# ManDown - A Social Drinking Management App

Diyar Alyasiri [da1313], Michael Hart [mh1613], Daryl Ma [dm2913], Max Poynton [map213],
Santiago Rubio [spr13], Arshan Shafiei [as2413], and Wei Cong Te [wct113]
GitHub: https://github.com/michael-hart/mhml_sensor_data

***Abstract*— Social drinking has become an accepted part of modern culture, and with that comes the possibility of intoxication due to overdrinking. In order to provide a safer drinking environment for social drinkers, our ManDown app would manage the alcohol intake of the user by providing a non-invasive measure of their blood alcohol content (BAC) levels. The app would then give a classification for the user indicating one of three states - drunk, tipsy, or sober. This report details the different components required in the design of the application. The overall system design, incorporating the data capture, gamification inputs, data management, machine learning and application design is discussed. An analysis into the accuracy of Mandown in its classification, as well as a study into its ease of use is also performed. Our results show that an accuracy of up to 96% for intoxification classification is possible, and that gamification inputs on diagnostics improves upon the accuracy, as well as being attractive for users.**

## I. Introduction

Social drinking has important sociocultural functions in modern day society. However, risks arise when people drink over the limit. Alcohol-related deaths in the UK have risen by 13% over the past decade, while hospital admissions due to alcohol-linked injuries have increased by 33% [1]. This report aims to address these tragic statistics with the development of ManDown, a social app with an emphasis on managing the alcohol intake of a user.

The approach suggested through this report would be to detect intoxication through the use of a smartwatch and a smartphone. In this case, intoxication is defined as the BAC threshold of 0.08% as per the regulations in the UK. When intoxication occurs, the body is affected through the reaction time, balance, walking gait as well as arm movements. Through a combination of passive sensing of physiological movements, as well as active inputs through gamification by the user, this report shows that it is possible to detect intoxication from a smartphone and wearables by monitoring these diagnostics. Additional features such as alerts and emergency calls are also implemented through the UI to manage situations when intoxication has occurred.

## II. Background

### A. Related Work on Smartphone Intoxication Detection

Related work on intoxication detection came from 3 main sources. These sources were useful in proving the reliability and accuracy of using passive sensing for intoxication detection, and was key in the design of the sensor. The first source was the field sobriety tests [2], as mentioned in the design report [3]. The Walk-and-Turn Test and the One-Leg Stand Test mentioned in [3] detects ataxic gait specifically. A report submitted to the US Department of Transport[4] for evaluating the accuracy of these tests showed that these tests were able to **discriminate BAC levels above and below 0.08% with an accuracy of up to 94%.**

*Arnold et al.* [5] displayed an app developed for identifying and displaying a user's level of alcohol intoxication based on their gait. The accelerometer in a smartphone was used for sensing the movements of a user, before machine learning was applied on the data obtained in order to produce an accurate prediction from the gait analysis. Their results displayed that there is **an average of 56% accuracy for classification**, which was impressive considering that a large number of factors influence gait. A limitation of their experiment was the lack of use of additional sensors in the phone.

*Nassi et al.*[6] investigated the use of several wearable devices such as a Google Glass, LG G-watch, Microsoft Band and a Samsung Galaxy S4 to detect intoxication levels. The motion data for each sensor is used in tandem with machine learning to provide a binary classification (drunk or sober). The BAC levels are also used to provide an indicator of the intoxication levels. Their results showed that combining various wearable devices for gait analysis can improve the detection of intoxication. However, the results obtained from just the use of a smartwatch and a smartphone were almost as reliable. This provided **a rationale for the wearables for intoxication detection to be a smartphone-smartwatch combination.**

### B. Related Mobile Apps

The idea of using everyday devices like a smartphone and a smartwatch to diagnose intoxication symptoms is a relatively novel idea. Most phone apps for alcohol management involve active inputs; the user constantly updates the app with information (units of alcohol drunk, time of consumption). *DrinkFree - Sobriety Counter* [7] is one such app that is marketed at drinkers, but has a **limitation on the possible information that can be gathered from an inebriated user.** By adding on diagnostics to our app, we hope to bypass this limitation to provide a greater level of convenience to the user.

A social app that is useful in providing a guide to our project is *GoodSam*[8]. The aim of this app is to provide first aid as quickly as possible to someone in danger. During an emergency, any app user in the vicinity of an affected person can react quickly through a notification from the app. **This**

**idea is something that we intend to implement through the social aspect of our app, to provide a level of safety to incapacitated users.**

## III. Research Hypothesis

The following hypotheses will be tested:

1) Data collected through a standard smartphone and wearables will provide good indicators for the level of intoxication of the user.
2) Users are more likely to regulate and reduce the amount drunk when using the app than without.
3) Gamification improves the accuracy of diagnostics compared to passive sensing alone.
4) Gamification improves user enjoyment and engagement compared to simple user inputs.

