AN INTELLIGENT MOBILE APPLICATION TO TEST SOBRIETY OF THE USER USING MACHINE LEARNING

Christopher Feng [1], Soroush Mirzaee [2]

[1] Hopkins School, 986 Forest Rd, New Haven, CT 06515
[2] Computer Science Department, California State Polytechnic University, Pomona, CA 91768

ABSTRACT

This project presents a sobriety detection system aimed at preventing drunk driving through a combination of image recognition, alcohol sensing, and speech evaluation tests. This project uses a Raspberry Pi device, equipped with an alcohol sensor and a camera to assess a user's sobriety by analyzing facial expressions and blood alcohol content (BAC) [9][10]. Additionally, a vocal test is conducted using the user's smartphone, with results processed and stored in a cloud database for further analysis. The system offers an affordable, reliable, and user-friendly alternative to traditional methods, such as breathalyzers, by detecting both alcohol and drug impairment. Through a series of experiments, the system demonstrated high accuracy, achieving a 97% success rate when combining all tests, highlighting its potential to reduce driving under the influence of incidents.

KEYWORDS

Drunk Driving, Sobriety, Raspberry Pi, Flutter

1. INTRODUCTION

Drunk driving has been an issue since motor vehicles were first created, exemplified by the growing access to cars over the years. Beginning in the 1830s, the temperance movement in the United States was the first anti-alcohol movement, and the first drunk driving arrest was made in 1897 [2]. Nearly 25 years ago, surveys showed that 90% of households had at least one car, and this number has grown to 92% according to United States Census data [1][4]. As more people have access to cars, this issue will continue to grow unless measures are taken to prevent it. Drunk driving is an issue that should, and in recent years, have been brought into the limelight, because according to data from the National Highway Traffic Safety Administration, in 2022, alcohol impaired traffic deaths resulted in 13,524 people dying, of which 2,337 had BAC(Blood Alcohol Concentration) of 0.01 to 0.07[5]. Although the federal limit for BAC is 0.08, we can see that even below this amount, driving is dangerous [3]. This problem affects everyone because anyone who is drunk on the road is a danger to themselves as well as everyone else around them. All of these deaths are preventable, so why not take a step to do so?

The first methodology uses a breathalyzer that uses infrared light to detect alcohol in a user's breath, as alcohol absorbs infrared light. This method identifies alcohol consumption, but my project has a broader scope and is capable of detecting impairment caused by both alcohol and

David C. Wyld et al. (Eds): ADCO, CMCA, NeCoM, NLPI, CSITEC, BDBS, DMML, SP – 2025
pp. 279-288, 2025. CS & IT - CSCP 2025 DOI: 10.5121/csit.2025.150522

drugs, which a breathalyzer cannot do. The second methodology examines drunk driving punishments based on BAC. However, my product not only accurately measures BAC but also uses vocal and facial features to guarantee accuracy. The final methodology proposes a prevention system using blood pressure, alcohol content, facial expressions, road data, and car acceleration. If three drunk driving indicators are detected, the ignition is cut and authorities are notified. However, their product's cost and effectiveness are unclear, while my product is affordable and effective.

The product created is directed to the individual and people who are concerned for them. We have created an application which can detect whether someone is sober or drunk. If a loved one is concerned for their safety, they can easily use this application and prevent them from driving if they are drunk, or permit them to go. The individual can use it themselves before driving, but if they are drunk, this will likely be ignored and someone else will have to do it for them. We trained a custom AI model for the Raspberry Pi to detect if someone is drunk based on their face, which is handled using the camera on it [11]. It is noteworthy to mention that it will also measure the blood alcohol sensor using an alcohol sensor on the same device to check for Blood Alcohol Content, and finally a vocal test by giving a sentence for the user to read out loud.

Speaking of which, the vocal test uses a deep gram model to identify how accurately the person pronounces the words that are displayed to them in the application by converting them into vowels and words to match with the original sentence.

