Research Manual

Human Activity Recognition

Aaron Finlay, C00226131, 4th year Soft Eng

Supervisor: Greg Doyle

Institiúid Teicneolaíochta Cheatharlach

INSTITUTE of TECHNOLOGY CARLOW

At the Heart of South Leinster

Table of Contents

Abstract

Human Activity Recognition (HAR) is a wide topic of discussion which is used in many areas today. Such areas include, Deep Learning, Video Surveillance Systems, Computer Vision. HAR systems are developed as part of a framework to enable continuous real-time monitoring of human behaviors in a number of areas, including sports injury detection, rehabilitation, elderly care. The use cases for this project are limitless. One use case may be for elderly people in care homes. If they have a fall you want to detect this and be alerted as soon as possible. Another use case might be a setting for a business, detecting whether people washed their hands or not upon entering the building. This proposed project will do exactly that. The purpose of this project is to identify key issues in this area, and to suggest modalities, technologies, algorithms, and techniques based around developing a system capable of recognition of humans, objects, events, and activities. alongside public datasets to develop a working proof of concept.

Prerequisites

In researching Human Activity Recognition (HAR), the findings were that in order to develop a fully functional HAR system, it is important to understand the following, in order to identify useful modalities, technologies and to be able to evaluate their proficiency.

The first important concept to understand is how machines “think” in this area. For example, with the human brain, it consists of billions of neurons which are all interconnected with “synapses”. If there is a considerable amount of synaptic inputs to the neuron, the neuron will fire up also known as an activation of neurons. This process is called thinking, and in order to demonstrate the same thinking for machines, machine learning alongside neural networks are paramount.

The following concepts should be understood firmly before implementing the fundamental theory behind HAR to develop a system capable of identifying human activities.

Machine Learning

ML enables computers to learn, typically, a user feeds instructions to computers so that the machine can act accordingly, well, with ML, a different approach is adopted. Rather than giving the computer instructions on what to do, one can give the machine instructions on how to do a task. For example, a system which classifies types of animals, such as cats, mouses, etc. Rather than going through the code manually and finding unique content for each animal and coding it all up, ML takes in an image, i.e. in the input layer, an image is loaded inside a neural network, it then finds correlating characteristics in the image from its own arsenal of pre-trained, modelled data and this passes into the next layer - the “hidden layer”. This entire process of teaching machines is referred to as training.

Deep Learning

This is a technique used for implementation of ML models. Essentially, Deep Learning uses neural networks to learn, but may also use, but not limited to, decision trees, which is also sometimes referred to as deep learning.

Neural Network

Neural networks can be viewed as a series of doors , where one enters each door one after the other, and the person entering can be considered the “input” and everytime the same person walks through a door, they end up a different person. By the time the last door is opened, the person can be very different. As he or she exits through the last door, that person is then the “output” of the network. So the doors, in this example, represent a layer. Thus, a neural network, therefore, is composed of layers that manipulate the input in some way to arrive at the final layer (Output layer) with an output.

Introduction

In 2008, according to a publication from the HSE, Ireland had an ageing population. Back then 11% of our population were aged 65 years or over (468,000) (HSE, 2008). By 2031 it was projected that proportion will increase to 18%, i.e. to over one million older people. The biggest increase will be among those who are aged 80 years or over. With the impact of COVID-19it is uncertain whether that data will still be as accurate, however, Ireland certainly is ageing as a population. Fall related injuries are a major problem for the elderly. It is estimated back then that one in every three people over the age of 65 years and one in two people over the age of 80 years suffer a traumatic fall every year.

Older people are more likely to suffer serious injuries, disability, psychological consequences and death following a serious fall rather than other age groups. Fall related injuries represent a large expenditure to the HSE and the problem was and still is increasing as our population continues to age. Being able to recognize when an elderly person has fallen is a mission-critical functionality. The elderly person's life may depend on being recognized as falling. Medical attention will arrive sooner, they will have a better knowledge basis of the patient's current condition.

The use cases for human activity recognition are plentiful. One other use case, which has gained prevalence due to the CoronaVirus (COVID-19) Pandemic, might be, that a business wants to detect when a person has entered a building and whether that person has used hand sanitization upon entering. Other use cases may be to prevent fraudulent claims, crowd control, enforcing social distancing guidelines, etc...

The purpose of this document is to evaluate the technologies, techniques, and modalities around modern human activity recognition (HAR) systems and to select the technologies which will be of most benefit to this study. Every topic discussed in this study may or may not be used, but are researched regardless to better understand HAR systems, the intricacies in some systems and where some work well. Essentially a mix & match of different models will be chosen, and this will work in a trial and error manner, which will allow for the algorithms, languages, and libraries that suit this study most to be chosen.

Technologies, Operating System, Topics & Algorithms Researched

Overview of the System

How it works:

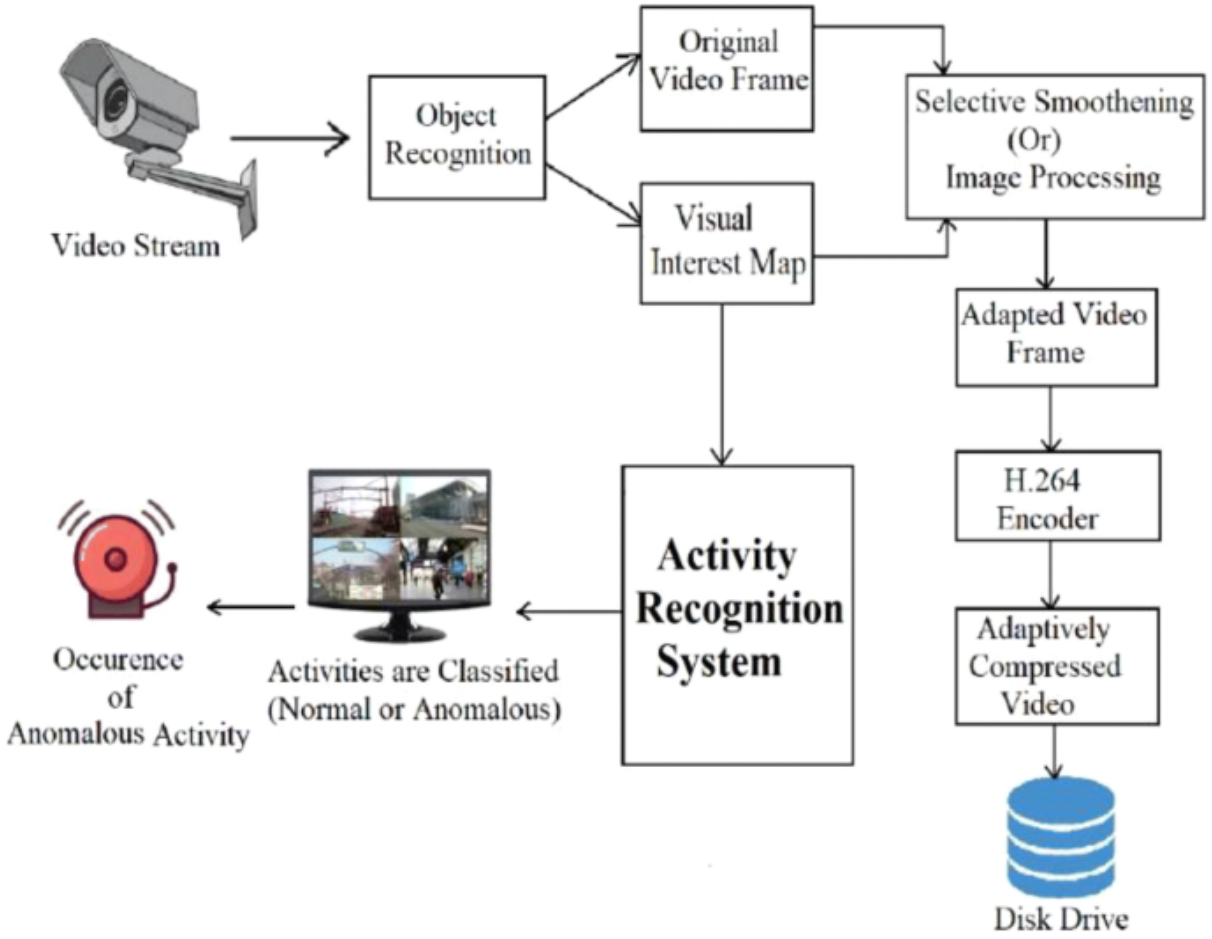

Figure 1: (Shreyas et Al, 2020)

The video surveillance records the activity occurring in the video, e.g. a person falling. Due to the pre-training and modelling of our data, our machine learning algorithm captures and identifies this activity (due to a variety of reasons, e.g. temporal, spatial locality, change in human posture, the degree of the x or y axis changing significantly) to be that of a fall, therefore, the activity in the image is classified accordingly.

If an action or activity occurs that is not fit to be classified according to our pre-trained classifications, it is considered an anomalous activity and is noted. The use of this anomalous classification is a way to mitigate against security

breaches, as an alert can be pulled from the instrumentation set up so that when an anomalous activity occurs in the video surveillance, the instrumentation setup knows to send an alert. The data is encoded, compressed (e.g. using the NumPy array) and stored to the local disk drive the executable is run on. The dataset which is used to train and model our system is referred back to when identifying actions, events, and activities in the image as all of the awareness to identify what is taking place in the images is derived from the pre-processing, pre-training and pre-modelling of our dataset.

Simply put, the system reads in an image in the input layer (the first layer in a neural network), the data structure is tuned and motivated throughout the "hidden layers" (logical layers of the network) the data then takes shape (motivated by the hidden layers) and when it reaches the output layer (the final layer of the network), the neurons which associate themselves with the input feature inside the image the most, are activated meaningfully and then the user will have the analysis and output of the image classified accordingly. We then have a high probability (depending on the accuracy of the data, algorithms etc.) that the feature inside the inputted image matches one of the classifications in our data, which is then outputted in the output layer. One such activity may include a person falling, which in our case, is what we want to identify as much as possible so that in a real world scenario an alert can be sent out quickly.

Convolutional Neural Networks

Neural networks provide people with the ability of having a serverless architecture to perform varied tasks, such as in this case, image recognition and classification. From being able to recognise an activity in an image, such as falling, it provides us with a rich monitoring with the ability to automatically detect patterns and images that are of interest to people.

Let's say somebody wishes to discover if a person is attempting to perform a fraudulent claim, well with video surveillance equipped with the necessary technologies and tools, i.e. a Convolutional Neural Network, we can tell if a person is indeed telling the truth or attempting fraud. For the reasons of CNNs providing invaluable resources to image recognition & classification, CNNs were chosen to be studied and implemented inside this study.

CNN image classifications take an input image, process it and classify it under certain categories (e.g., a human-object detection, such as a person holding an apple). Computers detect the input images by viewing them as an array of pixels and there are also dependencies on the image resolution. Based on the image resolution, the system will see h x w x d (h = Height, w = Width, d = Dimension) (Prabhu, 2018).

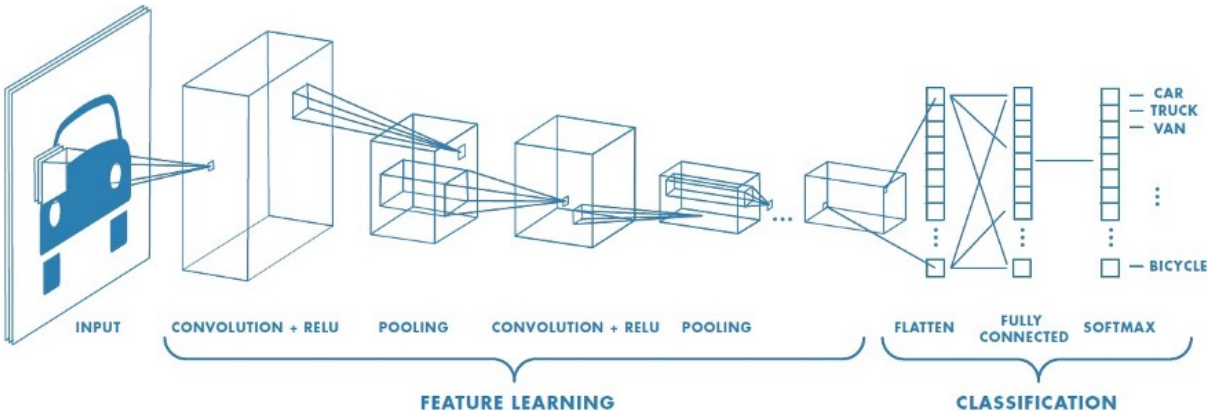

Figure 2: Neural network with many convolutional layers (Prabhu, 2018)

Deep learning Convolutional Neural Network models to train and test, each input image will pass the network through a series of convolutional layers with filters (Kernels), Pooling, fully connected layers (FC) and apply the Softmax function to classify an object with probabilistic values between 0 and 1 (0 being least probable, and one being most probable). The above figure visualised the

complete process flow of a CNN to process an input image and classify the object(s) based on the numerical values in the neurons fired up (Prabhu, 2018).

