Module Exam

Module code and Name	DE4-SIOT Sensing & IoT
Student CID	01353594
Student name	Aida Manzano Kharman
Assessment date	4pm 14th Jan 2021

Presentation URL (publicly accessible link):
Demo: https://www.youtube.com/watch?v=OaJr2Ap-1ss&feature=youtu.be

YOLOv4 weights real time prediction: https://youtu.be/3LaPwffMKmU

Code & Data (publicly accessible link):
Code is not publicly accessible as it contains personal data, the examiner has been made a collaborator
https://github.com/aidamanzano/IOT_Coursework

Coursework 1: Sensing

1. Introduction

The correlation between mood and sleep is not a new field of study, and there are numerous well documented studies that show the relationship between mood and sleep cycles. Some demonstrate the importance of sleep-wake rhythms in mental health disorders and prove that there is a significant correlation between triggering or augmenting mania and sleep loss [1]

Other studies have found significant correlations between negative mood and subjective sleep quality and have established a association between the two factors, as well as identifying poor sleep quality as an early recognition symptom of mania and depression in bipolar patients [2]

Mood and activity are also correlated, and this has been asserted by a high number of academic literature and medical studies that have proven the positive effect of a reasonable level of exercise in reducing the likelihood of depression, mania and low mood. [3], [4], [5], [6]

I'm interested in finding out if I can demonstrate this correlation on data of my own physical activity, sleep levels and mood.

2. Objectives

There are a number of objectives to achieve upon the completion of this project:

1. Finding out if there is a correlation between my mood, sleep and step count.
2. Creating an interactive and compelling way of visualising the data of my sleep steps and mood
3. Creating an application that can fuse the data visualisation with the image recognition algorithm that can return predictions of the user's mood in real time
4. Apply the techniques learnt in the Sensing and Internet of Things Module about data sensing, sampling, data correlation, storage and interaction with the data into an integrated working application.

3. Planning

There where two main dependency tasks that would draw out the length of the project. The first one was the training of the image recognition algorithm, and the second one was the data collection. However, because I could work on other tasks simultaneously to wearing the Fitbit for data collection, the first task was to set up all the requirements for the image recognition training. This involved first and foremost finding a good quality dataset that I could use. Upon research, I came across the AffectNet dataset, which contains about 1M facial images collected from the Internet by querying three major search engines using 1250 emotion related keywords in six different languages. [7] I submitted the request to use the AffectNet Dataset from University of Denver on the 29/10/2020 and received access on the 23/11/2020 and it took a little less than a week to download, as it was over 100Gb of size.

I could not download it to my local computer so I purchased a hard drive that took another week to arrive. I was able to start pre-processing the dataset by mid December.

As expected, the labels that the dataset had were not in the format required to train the image recognition algorithm YOLOv3 and as such, I had to do extensive pre processing in order to generate the necessary files and labels to train. These will be described in further detail in section 5.2

Prior to this, the data sensing set up was done relatively early on in the project, because I already owned the Fitbit and setting up the API to download the data of my sleep and steps was a straightforward task. I tested the set up by downloading the data and generating csv files of the relevant attributes and automating their saving locally onto my laptop. This is why these milestones are not shown in Figure 1, because in fact they were completed in mid November and the duration of the project is too long to include in one image. The project plan file is included in the Github repository with the rest of the code where the milestones can be seen in more depth.

As is shown in Figure 1, the project had three main tasks: the image recognition code, the data collection and processing, and finally the user interface. The data collection was the first task I began, and then proceeded with the image recognition section, to conclude on the user interface, which was the integration of the two previous sections.

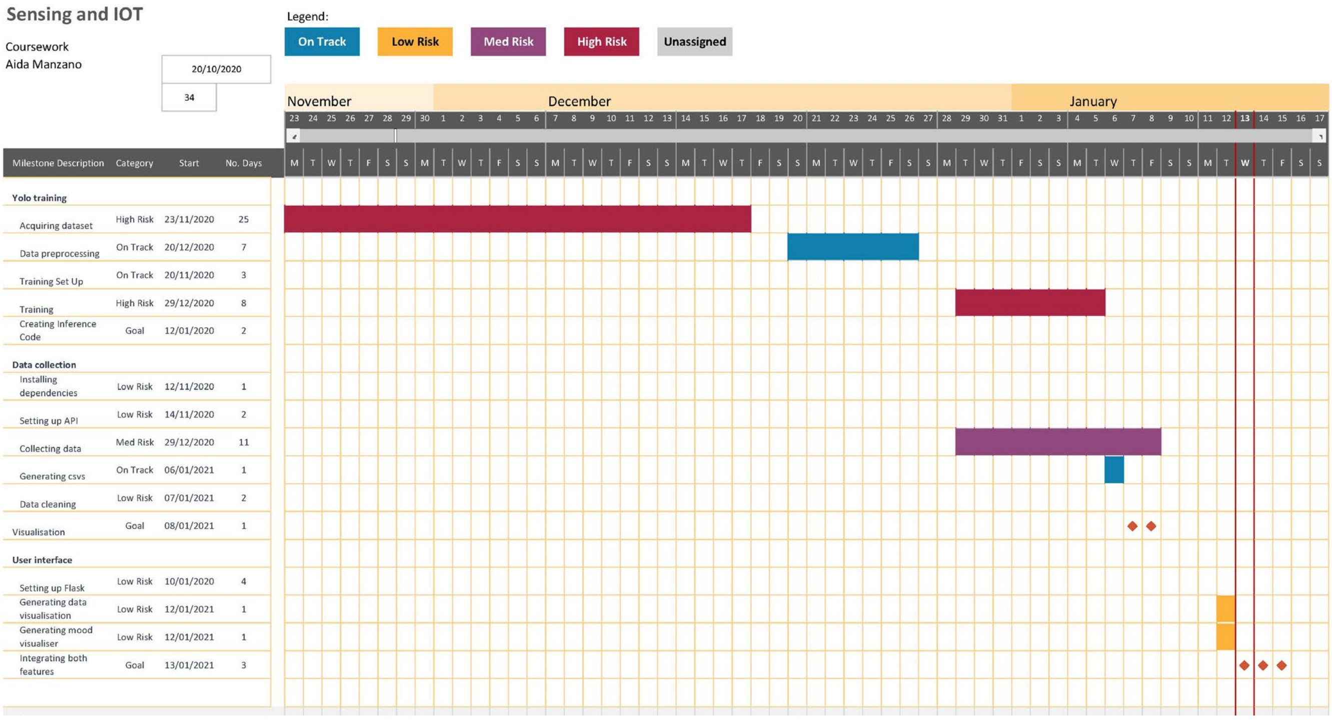


Figure 1: Section of Project Plan

4. Data Sources and sensing set-up

The data sources came entirely from myself, as I was measuring my own sleep, steps and facial expressions. I wore a Fitbit Ionic for a week, which had the sensors to monitor my step count at every minute and my sleep phases throughout the night. These varied from light sleep, deep sleep, REM sleep, restless and awake. This provided a dimension to the sleep data, as opposed to it being binary between awake or asleep. The set up was simple, as it was entirely relied on the Fitbit sensors, which are precise and accurate and have a high sampling rate.

