CA400 Final Year Computer Applications Project

Document Type: User Manual

Project Title: Human Activity Recognition using wrist-mounted sensors based on Symbolic Aggregate Approximation and Machine Vision

Student: Shane Creedon

Student ID: 15337356

Supervisor: Tomas Ward

Date of Completion: 17/05/2019

1 Abstract

This document is the user manual behind my fourth year project based on Human Activity Recognition. In this document I will provide a step-by-step approach involving how to setup, start, and utilise the application. For my particular software project, I have it split up into two components:

1. The Web Application
2. The Desktop Application

For each section in this user manual, insight and guidance for both of the above listed components will be discussed and mentioned. This document will also contain screenshots to help further with this process of comprehension.

2 Installation

Before I specify instructions relating to both components of my project, I want to provide some pre-initialization steps:

1. Clone repository from Gitlab.
2. Install all requirements in requirements.txt file (pip install -r requirements.txt)
3. It is generally a good idea to set up a Python Virtual Environment prior to installing all requirements. (virtualenv {environment_name}

Web Application Component:

Once all the requirement packages are installed using the above steps, running the web application becomes quite easy.

1. Navigate to src/code/website_module/website-implementation/
2. Run: 'python manage.py runserver'

The script should initialise the web application on port 8000. Simply open up your browser and enter: localhost:8000 or 127.0.0.1:8000 and you will see the web application i've built.

Alternatively, I am hosting the website online using Nginx and Gunicorn and can be viewed here: https://www.projectactivityrecognition.ml

Desktop Application Component

Similarly to the above component, it is essential all requirements are downloaded prior to using the application. Again, generally it is advised to use a virtual environment in these scenarios to prevent version conflicts with packages. It is required you spin up a client and server instance to run the desktop software sufficiently and I will detail both procedures below.

NB: It is essentially important that you install a piece of broker software to facilitate the MQTT protocol. To do this, navigate to https://mosquitto.org/ and download the mosquitto broker software on your desired operating system. By default once downloaded, the broker should be an active background process on your machine and will be accessible at **127.0.0.1:1883**. This step is crucial before testing out the desktop software. Both files will be runnable without this broker active, but the implemented functionality will not work.

Server:

1. Navigate to src/code/mqtt_protocol_module/
2. Run: ‘python server_connect.py’

The server should spin up and subscribe to various topics. Once this happens, you know the server is successfully active.

Client:

1. Navigate to src/code/desktop_gui_module/main/
2. Run: ‘python application.py’

The client application should spin up and appear as an application on your screen.

3 User Guide

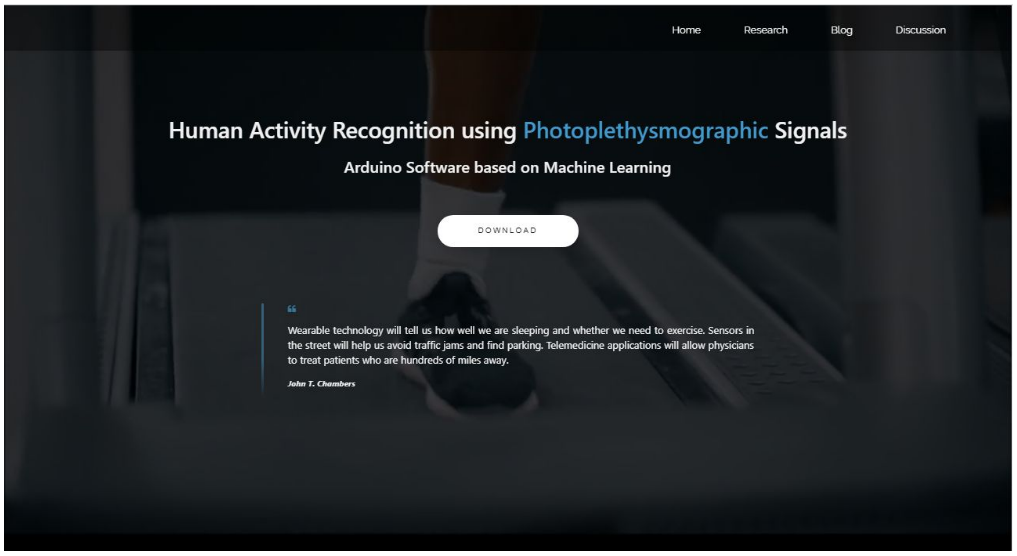
Website Instructions:

To successfully use the website, either access it using the above **installation** methods or as previously mentioned, access the website at: https://www.projectactivityrecognition.ml.