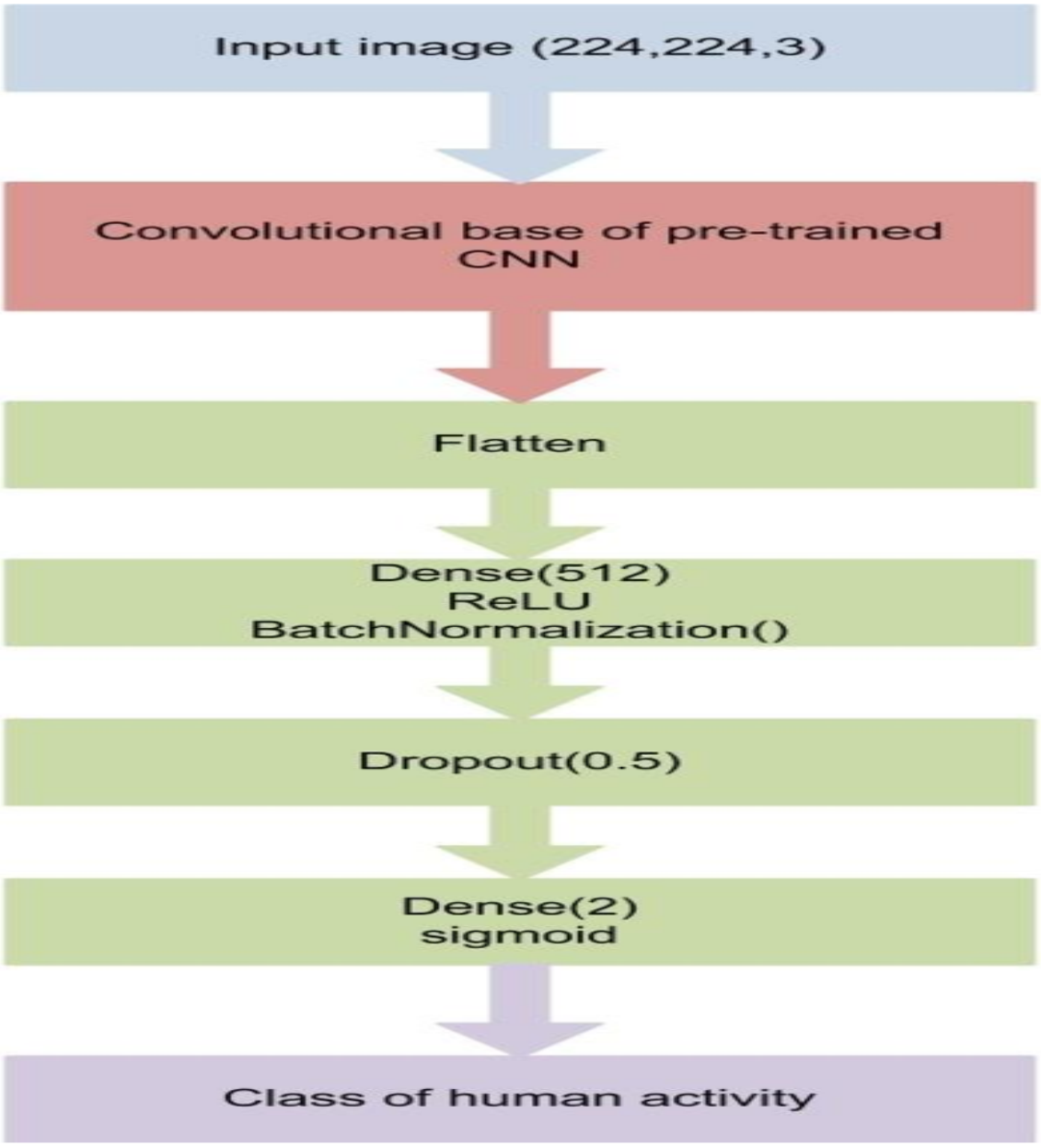

Fig. 1. Structure of used CNN model

Figure 3: Structure of used CNN model (Nikolaev, 2019)

The Convolutional Neural Network (CNN) used in this project is suitable for image recognition. The network is composed of neurons which contain numbers, and these numbers inside the neurons are called the neuronal activations. The first layer (input layer) is where the images get loaded in and for whatever criteria the image fits in, the neurons will fire up (activate) and will move on to the next layer (hidden layer(s)). These layers are based on how we

wish to motivate the structure of the data. What is also important about the network, is the activations in one layer determine the activations in the next layer, that is, each neuron fired up has a connection to the neuron(s) in the next layer, as well as the previous layer, and each neuron connection is assigned a weight value depicting the importance of each connection. Furthermore, the brightness of each pixel in the neuron is based on a loose depiction of the weight's value.

The heart of the network as an information processing mechanism comes down to exactly how the activations in one layer bring about the activations in the next layer. For example, when we feed the neurons in the first layer of the network an image, all neurons in that layer will fire up according to the brightness of each pixel in the image. This pattern of activations causes some very specific neurons in the next layer to fire up.

The brightest neuron for the output layer is the network's choice for what number in the neuron this image represents and the images are all pre-classified with numbers accordingly in order to represent what the image being processed actually is according to pre-historic data.

By knowing all of these details about CNNs it provides the ability to incorporate CNNs into the strategy of developing a fully functional HAR system.

Challenges & Solutions to Human Activity Recognition

Challenges

There are many issues around recognizing human actions and events. Partial occlusion, changes in scale, viewpoint lighting, cluttered background, frame resolution and a multitude of other problems challenge this area. Among various classification techniques, two main questions arise in recognizing events, or activities from humans which are: “what is the action?” known as the recognition problem and the other, “where in the video?” also known as the localization problem. The way in which humans perform activities depends entirely on their habits, and this makes the problem of identifying the underlying activity quite difficult to determine (Vrigkas et Al, 2015).

Furthermore, another issue arises when the system reads in a series of images in a video, how does the system know the difference between an action and an activity, i.e. if a person is running it is considered an action, however, if the person is running from someone, that is considered an activity. Behavioural patterns is another complexity among HAR systems. The way in which people perform certain activities, e.g. shaking hands comes down to their personal habits, the way they express their body movements and could vary entirely among thousands of different images stored in a database.

When attempting to recognize a human activity, one must be able to determine the kinetic state of a person, so that the system can efficiently trace & recognize the activity. It may be harder to detect an activity such as a person peeling an apple. Complex activities may be decomposed into other generic activities, which makes it easier to identify (Vrigkas et Al, 2015).

Human activity recognition is based on dynamic spatio-temporal relations, those of which are the components of more complicated activities and evolve dynamically over time. Furthermore, the description of a single human action and the modelling of the evolution of successive human actions are two major issues in human activity recognition.

Solutions

Typically, the detection of objects in a scene may help to better understand human activities as it may provide useful information about the ongoing event (Vrigkas et Al, 2015). In order to bypass these challenges and develop a functional HAR system, a task is required that consists of three components, collectively named, (i) background subtraction, (ii) human tracking, (iii) human action and object detection.

