National College of Ireland

Configuration Manual

MSc Research Project
MSc Data Analytics

Syed Ahmed Omer
Student ID: 22132295

School of Computing
National College of Ireland

Supervisor: Abdul Shahid

National College of Ireland

MSc Project Submission Sheet

School of Computing

Student Name:	Syed Ahmed Omer
Student ID:	22132295
Programme:	Msc Data Analytics **Year:** 2023
Module:	Msc Research Project
Lecturer:	Abdul Shahid
Submission Due Date:	14-12-2023
Project Title:	Euro Coin Recognition using YOLOv7
Word Count:	1092 **Page Count:** 6

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:	Syed Ahmed Omer
Date:	13-12-2023

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

Syed Ahmed Omer
Student ID: 22132295

1 Requirement Guide

Please follow the following steps to run the YOLOv7(Wang et al., 2022) code that was used in the research project. This document will discuss all the requirements that are required to run the code from scratch.

2 Machine Hardware Requirement

It is to be noted that the Deep Learning model requires GPU to train faster, using CPU could take a very long time, so it is recommended to use GPU while training the model. Google Colab was used for training the model as free GPU resources are available in Colab. Following are the details of the GPU used and the Processor and CPU.

GPU Model: Tesla T4
Processor: Intel(R) Xeon(R)
CPU @ 2.20GHz CPU Cores: 2

3 Machine Software Requirement

The Software Requirements used for the project are as follows:

- Environment used to run the code = Google Colab
- Google Drive is linked to the Colab to save the files.
- Programming language used = Python, version (3.10.12)
- Code was written in Jupyter Notebook format '.ipynb' files
- Other software includes Microsoft Excel to write the summary of research papers, overleaf to write the Report, and Microsoft Word to write the configuration file.

4 Environment Setup

Firstly, to train the YOLOv7 model the YOLOv7 repository is to be downloaded from GitHub, the link for the GitHub is: https://github.com/WongKinYiu/yolov7.git
It can directly be downloaded or

- "!git clone https://github.com/WongKinYiu/yolov7.git" command can be used in the colab notebook to clone the repo in colab environment.

Once the repository is downloaded, the next step is to change the directory inside the YOLOv7 repository then the dependencies are to be installed in the Python environment.

The below command is run in the colab environment, which downloads all the dependencies required to run this model.

```
!pip install -r requirements.txt
```

4.1 Libraries Required.

Below are all the required dependencies and libraries with their version as shown in the requirement.txt file in the below figure 1.

```
# Usage: pip install -r requirements.txt

# Base ------------------------------------------
matplotlib>=3.2.2
numpy>=1.18.5,<1.24.0
opencv-python>=4.1.1
Pillow>=7.1.2
PyYAML>=5.3.1
requests>=2.23.0
scipy>=1.4.1
torch>=1.7.0,!=1.12.0
torchvision>=0.8.1,!=0.13.0
tqdm>=4.41.0
protobuf<4.21.3

# Logging ---------------------------------------
tensorboard>=2.4.1
# wandb

# Plotting --------------------------------------
pandas>=1.1.4
seaborn>=0.11.0

# Export ----------------------------------------
# coremltools>=4.1  # CoreML export
# onnx>=1.9.0  # ONNX export
# onnx-simplifier>=0.3.6  # ONNX simplifier
# scikit-learn==0.19.2  # CoreML quantization
# tensorflow>=2.4.1  # TFLite export
# tensorflowjs>=3.9.0  # TF.js export
# openvino-dev  # OpenVINO export
```

Figure-1

Now, we have all the libraries that are required to train the model installed and our environment setup is completed.

5 Dataset:

The Euro Coins dataset that was used for this research can be found on the GitHub link, GitHub - SuperDiodo/euro-coin-dataset: Euro coin dataset used in Roboflow for object detection in Tensorflow. This is an open-source dataset, which can be used for any project.

The dataset contains the folders for each class of coins, within each folder the corresponding class of images and their labeled text files can be found.

We uploaded this dataset along with labels to Roboflow, then we checked the labeling quality of the dataset, it was found the ground truth was accurate and correctly labeled, Figure 2 shows an example of checking the labeled images on Roboflow.

