Machine learning model development and deployment for the task of Covid-19 contact tracing

Nam H. Trinh
ML-Labs, School of Computing
Dublin City University
Dublin, Ireland
nam.trinh2@mail.dcu.ie

Delwende Eliane Birba
ML-Labs, School of Computer Science
University College Dublin
Dublin, Ireland
delwende.birba@ucdconnect.ie

Ryan O'Connor
ML-Labs, School of Computer Science
University College Dublin
Dublin, Ireland
ryan.o-connor@ucdconnect.ie

Liliya Makhmutova
ML-Labs, School of Computer Science
Technological University Dublin
Dublin, Ireland
d21124385@mytudublin.ie

Ramin Ranjbarzadeh
ML-Labs, School of Computing
Dublin City University
Dublin, Ireland
ranjbar.ramin24@gmail.com

***Abstract*—A mobile phone app can help to control the COVID-19 epidemic by facilitating contact tracing and alerting close contacts. To do this, a record of proximity events between individuals must be created using data gathered from their mobile phones. In this paper, we attempt to attain an effective and flexible distance estimation between two phones. Firstly, we employ Bluetooth Low Energy (BLE) signals with other on-device sensors including gyroscope and accelerometer data from the NIST challenge dataset. To overcome the of effect of multi-path scattering, we calculate the mean values of these received BLE, gyroscope and accelerometer signals. Next, to explore complex data and classify each input sample, we utilise convolutional neural network, Catboost, and XGBoost models. We also propose a model deployment interface where the input to the interface is an event recording file in CSV format, and the output is the distance predicted by the deployed models. This model deployment interface is a novel step towards the end goal of these distance estimation models; Large scale contact tracing programmes.**

***Index Terms*—contact tracing, Bluetooth Low Energy signals, convolutional neural network, XGBoost, CATboost**

I. Introduction

Coronavirus disease 2019 (COVID-19), which has the ability to transmit rapidly between individuals has lead to an unprecedented global pandemic. COVID-19 is spread between individuals through fecal-oral contamination, virus particles, contamination of surfaces, aerosol, and exhaled droplets.

As vaccines are rarely 100% effective and not expected to control the epidemic within the coming year, further measures to control the virus' spread are needed. Some classical epidemic control approaches to stop the epidemic have been employed including hygiene measures, physical distancing, contact tracing, quarantine, and case isolation [4].

Diverse digital health strategies have been utilized to stop the spread of these viruses during the past pandemics and current COVID-19 pandemic. Among these approaches, contact tracing is one the most effective ways to control the pandemic. Contact tracing can be considered as the identification and isolation of an infected person, as well as the warning of their close contacts. A close contact is a person who has a face-to-face contact with an infected individual without wearing appropriate personal protection equipment (PPE), for a period of at least 15 minutes [2]. By having a temporary record of close contact events between individuals, a mobile phone app is able to trace the close contacts of an infected person and make an instantaneous notification upon case confirmation, prompting the close contact to self-isolate.

However, the contact tracing apps have sparked a heated debate on which sensor technology (i.e. quick response [QR], GPS, or Bluetooth) and deployment pipeline (i.e. decentralized versus centralized) can appropriately overcome critical challenges such as sustainability and effectiveness of restrictions. Private information can be obtained for public control in centralized framework, while gathering private information is not employed in the decentralized strategy [9].

There have been many strategies for proximity sensing, including QR codes [12], Bluetooth Low Energy (BLE) signals [16], [17], GPS [13], WiFi signals [3], [15], and Ultrasound [10]. Current techniques for automated contact tracing and notification employ BLE signals originating from smartphones (chirps) to discover close contact events between a healthy and an infected person.

The decentralized Bluetooth outline recently provided by Google and Apple provides contact tracing with notification to users. The outline also supports the preservation of privacy and provides guidance on how to respond if a close contact risk is recognized. However, as the non-distinguishable data are either deleted or changed occasionally, a fully effective and robust tracking system cannot be obtained. Moreover, the explanation of proximity connection is rather flexible; so, the precision and robustness of detecting an exposure may be different in diverse settings [1], [18].

*All authors contribute equally to the work

Current methods to calibrate Bluetooth signals only employ device-level information Google (2020) and calculate the difference of hardware-level between the devices. Additionally, the value of the received signal strength indicator (RSSI) related to Bluetooth chirps transmitted among phone devices is a noisy estimator of the actual distance among phone devices as they can be vividly disturbed by real-world conditions [6]. Also, RSSI suffers from the presence of other radio signals, physical objects, temperature, interferences due to air, multipath fading, and noise due to which deviations in RSSI happen. These deviations in RSSI produce distance estimation error and lead to a decrease in the value of distance estimation accuracy.