- (i) **Background subtraction** - in which the system attempts to separate the parts of the image that are invariant over time (background) from the objects that are moving or changing (foreground) (Vrigkas et Al, 2015).
- (ii) **Human tracking** - Whereby the system locates human motion over time.
- (iii) **Human action and object detection** - where the system is able to localize a human activity in an image.

It is important to identify the issues facing HAR, as they will be prevalent to this study, and by mitigating these issues with producing solutions, the development of the system will be more successful.

Events

Events are long-term temporal objects, which typically extend over dozens or hundreds of frames. Polana and Nelson (Polana and Nelson, 1993) separated the class of temporal events into three child classes essentially and recommended individual approaches for modeling and recognizing each:

1. Activities which are temporally periodic but spatially restricted (e.g., a person running).
2. Motion events which are isolated events that do not repeat either in space or time (e.g., a person winking).
3. Temporal textures which are of indefinite spatial and temporal extent, (e.g., lava flowing)

Dynamic events are considered as long-term temporal objects, which are distinguished by spatio-temporal features at multiple temporal scales. Dynamic events can prove to be invaluable for analysis of video data, including event-based video clustering, browsing, segmentation, and indexing. Analysis of events has typically been focused on the recognition of sets of predefined actions or events, or assumed restricted imaging environments (e.g., a fixed viewpoint from a camera), (Zelnik-Manor et Al, 2001).

By understanding the concept of an event and identifying approaches for modelling and recognizing events, it enables the workflow of a fully functioning HAR system.

Human Actions & Activities

An action can be considered like a sequence of **primitive actions** that fulfill a function or simple purpose, such as running, skipping, throwing a ball, etc. An activity contains sequences of actions over the duration of their spatio-temporal relationship, i.e. space and time. An additional feature of activities is that each activity is normally related between one or more people with objects of the surrounding environment.

It can sometimes be quite difficult to identify the difference between actions and activities, a person running may be considered an action, yet in a particular context, if the person was running towards something, it could be considered an activity. For this reason, a lot of the publicly made available datasets do not make the distinction between actions and activities.

Figure 2: Decomposition of Human Activities (Vrikgas et Al, 2015)

The image above illustrates the decomposition of human activities into its subclasses, which makes the process of identifying and isolating an activity easier.

It is important to understand the different activities and which class they belong to as when we train and model our data, since it has to be pre-trained, we need to predetermine actions or activities which are inputted into their correct class, this enables a solution to the common classification problem, i.e. “what is happening in the image?”.

Operating System - Windows 10

After conducting much research on which Operating System (OS) seemed suitable, Linux or a Mac OS distributor appeared to be the more desirable, developer, data science friendly OS. However, due to the variety of technologies being incorporated in this study, to keep things simpler, Windows 10 was chosen due to its user-friendly, popular community, and the ability of this OS to harness all technologies to perform the tasks necessary. For example, Anaconda, Jupyter Notebook, and more are all readily available on Windows 10, just like they are on Linux or Mac.

Data Science Platform - Anaconda

Anaconda is a free and open-source distribution of the Python programming language for scientific computing (data science, machine learning applications, predictive analytics, etc.), that aims to simplify package management and deployment.

Anaconda was chosen as a technology in this project because of its ability to make the importation of modules, packages and more easier since Windows 10 is being used, it can be hard compared to a Linux distribution to handle packages. Furthermore, the Anaconda Navigator application provides a centralised location where multiple apps such as the command line, jupyter notebook, visual studio code and more are stored.

The big difference in using anaconda is that the package dependencies are managed far superior to that of the pp package manager. When pip installs a package, it automatically installs any dependency Python packages without checking if these conflict with previously installed packages. It will install a package and any of its dependencies regardless of the state of the existing installation. In contrast, conda (the anaconda environment) analyses the current environment including everything installed, and, together with any version limitations specified, works out how to install a compatible set of dependencies, and shows a warning if this cannot be done.

Programming Language - Python

The analysis of the research conducted on which programming language would be suitable confirmed that Python would be most efficient. Python is a high level programming language and can be used for a multitude of complex scenarios.

The language is also a general-purpose programming language which means it can be used across domains and technologies.

Python has a large community, with the language being maintained and improved continuously. It is home to a rich stack of libraries, algorithmic functions, and a platoon of rich Data Science / Artificial Intelligence (AI) purposes built in. Some examples include the prebuilt library like Numpy for scientific computations. Another useful library is pandas, an open source library that provides users with easy to learn & utilise data structures and a variety of analytic tools specifically for Python.

One other library within Python that will be certainly used is scikit-learn which is an effective tool for data analytics as Python is used in parallel. It is also open source & a very popular general purpose machine learning library.

Though some minor experience of Python is acquired prior to embarking on this study, the level of comprehension in the language will no doubt be easier & efficient compared to using another worthy but narrow language like R used mostly for Statistical Computing.

Integrated Development Environment - Visual Studio Code

Visual Studio Code (VS Code) was chosen as the Integrated Development Environment (IDE) for this study due to the prior-experience, easy to use, Python Machine Learning / Data Science friendly environment.

VS Code enables the use of the command line interface (cli) while persisting the IDE. VS Code has intelli sense, it is easy to manage package dependencies within the IDE and you can perform powerful debugging, setting a watcher on variables, skipping through code, setting breakpoints, etc.

When loading up VS Code, if you open a Python file and are missing certain imports for modules, package dependencies or you don't have an executable loaded, VS Code will suggest fixes to the code, enabling a quicker, more efficient development of the system proposed, in order to develop a proof of concept for this study.

The knowledge of VS Code when embarking on the project can be considered as mediocre, but will surely improve and be advantageous over the course of this study.

JupyterLab & Jupyter Notebook

The **Jupyter Notebook** application is an open-source web app that allows users to construct and share documents which contain live code, algorithmic functions, equations, comments as well as visualizations.

The use cases for Jupyter Notebook are many, but in this instance, the machine learning application from Jupyter Notebook will be rolled into play. Jupyter supports over 40 programming languages, including Python, R, Scala etc. Notebook enables big data integration, in that you can leverage big data tools, for example, Apache Spark, from Python and R. You can explore the centralised data in your application with other tools such as pandas, TensorFlow, scikit-learn etc..

What makes Jupyter Notebook even more rich is the interactive output which is applicable to users using Notebook. This output type enables producing HTML, images, videos, custom Multipurpose Internet Mail Extensions (MIME) types and more. Why MIME's are useful is that it is an Internet standard that extends the format of email messages to support text in character sets which are not from the ASCII language. Furthermore, MIME supports attachments of audio, video, images and a variety of application programs.