Once you arrive at the website you will arrive at the homepage where you can interact with several key components/features on the website.

Home:

1. The **Download** functionality which is accessible through clicking the white button labeled ‘download’. A zip file contained the client-side software as well as instructions on how to use it will be downloaded upon click. This is available at the top of the home page.



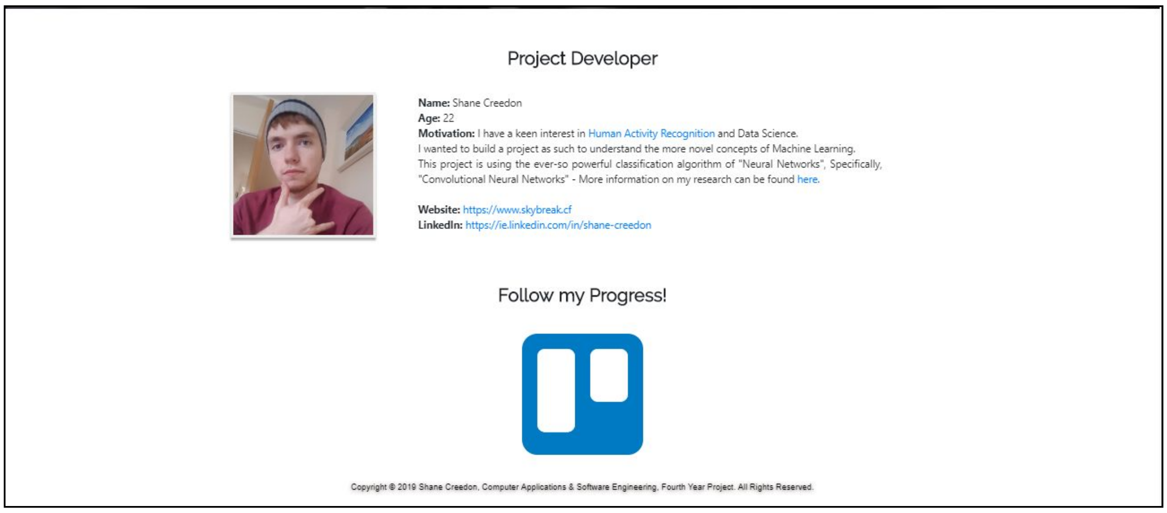
Human Activity Recognition using Photoplethysmographic Signals

Arduino Software based on Machine Learning

Wearable technology will tell us how well we are sleeping and whether we need to exercise. Sensors in the street will help us avoid traffic jams and find parking. Telemedicine applications will allow physicians to treat patients who are hundreds of miles away.

John T. Chambers

2. The **Developer** section, where by users can read a bit about me and why I wanted to get involved in data science and human activity recognition. This section is available at the bottom of the home page.



Project Developer

Name: Shane Creedon
Age: 22
Motivation: I have a keen interest in Human Activity Recognition and Data Science.
I wanted to build a project as such to understand the more novel concepts of Machine Learning.
This project is using the ever-so powerful classification algorithm of "Neural Networks", Specifically, "Convolutional Neural Networks" - More information on my research can be found here.

