An Exploration of Raspberry Pi's for Contact Tracing

Ojasw Upadhyay[a]

[a]Student Researcher at BWSI piPACT

This manuscript was compiled on February 1, 2021.

Contact tracing is one of the most efficacious applications of epidemiology in response to a viral epidemic or outbreak. Due to the current pandemic's extent and the speed at which the COVID-19 virus spreads, the current systems in place have often been overloaded, making the contemporary manual calls futile. In an effort to speed up contact tracing, the PACT team has come together to create a product that uses Bluetooth Low Energy signals on mobile devices. However, due to Bluetooth's greater prevalence, they have supported a project for high school students who will work towards a proof of concept using just two Raspberry Pi's.

contact tracing | Raspberry Pi | COVID-19 | algorithms | indoors

Introduction

The report covers the data analysis done to analyze the possibility of using Bluetooth towards contact tracing, which would reinforce the current PACT project in creating a robust and accurate way to allow a greater efficiency and speed while contact tracing.

Project Description. My project addresses the basic need for Bluetooth Received Signal Strength Indicator (RSSI), an estimated measure of power level, values to be used to determine the average distance from another Bluetooth device. This project is relevant to piPACT's goals as it sets a baseline for the accuracy, precision, and recall for a basic algorithm trained on around 1500 data points. This baseline is a way to quantify whether a contact tracing algorithm can be created initially that will correctly identify objects that were too close for any period of time. The project was executed by collecting data and then training six different types of models with the data to determine whether an alert should be given for being less than 6 feet next to someone who tested positive for the SARS-CoV-2 virus (1).

Background Information. Most of the information are either directly cited or are from the piPACT Project Webinars.

Raspberry Pi. The Raspberry Pi's (RPis) are set up with Bluetooth 5.0 with the newest model 4B. The basic specifications regarding the Raspberry Pi used are a high-performance 64-bit quad-core processor, 4GB of RAM, dual-band 2.4/5.0 GHz wireless LAN, Bluetooth 5.0, Gigabit Ethernet, and USB 3.0.

Bluetooth. Bluetooth has standards that are set by the Bluetooth Special Interest Group. This ensures that the devices will be interoperable. There are two types of Bluetooth: classic for audio, connecting to a car, etc. and low energy for really low power devices to support broadcasts, which allows to give data to any number of devices. Due to its high prevalence, most of the frequency ranges have been taken in the US, as shown in Figure 1.

The Bluetooth on all of the devices send out "chirps" that can be used to provide information regarding the signal strength of the chirp from another device. This is called "advertising."

Bluetooth Low Energy Advertising BLE devices send three short bursts close to each other and then waits for around 250 ms. The scanning process however, then needs to be as long as at least the waiting interval. The devices use three different frequencies: 2402 MHz, 2426 MHz, and 2480 MHz, as shown in Figure 2 (2).

GATT Generic Attribute Profile, also known as GATT, is the protocol for implementing Blue-

Author contributions: O.U. wrote the paper and did the analyses.

The author declares no conflict of interest.

[1]To whom correspondence should be addressed. The email address is ojaswupadhyay@email.com.