For the use case of this study, JupyterLab is being widely considered due to its advanced Notebook User Interface (UI). This app is a web-based interactive development environment for Notebooks, code, data and more. JupyterLab is flexible, you can configure and arrange the UI to support a wide range of workflows in data science, and machine learning. JupyterLab is easily extensible, modifiable, and modular. The app enables the user to write plugins which add new components and integrate the newer components with previously existing ones.

API - Keras

Keras is an API designed purposely for humans, rather than machines. This API follows best practices in order to reduce cognitive load. How this is done, is by the API offering consistent and simple APIs, it is then able to minimize the number of user actions required for many common use cases, and it provides clear & concise error mages, where the user can take action. It also has a large amount of documentation available to the public along with many developer guides and tutorials.

Keras contains an enormous ecosystem. The API is a central part of the closely connected TensorFlow Version 2.0 ecosystem. It paves the way for machine learning applications as it covers every step of the lifecycle process for a machine learning app, right from data management to hyperparameter training to deployment solutions. In Machine Learning, a **hyperparameter** is a parameter whose value is used to control the learning process.

Keras will provide a Python interface for artificial neural networks, it allows the use of distributed and training of deep-learning models on clusters of Graphics processing units (GPU) and tensor processing units (TPU). It wraps the efficient numerical computational libraries TensorFlow and Theano and essentially allows users to define the constraints of a neural network and train the models belonging to the network. This can all be done in just a few lines of code.

To conclude with Keras, it offers an easy to use, widely supported API. Keras makes deep learning a simpler process, and while the deep neural networks may be the hot topic of discussion, the complexity of the major frameworks can be considered to have hindered the experience of developers new to machine learning. Keras is a leader in high-level neural networks APIs and with there being a bundle of resources available using Keras, this is a paramount technology in this study.

Library - OpenCV

Open Source Computer Vision Library also known as OpenCV is as the name says, an open source computer vision and machine learning software library. It was built to provide a more common infrastructure for computer vision applications and to anticipate and accelerate the use of machine perception as such in the commercial platform of products.

OpenCV-Python is a library of Python bindings which were designed purposely to solve computer vision challenges. This technology combination makes use of the essential Numpy feature. Essentially, Numpy is a highly optimized library used for numerical operations with a MATLAB style syntax.

Essentially, Open-CV hosts a cross-platform, free for use (under a licence), made easy for developers to utilize the full stack of its power, including modifications to code, supported deep learning framework technologies such as TensorFlow, PyTorch, and various other frameworks.

The Open-CV library contains a vast amount of optimized algorithms, estimating at 2,500. What it also includes is a comprehensive set of state of the art machine learning algorithms which will be vital for the success of this study.

Furthermore, what also lies in the Open-CV armory is a vast amount of optimization algorithms, designed purposely for computer vision, and machine learning. Such algorithms may be Artificial neural networks, like the CNN which will be used in this proof of concept. Support vector machines (SVM), Deep neural networks (DNN), Gradient boosting trees, Decision learning trees and more are also provided with this library.

Library - TensorFlow

TensorFlow is an end-to-end open source machine learning platform used in an array of applications within the context of machine learning, deep learning, computer vision, etc. The TensorFlow framework was developed by Google purposely for creating Deep Learning models. Deep learning is a category of machine learning (ML) models (algorithms) that use multi-layer neural networks, i.e. input, hidden and output layers are what the neural network is composed of (Kofler, 2020).

Building and training machine learning (ML) models can be a staggering process at times, but with TensorFlow, the process of building ML models is easy in comparison to some techniques, as it models the data easily using advanced high-level APIs with modern technologies like Keras, which grants the user eager execution, which makes the process of model iterations more efficient, quicker (producing faster iterations of a product ensures maximum proficiency due to constant feedback for every iteration and incrementally, the project is decomposed into short, quicker deliverables, and it is easy to debug.

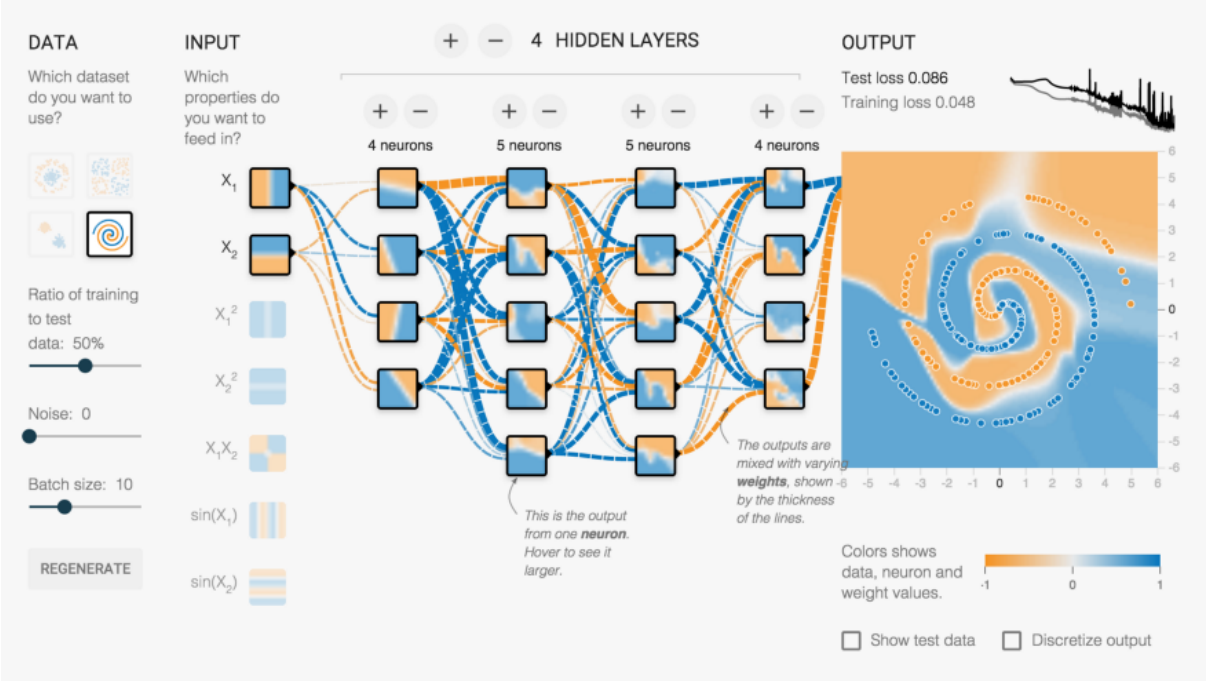

Figure 4: Deep learning concept of Tensorflow (Kofler, 2020).

In this image, the process of deep learning is illustrated, whereby, an image is inputted in the input layer of the neural network, this then passes through the "hidden" layers, arriving at the output layer, where the image is classified accordingly.