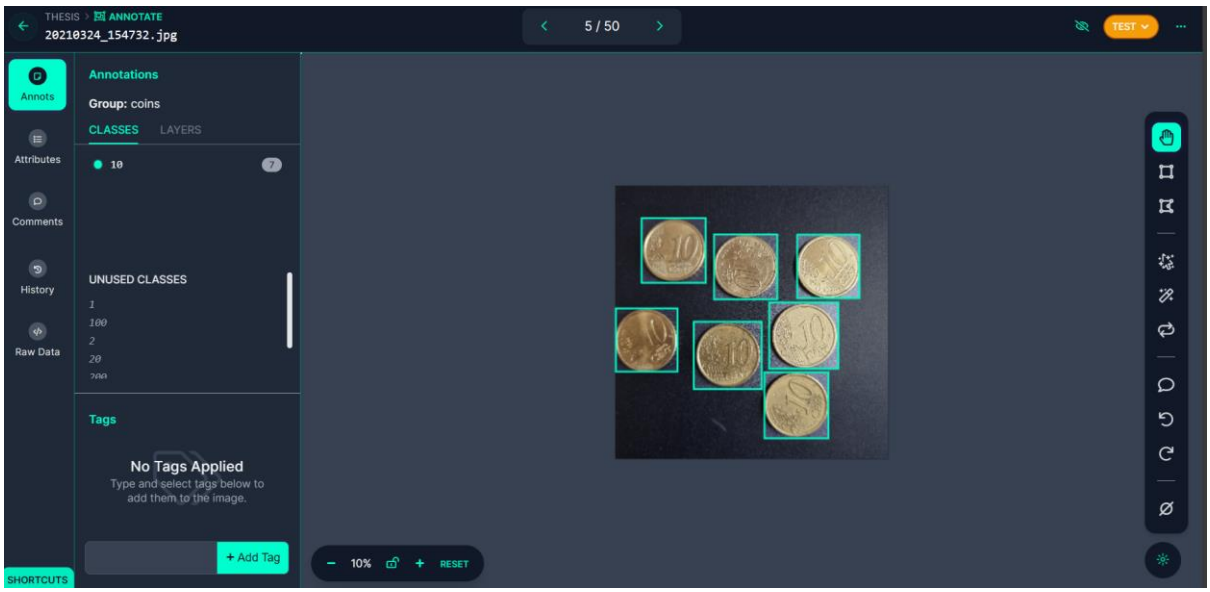

Figure-2

Now, that we have exported this data, The data was split into the train, test, and valid in the ratio of 70-20-10 as shown in the figure 3, then some preprocessing steps were applied which are essential for the proper orientation of images and bounding boxes and model training, figure 4 shows the preprocessing applied in the Roboflow.

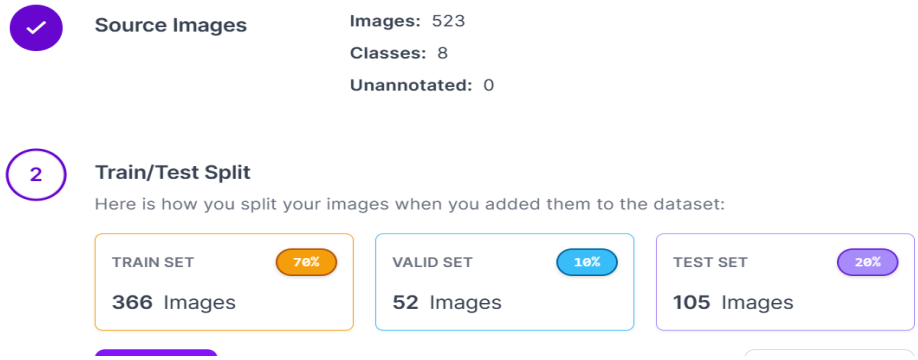

Figure-3

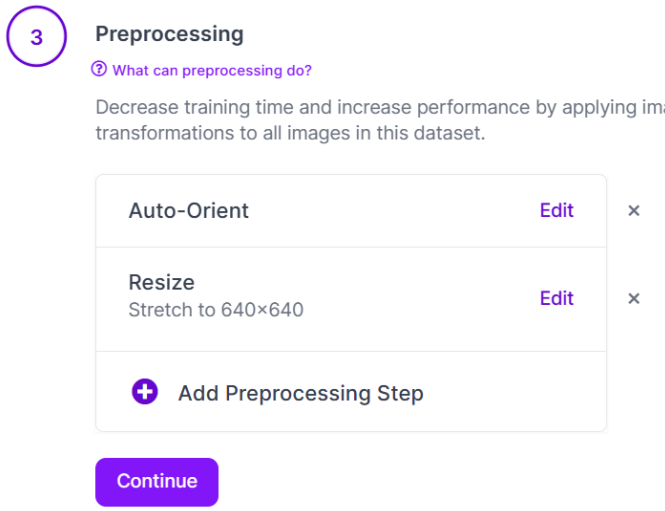

Figure-4

Next, we applied some augmentation techniques to the dataset which can be shown in figure 5. Which increases the size of train data and prevent the model from over fitting.