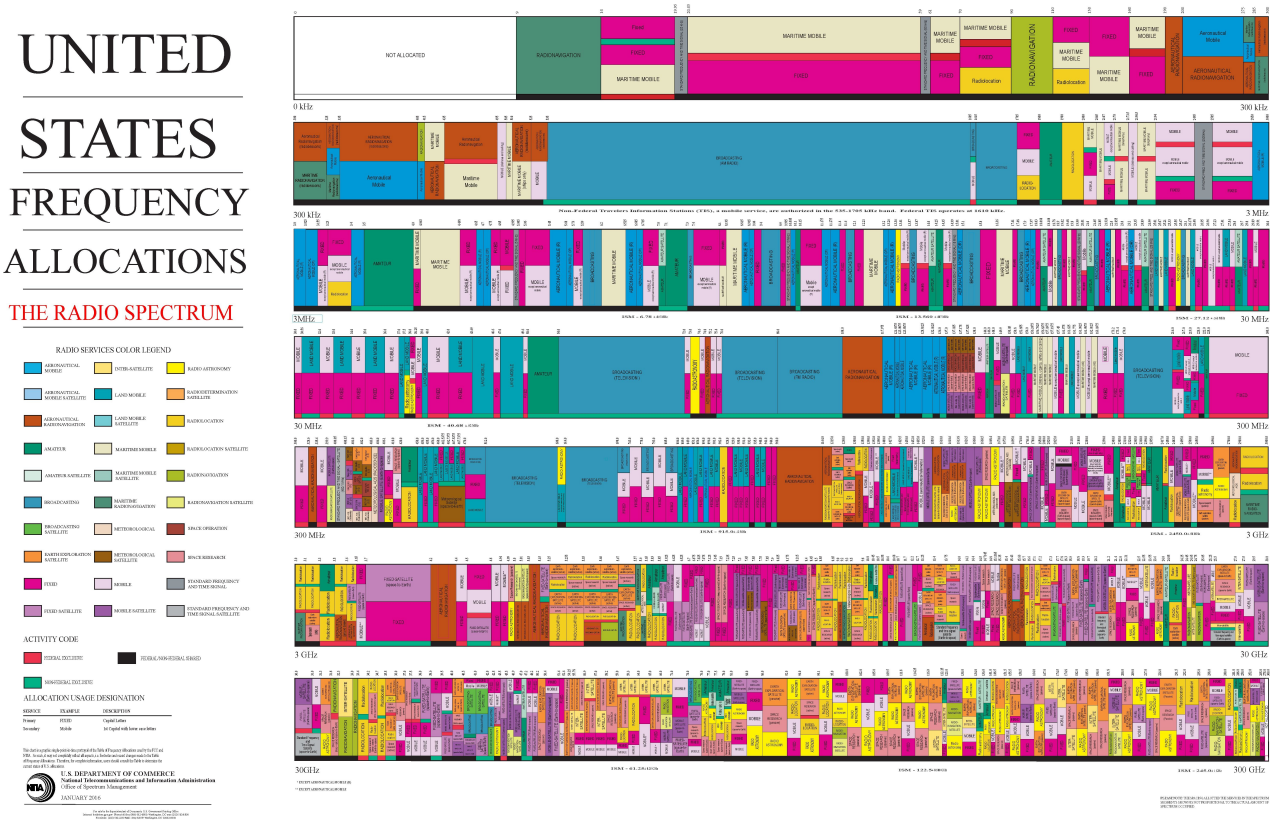

Fig. 1. The image shows a the complicated nature of the radio spectrum and the great number of standards. However, there is a portion of the band on the fifth row towards the right side, which is open to public use and is colored green/teal.

tooth Low Energy (BLE) devices, which was introduced in Bluetooth 4.0. This protocol implements and exposes services, which are a way to transfer data between two devices. Attribute Protocol (ATT) deals with the provisioning and packaging of the data. The data , or characteristic, is packaged within the service. The profile contains all of the services in the device (3). Within piPACT, GATT contains a service for beacons discovery. The discovery process would detect and identify each of the advertising beacons and keep track of all of the interactions.

Location, Obstructions, and the Environment All of my experiments were completed inside my house at the same location and time. Hence, the effect of multiple paths (i.e. Bluetooth signals bouncing off the roof or the walls) are all roughly kept constant. Of course, as the distance between the RPis increased, there was obviously a change in the effect of the walls and such. Moreover, the temperature, humidity, and other factors involving the environment were not addressed as there was no reasonable process by which these potentially confounding factors could be kept constant.

Hypothesis

My hypothesis regarding this project was that a machine learning model could be trained with minimal data to be more than 75% accurate in determining whether the two RPis are too close together, which in the case of COVID-19 is 6 feet or around 2 meters.

