Configuration Manual

MSc Research Project
Data Analytics

AKSHAY MENON
Student ID: x21173036

School of Computing
National College of Ireland

Supervisor:  Dr.Paul Stynes Dr.Pramod Pathak
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: |   |
| Date:      | 1st February 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): |
1 Initial Environment Setting

The software specification for the setup are described below in the Figure 1. :-

<table>
<thead>
<tr>
<th>Programming language</th>
<th>Python (v3.9)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cloud Platform</td>
<td>Google drive for storage</td>
</tr>
<tr>
<td>Device configuration</td>
<td>Intel(R) Core(TM) i5-1035G1 CPU @ 1.00GHz 1.19 GHz</td>
</tr>
<tr>
<td>GPU UTILIZATION</td>
<td>K 80 with 2xvCPU of 12 GB</td>
</tr>
<tr>
<td>CPU UTILIZATION</td>
<td>Intel(R) Core(TM) i5-1035G1 CPU @ 1.00GHz 1.19 GHz With 8GB</td>
</tr>
<tr>
<td>IDE</td>
<td>Jupyter Notebook, Google Colab</td>
</tr>
<tr>
<td>Image Labelling Tool</td>
<td>Microsoft Visual Object Tagging Tool</td>
</tr>
</tbody>
</table>

Figure 1: Environment Configuration

Once the environment with all valid specification has been setup we will be needing Jupyter notebook and Google Colab to run first module of the reseach project which is shuttlecock tracking.

1.1 Initiating Environment

Let us first initiate by running module on jupyter notebook.
1. Install and import all libraries for running shuttlecock tracking module which requires frame to image conversion as pre-requisite in Fig 2.

```
pip install argparse
pip install numpy
pip install matplotlib
pip install pillow
pip install scipy
pip install pydot
pip install keras
pip install tensorflow-gpu
pip install opencv-python
```

Figure 2: Model Libraries
2. The Frames from the video are converted into png format through the script in Fig 3.

```python
import cv2
import os
import sys
import shutil

try:
    videoName = sys.argv[1]  # Input path
    outputPath = sys.argv[2]  # Output path
    if (not videoName) or (not outputPath):
        raise Exception
    print('Usage: python3 Frame_Generator.py <videoPath> <outputFolder> -v')
except:
    exit()

if outputPath[-1] != '/':
    outputPath += '/'

if os.path.exists(outputPath):
    shutil.rmtree(outputPath)

os.makedirs(outputPath)

# Segment the video into frames
success, count = True, 0

while success:
    cv2.imwrite(outputPath + '%d.png' % (count), image)
    count += 1
    success, image = cap.read()
```

Figure 3: Frame generation

3. The generated frames are stored in local drive which is used for image labelling using Microsoft visual object tagging tool and saving output in csv format which has coordinates of the shuttlecock in the frame in Fig 4.

Figure 4: Image labelling

4. The labelled images are used for annotation using Heat Map prediction in Fig 5.
Figure 5: Export to CSV

Figure 6: Heat Map Prediction

```python
#1. Create heatmap as Ground Truth
import glob
import csv
import numpy
import matplotlib.pyplot as plt
import os

size = 20
# create gaussian heatmap:
def gaussian_kernel(variance):
    x, y = numpy.mgrid[-size:size+1, -size:size+1]
    g = numpy.exp((x**2 + y**2)/2 * float(2 * variance))
    return g

# make the Gaussian by calling the function
variance = 10
gaussian_kernel_array = gaussian_kernel(variance)
# rescale the value to 0-255
print(int((len(gaussian_kernel_array)-1)/2))
gaussian_kernel_array = gaussian_kernel_array * 255/gaussian_kernel_array[int((len(gaussian_kernel_array)-1)/2)][int((len(gaussian_kernel_array)-1)/2)]
# change type as integer
gaussian_kernel_array = gaussian_kernel_array.astype(int)

# show heatmap
plt.imshow(gaussian_kernel_array, cmap=plt.get_cmap('gray'), interpolation='nearest')
plt.colorbar()
```
5. The labelled Csv file and Heat map prediction are used to generate training and testing numpy array file in Fig 6.