To further elaborate on the latter hypotheses, games have been included in our app to enhance its appeal to the user and entice them into providing data. This is as opposed to non-game based user input, for example a menu to input what type of drinks user drank and how many.

## IV. System Design

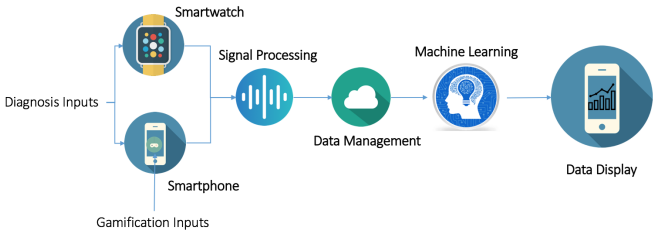



Fig. 1: System Diagram

### A. System Overview

The overall system consists of a smartwatch and a smartphone, with a mobile app encompassing both devices. The diagnostic inputs consist of sensors used in aforementioned devices to provide gait analysis data, while active gamification inputs are used to provide reaction time and balancing data.

These values are then stored into a cloud-based data management system for machine learning to be applied. The output of the machine learning is then stored back into the data management system, from which the data would be displayed back for the user in the UI.

The entire project is split into several modules to achieve easier integration. The modules consists of **Diagnostic Inputs**, **Gamification Inputs**, **Data Management**, **Machine Learning** and a **User Interface**. Each of these modules will be described in greater detail in the remainder of the section, in terms of their purpose, features and implementation.

### B. Diagnostic Inputs

The purpose of the **Diagnostics Inputs Module** is for diagnosing gait-based intoxication symptoms for ManDown, and be used to test the hypothesis that through the use of a smartphone and wearables, it is possible to accurately detect intoxication. The implementation for both of the devices would be explained below.

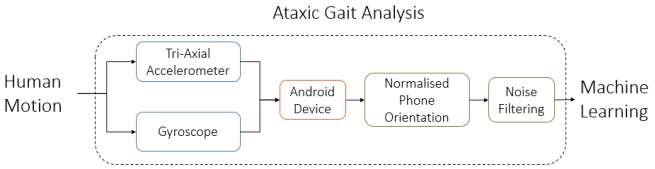



Fig. 2: Implementation of Ataxic Gait Diagnosis applied for the smartwatch and smartphone sensors

*1) Smartphone Sensor:* **Figure 2** illustrates the flow diagram for the ataxic gait diagnosis. The accelerometer and gyroscope sensors detect motion in the x, y and z domain, and the output signals from the accelerometer would provide an idea into the motion of the user. The gyroscope sensors would track the rotation or twist, and would be useful for normalizing the accelerometer data readings such that the gait features could be characterised regardless of the orientation or placement of the smartphone.

The machine learning classification would involve training the model to recognise a normal walking gait and differentiating it from ataxic gait. A normal walking gait would be expected to be periodic in nature, as walking is a repetitive motion. This fundamental frequency would be expected to be around 1-5Hz[5], which is verified from the experimental data.

As stated in [5], the average step length, step time as well as body sway area would change as the amount of alcohol ingested increases. In order to determine this, time-series windows of around 50s in length will be used for monitoring a number of characteristics: The number of steps taken, average step length, average time between the steps, average velocity, cadence and skewness of these signals. These are chosen based on contributions from the projects cited above. Our project would then analyse these factors to note whether a correlation with BAC levels exist. This would allow us to evaluate the hypothesis that the sensors on a smartphone and a wearable would be sufficient for intoxication detection.

*2) Smartwatch Sensor:* The purpose of the smartwatch sensor is to determine intoxication symptoms from a user's arm motion. As mentioned in [6], arm motion can be used to supplement gait detection on top of just using a smartphone. Features which were extracted from the data include frequency domain features such as SNR and top 4 frequency values as well as statistical features such as mean and variance.

An additional feature that was added was for user input to ManDown with information display via the wearable. This is possible with a smartwatch such as the Moto 360 provided

to us. This will allow for easier input of information such as drinks consumed to the main app as well as convenient display of information to the user such as intoxication levels.

*Wang et al.* found that the human body's range of motion is constrained to frequencies at or below 20Hz[9]. According to the Nyquist theorem, the required polling rate for the accelerometer is therefore at least 40Hz. To err on the side of caution, it was set to be at approximately 100Hz. The reason for this being that the polling rate is highly approximate due to the way in which Android interfaces with the sensors. Developers are allowed to set a polling period for the sensors (in our case, 10ms), but this period is only a suggestion for the system rather than a set rate. During testing, it was found that the data would be recorded at a non-deterministic rate which fluctuated around 100Hz (as much as $\pm$20Hz). At its slowest, 80Hz would still be well above the request Nyquist rate. It was concluded that a constantly changing polling rate will not adversely affect the results, especially when existing research have been successful in using such sensors in gait classification [5][6].