The sampling rate for both the sleep phase and the number of steps was every minute throughout the day and night, but upon analysing the data this was collapsed to hourly sampling. Minute sampling, whilst provided many datapoints, seemed redundant as often the step count value would be 0 and I would have many empty data points of mood throughout the hours I was sleeping. Hourly sampling was more appropriate, as this was the sampling rate used to collect photos of myself throughout the waking hours as well as reducing the amount of noise in the dataset.

The sensors that the Fitbit Ionic uses to measure data are: 3-axis accelerometer, 3-axis gyroscope, Optical heart rate monitor, Altimeter and Ambient light sensor. [8]

Fitbit have been quoted to have spent a high amount of resources on improving both the hardware sensors through prototyping and the algorithms they use to interpret the sensed data. [9] Because these algorithms are proprietary, I did not have access to them, but it could be observed from my own Fitbit Ionic that there is an optical heart rate monitor with three photo diodes and one LED set in the centre.

Figure 2: Fitbit Ionic Sensors (Credits: Engadget)

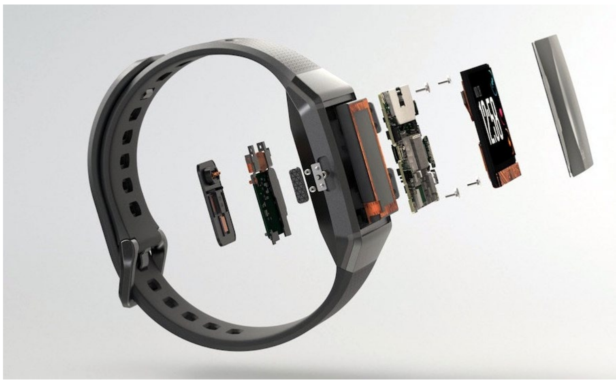


Figure 3: Fitbit Teardown (Credit: Fonearena)

5. Data collection and storage process

The data collection was carried out using the third party python-fitbit API. This is a well documented API which allowed me to have the data collection tested and trialled relatively early into the project.

Firstly, I set up my Fitbit account, and then I registered an App under the dev.fitbit.com site.

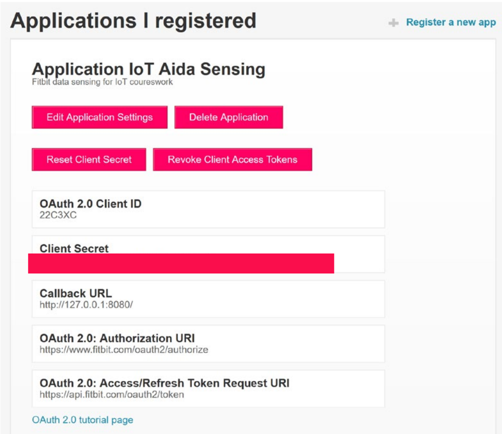


Figure 4: Fitbit Account Registered App

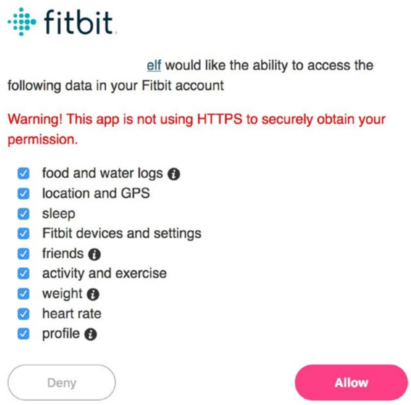


Figure 5: Access to the Fitbit account through unofficial python API

Using the OAuth 2.0 authentication type allows the third party Fitbit API to access my Fitbit account without having to share my Fitbit account password. Once this step was completed and the third party Fitbit API was installed the next step was to utilise the Client ID and Client Secret to access the app I registered and be able to access and refresh the tokens to download the data. [10] This code can be found under IOT_Coursework/Data Collection Scripts/Fitbit parser and API calls/ of the Github repo.

Once this was running, I granted access myself access to all the available data, despite only wanting to collect sleep and steps.

I then used the Fitbit's documentation to find out how to call the data the format in which the dataset is returned. The parser I wrote then scans the .json object that the API call returns and extracts the values of interest to create a pandas data frame and then save it as a .csv file locally. This was trialled with data from mid November and automated for the 7 days I was interested in, which was from 30/12/2020 to the 06/01/2021. This was the time interval by which I had trained the image recognition model and could use the final weights to collect the mood predictions of my daily photos.

For the image collection necessary to label the sleep and steps data to find if there is a correlation between all three, there was no appropriate way to automate the data collection that would ensure taking a picture of myself at every waking hour of the day. Using a Raspberry Pi to do this was an option, but the Raspberry Pi placement would have to be fixed, and I move locations throughout the day, so the images were taken manually with my phone every day from 11am to 11pm.

5.2 Selecting an Image Recognition Algorithm and Training it:

In order to achieve the second part of my data sensing which was to collect information about my mood during the day, I needed to utilise a suitable image recognition algorithm. Following the research I carried out on state-of-the-art image detection algorithms available, and as a result of the table shown in Figure 6, I decided to use YOLOv3 as the image detection algorithm. Its main benefit is computational speed, as is described in its name, You Only Look Once, so contrary to algorithms that perform multiple sweeps of the image, detection and classification can be carried out at much higher speed, without a significant loss of accuracy and precision. [11]

Training YOLOv3 is very time consuming without a GPU, and therefore it was necessary to install the cuDNN library and CUDA toolkit, as well as drivers for the GPU to be able to utilise my GeForce RTX 2070 MaxQ GPU during the training process. This required me to downgrade my Ubuntu installation to 18.04, as it was the latest long term support version that these toolkits and libraries were supported in.