This is an effective solution for the individual because there is easy access to the application, it is relatively cheap compared to the price of death or being arrested for driving while drunk, and it is better than other options like the Raspberry Pi 4 or the NVIDIA Jetson Nano Developer Kit [12].
The experiments focused on testing and improving a sobriety detection system using multiple methods: image recognition, breath analysis, and speech evaluation. The first experiment examined the accuracy of image recognition in determining sobriety, with dataset sizes of 500, 1,500, and 3,500 images. Larger datasets significantly improved model accuracy, from 68% to 94%. The second experiment tested a three-stage system combining breath analysis, speech evaluation, and image recognition. The system achieved a cumulative accuracy of 97%, with individual test accuracies ranging from 88% from speech to 94% from image recognition. The speech evaluation showed variability due to articulation and environmental factors. The combination of these methods was designed to address each of their limitations. The experiments demonstrated the importance of large datasets and complementary testing methods for fine-tuning effective sobriety detection.

2. CHALLENGES

In order to build the project, a few challenges have been identified as follows.

2.1. To create an AI model

To create an AI model that was consistent and accurate, we had to use a larger dataset to train with. To do so, we used Roboflow to find a dataset of 1800 images of people who were drunk and Teachable Machine to train the dataset. We then tested the model by seeing how the model would react when putting an image of a sober person and an image of a drunk person in front of the camera. Once satisfied with consistent results, the model was complete.

2.2. Finding the Right Hardware

Finding the right hardware takes into consideration the cost of use for the user and how good the performance of the hardware is. Therefore, the hardware with the best cost-to-performance ratio would be used. There were three main pieces of hardware to consider: The Raspberry Pi 5, the Raspberry Pi 4, and the NVIDIA Jetson Nano Developer Kit [13]. The NVIDIA Jetson Nano was easy to eliminate because it is too expensive to be feasible for most people and for the purposes of this project, sitting at $130. The Raspberry Pi 4 and Raspberry Pi 5 both cost around $60, which is less than half of the price of the NVIDIA Jetson Nano. Although they are both slower in processing speed, the tradeoff was worth it because of the difference in prices. The Raspberry Pi 5 is significantly faster in processing speed than the Raspberry Pi 4 while remaining at a similar price, so it is the best option.

2.3. The Selection of Tests

The final problem to be solved is the selection of tests to use for sobriety. The vocal test and taking a picture were the tests chosen. This is because the Raspberry Pi 5 has its own camera in the kit, so it is fitted for the Raspberry Pi and would therefore work better. The vocal test was chosen because all smartphones have a built-in microphone, which would take the load off the Raspberry Pi, thereby making the whole process faster and also slightly cheaper, as the user would not need to buy a separate microphone. The vocal test and taking a picture are also easy for the user to operate, as the vocal test automatically runs once the user decides to start the test, and while they are speaking into the phone, a picture is being taken and the AI model is being run on the Raspberry Pi to check for sobriety.

3. SOLUTION

The program officially starts on the picture test screen, during which the Raspberry Pi 5 takes an image of the user to check their eyes to determine if they are sober or drunk and uses the alcohol sensor on the i to check for the user's blood alcohol content. Both of these processes happen on the Pi. Next, it goes into the vocal test screen, which gives the user a phrase from which they would read it aloud and the data is stored in the final part of the program, the Firestore and Firebase storage area, where all the data from the vocal test, the picture test, and the alcohol sensor are both read to determine whether the user is sober or not. After getting a conclusion, the user is directed to the "Previous Tests" screen, which will show a card with their image, the date at which the test was taken, the alcohol level, confidence for the result, the verbal test result, and a green check or a red cross to show sober or not sober respectively.