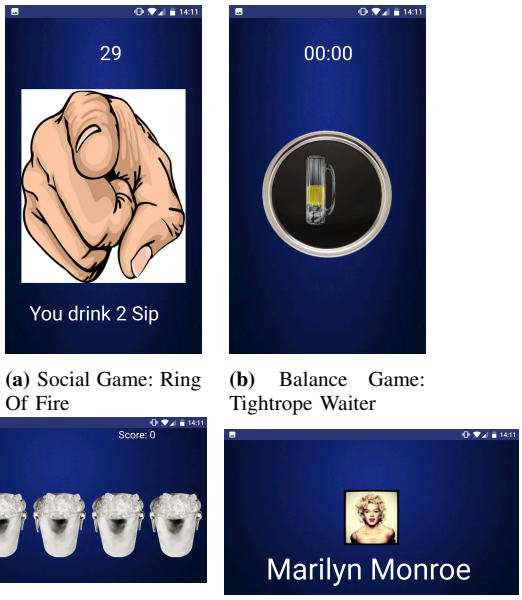
The implementation of the data collection is influenced by clean data collection and battery life. It was noted that first generation Android smartwatches such as this Moto 360 have poor battery life. This battery life is significantly affected by keeping the sensors on, constantly polling at a high rate. In terms of clean data collection, ideally the data from the smartwatch should only contain gait data. That is to say, it only records when the user is walking.

As a result, the final version that has been implemented is suitable only for testing purposes. A commercial version will require further tweaks which will be discussed in future works. Currently, the watch is programmed to start recording for 12 seconds whenever requested by the phone. Simultaneously, the phone will also record 12 seconds of sensor data in the user's trouser pocket. After 12 seconds, the data is all sent to the phone. Only the latest set of data is ever stored on the watch to minimise the impact on the Moto's limited storage.

### C. Gamification Inputs

The **Gamification Module** serves two purposes: to provide active user input for reaction time and balancing symptoms, while making the app enjoyable and social for both short and long term use. The games developed each had a specific purpose either in regards to obtaining diagnoses, as well as to maintain long term interest in the app. The four games are detailed below:

1) Whack-a-Beer: This game is used to **retrieve the reaction time** of the user by rapidly showing targets that the user should tap to increase a score. The score is used in both the machine learning and for users to compete with each other.
2) Tightrope Waiter: This game is used to **test the balance** of the user. It retrieves the same data as the passive data. However, compared to the passive data retrieval, the user has to walk using the heel-to-toe test as mentioned in [4]. The retrieved data is then used for an indicator of the user's balance.
3) Ring of Fire: Keeping in mind that the app has to be attractive in the long term, this social game **provides an incentive for users to use ManDown while out social drinking**. This is an important feature to attract users to continue using the app in the long run.
4) Who am I?: Similarly to Ring of Fire, this is a social game that allows the user to have fun with his/her friends while using the app.




**(a)** Social Game: Ring Of Fire

**(b)** Balance Game: Tightrope Waiter

**(c)** Reaction Time Game: Whack-A-Beer

**(d)** Social Game: Who Am I?

Fig. 3: Games Implemented in ManDown

These games allow the user to have social interactions among their friends, while allowing constant monitoring of their intoxication levels at the same time. **Figure 3** displays the screenshots of the games.

### D. Data Management

The **Data Management Module** serves the purpose of collecting and storing data from the user. This is done while considering the available memory, security and bandwidth constraints to eventually transfer data back to the front-end user.

*1) Database:* A cloud-based database is chosen as the preferred option due to the requirement of capturing all the passive and active data of the user. The *CAP*(Consistency, Availability and Partition Tolerance) Theorem is used to assess all available databases, from which Firebase is chosen. This is due to its functionalities of being able to store data locally, which allows the app to be responsive even when internet connection is lost, and ensures no data losses occur.

Firebase also supports real-time read and write access to the database via the *RESTful API*. Data in Firebase is stored in *NoSQL JSON* format and sent to machine learning for classification. Lastly, it offers adaptable user authentication techniques to restrict read and write access to the database.

The necessary Firebase function calls are implemented for ManDown in the *DBService* file to interface with other parts of the application. Each function call captures the relevant data and stores it in the correct hierarchical node in the database. Since we are consistently capturing passive sensor data through the smartphone and smartwatch, the database component is unable to handle all received calls from different components at the same time. To get around this, the database component uses *Intent* objects to queue the tasks asynchronously. The database schema was chosen to start with each user's unique Android ID followed by the sensor game data. Everytime a new data is available, the database stores the captured data under the captured timestamps in the corresponding column of that user. This is illustrated in **Figure 4**.