After this, the next step was to clone the latest fork of Darknet, and build it with OpenCV, CUDA and cuDNN. [12]

I had already obtained a good quality dataset that I could use for training, so the next step was to move on to the pre-processing.

Firstly, each image file must have a corresponding .txt labels file with the same name as the image, containing in each line the following information about each object in that image:
<object-class> <x_center> <y_center> <width> <height> where object class is a number that corresponds to the object type. This value references the obj.names file that contains the list of all the object classes, which in my case was the 11 moods the dataset contained.

x_center, y_center, width, height, are the relative x and y coordinates of the centre, height, and width of the boundary box containing the object. The labels I received with the dataset were absolute, and were missing the total image width and height, they only contained the object width and height. Thus, I had to write a number of scripts to firstly, read the pixel size of each image, store the values and append them to the csv with the absolute values. Then parse through the csv and calculate the relative values for x, y, width and height. Once this was done, I wrote another script to generate the .txt label files of each of the images as mentioned above.

Performance on the COCO Dataset

Model	Train	Test	mAP	FLOPS	FPS	Cfg	Weights
SSD300	COCO trainval	test-dev	41.2	-	46		link
SSD500	COCO trainval	test-dev	46.5	-	19		link
YOLOv2 608x608	COCO trainval	test-dev	48.1	62.94 Bn	40	cfg	weights
Tiny YOLO	COCO trainval	test-dev	23.7	5.41 Bn	244	cfg	weights
SSD321	COCO trainval	test-dev	45.4	-	16		link
DSSD321	COCO trainval	test-dev	46.1	-	12		link
R-FCN	COCO trainval	test-dev	51.9	-	12		link
SSD513	COCO trainval	test-dev	50.4	-	8		link
DSSD513	COCO trainval	test-dev	53.3	-	6		link
FPN FRCN	COCO trainval	test-dev	59.1	-	6		link
Retinanet-50-500	COCO trainval	test-dev	50.9	-	14		link
Retinanet-101-500	COCO trainval	test-dev	53.1	-	11		link
Retinanet-101-800	COCO trainval	test-dev	57.5	-	5		link
YOLOv3-320	COCO trainval	test-dev	51.5	38.97 Bn	45	cfg	weights
YOLOv3-416	COCO trainval	test-dev	55.3	65.86 Bn	35	cfg	weights
YOLOv3-608	COCO trainval	test-dev	57.9	140.69 Bn	20	cfg	weights
YOLOv3-tiny	COCO trainval	test-dev	33.1	5.56 Bn	220	cfg	weights
YOLOv3-spp	COCO trainval	test-dev	60.6	141.45 Bn	20	cfg	weights

Figure 6: Comparison of image recognition algorithms on COCO dataset

There also needs to be another two files named train.txt and test.txt that contain a new row for each image containing the path file of that image relative to the darknet directory. Train.txt should provide the location of the images that will be used to train YOLOv3 and test.txt, the location of the images to test it. [12] I generated these files with another script I wrote to iterate through all the directories, subdirectories and files in them, read them, refer to the main csv with all the label information that I had adapted as mentioned above, and then write in the main train.txt and test.txt files the path file of each image.

Furthermore, it is necessary to create a file with the class names, which is the obj.names file mentioned above, and as well as that, an obj.data file that contains the number of classes, the location of the train and the test set of images, and finally, the location of where the final weights will be stored.

Next, I had to download the pre-trained weights and place them in the correct folder in the Darknet directory and modify the .cfg file to update the correct number of classes and filters.[12]

I experimented with different parameters in the .cfg file to find a compromise between image resolution, batch size and speed of training, making sure to run the training process at the highest RAM capacity of my GPU without throwing the error of memory exceeded. I reduced the resolution and increased the batch size which did not cause any significant downgrade to the training because the images in the dataset were cropped to fit the face.

Finally, after all these steps, I was able to train the algorithm. I then assessed the mean average precision (mAP) and total average loss in the chart and selected the best performing set of weights from the first values that converged, to avoid overfitting. The total time to train was about 4 days, and due to the duration of the training, the mAP charts are discontinuous, because I experienced power cuts throughout the training.

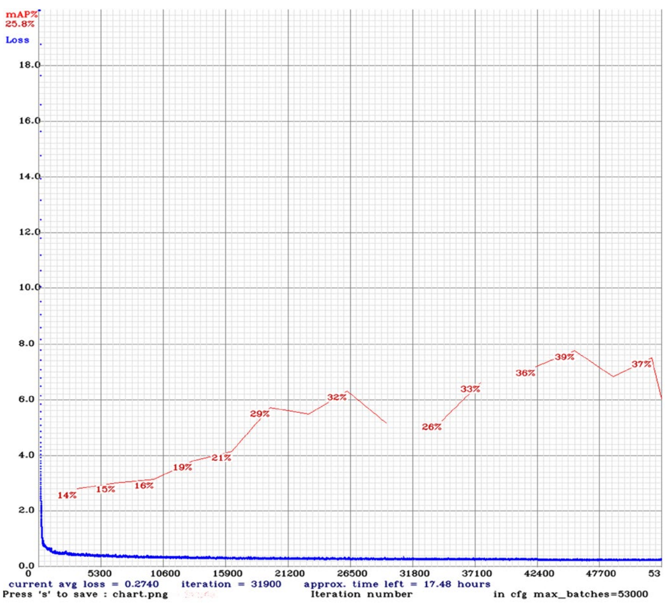


Figure 7: YOLOv3 training

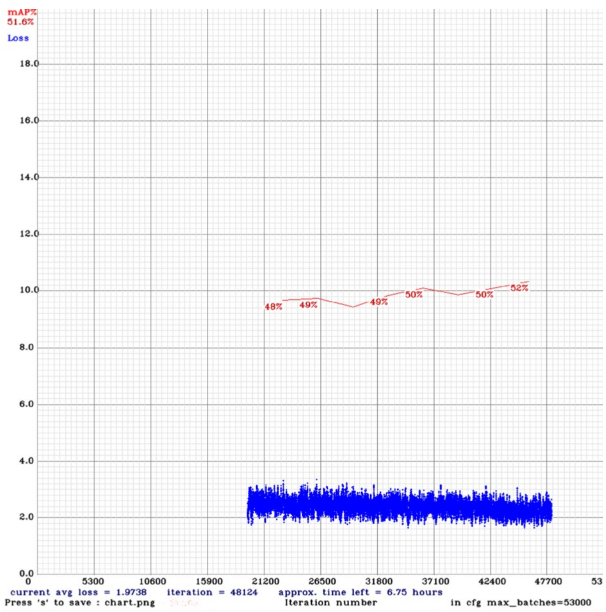


Figure 8: YOLOv4 training

Once I had finished the week of data collection for my images, I wrote another parser that would get the image metadata, including the date and time they were taken, and then run the image through the trained image recognition algorithm but calling it through the OpenCV Deep Neural Network module as opposed to locally through darknet. The output was the mood prediction and the confidence level as shown in Figure 9, that was then appended to a pandas data frame and saved as csv files automatically into my computer.