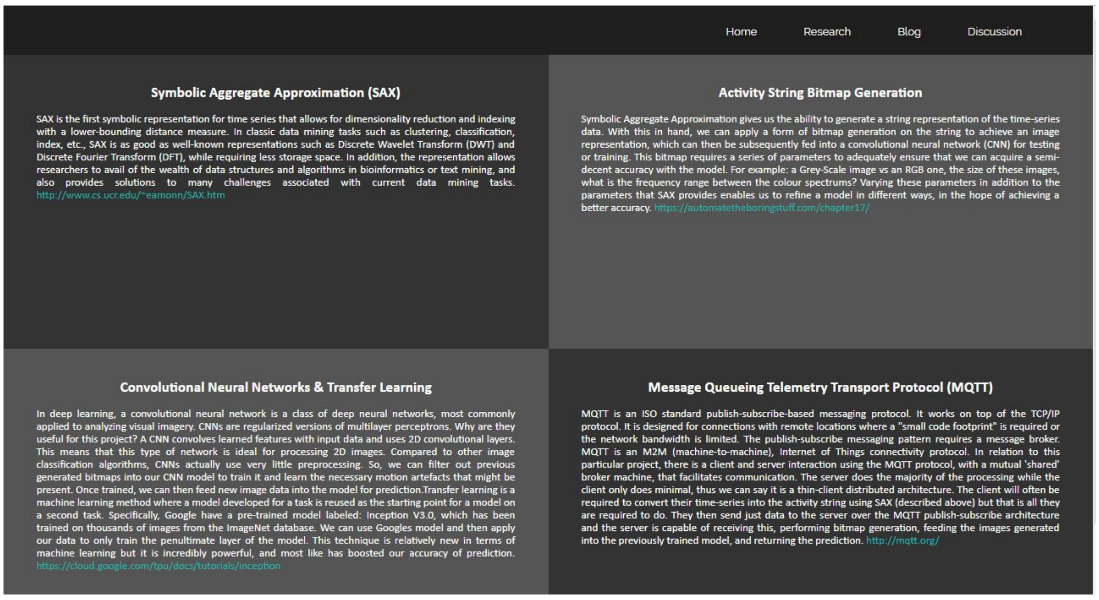
Website: https://www.skybreak.cf
LinkedIn: https://ie.linkedin.com/in/shane-creedon

Follow my Progress!

Copyright © 2019 Shane Creedon, Computer Applications & Software Engineering, Fourth Year Project. All Rights Reserved.

Research

1. The research section simply offers information on the different aspects and techniques I utilised to build my project. Once the tab is clicked, the page will load up in a grid pattern with each technique corresponding to a grid item. I also provide links for further information about any techniques used.



Symbolic Aggregate Approximation (SAX)

SAX is the first symbolic representation for time series that allows for dimensionality reduction and indexing with a lower-bounding distance measure. In classic data mining tasks such as clustering, classification, index, etc., SAX is as good as well-known representations such as Discrete Wavelet Transform (DWT) and Discrete Fourier Transform (DFT), while requiring less storage space. In addition, the representation allows researchers to avail of the wealth of data structures and algorithms in bioinformatics or text mining, and also provides solutions to many challenges associated with current data mining tasks. http://www.cs.ucr.edu/~eamonn/SAX.htm

Activity String Bitmap Generation

Symbolic Aggregate Approximation gives us the ability to generate a string representation of the time-series data. With this in hand, we can apply a form of bitmap generation on the string to achieve an image representation, which can then be subsequently fed into a convolutional neural network (CNN) for testing or training. This bitmap requires a series of parameters to adequately ensure that we can acquire a semi-decent accuracy with the model. For example: a Grey-Scale image vs an RGB one, the size of these images, what is the frequency range between the colour spectrums? Varying these parameters in addition to the parameters that SAX provides enables us to refine a model in different ways, in the hope of achieving a better accuracy. https://automatetheboringstuff.com/chapter17/

