A WEARABLE AND INTERNET-OF-THINGS (IOT) APPLICATION FOR SLEEP DETECTION AND LIGHTING CONTROL USING AI AND MACHINE LEARNING TECHNIQUES

William Ma[1] and Yu Sun[2]

[1]Crean Lutheran High School, 12500 Sand Canyon Ave, Irvine, CA 92618
[2]California State Polytechnic University,
Pomona, CA, 91768, Irvine, CA 92620

ABSTRACT

There are many apps that let you control hardwares with the application of the internet-of-thing today, however, I have seen none that lets you customly control the hardwares, most likely you can only use the few controls the developer of the hardware gives you, and there are very few auto control options. This paper designs an application to auto control the hardwares into desired state based on personal status detected [1]. We use a smart watch to detect the heart beat of the user and determine if they are asleep, and once they are asleep, we turn off a light switch in the user's room to create a total darkness environment. Much research done on sleeping quality shows that sleeping in total darkness gives much better sleeping quality, while many, out of fear or sleeping disorders, still leave little lights on when sleeping [3]. This software helps to give these people better sleeping quality with ease [2].

KEYWORDS

IOT, AI, Machine learning.

1. INTRODUCTION

I created this project on this topic because sleeping in total darkness means better sleeping quality. But sometimes it’s really hard for people to fall asleep in total darkness. I personally was really afraid of darkness when I was small, and I still somewhat do right now. For me it is really a pain to sleep in total darkness because it creates a lot of fear and I need a long time to actually fall asleep even though I know it’s good for my health. I also know some relatives with sleeping disorders also cannot fall asleep in total darkness. I searched up on the internet and I did find many people with this problem, and there were no real good solutions other than just overcoming the fear by force, which is hard, or using sleeping pills, for the ones with sleeping disorders, which is bad for your body and creates a reliance if used too much. I really want to solve this problem not only for myself, but also for others with similar problems around the world. This application can be used by anyone who cannot fall asleep in total darkness to increase their sleeping quality, whether it’s because of fear, sleeping disorders, or any other reasons. This application probably will not be used by the majority, but it will be very helpful for the ones struggling with sleeping problems [4]. One side benefit of using it is that it also saves electricity to not have one’s light on the whole night, and saving electricity means saving some money.

David C. Wyld et al. (Eds): FCST, CMIT, SE, SIPM, SAIM, SNLP - 2022
pp. 101-109, 2022. CS & IT - CSCP 2022 DOI: 10.5121/csit.2022.120808

Some existing methods on this topic include setting a time, and turning off the light when the time is detected, this allows the user to turn off the light after they are asleep if they approximately know when they are going to fall asleep. However, firstly, this is only available in a certain smart home software, and not usable if you use hardware from another company [5]. Secondly, they will need to set a different time on the app everyday if they do not sleep at the same time everyday, which is not very convenient because you only know that you are going to sleep when you feel really sleepy, and an extra thing to do when feeling sleepy is just inconvenient. They will also need to turn off the set time every morning if they are not sleeping at the same time everyday to prevent their lights from suddenly turning off when they are awake the next day, which adds more to the inconvenience. And people could also just forget to do it. Thirdly, it will be hard for people to know when they are asleep, especially for the ones with sleeping problems. I personally know relatives with sleeping problems, and they can go to bed at 10pm but are unable to fall asleep until 1am. It will be bad if they set the time too early, which can make them awake from the almost sleeping state if they find themselves awake in total darkness, and it can also stress them to sleep faster before the light is turned off, which will make them even less likely to fall asleep due to the stress [6]. It will also not be too good if they set the time too late because it means they will sleep with the light on for a while, and therefore enjoying less good sleeping quality.

Our goal is to detect the user's sleep. The method I use is detecting the heart rate from the user's and evaluating if the user is sleeping or not. I train the program to learn the sleeping heart rate and awake heart rate. Then I have the program compare how similar the heart rate sound of the user is to the trained model. Then I will have the AI determine if the user is asleep or not. If the user is asleep, I turn the light off. The first strength of my method is that I use the wearOS system in the method, and it can be integrated with switching lights off with, which is something that regular fitness bands cannot really do because they do not have an outside interface and cannot be connected to the internet [7]. The second strength is that my program builds it's AI model on top of the specific user's heart rate instead of heart rate from a big database [8]. Which is more personalized and can sometimes mean higher detection rate for people who have a more irregular heart rate pattern.