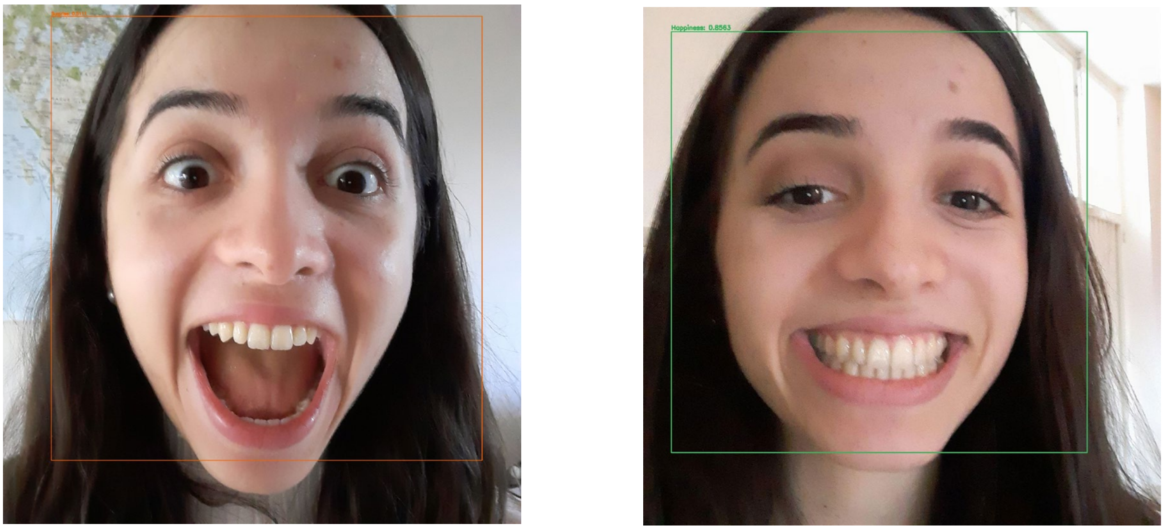


Figure 9: Examples of the outputs from the sensed data predictions by the parser (the left image shows a "Surprise" prediction of 51%, and the right image shows a prediction of "Happiness" of 85%)

6. Basic characteristics of the end-to-end systems set up and data

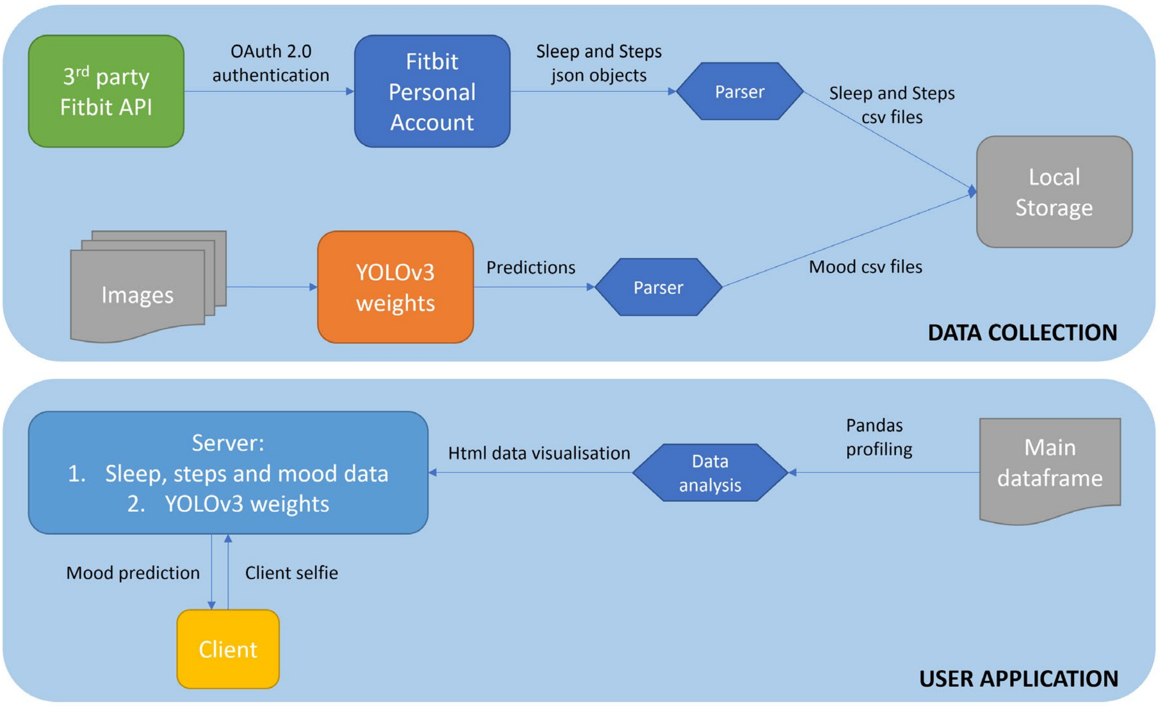


Figure 10: Data collection and User Application System Overview

Figure 4 depicts the system set up to collect the data that was described in the previous section. Extensive data analysis was carried out on the collected datasets, which ultimately included: REM sleep, Deep sleep, Light sleep, Restlessness, Awake minutes, Step count, and mood. The possible attributes of mood were Happiness, Sadness, Neutral, Sadness, Fear, Anger, Disgust, Contempt, Surprise, None and Uncertain.