Dataset - Kinetics 700

Finding the right dataset is an essential part of this study, in order to check if our system works as intended, we need test data to run it against, which is what Kinetics (Version 700) will provide us with. A good dataset should contain data to proportionally represent the inputted data being run against the system so that the output is accurate thanks to the pre-trained modelling of the test data being used.

The dataset for this study has to be carefully chosen so that there is enough data representation for each classification of activities involved. For that reason, Kinetics was the perfect choice for our test data. This hosts a collection of large-scale, high-quality datasets of URL links up to 650,000 video clips that cover 700 human action classes in the version we are using.

The videos include human-to-object interactions such as a person eating an apple, as well as human-to-human interactions such as shaking another's hand. Each action class holds approximately 700 video clips. Each clip is a single action class with a human annotating the behaviour and the duration is around 10 seconds.

Performance Metrics Evaluation

Once the construction of a ML algorithm has been completed, the next phase is to evaluate just how effective the model is based on the metrics and datasets. Varied performance metrics are used to conclude different ML algorithms. E.g., a classifier used to distinguish between images of different objects, we can use classification performance metrics, for example, Log-loss, Average Accuracy, etc. If the ML model is trying to predict housing prices, a function such as the root mean squared error - RMSE can be implemented to calculate how efficient the current model being used is. Not only are there different metrics used to measure the efficiency of algorithms, but also the test data being consumed duration of the study.

The vast majority of the importance of choosing the correct dataset in context with performance metrics comes down to the fact that the process of training the ML model has already tuned the expected outcome to the training dataset. Thus, in order to estimate the generalization error, the model must test a dataset which it is not yet aware of, therefore, gives life to the term of testing dataset.

Classification models predict the class labels for given input data, i.e. which class does the object in this image belong to. Accuracy is the measure of the correct % of predicted images. In other words, it is the ratio of the units of correct predictions and this is compared to the total number of predictions made by the different classifiers. Although, it is important to note, that accuracy is not always the solution for giving us the broader picture of the cost of misclassification or whether the testing data set was balanced or unbalanced, i.e. too much data on running, not enough on walking. This gives birth to the reasoning of why confusion matrices are used, which treats the failures of each image differently.

In order to understand confusion matrices, one must first understand the terms - **False Positive** & **False Negative.** For example, if a classifier labels a non-falling image as a falling image, it is the case of False Positive. Alternatively, when a classifier identifies a falling image as a non-falling image, it would then call an instance False Negative. The confusion matrix for any model created, takes into account all these details and offers a clear picture of incorrect & correct classifications for each class of objects.

Often enough, standard binary classification performance metrics can be easy to evaluate, e.g. a clear cut 0 signifying how unlikely the image is compared to the stored pre-trained data, and 1 signalling the image inputted is classified to that of a class local to the system. However, when the output of the classifier isn't as straight forward, and it is a numeric probability value signifying the accuracy of how likely that data is belonging to a certain class. In this scenario, a good performance metric is the **Logarithmic loss (log-loss).** This is a starter in measuring accuracy accounting for the probability of the confidence (where the data belongs to) of the classifier model. In order to improve the efficiency of the classifier, one must first minimize the cross-entropy of the model.

Essentially, the cross-entropy is a measure of the performance of a classification model whose output is a probability between 0 and 1. Essentially, it is a measure of the different numerical values between two probability distributions for a chosen or given variable which may be random or even a full set of events. The information produced and split into layers, to identify an image of it in the input layer, is what quantifies the number of bits required to encode and transmit an event. Cross-entropy loss increases as the predicted probability sways further from the actual classifier. As the predicted probability deflates, it is important to note, the log loss increases rapidly.

Algorithmic Description

For the maximum probability of success for creating a HAR system, with a working proof of concept, a various amount of ML algorithms were researched to evaluate which one(s) will be of most value to our needs. Furthermore, a deep dive analysis was taken on identifying which algorithms were better in performance criteria. Some may not yet have been investigated due to the multitude of models available.

It is known that an algorithm is a set of instructions where the main goal is to solve a specific problem, computation, etc. However, ML algorithms have a different definition. Instead of giving the computer instructions, in ML, we aim to train the computer to understand things like images inputted, and to know that image belongs to the correct classification.

Broadly, there are 3 types of Machine Learning Algorithms (Ray, 2017):

1. Supervised Learning - included, but not limited to:
    a. Support Vector Machines (SVM)
    b. Neural Networks
    c. Random Forest Models
    d. Regression
    e. Classification
2. Unsupervised Learning - included, but not limited to:
    a. Hidden Markov models
    b. Hierarchical clustering
    c. Self-organizing maps
    d. Gaussian mixture models
    e. k-Means clustering
3. Reinforcement Learning - included, but not limited to:
    a. Deep Q Neural Network (DQN)
    b. Q-learning or value-iteration methods
    c. Proximal Policy Optimization (PPO)
    d. Hybrid
    e. Policy Gradient (PG)

1) How **Supervised Learning** works (Ray, 2017) - the algorithm consists of a target / outcome variable which is to be expected from a given set of independent variables (predictors). Utilising these sets of variables,

enables the construction of a function that maps inputs to the desirable outputs. The process known as training then continues until the model attains a preferable stage of accuracy on the training test data. Such examples can be shown in Random Forest, Logistic Regression, Regression, Decision Tree etc.

2) **Reinforcement Learning** operates by training the machine to make specified actions and decisions. Logic behind it: the machine is revealed to an environment, one can describe it as a virtual laboratory environment type filled with trial and error, by this process the machine trains itself and learns every time. It uses past experiences to try to capture the best possible knowledge to make an accurate decision. Such examples may be shown in the algorithm - Markov Decision Process (Ray, 2017).

3) **Unsupervised Learning** - in this algorithm, there are no targets or outcome variables to estimate. It is used widely for clustering population in different classifiers, which is used mainly for segmenting customers from different classes for specific action. An example is in the Apriori algorithm, or even K-means (Ray, 2017).

There are a large number of algorithms that can be incorporated into the studies use case, but this study focuses particularly on a couple of algorithms.

Logistic Regression

A deceiving name for it's true purpose. Rather than being a regression algorithm, it is actually a classification one. This algorithm estimates discrete values (binary values such as, 0/1, true/false, etc.) based on the provided set of variables. In other words, it estimates the probability of occurrence of an event by putting data to a logit function. Thus, this algorithm is also known as logit regression. As it is trying to predict the probability, the output value will always lie in the range between 0 and 1 (Ray, 2017).

In the mathematical point of view, the log odds of the outcome is modeled as a linear combination of the variables purposely for predicting.

$$odds = p / (1 - p) = probability\ of\ event\ occurring / probability\ of\ event\ not\ o$$

$$ln(odds) = ln(p/(1 - p))$$

$$logit(p) = ln(p/(1-p)) = b0 + b1X1 + b2X2 + b3X3.... + bkXk$$

In the above formulae, p represents the probability of containing the characteristic value that is of interest. It selects parameters that it thinks will optimize the likelihood of observing the sample values rather than decrease the whole set of squared errors like in common - ordinary regression.