First, I tested the app on myself, and the app gives an 63% accuracy rate immediately after detecting sleep. It gives a 73% accuracy in 5 minutes of sleep, which shows the method is an effective method. I tested it in the scenario where I wear the wearable, and the smart plug I used is wyze, which is connected to IFTTT [9]. The smart plug is connected to the lamp in my room, and my room has minor background noises. In other scenarios where I trained the machine learning model of the app on myself for a week. In 73% of the trials, where the light immediately closes after sleep is detected, the user is asleep. And in 80% of the trails, where the light closes 5 minutes after sleep is detected, the user is asleep. The watch also trains data on the same person over time, therefore increasing the accuracy over time. Data will be fitted more and more throughout time, and accuracy will increase. The part where the signal sent to the smart plug currently always works, as long as there is internet, showing this is a very compatible process, and it also works with most of the plugs out there.

The paper is organized into six sections. The first section is this section, which is the introduction. The second section is challenges, it's about the challenge I face during the process of experimenting and creating the program. The third section is about my solution to the problems that I face in section two. The fourth section is about the details of my experiment and program.

The fifth section is about related works to my paper, and comparison between our work. The sixth section is about the conclusions, and it also points to what I will be adding on to this project and experiment in the future.

2. CHALLENGES

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

2.1. Finding how to detect people's sleep

The first way is using an accelerometer, but it is to irregular, people can have really different habits [10]. The second way is using a wrist band or smart watch to detect user's heartbeat. And when the heartbeat drops to a certain level, determine if the user is asleep. It could also be not so accurate because different people can have different heart rate. For example, awake older people can have lower heart rate than sleeping young man. Then I tried to use dynamic heart rate determination based on the user's normal heart rate, but it still does not work quite well because people's heart rate can flow up and down even during sleep and it is hard to actually determine if one is actually asleep or the heart rate just dropped by accident.

2.2. Finding ways to turn off light switch

My first thought of myself for turning off the light switch is trying to find an apis of smart home companies to have my code able to connect with the light switch. But that does not really work because they don't really have only open apis. I later learned it is because it would make the smart plugs too easy to hack with an open api. And it would cause security issues. Because most smart plugs are connected with google home. But that does not really work either as it requires code to run off of a computer or raspberry pi. But I am sending code off of a watch, and google home doesn't really support that. I also tried to look at android documentation of talking to google home on android device, but that documentation is outdated and cannot be used now.

2.3. Finding ways to send a signal from a watch to the server

I originally wanted to use a Firebase to connect with the watch. But the server I choose cannot detect Firebase changes. It however can detect changes in a set google sheet or an email sent to the server. I first tried with google sheet but it does not really work because works that alter google sheet cannot really be publicly used without a license. So I decided to send the server an email as a signal. The problem is that most of the demonstrations online and APIs are being demonstrated off of a phone. And when I try to use them on the watch, it shows no error but it does not work. There is really no demonstration of sending email off of a watch online because it is a less exposed area.

3. SOLUTION

The whole system works by first detecting the heart rate of the user by the use of a smart watch. Then we input the heart rate into a machine learning system that is written in python. I do that by using an implementation called chaquopy. The program will not be usable on the first day, as on the first day, the heart rate data will be written into two files, one for resting heart rate and one for sleeping heart rate. On the second day, the AI will be able to determine if the user is asleep or not based on the model created on the first day. Once the AI determines that the user is asleep, it will send an email to an outside app called IFTTT. To send an email, the first thing that you need to do is get host names, smtp services and more. Then I need the program to detect how the thing works. Then I will need to pass the username and password of the test email in to send it. After the email is sent, it triggers a trigger in the IFTTT, which can connect to any service or products that is already connected with IFTTT, including many smart home apps. Then just choose a light to close in the smart home product menu to stop the light in your home.