The user only interacts with the processed data after it has been run through pandas profiling to generate a html report. This html report is interactive and is rendered through a Python Flask App. It lies in the server along with the trained weights of YOLOv3. The interface also automatically runs a script that allows the user to take an image of themselves and then this image is processed through another script to return the mood predictions.

Coursework 2: Internet of Things

1. Data Interaction and visualisation platform

The data visualisation was carried out using pandas profiling. Pandas profiling is a library that extends on the functionality of existing prebuilt pandas functions to visualise data. This library returns the following information:

- **Type inference**: detect the types of columns in a dataframe.
- **Essentials**: type, unique values, missing values
- **Quantile statistics:** like minimum value, Q1, median, Q3, maximum, range, interquartile range
- **Descriptive statistics** like mean, mode, standard deviation, sum, median absolute deviation, coefficient of variation, kurtosis, skewness
- **Most frequent values**
- **Histograms**
- **Correlations** highlighting of highly correlated variables, Spearman, Pearson and Kendall matrices
- **Missing values** matrix, count, heatmap and dendrogram of missing values
- **Duplicate rows** Lists the most occurring duplicate rows
- **Text analysis** learn about categories (Uppercase, Space), scripts (Latin, Cyrillic) and blocks (ASCII) of text data

All of this information is displayed by the server that returns this visualisation through Flask by rendering a html report. The user can decide which data to toggle so the amount of information is not overwhelming.

Further to this, the client can send the image they took of themselves which is automatically cropped and passed through the trained weights of the image recognition algorithm, and the server returns the predictions, the confidences and the image visualisation with the bounding boxes. This is done by using the Deep Neural Networks module from OpenCV, because otherwise the way to run the predictions is locally on Darknet, and I wanted to be able to run them on a server in real time.

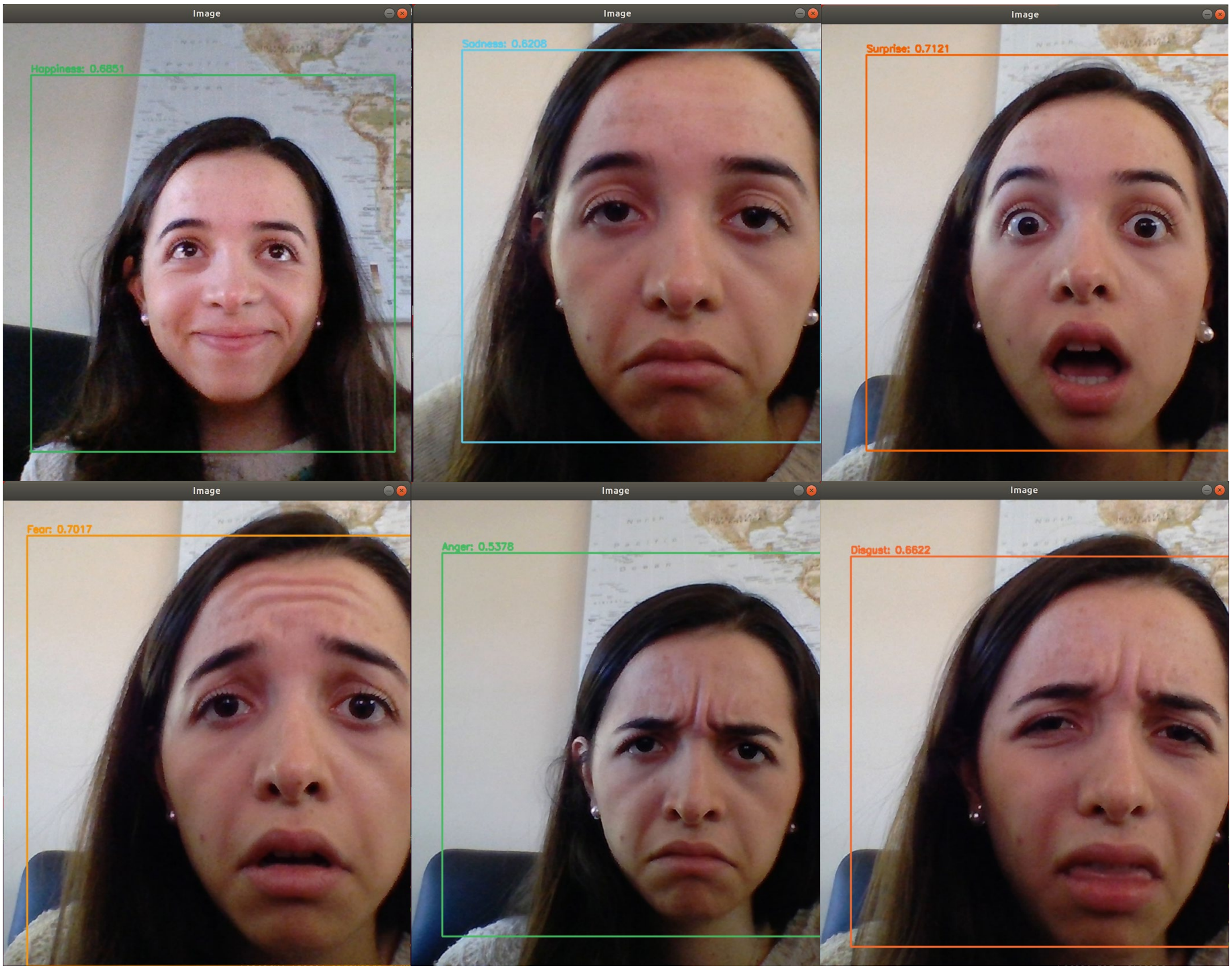


Figure 11: Examples of the client end of the user interface in real time

2. Data analytics, inferences and insights

Steps analysis:

Quantile statistics

Minimum	0
5-th percentile	0.9
Q1	111.5
median	283
Q3	482.5
95-th percentile	956.5
Maximum	3564
Range	3564
Interquartile range (IQR)	371

Descriptive statistics

Standard deviation	535.5039518
Coefficient of variation (CV)	1.350825505
Kurtosis	22.54127853
Mean	396.4271845
Median Absolute Deviation (MAD)	175
Skewness	4.238785232
Sum	40832
Variance	286764.4824
Monotocity	Not monotonic

Figure 12: Analysis of the step count data

The mean number of steps per hour is relatively low, and in fact the average number of steps for the week of data collected was 6009 steps, which is significantly below the recommended daily average of 10 000 steps. This can also be seen very clearly from figure 8, where the most common value for steps that hour was 0.

