May 2022, Baltimore, MA, US

Boss Sensor in Office

Detecting and predicting the movement of the boss to create a comfortable office environment

Chikai Shen, Qifan Yu
Whiting School of Engineering
Johns Hopkins University
Baltimore, MD, US
cshen23@jh.edu, qyu24@jhmi.edu

Abstract

We might all experienced this kind of situation where we really want to check out the latest game release or shopping website during work hours in the office, and, at the same time, we do not want to find our boss standing right behind us to be able to see our screens. In this report, we designed a human-robot interactive solution to prevent that from happening and provide a comfortable office environment that would set the salaries and bonuses free from penalties

This project is meant to be a pragmatic and open-source project that implements a human-robot interactive design to facilitate the user in an office environment. The hardware is designed to be cheap and accessible for implementation.

1. Introduction

The solution is to implant a sensor and notify users when someone is approaching. Based on the change of the distance (i.e. when someone goes through the door, the distance captured by the ultrasonic sensor will first decrease and then go back up), the sensor could tell if the boss is approaching and notify users by sending emails by wifi module. Once the NodeMCU detects this fluctuation in distance, it will send the notification through the WiFi module to the remote server, and the remote server will push the email notification to users' phones, and computers.

The fundamental statistical model used to predict whether the person passing through is the boss is based on Bayes' theorem of conditional probability and the assumption that the boss has a habit of accessing the office, times during the day when the boss is more likely to walk by the office.

2. Method

The project essentially consists of two stages, the data collection stage and functioning stage, and three components, the hardware support, the front-end user interface, and the back-end server and database management. In the data collection stage, a statistical model will be built by the end of the stage with the sensor inputs along with the user inputs. The functioning stage is where the project device is put to work with the established model. The hardware support is responsible for the input data from the ultrasonic sensor through the Wi-Fi connection. The front-end user interface is responsible for the input data from the user and the interactive notification for the user. The back-end management is responsible for server setup, data collection, and front-end support. The fundamental model of the project is based on Bayes' theorem of conditional probability which will be discussed in the algorithm section, and the project requires a certain amount of collected data to provide a satisfactory prediction.

2.1. NodeMCU and Ultrasonic PING)))

The hardware support includes NodeMCU and PING))) Ultrasonic Distance Sensor. The PING))) sensor with a resolution from 2cm to 400cm and an accuracy of 0.3cm (Santos, 2021) be used to detect a passing object/individual depending on the distance measured. In this project, a sudden decrease in distance measurement from the ultrasonic sensor indicates a person walking by. NodeMCU is a low-cost open source IoT platform with ESP8266 Wi-Fi SoC and other hardware which was based on the ESP-12 module manufactured by the Chinese company Espressif (Pieter, 2017), and it is mainly responsible for establishing the Wi-Fi connection for input/output data exchange through the network. The two units are connected on a breadboard with wiring indicated in figure 1.

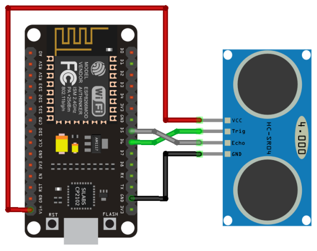
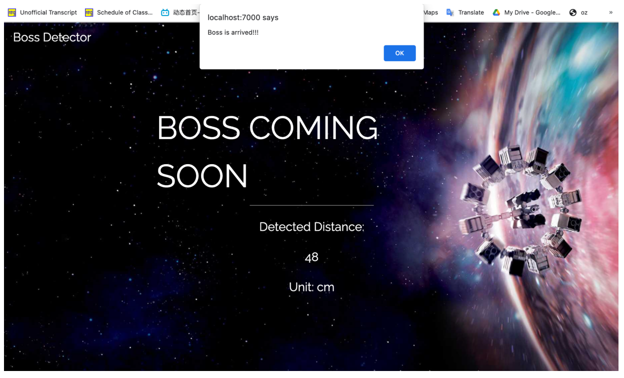


Figure 1: Hardware wire connection

2.2. Front-end User Interface

We used Apache Velocity to design the front-end user interface. Using Javascript to generate HTML with the updated detected distance sent from NodeMCU, We design the notification and pop-up window for user interactions and feedback inputs.



localhost:7000 says

Boss is arrived!!!