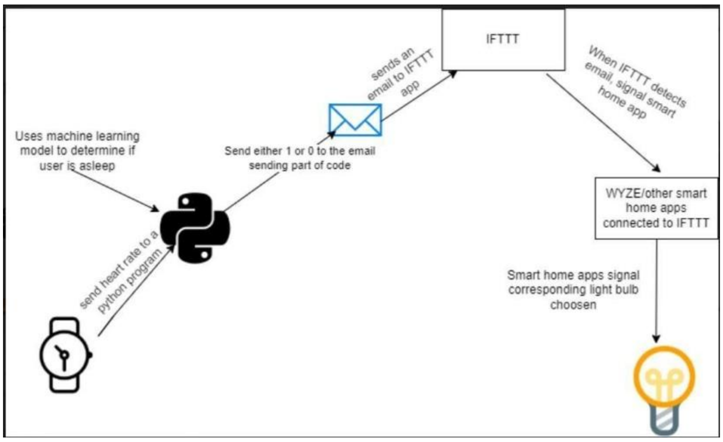

Figure 1. Overview of the system

The implementation of the heart rate detecting uses android permission and sensor system. It uses the sensorOnChange detection system. Which returns data every time that the sensor changes value, which is really fast. The implementation of machine learning is done by python with the chaquopy. There are two choices on the Java UI, one is detect rest heart rate and another is detect sleep heart rate. The python code takes in the heart rate detected from rest and sleep, and input them into two different clusters. Then train the model based on the clusters so that when a new heart rate comes in, the machine is able to determine if the user is asleep or not. The implementation of sending email uses three Java Jars: Mail.jar, activation.jar, and additional.jar. It creates smtp services to send email. The implementation of all 3 packs ensures that email sending is doable on all of the WearOS devices because some wearOS devices are different from others. The implementation of triggering light turning off is through the use of an outside program IFTTT. It is a program that can trigger actions in other registered apps. The user can set the triggered app to any smart home app registered on IFTTT. For testing purposes, I use a wyze application in the triggered application. Then you can set the trigger to anything you want. I set the trigger to email sent with a certain tag. The tag is an uuid given to the user username so the IFTTT can distinguish which user sends the email. The heart rate detection gives heart rate input to the chaquopy that's connected to a machine learning model in python every 10 seconds. Every 100 seconds, there will be 10 heart rates in the python program, and the program will put the 10 heart rates together into a machine learning model because sometimes people's heart rate can fall very low during awake state, so one heart rate cannot determine if one's asleep. The machine learning model will return either a 0 or a 1 after analysis. A 0 means the user is not asleep and 1 means the user is asleep. Once the Java side of the code gets back a 1 value. Then, the email sending code will be activated. It sends an email through the internet to the IFTTT trigger. Then as the IFTTT trigger detects the email with a tag in it's application, it will signal the triggered element the email tag trigger connects to. It uses the tag on the email I send out to specify which specific triggered element is wanted to trigger, then it sends the signal to the right smart home plug. The plug will turn it's light off as it receives this signal.