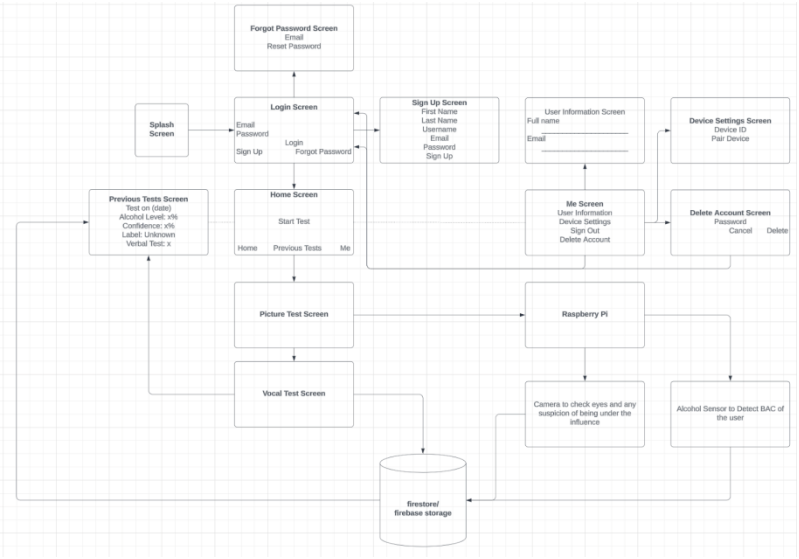

Figure 1. Overview of the solution

The Raspberry Pi is used for the picture test and the alcohol sensor. The picture test determines whether someone is sober or not through an image taken by the camera on the Raspberry Pi and running a program on it which uses an AImodel trained with 1800 images of drunk and sober people through the website Teachable Machine [14]. The alcohol sensor checks the blood alcohol content of the user and stores the information.

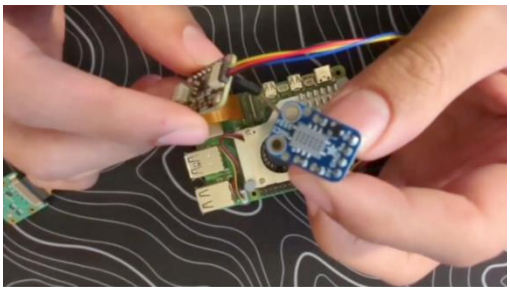

Figure 2. Picture of the component

```
# ADS1115 setup
def setup_ads1115():
    i2c = busio.I2C(board.SCL, board.SDA)
    ads = ADS.ADS1115(i2c)
    return ads

def read_alcohol_level(ads, channel=0):
    chan = AnalogIn(ads, getattr(ADS, f"P{channel}"))
    return chan.voltage
```

Figure 3. Screenshot of code 1

The MICS 5524 sensor module takes care of the measurements for the blood alcohol content. Using this sensor with Raspberry Pi is only feasible through an ADS1115 Analog to Digital Convertor module that allows us to connect the analog sensor to the Pi and read values. Then from the thresholds initialized by the creators of the sensor, the alcohol content is measured. This data will be sent to the Firestore Database alongside the image captured for the test completed before being sent to the phone app for the verbal test.

Moreover, SoberAI utilizes the user's smartphone to make the vocal prompt possible. After they begin the test on their phone, and after their facial test is done on the Raspberry Pi, the user will be prompted with a sentence that might be a little tough in the first place., They will have to recite it and record their voice doing so. In the end, the recording of the sentence will be sent to Google Cloud using Deepgram's voice-to-speech, then converted and evaluated on the server.

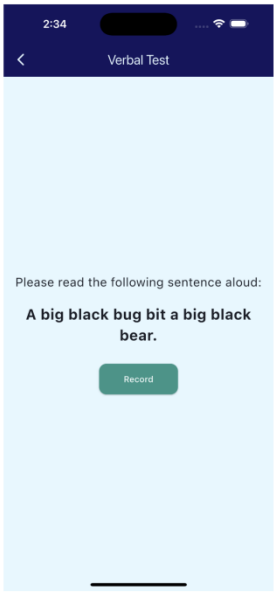

Figure 4. Screenshot of the APP

```
// Specify the audio source URL
const audioSource = data.data.url.toString();

console.log("URL: ", audioSource);

// Use v3 SDK method to transcribe audio
const { result, error } = await deepgram.listen.prerecorded.transcribeUrl({ url: audioSource }, {
    model: 'nova-2',
    language: 'en',
});

// Check if there is an error in the response
if (error) {
    console.error(error);
    return error;
} else {
    console.dir(result, { depth: null });
    return result;
}
```

Figure 5. Screenshot of code 2

After the user is done recording them reading over the sentence/paragraph, the recording is sent to Google Cloud Function Run and it will invoke the method above [15]. This method will then send the audio recording to the Deepgram for deconstruction to text. The text then is fed back into the function, comparing results with the original sentence and saving passing or not passing in Firestore.