6. The training of the shuttlecock will be done on google colab with K8 GPU memory for the same once initiating Google colab notebook change runtime to GPU [Kosaian et al. (2021)] and mount drive on your google drive in Fig 7 and Fig 8.

7. Upload NPY from local device to google drive for training purpose in Fig 9.

8. From the Google colab run the TrackNet model script in Fig 10.

9. The Model training script needs to be run using labelled csv files and Numpy array files at epoch 30 with tolerance 4 with folder for saving weight in Fig 11.

10. The trained images are predicted using saving weight for the accuracy script has to be run in Fig 12.
Figure 8: GPU

Figure 9: Mount Google Drive

Figure 10: Npy file uploading to G drive
Figure 11: TrackNet Model loading

```python
from keras.models import *
from keras.layers import *
from keras.activations import *

def TrackNet( input_height, input_width ): #input_height = 288, input_width = 512
    inputs = Input(shape=(9,input_height,input_width))
```

Figure 12: Training Images

```python
try:
    (opts, args) = getopt.getopt(sys.argv[1:], '', [
        'load_weights=',
        'save_weights=',
        'dataDir=',
        'epochs=',
        'tol='
    ])
    if len(opts) < 4:
        raise
except:
    print('''usage: python3 train_TackNet.py --load_weights<previousWeightPath> --save_weights<newWeightPath> -
    print('argument --load_weights is required only if you want to retrain the model')
    exit()''')

paramCount=
    'load_weights='
```

Figure 13: Model Accuracy Prediction

```bash
python3 '/content/TrackNetV2/3_in_1_out/accuracy.py' --load_weights='/content/TrackNetV2/3_in_1_out/model_33
```

```
2022-12-14 23:36:44.692325: I tensorflow/core/platform/cpu_feature_guard.cc:195] This TensorFlow binary is opt 
To enable them in other operations, rebuild TensorFlow with the appropriate compiler flags.
2022-12-14 23:36:45.629644: W tensorflow/compiler/tf2/python/utils/cc38] TF-TRT Warning: Cannot di 
2022-12-14 23:36:45.671906: W tensorflow/core/common_runtime/gpu/bfc_allocator.cc:42] Overriding orig_v 
Beginning evaluating.....

================================================================
156/156 [%%%%%%%%%%%%%%%%%%]. 135 Sims/step
Finish evaluating data: (TP, TN, FP1, PP2, FN)=(134, 7, 1, 3, 11)
================================================================
Number of true positive: 134
Number of true negative: 7
Number of false positive PP1: 1
Number of false positive PP2: 3
Number of false negative: 11
accuracy: 0.9038461538461539
```
2 Initiating Environment for Player Service Fault Detection

1. The Machine learning framework for player pose detection need to install following libraries including google mediapipe pose [Lugaresi et al. (2019)] in fig 14.

![Figure 14: Media pipe Library](image)

2. Post implementing relevant libraries Youtube video is uploaded into the script using OpenCV to read frame and generate body landmark in fig 15.

```python
fig = cv2.VideoCapture('D:\research\UMAR-UM\F h e a  n g\scripts\mp_pose_video\video_frames\Recording Video.mp4')
if not fig.isOpened():
    print('Error opening video stream or file')
while fig.isOpened():
    ret, frame = fig.read()
    if not ret:
        break
    # Process image to draw
    image = cv2.cvtColor(frame, cv2.COLOR_RGB2BGR)
    image.flags.writeable = False
    # Make detection
    results = pose.process(image)
    # Process back to draw
    image.flags.writeable = True
    image = cv2.cvtColor(image, cv2.COLOR_RGB2BGR)
    # Extract landmarks
    try:
        landmarks = results.pose_landmarks.landmark
        for landmark in landmarks.landmark:
            print(landmarks)
    except:
        print('No landmarks found')

fig.release()
```

![Figure 15: mediapipe script](image)

3. We have made logical expression for calculating angle from mediapipe script in fig 16.
4. Body Landmark is generated and recorded in fig 17.
5. The generated landmark in the form of data frame undergoes training in fig 18.


7. The best model has been selected as Random Classifier which will be used for saving model and for predicting outcome of the machine learning framework in fig 20.
Figure 19: Best Model Fit

Figure 20: Test using Saved model
References