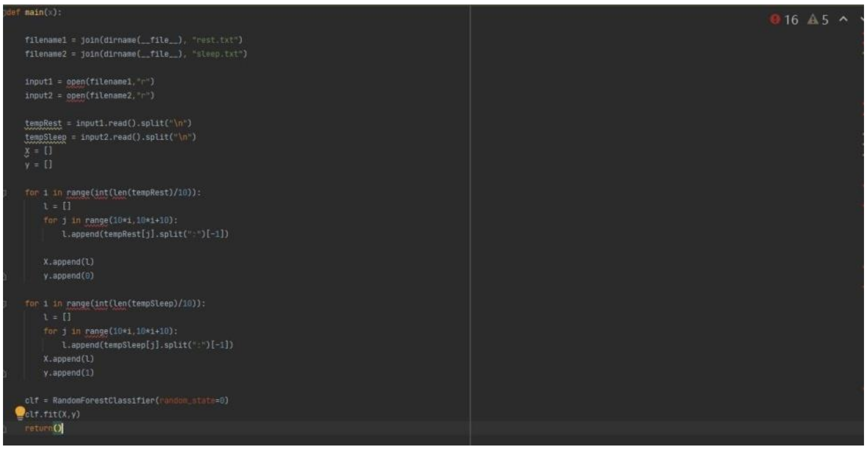

Figure 2. Python code that inputs data into the model

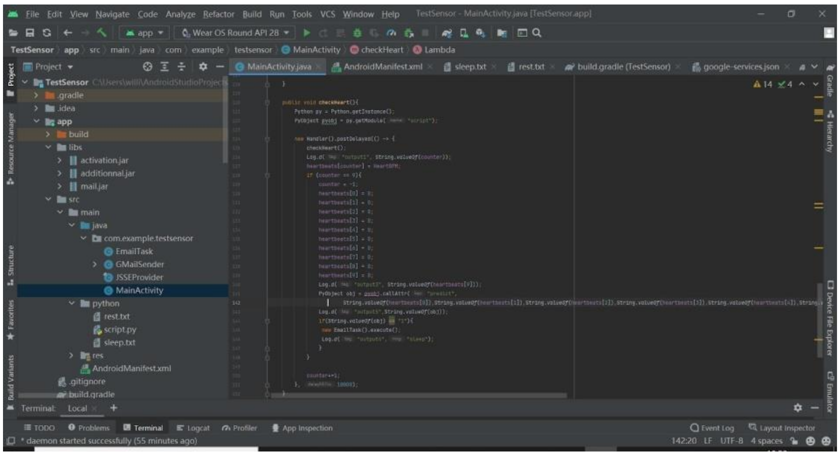

Figure 3. Java code that inputs prediction into python side

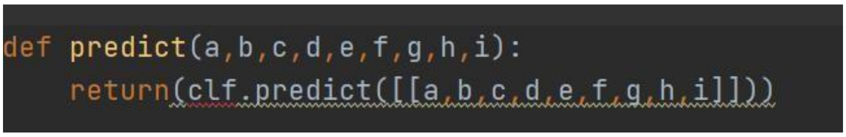

Figure 4. Python that predicts if user is asleep

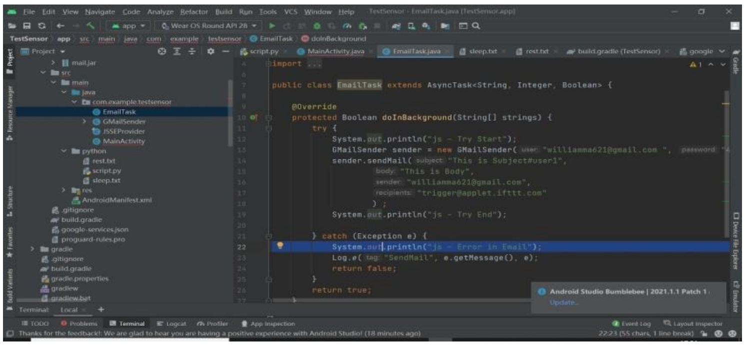

Figure 5. Email Task Code

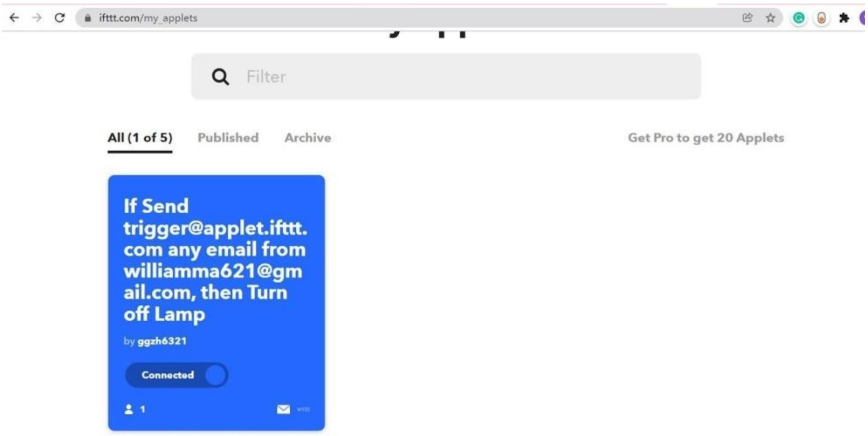

Figure 6. IFTTT

4. EXPERIMENT

4.1. Experiment 1

My solution solves the problems by having a good accuracy rate for detecting sleep and having a very good rate of success in connecting to the smart plug to turn off the light. The experiment is scientific because it's repeatable and there are statistically enough trials taken to show a good accuracy. The first day I detected the user's resting heart rate using the program when he's awake. Then I detected the user's sleeping heart rate by detecting the user's heart rate throughout the night, and taking the heart rate collected at intervals of time in which the user is sure he is asleep. Then on the second day, the program will analyze the data collected on the first day to decide whether the user is asleep, and send a signal immediately when the program model decides that the user is asleep to the light bulb. Due to experimental purposes, the program will also produce a sound to wake the user up. After he is woken up, he will state if he was asleep or not when the sound is played. In an attempt to increase the rate in which the user is asleep when the light is turned off, we also tested sending a signal to the light bulb and playing the sound 5 minutes after the model declared the user is asleep.

Table 1. User sleep rate when triggered immediately after the program detects sleep

Trials	Success	Rate
30	19	63.3%

Table 2. User sleep rate when triggered 5 minutes after the program detects sleep

Trials	Success	Rate
30	22	73.3%

Summary: when the program sends a signal to the light bulb 5 minutes after sleep, there is a high rate that the user is already asleep and the user would not be awake.

4.2. Experiment 2

The second experiment uses the same setting as the first experiment, but this time, the program gets the data from the whole week of data on the user's resting heart rate, sleeping heart rate and user's feedback on the machine. This time, the number of trials is lesser because the amount of time required to take enough data for each trial. And below is the result of the experiment on the 8th day.

Table 3. User sleep rate when triggered immediately after the program detects sleep

Trials	Success	Rate
15	11	73%

Table 4. User sleep rate when triggered 5 minutes after the program detects sleep

Trials	Success	Rate
15	12	80%

Summary: when the program sends a signal to the light bulb 5 minutes after sleep, there is a high rate that the user is already asleep and the user would not be awake.