Convolutional Neural Networks & Transfer Learning

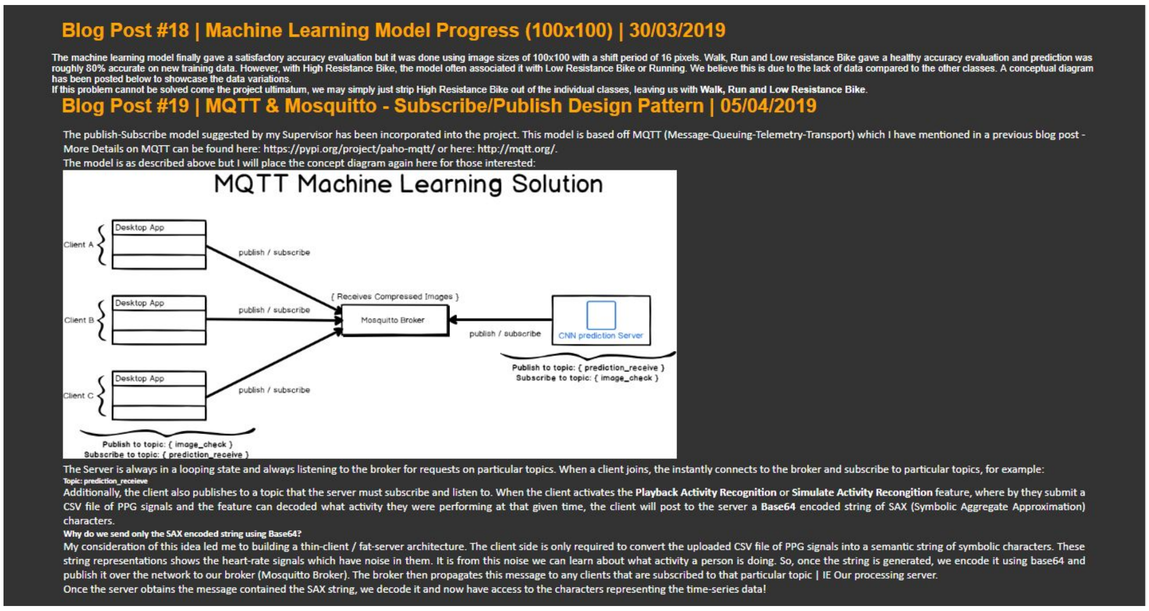
In deep learning, a convolutional neural network is a class of deep neural networks, most commonly applied to analyzing visual imagery. CNNs are regularized versions of multilayer perceptrons. Why are they useful for this project? A CNN convolves learned features with input data and uses 2D convolutional layers. This means that this type of network is ideal for processing 2D images. Compared to other image classification algorithms, CNNs actually use very little preprocessing. So, we can filter out previous generated bitmaps into our CNN model to train it and learn the necessary motion artefacts that might be present. Once trained, we can then feed new image data into the model for prediction.Transfer learning is a machine learning method where a model developed for a task is reused as the starting point for a model on a second task. Specifically, Google have a pre-trained model labeled: Inception V3.0, which has been trained on thousands of images from the ImageNet database. We can use Googles model and then apply our data to only train the penultimate layer of the model. This technique is relatively new in terms of machine learning but it is incredibly powerful, and most like has boosted our accuracy of prediction. https://cloud.google.com/tpu/docs/tutorials/inception

Message Queueing Telemetry Transport Protocol (MQTT)

MQTT is an ISO standard publish-subscribe-based messaging protocol. It works on top of the TCP/IP protocol. It is designed for connections with remote locations where a "small code footprint" is required or the network bandwidth is limited. The publish-subscribe messaging pattern requires a message broker. MQTT is an M2M (machine-to-machine), Internet of Things connectivity protocol. In relation to this particular project, there is a client and server interaction using the MQTT protocol, with a mutual 'shared' broker machine, that facilitates communication. The server does the majority of the processing while the client only does minimal, thus we can say it is a thin-client distributed architecture. The client will often be required to convert their time-series into the activity string using SAX (described above) but that is all they are required to do. They then send just data to the server over the MQTT publish-subscribe architecture and the server is capable of receiving this, performing bitmap generation, feeding the images generated into the previously trained model, and returning the prediction. http://mqtt.org/

Blog

1. The blog also offers no real interaction other than researching and understanding about how the project works as well as the process that was undertaken to create it. I have over 35 blog entries describing the process from start to finish and also providing details about how someone can use the knowledge i've gained in their own projects / technical ventures. Simply scroll up and down to read about my project milestones with insightful images to aid.

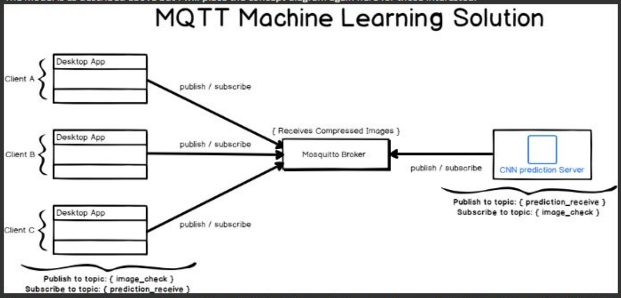


Blog Post #18 | Machine Learning Model Progress (100x100) | 30/03/2019

The machine learning model finally gave a satisfactory accuracy evaluation but it was done using image sizes of 100x100 with a shift period of 16 pixels. Walk, Run and Low resistance Bike gave a healthy accuracy evaluation and prediction was roughly 80% accurate on new training data. However, with High Resistance Bike, the model often associated it with Low Resistance Bike or Running. We believe this is due to the lack of data compared to the other classes. A conceptual diagram has been posted below to showcase the data variations.
If this problem cannot be solved come the project ultimatum, we may simply just strip High Resistance Bike out of the individual classes, leaving us with **Walk, Run and Low Resistance Bike**.