Finally is the storage section of the program. This application uses Firebase as its main cloud storage service. In Firebase, the pictures taken by the camera on the Raspberry Pi are stored. Firestore stores all the other information like the result from the vocal test, the blood alcohol content, user information, date of the test, confidence of the result, and everything else to come up with a final conclusion of whether or not the user is sober.

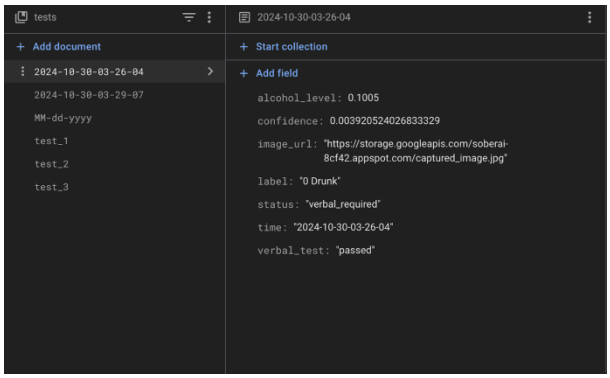

Figure 6. Screenshot of the test

```python
# Upload image to Firebase Storage and get the download URL
def upload_image_to_firebase(image_path):
    blob = bucket.blob(os.path.basename(image_path))
    blob.upload_from_filename(image_path)
    return blob.public_url

def store_in_firestore(alcohol_level, confidence, label, image_url, user_id):
    # Create a unique identifier for the test document (user ID + timestamp)
    timestamp = time.strftime("%Y-%m-%d-%H-%M-%S", time.localtime())
    doc_name = f"{timestamp}"

    # Firestore path: users/{user_id}/tests/{doc_name}
    doc_ref = db.collection('users').document(user_id).collection('tests').document(doc_name)

    test_data = {
        "time": timestamp,
        "alcohol_level": alcohol_level,
        "confidence": confidence,
        "label": label,
        "image_url": image_url,
        "verbal_test": "",  # Placeholder for verbal test result
        "status": "verbal_required"  # Mark as waiting for verbal test
    }
    doc_ref.set(test_data)
    print(f"Data stored in Firestore under document: {doc_name}")
```

Figure 7. Screenshot of code 3

4. EXPERIMENT

4.1. Experiment 1

One of the possible blind spots in the program is its performance in determining sobriety under varied lighting or environmental conditions. Accurate detection is essential to ensure safety and reliability.

The experiment involves testing a trained model's ability to classify sobriety using image recognition. Controlled datasets of varying sizes, including 500, 1,500, and 3,500 images, are used. These datasets cover diverse scenarios to mimic real-world variability, such as different lighting, facial expressions, and postures. Control data is sourced from labeled images verified by experts. The setup evaluates accuracy, false positives, and false negatives to ensure robustness. By systematically increasing dataset size, the experiment identifies trends and limitations, ultimately ensuring reliable performance under diverse conditions. This setup mimics real-world scenarios and ensures practical application.

One of the tests that the user has to do is the picture test, where a trained model to determine whether a person appeared sober or under the influence, aiming to decide if they were fit to drive.

Initially, we explored comparing eye features between their sober and influenced states. However, this approach proved to be inconsistent and inaccurate due to the variability in eye characteristics across individuals and situations.