The experiment result shows that there is a higher rate of success when the signal is sent 5 minutes after the user's sleep. It also shows that accuracy rates have an increasing trend. After 1 week, the rate of success in sending signals five minutes after detecting sleep is good as it reaches 80%. This also means that it is probable for the program to continue growing in success rate as more and more data is trained. The success rate of turning lights off is also very good. There are only two cases throughout the entire experiment in which the light bulb did not turn off. After some investigation, both cases are caused by some sort of accidental internet connection failure. Because both the rate of sleep detection and sending light signals is good, it proves the program is useful.

5. RELATED WORK

Sleep detection using an accelerator [11]. This work is great, it uses an accelerator and studies an individual over a long time. It is very accurate. But it requires the user to have a certain LSAD wearable on. The data cannot really be further passed on to other devices. Although mine is less accurate, it can be put on any wearable devices installed with wearOS, and it can be used on many different devices. The main difference is different methods of detection. This work uses an accelerator to study the person over a long period of time, which is hard to integrate with other things. While what I did is use a program that runs on a wearable, and possibly even on a phone in the future, that detects heart rate, and can be integrated to close light or do other things related to sleep.

This work uses sleep detection using EEG, which is certainly very accurate as it directly detects brain waves [12]. This method is very strengthful in it's accuracy and it can also gather data of many other parts about sleep besides falling asleep such as how good the sleep is, and how fast the different stages of sleep cycle in the user. But it is hard to pass data from this headset to do other things on the internet. The headset is also difficult to get. While my method certainly does not have as good of accuracy and detection of many other important things, it has enough accuracy and is compatible with turning lights off when sleep is detected.

This work uses many sensors on both phone and wearable, and passes data between phone and wearable to provide very accurate detection [13]. This is great in it's accuracy and multitude of detection, but this requires users to have too many devices to close the light when sleep is detected. It requires setup on the phone, smart watch, and plug to inter-connect all 3 to work together. This is an overkill on both the phone and the smart watch, as both devices need to stay awake and close to the user, which uses a lot of battery power. My method is easier in doing this because I only require connection between a wearable and any plug, which is much easier although with less accuracy.