Blog Post #19 | MQTT & Mosquitto - Subscribe/Publish Design Pattern | 05/04/2019

The publish-Subscribe model suggested by my Supervisor has been incorporated into the project. This model is based off MQTT (Message-Queuing-Telemetry-Transport) which I have mentioned in a previous blog post - More Details on MQTT can be found here: https://pypi.org/project/paho-mqtt/ or here: http://mqtt.org/.
The model is as described above but I will place the concept diagram again here for those interested:

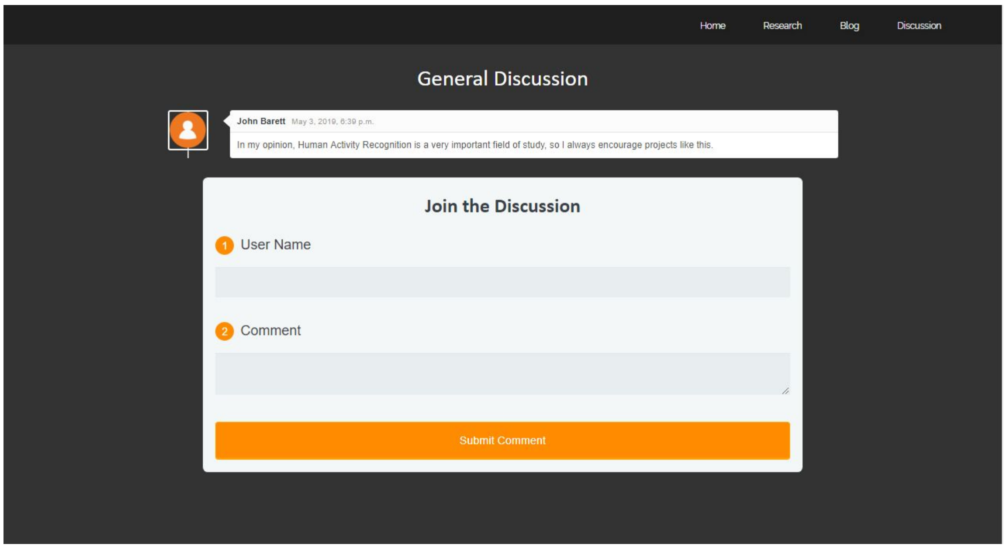

MQTT Machine Learning Solution
Client A
Desktop App
Client B
Desktop App
Client C
Desktop App
publish / subscribe
publish / subscribe
publish / subscribe
{ Receives Compressed Images }
Mosquitto Broker
publish / subscribe
CNN prediction Server
Publish to topic: { prediction_receive }
Subscribe to topic: { image_check }
Publish to topic: { image_check }
Subscribe to topic: { prediction_receive }

The Server is always in a looping state and always listening to the broker for requests on particular topics. When a client joins, the instantly connects to the broker and subscribe to particular topics, for example:
Topic: prediction_receieve
Additionally, the client also publishes to a topic that the server must subscribe and listen to. When the client activates the **Playback Activity Recognition** or **Simulate Activity Recongition** feature, where by they submit a CSV file of PPG signals and the feature can decoded what activity they were performing at that given time, the client will post to the server a **Base64** encoded string of SAX (Symbolic Aggregate Approximation) characters.
Why do we send only the SAX encoded string using Base64?
My consideration of this idea led me to building a thin-client / fat-server architecture. The client side is only required to convert the uploaded CSV file of PPG signals into a semantic string of symbolic characters. These string representations shows the heart-rate signals which have noise in them. It is from this noise we can learn about what activity a person is doing. So, once the string is generated, we encode it using base64 and publish it over the network to our broker (Mosquitto Broker). The broker then propagates this message to any clients that are subscribed to that particular topic | IE Our processing server.
Once the server obtains the message contained the SAX string, we decode it and now have access to the characters representing the time-series data!

Discussion

1. The discussion section is dramatically more interactive offering the ability for users to join in on a discussion with other users in relation to human activity recognition. Users will be able to view other users comments, with each comment containing a particular author, comment date, comment text and comment time.

2. Users can join the discussion by entering a form name into the name box below and also provide relative comment text. Clicking the submit button will post their comment up for all to see.