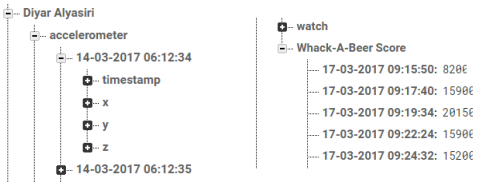



Fig. 4: Screen capture of Firebase. If user is not logged in via Gmail, their unique Android device ID is captured.

*2) Data Security:* As the data handled is confidential data, it is important to implement security protocols. The Firebase console where data is stored is only accessible by the database administrator and encrypted by a secure username and password only available to the administrator. In order to ensure data privacy among users, a user authentication technique has been implemented via Gmail sign in. When users log in to the application, they are prompted to log in using their Gmail details. By adding security rules to the Firebase settings, the database ensures each user's data is only accessible by him/her self only.

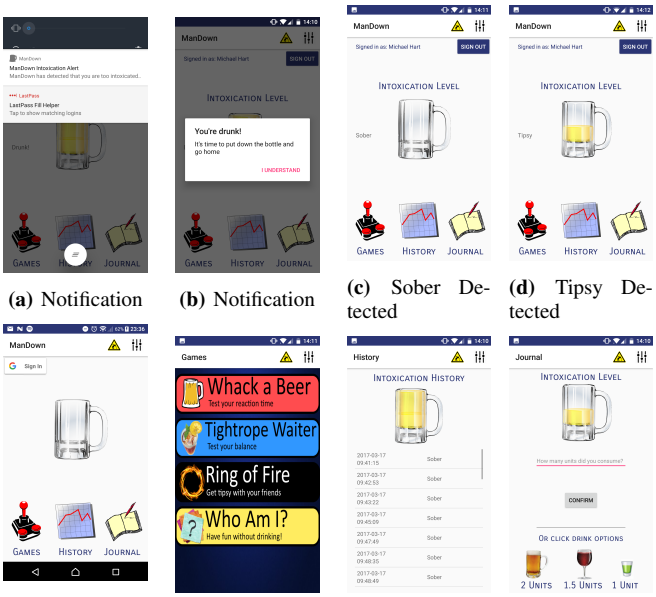
### *E. Smartphone User Interface*

The **User Interface** serves the purpose of providing feedback to the user regarding their intoxication levels, as well as offering management options in the event that intoxication occurs. **Figure 5** displays the different interfaces for the app.

*1) Feature Requirements:* The smartphone UI contains 4 key features that are outlined below. These features were determined after careful consultation among group members, and by understanding the target demographic.

1) Intoxication level display: The estimated intoxication level is available for users to view as seen in **Figure 5c and 5d**.
2) Trends display: The past history of a user's intoxication and consumption levels must also be accessible to users. This aids users in appreciating the patterns of their drinking which could help them self-manage their consumptions as shown in **Figure 5g**.
3) Managing output: The app must also offer users management techniques in the case when intoxication occurs. This can be seen from the emergency button on the homepage, in which an emergency call is dialed if pressed. This function can be modified to call a close friend as well.
4) Option personalization: Lastly, users should be able to update their display settings to suit their needs as well as their maximum intoxication threshold. Giving users a greater sense of ownership and control over the app.

*2) Design:* **Figure 5e** displays ManDown's Home Page. A top-down approach was adopted which presents high-level information first and allows users to decide whether or not to retrieve the details. The current intoxication level is displayed graphically by a pint of beer, with the level of beer in the glass representing how inebriated the user is. A full glass would indicate that the alcohol intake limit has been reached. Clicking the beer glass would direct users to another screen displaying further information in regards to the current drinking session, including units consumed, preferred drink (if input), and how many more units can be safely consumed. An example of the varying beer glass values can be seen in **Figure 5c and 5d**.




**(a)** Notification **(b)** Notification **(c)** Sober Detected **(d)** Tipsy Detected

**(e)** ManDown Welcome Screen **(f)** Game Menu **(g)** History Page **(h)** Journal Page

Fig. 5: ManDown UI

Additionally 3 graphical buttons below the intoxication display are used which direct to *Games*, *History*, and the consumption *Journal*. Again graphical buttons are used to make it easy for users to identify the conceptual model of the system, the alternate actions that can be taken, and their resulting execution. At the top right of the page we have an *Options* icon, which would direct users to another screen displaying the different settings that be amended. One possible means of personalization, would be offering different beverages, such as glass of wine, or a cocktail, as the graphic which displays the current level of intoxication. On the top right we have a contact help button.

*3) Active Input Processes:* The application interface will involve users manually logging consumption quantities as well as data collected from the user engaging with games.