To overcome the aforementioned problems, several strategies try to classify the distance between individuals using BLE signals in the literature. Work in [16] proposed the combination of BLE signals with other on-device sensors (gyroscope, magnetometer, and accelerometer) for contact tracing. By segmenting each 4-second interval into several time-steps, they modelled each experiment as a time series to explore the temporal characteristics of the dataset. In the pre-processing step, the mean value of input signals per each 4-second interval (150) has been selected as the fixed length of each time series. This was chosen in order to diminish the effect of noise in the data, and the need for under-sampling and over-sampling readings. They indicated each time-step as a normalized fixed-length feature vector to demonstrate the most recent values achieved from each sensor. In order to eliminate the impact of presenting noise caused by the multipath effect, they represent the BLE RSSI values in a histogram plot. Therefore, by calculating the mean value from this histogram the effect of multi-path scattering was decreased. Finally, to model this complex data and classify each input sample, several classifiers including Naive Bayes, support vector machines (SVM), and XGBoost have been employed. Work in [5] applied a standardization approach to all events and created variables from the input samples for the pre-processing step. This is due to the fact that the IMU and Bluetooth data are not updated in regular order, and the total number of data points recorded within each four-second window varies between events. Moreover, they applied some distance predictions for each event by generating a row for each new RSSI value and attributing values for all other variables. Then to obtain the final distance calculation, a mean operation (or max/average) was applied to the output distance for each event. Finally, to classify each sample they employ Multi-Layer Perceptron (MLP) and Gradient Boosting Machine (GBM) classifiers. Work in [11] investigated several approaches to solve the problem of non-linearity in Bluetooth Low Energy (BLE) beacons. They employed three Bayesian filters, namely, the Nonparametric Information (NI) filter, the particle filter, and the Kalman filter to improve the accuracy of BLE beacons on a mobile application. They indicated that the most common and simplest filtering approach to diminish the effect of scattering is a simple moving average (SMA). Due to path loss, RSSI can be considered as a nonparametric variable. However, easily tuned parameters and computation make the Kalman filter beneficial for proximity estimation using RSSI strategies in a noisy environment for measuring and processing the noise. Moreover, a particle filter can be implemented as a nonlinear filter to work in a noisy environment. This filter is able to estimate the current state by employing all observations k until time t. Also, to handle non-Gaussian and nonlinear conditions, the NI filter is a good option for the non-parametric path loss model. Work in [8] proposed a combination of a Kalman filter (KF) and an Optimized Support Vector Machine (O-SVM) regression (O-SVM-KF) to increase the performance of a distance estimation system at both short-range and long-range distances. When many phones transmit their received signal strength (RSS) values to the server at the same time, a Kalman filter causes heavy traffic load to the server. As a result, they applied KF to smooth the RSS measurements before any further processing. Then, to train the model based on de-noised samples, they employed an optimized SVM classifier.

In this study, we aim to calculate the distance between two phones using data from mobile sensors and the RSSI values from BLE signal logs. Firstly, to investigate the extent of noise present in this data distribution and the contribution of several sensors on the overall task, we analyze and discuss different approaches associated with the data distribution. Employing datasets collected by NIST (collected in coordination with the MIT PACT project), we implement several ML techniques for learning the representations of the data, obtaining the most favorable outcomes by utilizing convolutional neural network, Catboost, and XGBoost models.

II. Data Exploration

A. Dataset

In this study, the MITRE Range Angle Structured (MITRE) and the NIST dataset (collected in coordination with the MIT PACT project) datasets have been employed. In order to obtain all samples, the Structured Contact Tracing Protocol V 2.0 Laboratory (2020) is used which deliver RSSI samples between two phones at varying distances including 1.2, 1.8, 3.0, and 4.5 meters.

These four distances are used as different classes. Moreover, we use gyroscope (angular velocity) and accelerometer (linear acceleration) as other phone sensor data, and the device model, power, and carriage state as the metadata.

B. Data Exploration

During each session of TC4TL challenge many Bluetooth values were obtained. We first implement a data exploration of the Bluetooth values time series. We test the sequences for normality and explore their distributions. Several histograms of Bluetooth values are shown in Figure 1. We also implement several statistical tests on the data including Shapiro-Wilk and Kolmogorov-Smirnov to test the normality of the data. The results from both of these tests are that the statistical values are close to 1 while the p-value is extremely small and close

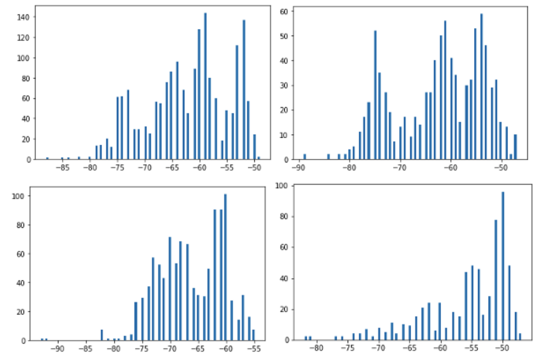

Fig. 1. Bluetooth value distribution.