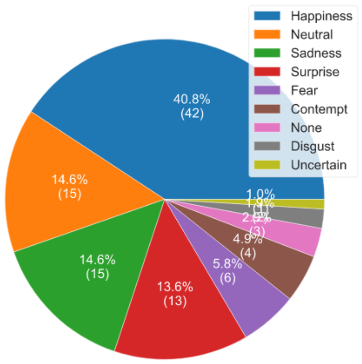


Figure 13: Mood frequency during the week

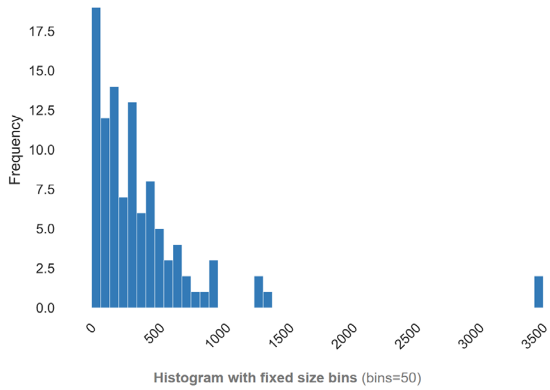


Figure 14: Step count histogram

Mood analysis:

Predictions:

The most frequent mood that I experienced during the week was Happiness, followed by Neutral and Sadness. However, I was only happy for 40.8% of time so it is possible to draw the observation that the frequency of my happiness count was low, as well my average step count. This could lead to the hypothesis that potentially an increase in average step count could increase the number of times I am happy throughout the week, although this would require collecting data for a longer period of time and identifying if there is indeed a trend between the two.

The week I carried out my data collection was an outlier for my mood and physical activity because of the density of work I had to carry out, so I often spent long periods of time studying in my desk, and as a result, because mood is a complex variable with many factors playing a key role in it, it is not possible to conclude that my happiness, or lack thereof, is caused by my physical activity. Both of these attributes are likely to be a cause of other external features that are not represented in this dataset.

Confidence:

This value represents the confidence of the mood prediction and the results show that the model I built has a mean average confidence prediction of 66%. Verifying that indeed all the predictions of outputted by the trained weights were correct for the images I took throughout the week, this reflects that the model performs well and it has a positive skewness, meaning its values lean towards a higher confidence of prediction. Furthermore, the values represented are for YOLOv3 and not YOLOv4, which had a 13% higher mean average precision than YOLOv3, so this demonstrates that even the worst models of the two performs to a very satisfactory level.

Descriptive statistics

Standard deviation	0.2101734128
Coefficient of variation (CV)	0.3205819041
Kurtosis	-1.236405819
Mean	0.6555997393
Median Absolute Deviation (MAD)	0.176820755
Skewness	0.04731131316
Sum	67.52677315
Variance	0.04417286345
Monotocity	Not monotonic

Figure 15: Analysis of confidence data

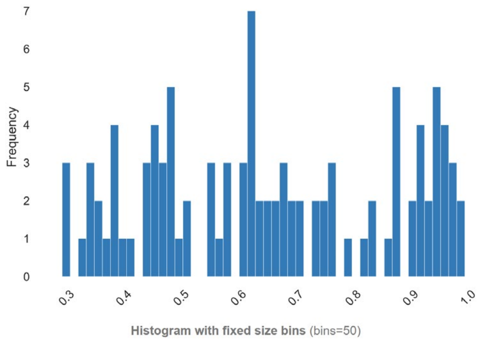


Figure 16: Histogram of confidence values

Sleep Analysis:

Given that I was measuring my mood throughout every hour of the day, the sleep variable was kept constant and was an attribute of the previous day. This decision was made after carrying out the first round of data analysis and finding that if I kept the sleep per minute data points, it skewed the results towards the “no-face” mood prediction, as I did not have any mood predictions whilst I was sleeping.

Quantile statistics

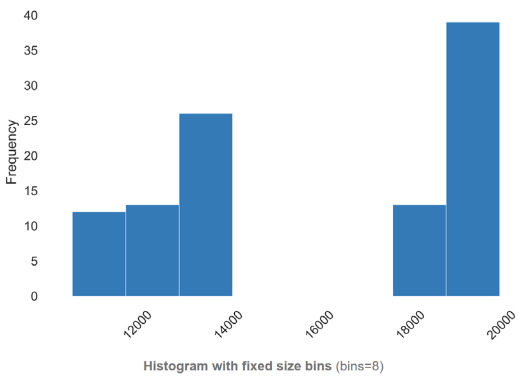
Minimum	4590
5-th percentile	4590
Q1	5070
median	5130
Q3	5910
95-th percentile	7350
Maximum	7350
Range	2760
Interquartile range (IQR)	840

Descriptive statistics

Standard deviation	826.1490622
Coefficient of variation (CV)	0.1464273973
Kurtosis	-0.1525327238
Mean	5642.038835
Median Absolute Deviation (MAD)	540
Skewness	0.8076054098
Sum	581130
Variance	682522.273
Monotocity	Not monotonic

Figure 17: Statistical information of Deep Sleep

Figure 17 shows the statistical information of my deep sleep, and the user app displays along side this category REM sleep, light sleep, restlessness and awake as well.



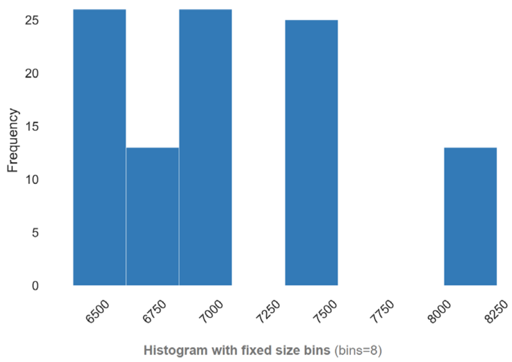


Figure 18: Light sleep histogram

Figure 19: REM sleep histogram

Light sleep comprised the majority of my sleep percentage in the night, with an average of 52% of my night spent in light sleep. This can be clearly observed from Figures 18 and 19, where the minutes in the light sleep histogram are significantly higher than in REM sleep. This is also the case for deep sleep.