A query, prompted by an event which suggests that the user may be consuming alcohol, will be triggered to ask the user whether they are indeed having an alcoholic beverage and if so they will be asked to enter the number of alcoholic units it contains (a unit being 10ml of ethanol). This could be done in the form of tapping a generic drink, such as a pint of beer, glass of wine, or shot of liqueur. The different drinks will have the standard number of units encoded. This is shown in **5h**.

This graphical approach will be adopted to simplify the process for users, so that the need to type in values is eliminated, thereby making it easier for users to comply with the query, no matter where they may lie on the spectrum of intoxication. A similar approach will be used for the manual entry journal, however this would offer the users numerical input as well for greater flexibility if they so desire.

*4) Drink Management:*

- User Warnings: Health risks of alcohol consumption are minimised by consuming a maximum of 14 units per week, and no more than 5 units in a single sitting [10]. To manage consumption within these limits, ManDown provides intake recommendations to warn users if their consumption approaches either threshold, displaying how many more units they can consume safely. The recommendation system could be personalised by providing specific warnings to each user. If a user has consistently consumed 5 units of alcohol on both Saturday and Sunday evening, then a 'Stop drinking tonight if you want to enjoy the Weekend!' notification appears if 6 units are consumed over the current working week. These notifications can be seen in **Figures 5a and 5b**.
- High Alert Intervention: The consumption warning system relies on direct user input of alcohol intake, which users may or may not provide. Consequently, a more robust management system will be employed to mitigate the risks of extreme intoxication. When high levels of intoxication are identified via autonomous methods, and users are found to be motionless, indicating they have been incapacitated, ManDown will inform the user's friends of their whereabouts and their need for help. An additional solution would be to contact emergency health services; however, this approach will be disregarded for the current evaluation and testing scope.

### *F. Smartwatch User Interface*

The implementation of the user-interface on the Moto 360 follows a minimalist concept. This allows for a no-frills interface which maximises the usage of the very limited screen space. An important consideration when designing the app for the watch was that this is an app which will be used when the user is inebriated (potentially even when the user requires medical aid). As such, the information needs to be clear and concise, with the functions easily accessible


Fig. 6: Watch App. Top left: App Icon. Top Right: Main page. Bottom Left: Confirmation of SOS. Bottom Right: Input menu for drinks

even by a drunken person. This can be seen in **Figure 6**, where only very few large icons are used against a black background to ensure good visibility and easy accessibility. The icons are the same as that in the mobile app to minimise confusion and ensure consistency of experience.

On the main page, the user is able to quickly see his/her intoxication level via the beer glass. This beer glass will change level accordingly with the main one in the mobile app (the smartphone sends a signal to the watch to notify of a level change). As such, it shows empty for sober, half-filled for tipsy and full for drunk.

The yellow emergency icon is also featured here. Upon clicking it, the confirmation page is brought up as shown in **Figure 6**. Again, minimal text is used and simple symbols and differently coloured circles are used to allow the user to make the confirmation. Upon confirmation, a message is sent to the smartphone and the smartphone will start the Android intent to dial emergency services. This is especially useful for users who use headsets and will allow them to call for help easily without pulling out their smartphone.

Finally, the notebook leads users to the drinks input page. Unlike the smartphone, the watch app simplifies it to simple inputs of common drinks found at pubs and bars and does not allow for custom input of alcohol units. Upon selection, the watch sends a message to the smartphone which is responsible for keeping track of the total units drunk by the user. It should be noted that the 2 text boxes are there to supplement the icons but are not meant to be integral to the user's understanding of the app beyond first usage.

### *G. Machine Learning and Classification*

*1) Machine Learning Specification:* The purpose of the **Machine Learning Module** is to obtain a state of either drunk, tipsy, or sober from the user inputs.

To obtain this state, a machine learning model needs to be trained using a combination of different sensor data,

game scores, and statistics. This data set is collected using smartphone sensors, wearable sensors, and mobile games. This model can be tested with any subset of the training data to generate a prediction as to whether the user is sober, tipsy, or drunk.

Furthermore, since the app is to be run on a smartphone, the machine learning should not be intrusive or take up processing power. With these specifications in mind, the final implementation uses the Amazon Web Services (AWS) Machine Learning Cloud platform to generate and store the model, and its REST API to make requests for predictions on particular user data as it is measured/required.

*2) Training AWS Model:* AWS Machine Learning takes as input a csv (comma-separated values) file with rows of training examples, columns of attributes to use to distinguish each example, and a final column which denotes the actual target classification for that example. The csv is uploaded to an Amazon S3 storage location, and from there a datasource is created which is essentially an ID the machine learning service uses to locate the file.

From the data-gathering experiments, csv files containing examples were constructed. These examples were used to train a machine learning model which is stored on the AWS server and accessible through a "real-time endpoint".