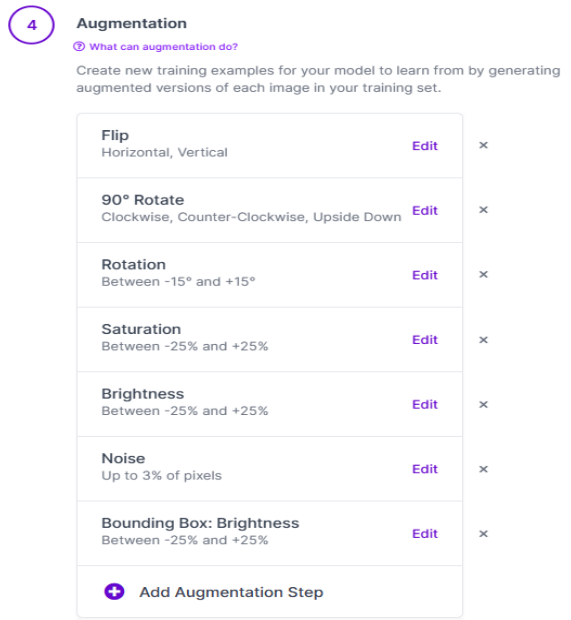

Figure 5

Then we export the data in YOLOv7 format from the Roboflow, then the code snippet shown in Figure 6 is generated, which can be used to download the dataset in Colab by just running the code in the Colab cell.

```
!pip install roboflow

from roboflow import Roboflow
rf = Roboflow(api_key="ncGSEcE9WEou5vvd4SrF")
project = rf.workspace("dhbw-mannheim").project("eurocoindec")
dataset = project.version(1).download("yolov7")
```

Figure 6

Now by running the code snippet in figure-6, we get the data downloaded. Now we have our dataset ready in Colab to be used for training.

6 Training the model:

We start by downloading the COCO weights of YOLOv7 as we are doing transfer learning, and then we will tune these weights for our use case. The code snippet in Figure 7 is used to download the weights in the repository.

```
%cd /content/yolov7
!wget https://github.com/WongKinYiu/yolov7/releases/download/v0.1/yolov7_training.pt
```

Figure 7

Now we modify the configuration file which is coco,yaml as per our use case, we give the path of train, test, and valid data and define number of classes and their names, which is shown in Figure 8.

```
#train and val data as 1) directory: path/images/, 2) file: path/im
train: D:\NCI\Thesis_code\data_aug_exp\data\train
val: D:\NCI\Thesis_code\data_aug_exp\data\valid
val: D:\NCI\Thesis_code\data_aug_exp\data\test


nc: 8
names: ['1', '10', '100', '2', '20', '200', '5', '50']
```

Figure 8

Now, we are ready to start the training of the model, the command below in figure 9 is used to run the training, where we give batch size, number of epochs, and path to coco.yaml file and the weights we downloaded and device is ‘0’ when we trian on GPU.

```
!python train.py --batch 16 --epochs 100 --data /content/yolov7/data/coco.yaml --weights 'yolov7_training.pt' --device 0
```

Figure 9

Once the model is done running, the weight of the model is saved as “best.pt” in the runs folder, which is to be saved in Google Drive because once runtime is expired all the data is lost on Colab. Now, to evaluate the model same command with test.py is run in the cell, with the weights path pointing to our “best.pt” model obtained from training for 100 epochs. This will generate all the graphs and predictions for the test dataset.

7 Code and artifact submitted:

This Section describes the directory structure of the code and Artifact file submitted and the use of the code file:
Directory structure:

1. Images_used_in_the_report: This folder contains all the images that were generated during the research and are used in the report.

2. Yolo_running_scripts: This folder has the following files.
    a. Start training yolov7: This code was used to train the yolov7 model, this contains the code snippets which are described in the configuration manual.
    b. Check_data_quality: This piece of code was used to draw the bounding boxes around the image to check the quality of labeling on the dataset.
    c. Data_split_check: This piece of code was used to generate the distribution of classes in the train, test, and valid data, the images obtained are used in the report.
    d. Calculating_total: This piece of code is used to calculate the total amount of coins in an image from the predictions obtained from the yolov7 model, this is mentioned in future works in the paper.
3. Yolov5_repo: This is the YOLOv5 repository which contains the results from the mode on test data after training, which can be found in "runs/val/data2_test_data" inside the repository.
4. Yolov7_repo: This is the YOLOv5 repository which contains the results from the v7 model on test data after training it, the results can be found in "runs/test/yolov7_data1_final"

8 References:

Wang, C.-Y., Bochkovskiy, A., Liao, H.-Y.M., 2022. YOLOv7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors [WWW Document]. arXiv.org. URL https://arxiv.org/abs/2207.02696v1 (accessed 12.12.23).