Inferences and Correlations:

Because the predictions data is categorical, the Phi K correlation coefficient was used. This coefficient is a new correlation coefficient between categorical, ordinal and interval variables with Pearson characteristics and works consistently between categorical, ordinal and interval variables. [13] It also captures non-linear dependency, and it reverts to the Pearson correlation coefficient in case of a bi-variate normal input distribution. As such it makes it a suitable correlation coefficient to use for variables of mixed types. [13]

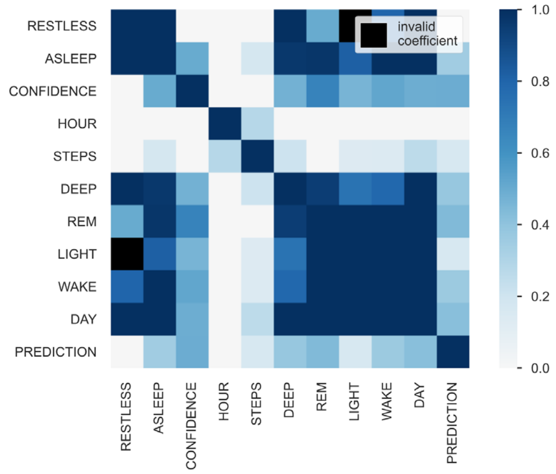


Figure 20: Phi K correlation matrix

Figure 20 shows that there is a correlation between Predictions and Confidence, which is of course expected, but interestingly it shows that the coefficient of correlation between REM sleep and mood is around 0.5 to 0.6. Further data collection would aid to clarify if this correlation sustains over a prolonged period of time.

Figure 21 takes a closer look at the mood and steps data. It shows the count of the times a mood appeared in that day as a percentage of the times it appeared in total. It is worth noticing that the highest frequency of the happiness mood in the week appears between 12:00 at 15:00 in the day, which is also the time window when I walked the highest number of steps consistently throughout the week.

Cuenta de CONFIDENCE	**Etiquetas de columna**									
Etiquetas de fila	**Contempt**	**Disgust**	**Fear**	**Happiness**	**Neutral**	**NoFace**	**None**	**Sadness**	**Surprise**	**Uncertain**	**Total general**
0:00:00	0.00%	0.00%	0.00%	0.00%	0.00%	100.00%	0.00%	0.00%	0.00%	0.00%	100.00%
1:00:00	0.00%	0.00%	0.00%	0.00%	0.00%	100.00%	0.00%	0.00%	0.00%	0.00%	100.00%
2:00:00	0.00%	0.00%	0.00%	0.00%	0.00%	100.00%	0.00%	0.00%	0.00%	0.00%	100.00%
3:00:00	0.00%	0.00%	0.00%	0.00%	0.00%	100.00%	0.00%	0.00%	0.00%	0.00%	100.00%
4:00:00	0.00%	0.00%	0.00%	0.00%	0.00%	100.00%	0.00%	0.00%	0.00%	0.00%	100.00%
5:00:00	0.00%	0.00%	0.00%	0.00%	0.00%	100.00%	0.00%	0.00%	0.00%	0.00%	100.00%
6:00:00	0.00%	0.00%	0.00%	0.00%	0.00%	100.00%	0.00%	0.00%	0.00%	0.00%	100.00%
7:00:00	0.00%	0.00%	0.00%	0.00%	0.00%	100.00%	0.00%	0.00%	0.00%	0.00%	100.00%
8:00:00	0.00%	0.00%	0.00%	0.00%	0.00%	100.00%	0.00%	0.00%	0.00%	0.00%	100.00%
9:00:00	0.00%	0.00%	0.00%	0.00%	0.00%	100.00%	0.00%	0.00%	0.00%	0.00%	100.00%
10:00:00	0.00%	0.00%	0.00%	0.00%	0.00%	100.00%	0.00%	0.00%	0.00%	0.00%	100.00%
11:00:00	0.00%	12.50%	12.50%	12.50%	12.50%	12.50%	0.00%	25.00%	12.50%	0.00%	100.00%
12:00:00	0.00%	0.00%	12.50%	50.00%	12.50%	0.00%	0.00%	12.50%	12.50%	0.00%	100.00%
13:00:00	25.00%	0.00%	0.00%	50.00%	12.50%	0.00%	0.00%	0.00%	12.50%	0.00%	100.00%
14:00:00	12.50%	12.50%	0.00%	50.00%	25.00%	0.00%	0.00%	0.00%	0.00%	0.00%	100.00%
15:00:00	0.00%	0.00%	12.50%	37.50%	12.50%	0.00%	0.00%	12.50%	25.00%	0.00%	100.00%
16:00:00	0.00%	0.00%	12.50%	37.50%	12.50%	0.00%	0.00%	25.00%	12.50%	0.00%	100.00%
17:00:00	0.00%	0.00%	0.00%	50.00%	12.50%	0.00%	12.50%	25.00%	0.00%	0.00%	100.00%
18:00:00	0.00%	0.00%	0.00%	25.00%	12.50%	0.00%	12.50%	25.00%	12.50%	12.50%	100.00%
19:00:00	12.50%	0.00%	0.00%	37.50%	25.00%	0.00%	0.00%	12.50%	12.50%	0.00%	100.00%
20:00:00	0.00%	0.00%	0.00%	37.50%	12.50%	0.00%	12.50%	12.50%	25.00%	0.00%	100.00%
21:00:00	12.50%	0.00%	0.00%	25.00%	25.00%	0.00%	0.00%	12.50%	25.00%	0.00%	100.00%
22:00:00	0.00%	0.00%	25.00%	37.50%	12.50%	0.00%	0.00%	12.50%	12.50%	0.00%	100.00%
23:00:00	0.00%	0.00%	0.00%	75.00%	0.00%	0.00%	0.00%	12.50%	12.50%	0.00%	100.00%
Total general	**2.60%**	**1.04%**	**3.13%**	**21.88%**	**7.81%**	**46.35%**	**1.56%**	**7.81%**	**7.29%**	**0.52%**	**100.00%**

Figure 21: Mood percentage distribution throughout the day

Figure 20 and figure 21 show that there is a basis in the hypothesis that the minutes of REM sleep and the hourly step count have a positive correlation with mood, but 7 days of data points are not enough to verify if this claim holds in longer time intervals.

3. Discussions on the important aspects of the project

Training YOLO was a crucial aspect of this project. I could have used pre trained weights or other existing models or neural networks but carrying out this task from start to finish myself was extremely rewarding. I had a particular desire to do this because I had previously trained YOLOv3 on a much smaller dataset to detect and sort different classes of recycling, but the dataset of merely 5000 images which is only 9% of the size compared to the one I have used this time.