Boss Detector

BOSS COMING SOON

Detected Distance:

48

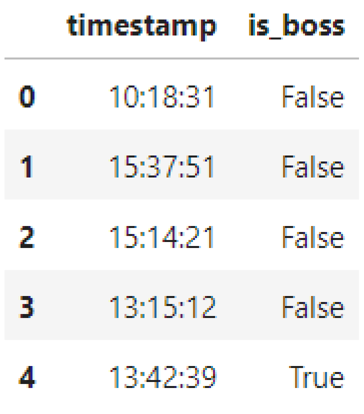
Unit: cm

Figure 2: Front-end interface design

We ask our user to register an account and to input for the 'is_boss' feedback if the current person passed by is the boss upon detection. The feedback will be stored in our database along with a timestamp in the format of "hh:mm:ss" where we look at the data at the day-to-day time frame.

2.3. Back-end Server and Database

Based on user selection, the back-end algorithm will persist the generated timestamp to the database. For backend databases, we use OrmLite to set up classes and tables for the database and spark to set up CRUD API (figure 4). The data stored is in the format demonstrated in figure 3 below.



	timestamp	is_boss
0	10:18:31	False
1	15:37:51	False
2	15:14:21	False
3	13:15:12	False
4	13:42:39	True

Figure 3: Sample of timestamp in the database

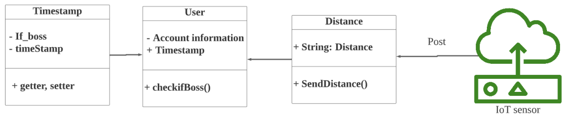


Figure 4: Class diagram for boss sensor

The distance class gets information from NodeMCU and the PING))) sensor and could interact with the user class. The open-source code could be found on our Github at:

https://github.com/qifanyyy/Boss-Sensor

3. Algorithm

The algorithm is based on Bayes' theorem of conditional probability. It builds a statistical model with the time inputs 'timestamp' and user inputs 'is_boss'. Assuming an 8-hour working period from 9:00 to 17:00 (this could be tailored to personal needs in the source code), the time span is divided into 32 15-minute time blocks labeled from 0 to 31 such that the probability of a time block Ti becomes one thirty-second.

$$P\,(Time\ = T_i) = \frac{1}{32}$$

The probability of the person walking by is the boss is calculated by the data collected on the ratio of the number of rows where the boss is spotted.

$$P\,(Boss\ = True)\ = \frac{\#rows(Boss=True)}{\#rows}$$

To predict the probability of the presence of the boss given a certain time block Ti, according to the Bayes' theorem of conditional probability, we need the probability of the certain time block Ti given the presence of the boss, the prior probability of the presence of the boss, and the probability of the time block Ti.

$$P(Time\ = T_i\,|\,Boss\ = True)\ = \frac{\#rows(Boss = True \cap Time = T_i)}{\#rows(Boss=True\ )}$$

Through the collected data, we were able to calculate the above three required probabilities to build our predicted model, shown below, of the probability of the presence of the boss given a certain time block Ti.

$$P(Boss\ = True\,|\,Time\ = T_i) = \frac{P(Time = T_i\,|\,Boss = True) \bullet P(Boss = True)}{P(Time = T_i)}$$

We set an adjustable threshold probability for the model at 0.7 so that if the model predicts the probability will be over 0.7, the boss is believed to be present, and the server will then send out the email notification.

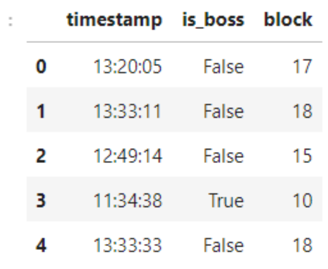
4. Measurement

4.1. Experimental Setup and Data Collection

The hardware units were assembled on a breadboard and put in a paper box for camouflage so that it does not catch as much attention. The sensor was left on a passage where it tends to experience a more selected group of individuals (including the boss). The passage is measured to be 122 cm, so we set the threshold distance at 60 cm, a distance measurement of 60 cm from the ultrasonic sensor indicates an individual walking by. We were able to set up the device and have one participant inputting the feedback of the presence of the boss for two days. Within the two days, we were able to collect 536 rows of inputs of the 'timestamp' and 'is_boss' in the database .

4.2. Data Processing

The main job on data processing is to create a new column 'block', shown in figure 5, to describe the timestamp generated from the server as one of the 32 time blocks we designed. We essentially converted the hours of T_i (subtracted by nine as the working hour starts at 9:00) to minutes and added it with the minutes of T_i to get the total amount of minutes. Since the 8-hour of working time converts to 480 minutes and we divided them into 32 15-minute time blocks, we divided and floored the total minutes of T_i by 15 to acquire the block label from 0 to 31.