6. CONCLUSIONS

In this research, the thing I did is using an android application that detects heart rate, and then, when the machine learning model finds that the heart rate is low enough, it indicates that the user is asleep [14]. As the user is asleep I send an email using 3 implementations, Mail, activation, and additional. They first connect to an smtp online, then send email with the username and password I provided. As I applied the application to experiment, my result contains an 80% accuracy after it is trained on the user for a whole week, and the program sends the signal 5

minutes after detection. Although it does not have 100% accuracy, it proves that this method is a workable approach. The Experiment result indicates that this is an effective way to solve the problem. Although it's accuracy is not entirely accurate, it is sufficient to close the light, and the thing is it is compatible with many different smart home plugs. You do not need to buy a specific one if you already have it. It's a cheap and compatible way to solve the problem without causing too much money and stuff.

The accuracy really needs to be increased as it is the most important part of this project. It's low when first trained, and not so high even when trained after a week. Although I can now still handle this problem by just delaying the time when the program closes the light to increase accuracy, this does not give the user optimal experience. It's also not as practical because it requires both a wrist band and a smart plug, in which many people probably do not have both and would need to buy them to use the program.

In the future, I plan to use machine learning algorithms on a computer to study the user's breathing pattern on a computer [15]. This way, the user will not need to wear a wrist band. And I probably would not need to use IFTTT as an interconnecter between my application and the smart home application because accessing permissions on the computer is much easier.

REFERENCES

[1] Rezk, Mario. "Auto-control: nociones básicas e investigación fundamental." Revista latinoamericana de psicología 8.3 (1976): 389-397.

[2] Malvy, D., and François Chappuis. "Sleeping sickness." Clinical Microbiology and Infection 17.7 (2011): 986-995.

[3] De Santo, Rosa Maria, et al. "Sleeping disorders in early chronic kidney disease." Seminars in nephrology. Vol. 26. No. 1. WB Saunders, 2006.

[4] Léger, Damien, et al. "An international survey of sleeping problems in the general population." Current medical research and opinion 24.1 (2008): 307-317.

[5] Xu, Ke, et al. "Toward software defined smart home." IEEE Communications Magazine 54.5 (2016): 116- 122.

[6] Baum, Andrew, Jerome E. Singer, and Carlene S. Baum. "Stress and the environment." Journal of social issues 37.1 (1981): 4-35.

[7] Liu, Renju, and Felix Xiaozhu Lin. "Understanding the characteristics of android wear os." Proceedings of the 14th annual international conference on mobile systems, applications, and services. 2016.

[8] van Ravenswaaij-Arts, Conny MA, et al. "Heart rate variability." Annals of internal medicine 118.6 (1993): 436-447.

[9] Ovadia, Steven. "Automate the internet with “if this then that”(IFTTT)." Behavioral & social sciences librarian 33.4 (2014): 208-211.

[10] Ravi, Nishkam, et al. "Activity recognition from accelerometer data." Aaai. Vol. 5. No. 2005. 2005.

[11] Girardin Jean-Louis, Daniel F Kripke, Roger J Cole, Joseph D Assmus, Robert D Langer. Sleep detection with an accelerometer actigraph: comparisons with polysomnography, Science Direct, Jan. 2001, www.sciencedirect.com/science/article/abs/pii/S0031938400003553.

[12] Hal, Bryan V., et al. Low-cost EEG-based sleep detection, IEEE, 2014, ieeexplore.ieee.org/abstract/document/6944641.

[13] Martinez, Gonzalo J., et al. Improved Sleep Detection Through the Fusion of Phone Agent and Wearable Data Streams, IEEE, 2020, ieeexplore.ieee.org/abstract/document/9156211/authors#authors.

[14] Enck, William, et al. "A study of android application security." USENIX security symposium. Vol. 2. No. 2. 2011.

[15] Mahesh, Batta. "Machine learning algorithms-a review." International Journal of Science and Research (IJSR).[Internet] 9 (2020): 381-386.