This relates to the overall mission of piPACT as it uses the data collected to create an algorithm that would reliably have a high true positive rate and a low false positive rate. Moreover, this is a proof of concept for a further study in using the "Too Close for Too Long" (TC4TL) concept. Moreover, due to its simplicity and lack of confounding variables, which need to be factored

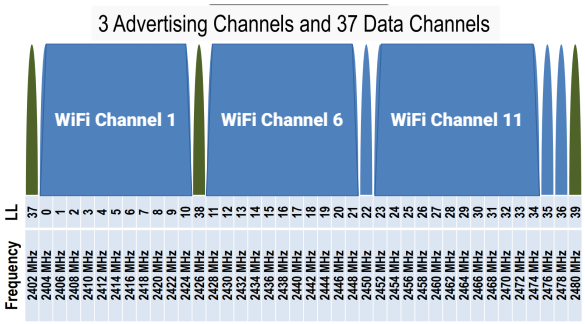

Fig. 2. The image shows all of the available channels within the 2.4 GHz range. The WiFi channels can be grouped as shown in blue. The olive-colored frequencies are the three frequencies at which advertising occurs.

into the models, it creates a better binary classification. However, the architecture of each of the models allow for other aspects to be included despite them not being included.

Most of the investigation came from the optimization of each of the six models trained and the final decision for which model was best suited for the binary classification task using both the "clean" RSSI values as well as those which included noise. The six models that I explored were the Logistical Regression, Decision Tree, Random Forest, Support Vector Machine, Kth Nearest Neighbor, and Two-Class Bayes.

Experiments and Data Collections

Table 1. Experiment Overview

Hypothesis	Reason	Repetitions
The effect of increasing distances on the RSSI values is logarithmic	Empirical quantification of the effect and training data for the models	3 repeats per distance interval for 48 total

The data is shown in Figure 3

Plan and Execution. The plan and execution of the experiment and the data collection are:

1. Measure out the distance that is to be measured. Using a meter stick or a ruler when necessary, between each run make sure that the distance is retained.
2. Ensure that each of the Raspberry Pi's are facing the same direction with them horizontally in line at the same elevation (using textbooks).
3. The advertiser is to remain in the same place while the receiver moves to prevent any changes in how the signal travels.
4. Collect data using the reference code (4).
5. While the approximate times were the same every day, due to the unpredictability of nature, the temperature, humidity, exact composition of the air, or the exact configuration of the room were beyond the scope of the experiment and were not controlled for.

Data Relevance. The only data collected was the distance and RSSI data. This data was essential for creating the training and testing data set for the model. This data would be split into 3 main groups. The training, which is 60% of the data, testing, which is 20%, and validation, which is the last 20% of the data. This breakdown ensures that there is enough data for the training set as the number of data points is low. However, by keeping a large amount for testing and cross-validation, the models can be robustly tested at each of the distances, allowing for a better measure of their reliability.

Examples. The histogram below is representative of all the samples of the data with bearing to the distance, which is shown by the color of the bar and the RSSI value, which is plotted on the x-axis. This is informative as it follows the main premise of my hypothesis that a greater distance will decrease the RSSI. Moreover the higher degree of overlap as the distances increase matches the behavior a logarithmic scale.

Analysis and Algorithms

Description. The creation of the data followed two main phases: processing of the data and the

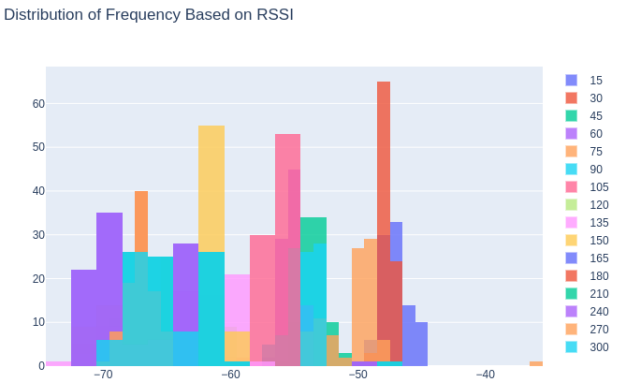

Fig. 3. The figure is a histogram of each of the frequencies of the RSSI values. Each of the colors are a different distance. We can see that, with a few exceptions, an increase in distance moves the values to the left.

machine learning models. The processing of the data followed the sequence below:

1. There are 16 files: each representing a specific distance. Each are fed through a for loop and both the data and file name are added.