General Discussion

John Barett May 3, 2019, 6:39 p.m.
In my opinion, Human Activity Recognition is a very important field of study, so I always encourage projects like this.

Join the Discussion

1 User Name

2 Comment

Desktop Application Instructions:

Open the desktop application using the above installation instructions, spin up both the server application and the client application, in that order. Users should subsequently be greeted with the desktop application overview screen.

Overview:

1. The overview window is split into 4 sub window panes which act as summarizations of the 4 tab choices at the top of the application frame. Users can only interact with the bottom-left pane which is denoted as the 'controller' pane, since it controls the other window panes.

2. Users will notice two flashing red or green icons at the bottom of the desktop software which are indicative of whether the client is successfully connected to the broker and whether the client has a successful PPG connection.

   Note: If you have an arduino PPG connected and this icon has not changed to green, you may need to find out what port the arduino is connected on and set the application to that port. This can be done in the port settings in the menu bar at the top of the software.

   Once the Arduino PPG is connected, you should see the top-left pane (Graph Pane) start picking up your heart-rate and motion artefacts from any inertial movements.

3. Users can select the visibly green button in the controller pane (bottom-left) to activate activity recognition playback. This function when clicked will bring up a file explorer dialog box, where by users can select any PPG recording files in the form of a csv. It is essential there is an active server to facilitate this request, so if you receive no feedback it is likely the server is not available or is not online.

   Users will notice a loading icon appear and the loading bar appear start to progress in relation to the length of the csv file submitted. After several seconds of waiting, (3-4 seconds) the server will start responding to the client with what activity it thinks the client submitted. The server will go through the submitted csv file and adequately "guess" what it thinks activity had been performed.

   Users will notice the UI update to reflect what activity the server believes is being performed. The UI avatar (mario) will update to reflect the 4 cardinal activities of Walking, Running, Slow Cycle and Fast Cycle respectively. Additionally, users can view the time of the activity as well as the predictive accuracy. The graph view pane (Top-Right) will also update to show the PPG recording signal at that particular moment in time.

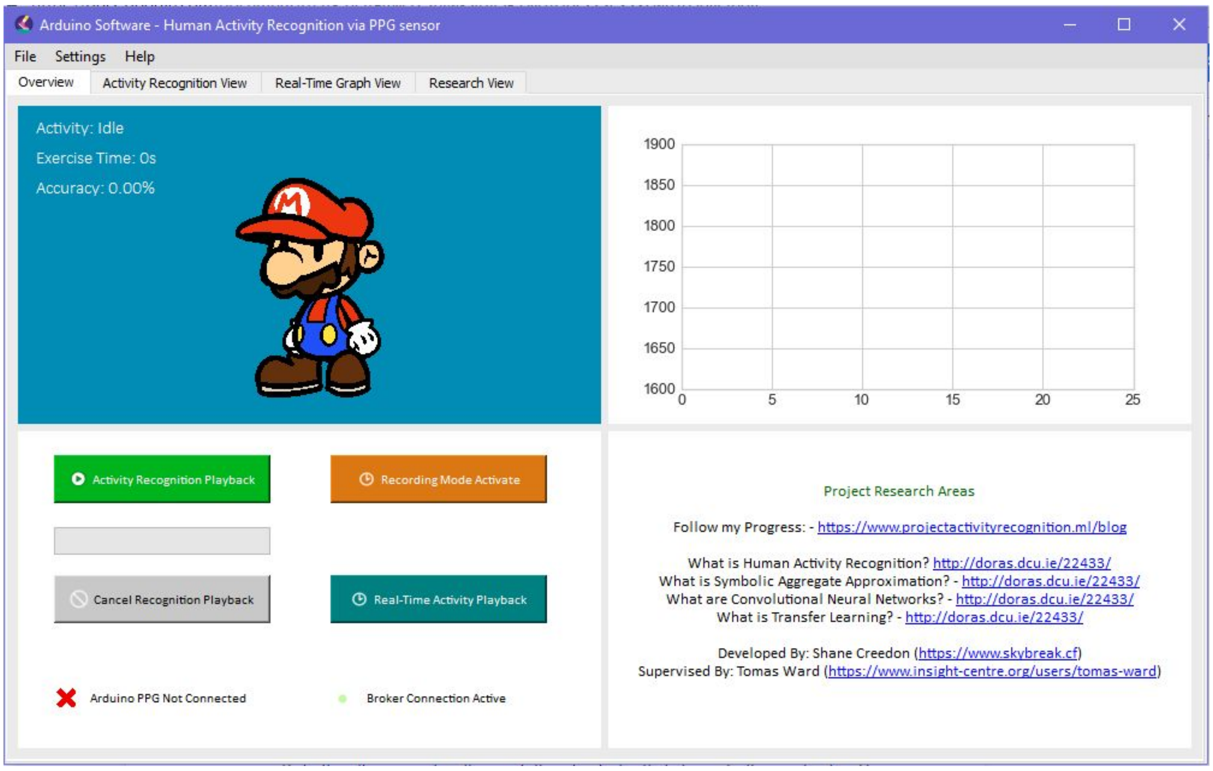
4. When the previous step is activated, all the buttons in the controller pane will be 'greyed out' indicating they cannot be clicked. This is the case for all buttons except for one labeled 'cancel activity playback'. This button can be clicked by users to cancel an in-progress playback function. This is crucial to end the process early and try out other aspects of the application.

