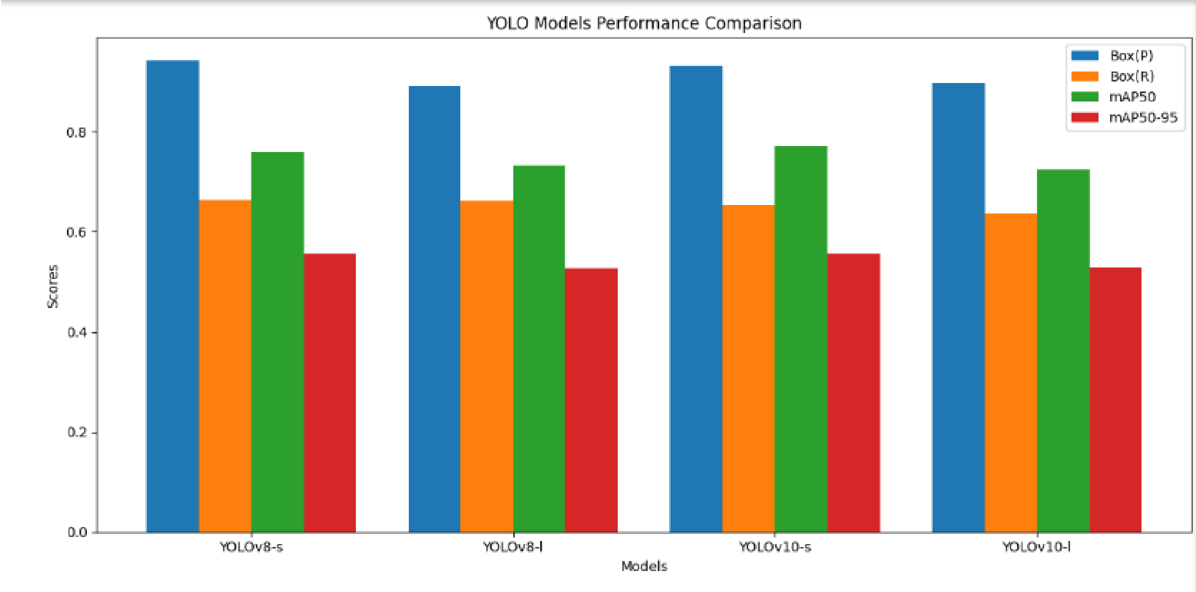

Figure 11: Yolo Metrics Comparison