*3) Predicting using AWS Model:* In order to request predictions for further examples from the user from the trained model, the AWS API is included in the Android Gradle build file and certain permissions allowed in the manifest file. The main bulk of the implementation lies in the RealtimePrediction Java file. This file uses AWS credentials, along with the model identifier, to create a handle to the model real-time endpoint. From this, a request function is called with a map of attributes and values as input to the model and the return value from the API is a classification label, with prediction confidence.

The AWS Machine Learning console allows customisation of the model, for example, thresholds of confidence can be adjusted. This will only allow the model to assign a class to an example if its confidence that that example belongs to that class surpassed the threshold. During training the number of epochs used and other optimisation parameters such as whether early stopping is used can also be set. This allows fine-tuning of our model.

*4) Model Limitations:* Due to the AWS architecture, in order to update a model, it is necessary to complete the entire training process from uploading to S3 storage to identifying the schema, etc. rather than just sending the extra example to include in the pre-existing model. Not only is this prohibitively challenging to do using the API, but the training itself takes an unpredictable and long enough time to complete that it would not be feasible to do this every time the user had some new data they wanted to add to the model. For future iterations, it would be desirable to develop our own server which could store the model, and use a machine learning technique which facilitates the addition of new examples, such as a neural network. Doing so would also aid fine-tuning of the model since we can better explore which attributes exactly are causing misclassification. Most importantly, the machine learning model would need to be unsupervised, as the user will not be taking and storing breathalyser measurements.

## V. Experiment Outline

### A. Field Tests For Intoxication Level

To be able to evaluate the hypotheses given in **Section III**, an experiment was undertaken involving students. All of the students were social drinkers aged between 21-23. Each participant would undertake the instructions displayed below, leaving half an hour between each repetition to allow consumed alcohol to enter the bloodstream and breath for more accurate readings. The data collected allowed us to test the accuracy of the machine learning algorithm (Hypothesis 1) and whether the Gamification inputs improves the accuracy of the diagnostics (Hypothesis 3).

1) Note a breathalyser result, with the corresponding time to be matched with BAC data taken.
2) Play the Whack-A-Beer game three times, allowing the score and reaction times to be stored in the database.
3) Play the Tightrope Waiter game three times, allowing the accelerometer data to be stored in the database.
4) Walk in a straight line with a normal gait while wearing the watch three times, recording the data using the ManDown app.
5) Consume a fixed amount of units of alcohol.

This set of readings collected both passive and active data for later analysis, while the breathalyser reading was the true value of the user's intoxication level.

### B. Survey on User Interface

In addition to the experiments, a survey was conducted in order to gain an understanding of the efficacy of the app in drink management (Hypothesis 2) and improving user enjoyment in relation to simple user inputs (Hypothesis 4).

The survey was conducted upon students in the target age range of 20-24, or around student age. All the students were in the Electrical Engineering department building of Imperial College London. Each student was allowed to use the app until they felt familiar with it, at which time the drunk notifications were shown to them; the user then filled out the survey questions. The survey questions and results were taken using Google Forms, and can be found linked to in the README on GitHub, or at the Google Forms link [11].

## VI. System Evaluation

### A. Intoxication Detection Accuracy

Even upon manual inspection, the data is interesting as general trends can be seen when the user's blood alcohol content increased. As BAC increased, the minimum variance of the accelerometer data during the tightrope waiter game decreased noticeably as the user passed from the sober, tipsy and drunk thresholds. This is to be expected, as fine motor control is lost in favour of more general swaying motions.

The number of sober, tipsy and drunk examples recorded were 170, 77 and 54 respectively, for a total of 301 samples. These numbers are different enough from each other such that one class is overrepresented in the data. This will likely cause classification rate to go down due to the effect of the majority class. Furthermore, precision goes down for the minority class. To prevent this, we employed uniform random downsampling of the majority class.

*1) Accuracy of Intoxication Detection:* During the development timeline, multiple models were trained and evaluated based on AWS's internal evaluation. The training process involves splitting the data into 30% testing and validation data, and using the remaining 70% as training data. This helps to prevent overfitting. Cross-validation is also performed, and various indications of the model's performance can be produced. The accuracy of the model was measured as an F1 score, which is an accuracy measure accounting for accurate results and false positives/negatives [12]. A higher F1 score indicates a higher accuracy.

The final model produced an F1 score of 0.96, which is surprisingly high. There is a very high proportion of TP (true positive) and almost no FP/FN (false positives/negatives). The performance can be seen in the confusion matrix illustrated in **Figure 7a**. Hence, **Hypothesis 1 is validated as it is possible to accurately detect intoxication symptoms with a very high accuracy of 96%.** The algorithm, multinomial logistic regression, was automatically selected as the best performing by Amazon AWS.