The final and more effective approach utilized image recognition techniques. We tested datasets of various sizes, eventually settling on a dataset containing approximately 3,500 images. This dataset provided a diverse range of examples that significantly improved the model's accuracy and reliability.
Below is a hypothetical graph showing the relationship between dataset size and the estimated accuracy of the model:

Dataset Size	Estimated Accuracy (%)
500 Images	68%
1,500 Images	80%
3,500 Images	94%

Figure 8. Graph of the relationship

The graph illustrates how increasing the dataset size contributed to a noticeable improvement in the model's performance.

The experiment's results highlighted several important findings. The mean and median performance of the model improved consistently as the dataset size increased. The lowest observed accuracy, around 68%, was with the 500-image dataset, while the highest accuracy, approximately 94%, was achieved with the 3,500-image dataset.

The shift from using eye-based features to image recognition was a pivotal decision. Eye feature analysis yielded inconsistent results due to individual variability, whereas image recognition provided a more generalizable and robust method. The dataset size had the most significant impact on results, with larger datasets offering a wider variety of training examples and reducing overfitting.

One surprising observation was how quickly accuracy improved between the smaller datasets and the 3,500-image dataset, emphasizing the importance of comprehensive training data. These results underscore the value of robust datasets and the limitations of relying solely on feature-specific methods for nuanced tasks like sobriety detection.

4.2. Experiment 2

To ensure accurate sobriety detection, a series of tests is used, combining breath analysis, speech evaluation, and image recognition. These methods are essential for comprehensive and reliable assessments.

The system employs a three-stage testing process. First, a MiCS5524 module detects alcohol in the driver's breath as they enter the car. If no significant alcohol is detected, the driver is prompted to read a sentence aloud to evaluate speech clarity and identify slurring. Finally, an image recognition model assesses physical signs of sobriety. Each test provides an independent evaluation, with all three required to pass for driving approval.

This design ensures redundancy, addressing potential inaccuracies of individual tests. The chosen methods prioritize reliability, accessibility, and user compliance, avoiding intrusive or impractical alternatives. Control data comes from labeled samples of sober and under-the-influence states.
The experiment involves testing the performance of the three-stage system on a dataset of 100 test cases. Each case includes a combination of breath analysis, speech evaluation, and image recognition results, labeled as either “Pass” or “Fail.” Below is a hypothetical table summarizing the outcomes:

Test Type	Accuracy (%)	False Positives (%)	False Negatives (%)
Breath Analysis	90%	5%	5%
Speech Evaluation	88%	7%	5%
Image Recognition	94%	3%	3%

Figure 9. Table of the outcomes

The graph below visualizes the cumulative system accuracy when combining results from all three tests:

Combined Tests	Accuracy (%)
Breath Only	90%
Breath + Speech	94%
All Three	97%

Figure 10. Table of the accuracy

The three-stage testing process demonstrated high cumulative accuracy, achieving an overall success rate of 97% when all methods were combined. Individual test accuracies ranged from 88% for speech evaluation to 94% for image recognition. The lowest observed accuracy was from speech evaluation, likely due to variability in user articulation and environmental noise. Breath analysis and image recognition performed consistently well, with minimal false positives and negatives.

The decision to use these specific methods was driven by their complementary nature. Breath analysis addresses immediate alcohol detection, speech evaluation highlights cognitive impairment and image recognition detects physical signs. Combined, they provide a robust and layered assessment, mitigating the weaknesses of individual methods.

Other potential approaches, such as invasive testing or behavioral observation, were excluded due to impracticality and lower user acceptance. The series of tests was chosen to balance effectiveness, user compliance, and feasibility, ensuring a reliable system for real-world application.

5. RELATED WORK

T. Shyam Ramanath et al. use infrared light to test whether the user is drunk or not by testing the breath of the user. If the user is drunk, their breath will contain some alcohol content and alcohol as a gas has the property of absorbing infrared light. The sensor they use will then detect the missing infrared light and be able to detect if the user has consumed any alcohol. Although this method is effective in detecting whether someone is drunk, my project has a wider range, able to detect drivers impaired by alcohol and drugs, which they are unable to do [6].