5. Once there is an active arduino PPG connected to the correct port, users can record that activate connection using the 'Recording Mode' button available in the controller pane. This will take the microvolt signal generated and record it into a csv file, specifically labeled: 'voltages01/02/03.csv' etc. The more recordings the user takes, the higher the increment. Such that files will not be overwritten. This file will be stored in the downloads section of a users file structure.

6. With an actively connected arduino PPG device, users can check what activity they are doing in real-time. This feature is only working to a degree and still has problems that need to be ironed out.

7. Users can read the research on the project and learn more about the inspirations from where it came.

Fig 3.0 - Desktop Application Overview Screen


Arduino Software - Human Activity Recognition via PPG sensor
File Settings Help
Overview
Activity Recognition View
Real-Time Graph View
Research View
Activity Recognition Playback
Recording Mode Activate
Cancel Recognition Playback
Real-Time Activity Playback
Arduino PPG Not Connected
Broker Connection Active

Activity: Idle
Exercise Time: 0s
Accuracy: 0.00%

Project Research Areas

Follow my Progress: - https://www.projectactivityrecognition.ml/blog

What is Human Activity Recognition? http://doras.dcu.ie/22433/
What is Symbolic Aggregate Approximation? - http://doras.dcu.ie/22433/
What are Convolutional Neural Networks? - http://doras.dcu.ie/22433/
What is Transfer Learning? - http://doras.dcu.ie/22433/

Developed By: Shane Creedon (https://www.skybreak.cf)
Supervised By: Tomas Ward (https://www.insight-centre.org/users/tomas-ward)

Menu Bar Features:

Users can utilise the menu bar for several different purposes:

1. **File**

   Under the file menu item, there exist the options for 'settings', 'about' and 'exit' each of which should be intuitive for a user to comprehend by naming convention alone. 'Settings' holds the ability to update the port for the arduino connection. Clicking 'About' will pop up another window explaining the mission behind the application in-depth. 'Exit' as the name suggest will close the application.

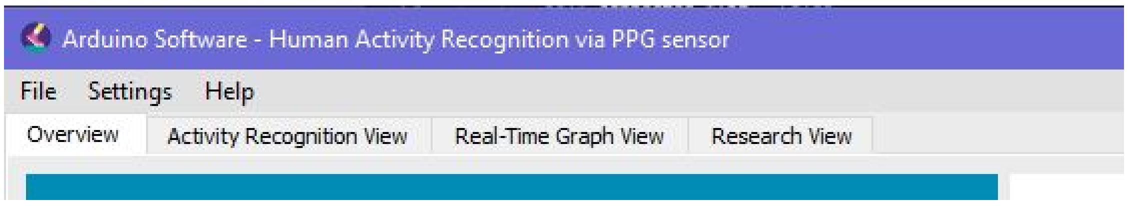
2. **Settings**

   Under the settings menu item, there exists the option for 'port' which provides the same functionality as 'settings' under the file menu. Again, this is just for usability and ease of access of features.

3. **Help**

   Under the help menu item, there exists the option for 'application guide' which as the name suggests provides a pop-up window with instructions and intricacies about using the application successfully.

Fig 3.1 - Menu Bar Items & Tab Items


Arduino Software - Human Activity Recognition via PPG sensor
File Settings Help
Overview
Activity Recognition View
Real-Time Graph View
Research View