2. Using a random seed, a random number is created and the `RSSI_GAUSSIAN_NOISE` column is created based on the RSSI column by adding the random number on a Gaussian curve starting from 0 to 2, which should provide sufficient variance.

3. An `ALERT` column is based on the real distance and is the output that we hope to train the model on.

4. Finally, the `ADDRESS`, `TIMESTAMP`, `UUID`, `MAJOR`, `MINOR`, and `TX POWER` as they were either kept constant or not relevant to the hypothesis.

	RSSI	DISTANCE	ALERT	RSSI_GAUSSIAN_NOISE
count	1480.000000	1480.000000	1480.000000	1480.000000
mean	-60.689189	136.976351	0.751351	-60.707516
std	8.636488	83.201021	0.432376	8.651837
min	-87.000000	15.000000	0.000000	-87.719973
25%	-68.000000	75.000000	1.000000	-68.050383
50%	-61.500000	135.000000	1.000000	-61.700351
75%	-54.000000	180.000000	1.000000	-53.773296
max	-36.000000	300.000000	1.000000	-36.085616

Fig. 4. The above table describes the data collected and the processed columns statistically.

Each of the models were run through both the "clean" data and the data with added noise to test the model's robustness.

Logistical Regression. Logistic regression is a linear model for classification. In this model, the probabilities describing the possible outcomes of a single trial are modeled using a logistic function, a sigmoid curve with a general function $f(x) = \frac{L}{1+e^{-k(\Delta x)}}$ (5).

As an optimization problem, binary class ℓ_2 penalized logistic regression minimizes the following cost function:

$$\min_{w,c} \frac{1}{2} w^T w + C \sum_{i=1}^{n} \log(\exp(-y_i(X_i^T w + c)) + 1).$$

Decision Trees. Decision Trees are a non-parametric supervised learning method used for classification and regression. The goal is to create a model that predicts the value of a target variable by learning simple decision rules inferred from the data features. Given training vectors $x_i \in R^n$, $i = 1, \ldots, l$ and a label vector $y \in R^l$, a decision tree recursively partitions the space such that the samples with the same labels are grouped together (6).

Random Forests. In random forests, each tree in the ensemble is built from a sample drawn with replacement from the training set. Furthermore, when splitting each node during the construction of a tree, the best split is found either from all input features or a random subset of size `max_features` (7).

Support Vector Machine. Given training vectors $x_i \in \mathbb{R}^p$, $i = 1, \ldots, n$, in two classes, and a vector $y \in \{1, -1\}^n$, our goal is to find $w \in \mathbb{R}^p$ and $b \in \mathbb{R}$ such that the prediction given by $\text{sign}(w^T \phi(x) + b)$ is correct for most samples (8).

SVC solves the following problem:

$$\min_{w,b} \frac{1}{2} w^T w + C \sum_{i=1} \max(0, y_i(w^T \phi(x_i) + b)).$$

Kth Nearest Neighbor. Neighbors-based classification is a type of instance-based learning or non-generalizing learning: it does not attempt to construct a general internal model, but simply stores

instances of the training data. Classification is computed from a simple majority vote of the nearest neighbors of each point: a query point is assigned the data class which has the most representatives within the nearest neighbors of the point. `KNeighborsClassifier` implements learning based on the nearest neighbors of each query point, where is an integer value specified by the user (9).

Naive Bayes. `GaussianNB` implements the Gaussian Naive Bayes algorithm for classification. The likelihood of the features is assumed to be Gaussian:

$$P(x_i \mid y) = \frac{1}{\sqrt{2\pi\sigma_y^2}} \exp\left(-\frac{(x_i - \mu_y)^2}{2\sigma_y^2}\right)$$

The parameters σ_y and μ_y are estimated using maximum likelihood (10).