Benjamin Hansen examines the punishments against drunk driving, focusing on the BAC of the individual. However, my product is not only able to check the BAC of the user and report it accurately with an alcohol sensor, it also acts as a prevention mechanism and an extra layer of safety [7].

Koneti Sandeep et al. proposed a prevention mechanism utilizing the blood pressure, alcohol content, facial expressions, road information, and acceleration of the car. If three qualities of drunk driving are met, the ignition will be cut and reported to authorities. However, there is no mention of the cost of these products and how effective it may be, whereas my product is both cheap and effective [8].

6. CONCLUSIONS

A potential limitation to the project is that for it to be used as a killswitch, it would have to be manually connected to a car, which is not easily accessible for some people. Because of this, we chose not to implement this feature in the project. To make the process more user-friendly, we could automate the entire process once the user gets in a car to do the scan and make a report rather than the user having to click the “start test” button.

Although my project has the limitations mentioned above, I also know that this project will allow a user’s loved ones to be less worried about them because they will be informed with guaranteed accuracy if the user is in danger while driving on the road.

REFERENCES

[1] Wright, Nicholas A., and La-Troy Lee. "Alcohol-related traffic laws and drunk-driving fatal accidents." Accident Analysis & Prevention 161 (2021): 106358.
[2] Lundman, Richard J. "City police and drunk driving: Baseline data." Justice Quarterly 15.3 (1998): 527-546.
[3] Eisenberg, Daniel. "Evaluating the effectiveness of policies related to drunk driving." Journal of policy Analysis and management 22.2 (2003): 249-274.
[4] Buneman, Peter. "Semistructured data." Proceedings of the sixteenth ACM SIGACT-SIGMOD-SIGART symposium on Principles of database systems. 1997.
[5] Yellman, Merissa A. "Motor vehicle crash deaths—United States and 28 other high-income countries, 2015 and 2019." MMWR. Morbidity and Mortality Weekly Report 71 (2022).
[6] Ramanath, T. Shyam, A. Sudharsan, and U. Felix Udhayaraj. "Drunken driving and rash driving prevention system." 2010 International Conference on Mechanical and Electrical Technology. IEEE, 2010.
[7] Hansen, Benjamin. "Punishment and deterrence: Evidence from drunk driving." American Economic Review 105.4 (2015): 1581-1617.
[8] Sandeep, Koneti, Ponnam Ravikumar, and Sura Ranjith. "Novel drunken driving detection and prevention models using Internet of things." 2017 International Conference on Recent Trends in Electrical, Electronics and Computing Technologies (ICRTEECT). IEEE, 2017.

[9] Eisert, Peter, and Bernd Girod. "Analyzing facial expressions for virtual conferencing." IEEE Computer Graphics and Applications 18.5 (1998): 70-78.

[10] Albalate, Daniel. "Lowering blood alcohol content levels to save lives: the European experience." Journal of Policy Analysis and Management: The Journal of the Association for Public Policy Analysis and Management 27.1 (2008): 20-39.

[11] Jolles, Jolle W. "Broad-scale applications of the Raspberry Pi: A review and guide for biologists." Methods in Ecology and Evolution 12.9 (2021): 1562-1579.

[12] Basulto-Lantsova, Artiom, et al. "Performance comparative of OpenCV Template Matching method on Jetson TX2 and Jetson Nano developer kits." 2020 10th Annual Computing and Communication Workshop and Conference (CCWC). IEEE, 2020.

[13] Gh, Mohammad Basman. "A novel Face Recognition System based on Jetson Nano developer kit." IOP Conference Series: Materials Science and Engineering. Vol. 928. No. 3. IOP Publishing, 2020.

[14] Carney, Michelle, et al. "Teachable machine: Approachable Web-based tool for exploring machine learning classification." Extended abstracts of the 2020 CHI conference on human factors in computing systems. 2020.

[15] Rose, Richard. Hands-on serverless computing with Google Cloud: build, deploy, and containerize apps using Cloud Functions, Cloud Run, and cloud-native technologies. Packt Publishing Ltd, 2020.