Overall, Logistic Regression uses a varied method for estimating the parameterization values, which yields more efficient results - clearer meaning it is more unbiased, with a reduction in variances. It enables users to reference this technique when attempting to understand specific functions, like logit link functions.

Support Vector Machine

The objective of the support vector machine (SVM) algorithm is to find a hyperplane in an N-dimensional space(N - the number of features) that distinguish the different data points that classify the feature. A SVM is a supervised machine learning algorithm that can be used for both regression and classification tasks, but is used mainly in classification objectives (Gandhi, 2018). To separate the classes of data points chosen, there are a multitude of hyperplanes that could possibly be chosen. The goal is to find a plan that offers the maximum margin, i.e. the maximum distance between the data points of the classes. Maximizing the margin distance enables some reinforcement so that future data points can be classified accordingly with more confidence.

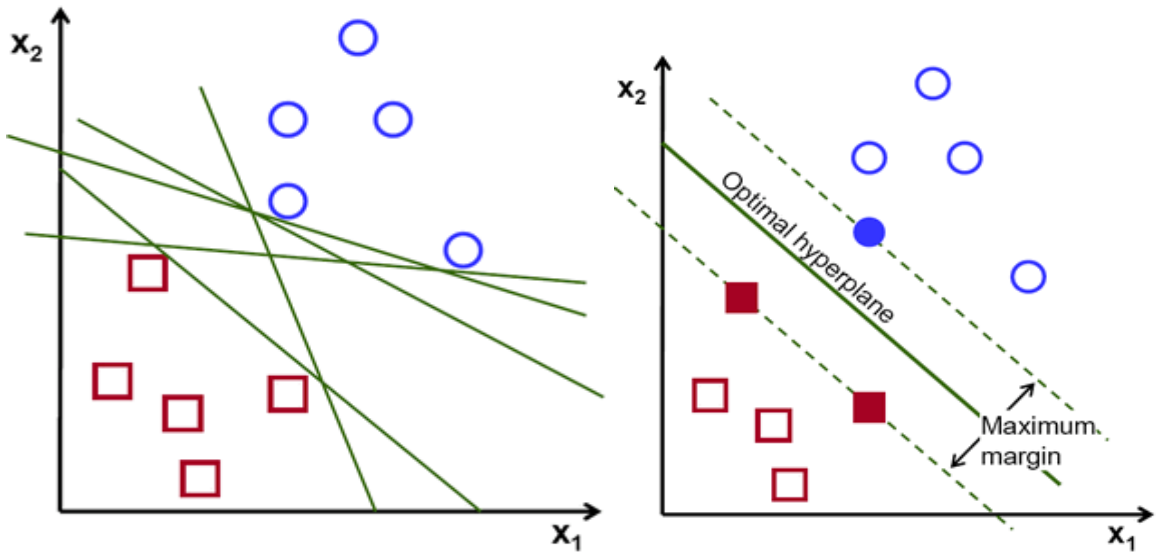

Figure 5: Possible hyperplanes (Gandhi, 2018).

Essentially, hyperplanes are decision boundaries that help classify data points. Data points which lie on either side of the hyperplane can be specified to different classes (Gandhi, 2018). What is also important to note is that the dimension of the hyperplane depends upon the number of features. For example, if the number of input features is 3 then the hyperplane becomes a two-dimensional plane. If the number of input features is 2, then the hyperplane is simply a line. It is difficult to imagine the hyperplane if the number of input features exceeds 3. **FiF**

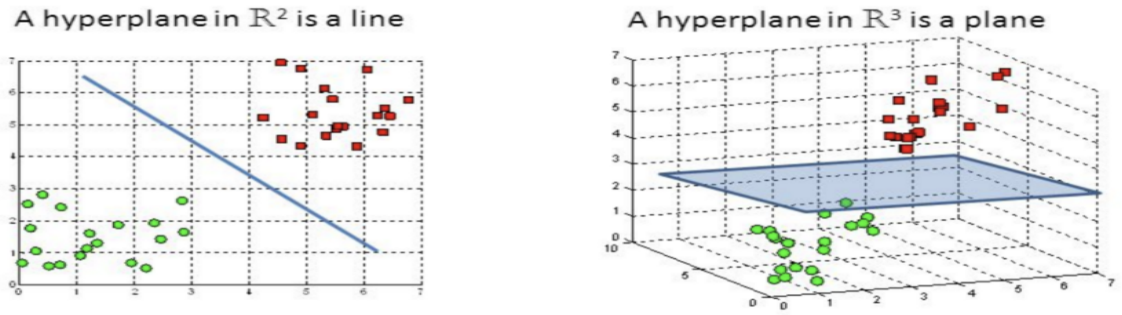

Figure 6: Hyperplanes in 2D and 3D feature space (Gandhi, 2018).

In order to find the correct hyperplane, segregation of the two classes within the data is the best option. The distance between the hyperplane and the nearest data point from either set is known as the margin. The objective is to select a hyperplane with the largest possible margin and any point within the training set, thus giving the SVM a greater chance of new data being classified accordingly. If there is no clear hyperplane, it is best to move away from a 2D view of the data to a 3D view. For example, imagine that there are two sets of cubes which are sitting on a sheet and this sheet is lifted suddenly, launching the cubes into the air. When these cubes are up in the air, the sheet is used to separate them. This ‘ascension’ of the cubes represents the mapping of data into a higher dimension. This process is known as kernelling. Hence, the idea is that the data will continuously be mapped into higher dimensions until a hyperplane can be formed to segregate it.

K-means Clustering

K-means clustering is widely used in unsupervised machine learning algorithms. Usually, unsupervised algorithms make inferences from datasets

using only input vectors without referring to known, or labelled, outcomes. The objective of K-means is to group similar data points together and discover underlying patterns. To achieve this mission, K-means scans for a fixed number (k) of clusters in a dataset. A cluster is a collection of data points aggregated together because of certain characteristics shared (Garbade, 2018).

The user will define a target number k, which is the number of centroids needed in a dataset. A centroid is an imaginary representation of the centre of the cluster. Every data point is allocated to each of the clusters through a reduction in the in-cluster sum of squares. Basically, the process of the K-means algorithm is defined as: identifying k number of centroids, allocate each data point to its nearest cluster, meanwhile, maintain the centroids to be as small as possible (Garbade, 2018).

The means part of “K-means” refers to the averaging of the data, meaning, finding the centroid. To process the learning data, the K-means algorithm in data mining starts first, with a group of randomly selected centroids, which are then used as the entry points for every cluster, and then iterates repetitively, until the algorithm summises the calculations needed to optimize the positions of the centroids.