:	timestamp	is_boss	block
0	13:20:05	False	17
1	13:33:11	False	18
2	12:49:14	False	15
3	11:34:38	True	10
4	13:33:33	False	18

Figure 5: Sample of processed data

4.3. Measurement Standard

Since asking people to input for the project during their working hours is a very demanding task, we did not get to collect more data or do a test round on our established model. Therefore, we splitted the collected 536 rows of data into a training and test sets with 10% used for testing and the rest used for training. This resulted in 482 rows of data for training and 54 rows of data for testing, which could potentially introduce biases, deviations, and edge cases.

5. Results

With a limited amount of data collected, the model turned out to have an acceptable prediction accuracy of 57.4% as an outcome of two-day data. From the feedback of the participant, the front-end user interface was easy to understand and interact with. However, due to the nature of the data collection stage of the project requiring constant attention from the user, the frequent pop-ups upon sensor detection asking for user input were cutting the participant's working progress and effectiveness. The foremost and most important outcome is that our sensor could connect to the wifi module, send emails, and implement the sensor correctly. We are devoted to delivering the best experience to users, thus some functions may not be intended to resolve the problems mentioned in motivation. We will probably implement a concise user interaction interface that provides a better user experience. The UI will allow users to restart and stop the instance. The final demo is presented with a short experiment presenting our solutio\n. The attendees of the experiments are our teammates.

6. Discussion

Limited by the nature of the data collection stage, the current design requires the user to sacrifice a few days of working quality to build a statistical model for prediction. To improve user experience during the data collection stage, there could be a customizable shortcut to input user feedback on whether the boss is presented so that this constant user interaction does not hinder the working efficiency as much. The original plan was to also develop an application for the project as it allows more freedom and functionalities, but it was not completed due to the time limitation. Thus, another improvement could be a customizable mobile/desktop application that offers different notification options.

A more sophisticated upgrade to completely eliminate the data collection stage is to incorporate a camera for facial recognition machine learning. The user will then be free from being responsible for inputting the status for 'is_boss', but instead, an facial recognition algorithm could be utilized to handle the process with high accuracy. However, since the project was set out to be cheap and accessible for implementation, a high-resolution camera might hurt the budget, and thus was not selected in the first place.

While the straightforward approach of improving the accuracy of the model by spending more time and days collecting data, the training process could be refined by eliminating noise data, specifically removing outliers and reducing duplicates. These approaches allow the model to be trained more effectively without being affected by biases. Another easy modification to potentially improve the model is to adjust the time span of a block. The block is originally set to be 15 minutes, and given different working circumstances, the duration of the block can be modified to fit the environment considering the cultural and habitual factors at different workplaces.

References:

Pieter. (2017, August 3). Establishing a Wi-Fi connection. A beginner's guide to the ESP8266. from https://tttapa.github.io/ESP8266/Chap07%20-%20Wi-Fi%20Connections.html

Santos, R. (2021, October 07). Complete Guide for Ultrasonic sensor HC-SR04 with Arduino. Retrieved May 7, 2022, from https://randomnerdtutorials.com/complete-guide-for-ultrasonic-sensor-hc-sr04/

Appendix:

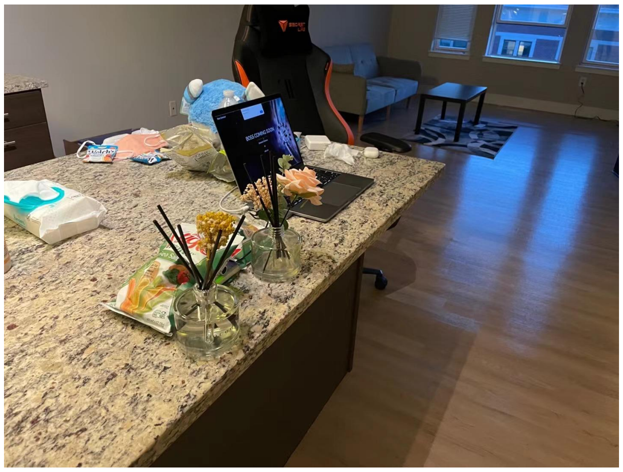


Figure 6: Boss-sensor in a real-life scenario

Boss sensor demo link: https://drive.google.com/file/d/12juEchcf2Bapi2NEIE3cpnhkbjUKBbSu/view?resourcekey