I realised that despite only using 5000 images for 5 classes (you should use at least 2000 images per class) when I trained it on the Recycling dataset, YOLO performed incredibly well compared to other image recognition algorithms as is described in figure 6. This is especially noticeable with live stream video, averaging a little less than 10FPS on real time predictions.

I was genuinely excited to see the capabilities YOLO has with a much larger dataset like AffectNet, and especially in the context of an application that is very interesting and stimulating to me, because it is looking at integrating my own sleep and step count data with my mood.

I also got the chance to experiment with YOLOv4, which is the very latest version of YOLO, which was not available this summer, when I first implemented it. Because I planned ahead, knowing that the data acquiring and pre processing would take a significant portion of time, I was actually able to train both YOLOv3 and YOLOv4 with a total of 53,000 images each. This was almost 5000 per class, and they took approximately 4 days to train each. I was able to witness first hand how YOLOv4 achieved 13% higher mean average precision on the same dataset and training time. I had to use YOLOv3 for the Flask client server application because YOLOv4 is not supported by the OpenCV version I have installed, which needs to be compiled with CUDA and cuDNN to be able to call the weights with the OpenCV Deep Neural Network module.

I also upgraded my skills using Flask, because prior to this the front end of my server-client app was extremely rudimentary, if existent at all. I am pleased with the fact that my front-end interface for the data visualisation is user friendly whilst still carrying out the data analysis and fusion that I set to achieve at the start of the project.

Finally, it is the first time I have encountered significant challenges thinking about the efficiency and time complexity of my code. I had to write numerous parsers to clean, pre process and extract key values from my datasets, and in particular with the AffectNet dataset, because the number of files was so large, I ran into unmanageable run times with my parsers when the code was not written efficiently. This made me take the time to learn about the logic behind my scripts and how to optimise their run time. Prior to this, I never really had to consider if my code was efficient so long as it ran.

4. Avenues for future work and potential impact

In the future, I would like to utilise more of the information provided by the AffectNet dataset, in particular the valence level attribute. It would provide further dimension to the prediction data to not only see how confident that prediction is, but how intense the emotion is. Further to this, I would like to automate the image collection in a way where the client does not have to even take the image themselves. I noticed that having to take the image made me conscious of what emotion I was feeling, and because I was the one collecting the data as well as the source of data, I believe this may have caused a skew in the results. Seeing how the predictions vary when the user is aware but not conscious that an image is being taken at a specific sample rate may provide more representative results.

I would also like to upgrade the compatibility of the sleep and steps measurements to be able to include not only Fitbit smartwatches, but a wider range of activity trackers. If I had a greater budget I would utilise AWS to create a model where the data that is acquired from the smart trackers is also used as training data to iteratively improve the model of correlation over time.

Whilst this is a light hearted application of a sensing and internet of things project, the clinical studies mentioned in the introduction show that there is a potential for greater impact in a medical context to be able to predict and therefore prevent the onset of manic attacks or depression in patients with mental illnesses. A much wider range of data categories should be acquired, and over a greater time period than just a week, and if this were achieved, a model could be derived that is customised to each patient to predict the onset of mania and depression. This application poses great ethical concerns however, as there is the utmost need for data privacy to prevent malicious use of the data and leakage to third party data collectors. Privacy and security have not been greatly considered thus far in the project given that the data source was myself, I was collecting the data and I was the sole owner of the data. I purposefully took the decision to save the sensed data locally because of the nature of it. Scaling this application would involve further research into privacy in IoT applications, such as proposals like the Databox platform.

References:

1. Bauer, M., Grof, P., Rasgon, N., Bschor, T., Glenn, T. and Whybrow, P.C., 2006. Temporal relation between sleep and mood in patients with bipolar disorder. *Bipolar Disorders*, *8*(2), pp.160-167.
2. Thomsen, D.K., Mehlsen, M.Y., Christensen, S. and Zachariae, R., 2003. Rumination—relationship with negative mood and sleep quality. *Personality and Individual Differences*, *34*(7), pp.1293-1301.
3. Miles, L., 2007. Physical activity and health. *Nutrition bulletin*, *32*(4), pp.314-363.
4. Silveira, H., Moraes, H., Oliveira, N., Coutinho, E.S.F., Laks, J. and Deslandes, A., 2013. Physical exercise and clinically depressed patients: a systematic review and meta-analysis. *Neuropsychobiology*, *67*(2), pp.61-68.
5. Ang, E.T. and Gomez-Pinilla, F., 2007. Potential therapeutic effects of exercise to the brain. *Current medicinal chemistry*, *14*(24), pp.2564-2571.
6. Sharma, A., Madaan, V. and Petty, F.D., 2006. Exercise for mental health. *Primary care companion to the Journal of clinical psychiatry*, *8*(2), p.106.
7. Ali Mollahosseini, Behzad Hasani, and Mohammad H. Mahoor, "AffectNet: A New Database for Facial Expression, Valence, and Arousal Computation in the Wild", IEEE Transactions on Affective Computing, 2017.
8. Sridhar, S. 2017, "Fitbit Ionic fitness smartwatch with swim tracking, GPS announced", Fitness Trackers News. Available online: https://www.fonearena.com/blog/228062/fitbit-ionic-fitness-smartwatch-with-swim-tracking-gps-announced.html
9. DC Rainmaker, 2017. "The Fitbit Ionic: Everything You Need to Know" Available online: https://www.dcrainmaker.com/2017/08/fitbit-ionic-gps-smartwatch-all-the-details.html
10. Hsu, S., 2018, "Collect Your Own Fitbit Data with Python", Towards Data Science. Available online: https://towardsdatascience.com/collect-your-own-fitbit-data-with-python-ff145fa10873
11. Redmon J., Farhadi Ali. 2016, "YOLOv3: An Incremental Improvement" Available online: https://pjreddie.com/media/files/papers/YOLOv3.pdf
12. Bochkovskiy, A., Wang, C.Y. and Liao, H.Y.M., 2020. YOLOv4: Optimal Speed and Accuracy of Object Detection. *arXiv preprint arXiv:2004.10934*.
13. Baak, M., Koopman, R., Snoek, H. and Klous, S., 2020. A new correlation coefficient between categorical, ordinal and interval variables with Pearson characteristics. *Computational Statistics & Data Analysis*, *152*, p.107043.