The algorithm ceases making and fine tuning clusters when either, the centroids have stabilized, i.e there is no change in their values because the clustering operation was successful, or the defined number of iterations has been achieved (Garbade, 2018).

Overall, K-means is an easy to understand algorithm, with plenty of tutorials out there to train the user. Furthermore, it delivers training results quickly. However, the performance of K-means is usually not as competitive as the other variations of the sophisticated clustering techniques due to slight variations in the data could lead to high variance. What must be noted is that clusters are assumed to be shaped spherically and evenly sized, something that could hinder the accuracy of the K-means clustering results (Garbade, 2018).

There is an arsenal of algorithms that can easily be integrated into this study, but for the meantime, the use of CNNs, SVMs, Logistic Regression and K-Means clustering will be widely adopted.

Conclusion

In this document, there has been a set of objectives given to the system proposed. Alongside this, the different technologies, how they fit into the picture, their usefulness, what they integrate with, have also been discussed. There are a variety of models out there for developing a HAR system, capable of recognizing images such as people falling, classifying this image appropriately, but choosing the right model was one of the most challenging tasks. With there being a variety of open source models, frameworks at disposal, choosing the first and easiest option is not always the most quantifiable approach. Similar to object oriented design patterns, it is important to evaluate each part of the stack, and choose which ones offer the optimal solutions, be it overhead, some of the metrics described in the Performance Metrics section, and the accuracy of the set of modalities.

The depth of the theory around this area was truly identified when beginning research, as the theory behind HAR is significant. It was also challenging to learn at the same time the basic principles, models, and implementations of machine learning, so that the learnings could be incorporated into this use case. However, the minor experience of Python, with Anaconda, VS Code, helped significantly as it was less difficult to install modules, import packages and follow tutorials, Additionally, what also helped was OpenCV library, with the wide amount of documentation, tutorials, and easy integrability with Python, the learning curve was reduced.

Each model offers a different range of accuracy, by investigating each algorithm, technology, it enables a view which displays which technologies are better integratable with other counterparts, i.e. python works very well with OpenCV for machine learning applications, such as this studies scenario. It’s also important to find a dataset with enough data representation for each class inside the dataset, so that when an image feature is inputted, the different classifiers are capable of identifying and isolating the image and the class the image belongs to.

The problem with developing HAR systems is not the actual coding phase, or even selecting the correct algorithm. Most of the time it comes down to the actual live data being tested against. If there is a lack of representation in a

specific activity - the algorithm(s) used will be less effective, the more data fed to the algorithm, the better it becomes, this is the same concept for HAR.

The development of a HAR system has proven in the analysis of this study not to be the most difficult phase, but gathering the data and training the system to work with it, was actually a harder task. However, that task was not as difficult as the beginning, with all the researching, coding tutorials, and a big learning curve, it felt as though the horizon was far, but over time proved to be doable. The theory learned, tutorials completed, along with the Python code integrated with various machine learning libraries and frameworks will prove to be invaluable for the understanding around HAR.

To conclude, the area of human activity recognition is vast and ever-emerging, with there being an arsenal of technologies at a user's disposal, the challenge nowadays may not be development of a system, but rather an efficient and elegant design. One which caters for problems, is accurate and has good classifiers. When attempting to create a HAR system, a series of technologies, modalities, public datasets, all must be investigated in choosing the most intuitive models for training and modelling the data at hand.

Bibliography

Vrigkas, M., Nikou, C. and Kakadiaris, I., (2015) *Review Of Human Activity Recognition Methods* Available at: https://www.frontiersin.org/articles/10.3389/frobt.2015.00028/full (Accessed on 10th October, 2020).

Zelnik-Manor, L. and Irani, M. (2001) "Event-based analysis of video", *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition.* Available at: https://ieeexplore.ieee.org/document/990935 (Accessed on 14th October, 2020).

HSE, (2008) *Strategy to Prevent Falls and Fractures in Ireland's Ageing Population.* Available at: https://www.hse.ie/eng/services/publications/olderpeople/strategy-to-prevent-falls-and-fractures-in-irelands-ageing-population.html (Accessed on 17th October, 2020).

Prabhu, (2018) *Understanding of Convolutional Neural Network (CNN) -- Deep Learning.* Available at: https://medium.com/@RaghavPrabhu/understanding-of-convolutional-neural-network-cnn-deep-learning-99760835f148 (Accessed on 19th October, 2020).

Polana, R. and Nelson, R. (1993) "Detecting activities", *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition.* Available at: https://ieeexplore.ieee.org/abstract/document/341009 (Accessed on 21st October, 2020).

Nikolaev, P. (2019) "Multi-Label Human Activity Recognition on Image using Deep Learning", *Proceedings of the 7th Scientific Conference on Information Technologies for Intelligent Decision Making Support (ITIDS 2019).* Available at: https://www.atlantis-press.com/proceedings/itids-19/125908972 (Accessed on 22nd October, 2020).

Kofler, M. (2017) Deep Learning with Tensorflow: Part 1 — theory and setup. Available at:

https://towardsdatascience.com/deep-learning-with-tensorflow-part-1-b19ce78

03428 (Accessed on 28th October, 2020).

Ray, S. (2017) Commonly Used Machine Learning Algorithms | Data Science Analytics Vidhya. Available at: https://www.analyticsvidhya.com/blog/2017/09/common-machine-learning-algorithms/ (Accessed on 2nd November, 2020).

Gandhi, R. (2018) Support Vector Machine --- Introduction to Machine Learning Algorithms. Available at:https://towardsdatascience.com/support-vector-machine-introduction-to-machine-learning-algorithms-934a444fca47 (Accessed on 4th November, 2020).

Garbade, M. (2018) Understanding K-means Clustering in Machine Learning. Available at: https://towardsdatascience.com/understanding-k-means-clustering-in-machine-learning-6a6e67336aa1#:~:text=5%20min%20read-,K%2Dmeans%20clustering%20is%20one%20of%20the%20simplest,popular%20unsupervised%20machine%20learning%20algorithms.&text=In%20other%20words%2C%20the%20K,centroids%20as%20small%20as%20possible. (Accessed on 7th November, 2020).

Plagiarism Declaration

*I declare that all material in this submission e.g. thesis/essay/project/assignment is entirely my/our own work except where duly acknowledged.

*I have cited the sources of all quotations, paraphrases, summaries of information, tables, diagrams or other material; including software and other electronic media in which intellectual property rights may reside.

*I have provided a complete bibliography of all works and sources used in the preparation of this submission.

*I understand that failure to comply with the Institute's regulations governing plagiarism constitutes a serious offence.

Student Name: (Printed) AARON FINLAY

Student Number(s): C00226131

Signature(s): Aaron Finlay

Date: 30th of April, 2021