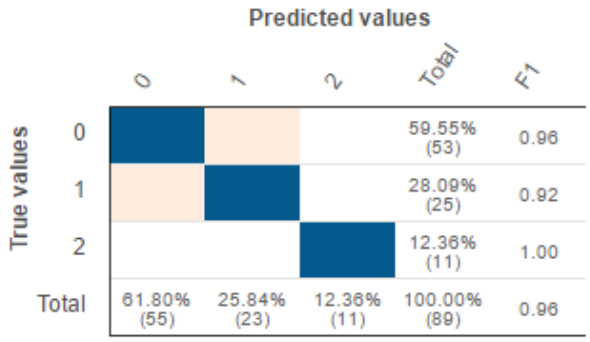
*2) Gamification vs. Passive:* To test whether the gamification improved the classification accuracy, the same steps as in the previous section were performed except that data collected using games was ignored. This left only passive data from walking and from background sensing.

A model just using this sensor data was produced and evaluated. The confusion matrix can be seen in **Figure 7b**

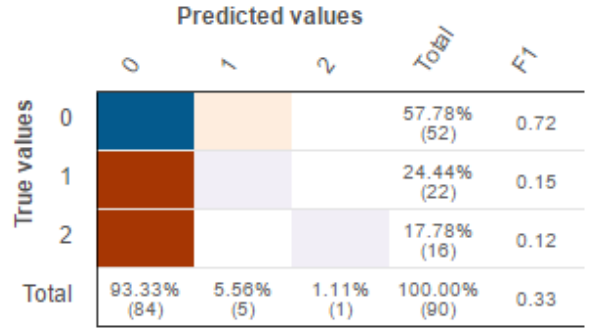
This model performed very poorly overall, with an F1 score of <=0.15 for two of the three classes. The reason for this is that the variance measure we used to generate the examples for training is not representative of the states of drunkenness after sober. i.e. the data can be used to distinguish between sober and non sober, but not different levels of BAC after that. The model did correctly classify class 0 (sober), however. In fact, this can be seen by eye from the data - if the examples are sorted by the different attributes, the examples of the sober class can be clearly separated, whereas the other two are mixed together. This leads to a mix of TP and FP for class 0, with very few examples being classified as belonging to the other two classes.

Therefore, **Hypothesis 3, which states that gamification improves diagnostic accuracy, is confirmed. The model using game data performed much more strongly than the model without.**

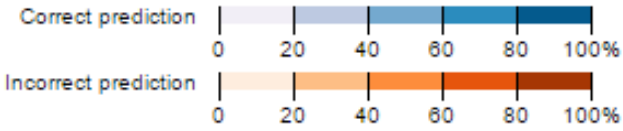
The mean and variance reaction times from Whack-A-Beer in particular were strong indicators of the user's BAC. The information gain upon dividing the data on these attributes was particularly high.




**(a)** The confusion matrix for the model using all data showing the F1 scores and number of examples in each class




**(b)** The confusion matrix for the model using just sensor data




**(c)** Confusion Matrix Key

Fig. 7: Confusion Matrices from Machine Learning Models

## B. User Interface Evaluation

The results from the survey were used to determine Hypotheses 2 and 4. Q6 from the survey gave an indication of the user enjoyment when using gamification to input data, and evaluated Hypothesis 4. Q8 from the survey determined whether users were more likely to manage their drinking habits with the app, and evaluated Hypothesis 2. The respective results can be observed in **Figure 8**. As 31 users responded to the survey, these results are considered valid.

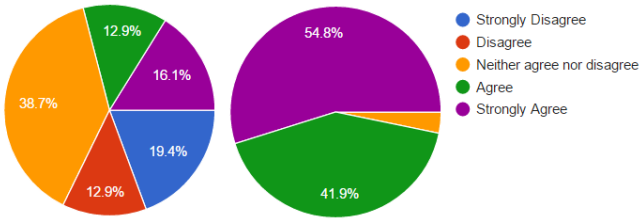



Fig. 8: Left: Q8 Results: I would consume less alcohol due to using this app. Right: Q6 Results: The games were engaging and fun.

*1) Managing Alcohol Consumption with the App:* The results from Q8 are slightly mixed, with 32.3% of participants disagreeing that they would consume less alcohol, while 29% of participants finding the app helpful in managing their alcohol consumption. The remainder of the participants were undecided. **This shows that the Hypothesis 2 is incorrect.** However, this may be due to the fact that certain users were sober while evaluating the user interface instead of using it while drinking. This can be circumvented by taking more data on users who drank while using the app.

*2) User App Engagement:* The results from Q6 show a strong positive response to the use of games in the app, with 54.8% of participants strongly agreeing, and 41.9% agreeing. Only one user out of all participants was unsure about game engagement. This shows that **Hypothesis 4 is correct**. During the survey when answering question 9, a significant number of users said that they would use the app while drinking for the games alone.