Results and Examples. In the Figures 5 and 6, the results of all of the models are tabulated and summarized based on their accuracy, precision, recall, F1 score (a weighted average of the precision and recall), and the cross-validated precision. From the table, it is clear that the decision tree did the best overall with an especially strong performance on the data that had Gaussian noise added. An interesting artifact is the success of the the Support Vector Machine without the noise. This would need further research into why it is so sensitive to even small changes, which may potentially mean that the model overfitted the data. Moreover, the Random Forest model does surprisingly well with the Gaussian Noise data while doing quite poorly comparatively with the clean data. These are two interesting cases that require more investigation as to what can be done to allow for better results.

	Model	Accuracy	Precision	recall	F1	cv_precision
3	SVM2	0.837838	0.830935	0.995690	0.905882	0.834261
1	d.Tree2	0.831081	0.879167	0.909483	0.894068	0.887436
5	Bayes2	0.831081	0.909910	0.870690	0.889868	0.879042
0	logit2	0.824324	-1.000000	-1.000000	-1.000000	-1.000000
4	kNN2	0.787162	0.862661	0.866379	0.864516	0.879042
2	r.f.2	0.783784	0.865217	0.857759	0.861472	0.879313

Fig. 5. The results are tabulated to show the Accuracy, Precision, Recall, F1 score, and the CV Precision of the six types of machine learning models. These models were trained on data that *did not have any noise added*.

	Model	Accuracy	Precision	recall	F1	cv_precision
1	d.Tree	0.858108	0.886179	0.939655	0.912134	0.871443
2	r.f.	0.858108	0.886179	0.939655	0.912134	0.868558
0	logit	0.837838	-1.000000	-1.000000	-1.000000	-1.000000
5	Bayes	0.834459	0.914027	0.870690	0.891832	0.873756
3	SVM	0.817568	0.834586	0.956897	0.891566	0.833507
4	kNN	0.783784	0.903846	0.810345	0.854545	0.873756

Fig. 6. The results are tabulated to show the Accuracy, Precision, Recall, F1 score, and the CV Precision of the six types of machine learning models. These models were trained on data that *did have Gaussian noise added*.

Conclusions

Hypothesis Evaluation. My hypothesis was found by me to be completely true especially when using a decision tree model which has an accuracy up to 85% and an F1 score above 0.9, which is completely astonishing considering that there were only 1,480 data points to train, test, and validate.

Noteworthy Conclusions. My noteworthy conclusion is that using a machine learning model in binary classification has proved to be highly effective in determining whether the two devices are closer than 6 ft or approximately 2 meters.

General Lessons Learned. Due to my conclusions above, the plausibility and efficacy of Bluetooth-based contact tracing is validated and encouraged.

Next Steps

I am confident that with more data (to refine the models) and more variables added (such as time as well as obstructions and weather to improve versatility), machine learning models provide an exciting way to make a reliable contact tracing algorithm.

ACKNOWLEDGMENTS. This report has made use of the Raspberry Pi Model 4B, provided by Lockheed Martin, the reference code from the BWSI team, and Google Colaboratory for the computations. The project would not have been possible without the great guidance from all of my classmates, Mr. Bhagavatula, and the entire BWSI team; I am deeply indebted to all of you.

1. (2020) *Naming The Coronavirus Disease (COVID-19) and the Virus that Causes It.*

2. Heydon R, Hunn N (2012) Bluetooth low energy. *CSR Presentation, Bluetooth SIG https://www. bluetooth. org/DocMan/handlers/DownloadDoc.ashx*.
3. Townsend K (2020) *Introduction to Bluetooth Low Energy*.
4. Bhagavatula R (2020) *BWSI piPACT Reference Code* (GitHub).
5. (2020) *Linear Models: Logistic Regression*.
6. (2020) *Decision Trees*.
7. (2020) *Random Forests*.
8. (2020) *SVM Classification*.
9. (2020) *Nearest Neighbors*.
10. (2020) *Gaussian Naive Bayes*.

List of Tables

List of Figures