## VII. Future Work

### A. Diagnostic Inputs Module

- Detection of other Intoxication Symptoms: Nystagmus, a symptom involving eye motion, could be detected using a smartphone camera.
- Improving Upon Ataxic Gait Detection: Distributing the polling intervals for sensor data between the phone and watch to optimise battery life.
- Improving Upon Arm Motion Detection: Other models [13][14] exist which could be used to improve upon the arm motion detection.
- Detecting Drinking Motions: Instead of the user having to manually input drinking data, the smartwatch could be used to detect whenever the user is drinking through the hand motions.

### B. Gamification Inputs Module

- Retrieving Data From Social Games: Data from social games could be used to determine how much a user has drunk. This can be worked into the game subtly, and the data could be used for machine learning.

### C. Database Module

- Group Formation/Classification: By separating different groups of users in Firebase(e.g how much they drink), it is possible to perform machine learning classification on individual groups for improved model training. This can also allow for personalised feedback on various groups.

### D. User Interface

- General Improvements: Changes such as making the emergency call button more visible, and the journal page easier to use, would encourage users to use the app.
- Social Tracking for Friends: Allowing users to form groups and sharing games, drinking data, and location data would improve the social aspect of the game and improve enjoyment.

### E. Machine Learning

- Extensive Data Collection: the model could be improved further by taking more extensive data samples and investigating other statistical analyses of the passive sensor data.
- Personal Customisation: the app could construct a model on a per-user basis, based on a general model, which could adapt to the user according to drinking preferences and how intoxication affects each user.

## VIII. Conclusion

ManDown exhibits the possibility of using a simple smartphone app with games in order to detect intoxication. The results are substantially accurate in detecting key BAC levels, as well as being able to provide a level of satisfaction to users in order to maintain interest in it over time.

In order to make this app fully viable for mobile healthcare, more work needs to be done on the management side in order to deter users from overdrinking. However, the implementation of games and the detection of intoxication are an excellent foundation into developing an app capable of managing social drinkers in the future.

## References


[1] F. Modig, P.-A. Fransson, M. Magnusson, and M. Patel, "Blood alcohol concentration at 0.06 and 0.10% causes a complex multifaceted deterioration of body movement control," *Alcohol*, vol. 46, no. 1, pp. 75–88, 2012.

[2] D. J. Link. Standardized field sobriety test. [Online]. Available: http://duijusticelink.aaa.com/issues/detection/standard-field-sobriety-test-sfst-and-admissibility/

[3] D. Alyasiri, M. Poynton, D. Ma, A. Shafiei, W. Te, S. Rubio, and M. Hart, "Mandown - a social drinking management app," January 2015, [Published privately; contact authors for copy].

[4] J. Stuster and M. Burns. Validation of the standardized field sobriety test battery at bacs below 0.10 percent. [Online]. Available: https://www.ncjrs.gov/pdffiles1/Photocopy/197439NCJRS.pdf

[5] H.-L. Kao, B.-J. Ho, A. C. Lin, and H.-H. Chu, "Phone-based gait analysis to detect alcohol usage," *Proceedings of the 2012 ACM Conference on Ubiquitous Computing*, pp. 661–662, 2012.

[6] Z. Arnold, D. LaRose, and E. Agu, "Smartphone inference of alcohol consumption levels from gait," *2015 International Conference on Healthcare Informatics (ICHI)*, pp. 417–426, 2015.

[7] Google Play, "Drinkfree - sobriety counter." [Online]. Available: https://play.google.com/store/apps/details?id=novusapp.drinkfree&hl=en_GB

[8] GoodSAM, "Goodsam website." [Online]. Available: https://www.goodsamapp.org/

[9] W. zhong Wang, Y. wei Guo, B. yu Huang, and G. ru Zhao, "Analysis of filtering methods for 3d acceleration signals in body sensor network," in *Bioelectronics and Bioinformatics (ISBB), 2011 International Symposium on*, 11 2011.

[10] National Health Service, "New alcohol advice issued," [Online; accessed on 2017-02-02]. [Online]. Available: http://www.nhs.uk/news/2016/01January/Pages/New-alcohol-advice-issued.aspx

[11] G. Forms. Mandown interface questionnaire. [Online]. Available: https://drive.google.com/open?id=17OrrTdKJB31FA79Uh3R_s4NaZMptEC2HQYk8HqucgSI

[12] K. Dembczynski, A. Jachnik, W. Kotlowski, W. Waegeman, and E. Hüllermeier, "Optimizing the f-measure in multi-label classification: Plug-in rule approach versus structured loss minimization." *ICML (3)*, vol. 28, pp. 1130–1138, 2013.

[13] Google. Detectedactivity. [Online]. Available: https://developers.google.com/android/reference/com/google/android/gms/location/DetectedActivity

[14] W. W, D. S, R. EE, P. C, and N. GJ, "Classification accuracies of physical activities using smartphone motion sensors," *Journal of Medical Internet Research*, vol. 14, no. 5, 10 2012.