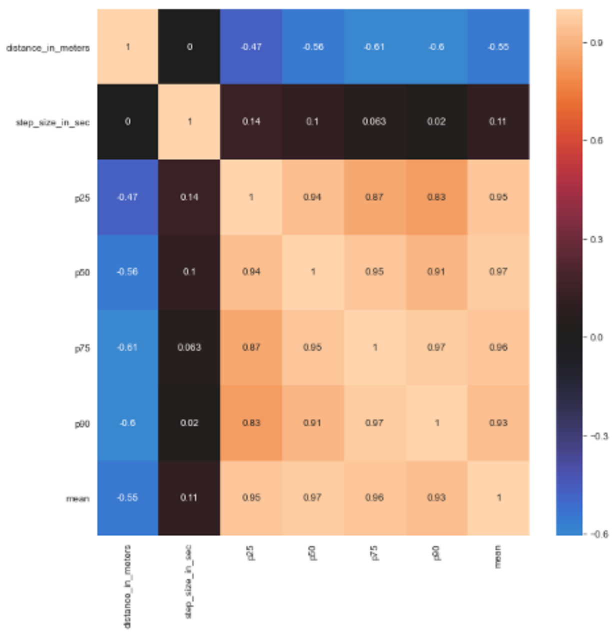

Fig. 2. The correlation between Bluetooth statistical values and the distance

to 0. These results indicate that the Bluetooth series values do not follow a normal distribution.

We subsequently explore the correlation of the Bluetooth signal features and the distance. We pick several statistical values from Bluetooth signals such as 25^{th} percentile, 50^{th} percentile, 75^{th} percentile, 90^{th} percentile, maximum and minimum. We then use the Spearman correlation to compute the correlation matrix between statistical Bluetooth values and the distance. We display such correlation matrix in Figure 2. We can also see on Figure 3 that 25^{th} percentile varies significantly across different distance categories, which indicates that this summary statistic could possibly add discriminatory value to our classifiers.

III. Methodology

To tackle the problem of distance prediction from each contact tracing event, we implement several methods including CatBoost, extreme gradient boosting (XGBoost) and a convolutional neural network (CNN). For each method, the provided TC4TL training dataset is used to explore the feature space and train the classifiers. Testing is carried out on the test dataset which has also been provided for the TC4TL challenge. We use the normalised decision cost function (nDCF) [14] as our evaluation metric, as provided from NIST's scoring software.

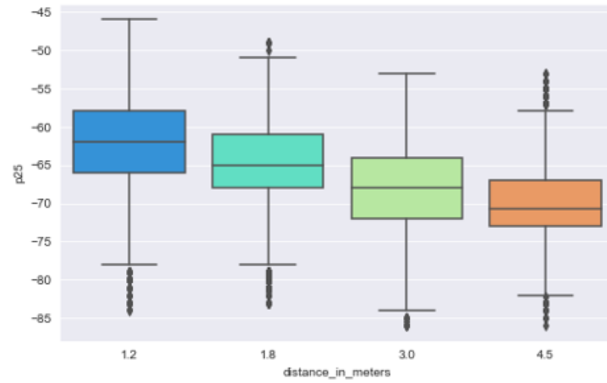

Fig. 3. The boxplot between Bluetooth 25^{th} percentile values and the distance

A. CatBoost

The first method we develop is based on the statistical Bluetooth data from our data exploration (see Section II). We choose the following statiscal Bluetooth RSSI values for our CatBoost classifier: step size in sec, coarse grain, minimum, maximum, 25^{th} percentile, 50^{th}, percentile, 75^{th} percentile, 90^{th} percentile, mean and pose.

The results of the CatBoost model are displayed in Table I.

TABLE I
Output results with the CatBoost model

Subset	D	P_miss	P_fa	nDCF
fine grain	1.2	0.47	0.19	0.66
fine grain	1.8	0.29	0.37	0.66
fine grain	3.0	0.24	0.58	0.82
coarse grain	1.8	0.29	0.20	0.50
Average		0.32	0.34	**0.66**

B. XGBoost

To develop the Extreme Gradient Boosting (XGBoost) model, we first extract the data from the event file by calculating the mean values of Bluetooth RSSI, accelerometer and gyroscope information. We also add a feature called path loss attenuation to the feature. The path loss attenuation equation is shown in Equation 1 where PL is the path loss attenuation, $TXPower$ is the power of the transmitter and MB is the mean of Bluetooth values.

$$PL = TXPower - 41 - MB \tag{1}$$

After extracting the mean values of the sensor, we normalise such mean values to the interval [-1, 1] before feeding the normalised values to the XGBoost model.

The results of the XGBoost model are shown in Table II.

TABLE II
OUTPUT RESULTS WITH THE XGBOOST MODEL

Subset	D	P_miss	P_fa	nDCF
fine grain	1.2	0.31	0.42	0.73
fine grain	1.8	0.05	0.79	0.84
fine grain	3.0	0.05	0.82	0.87
coarse grain	1.8	0.04	0.72	0.76
Average		0.11	0.69	**0.8**

C. Convolutional Neural Network

Noise is prevalent in Bluetooth RSSI and other sensor data in the NIST challenge dataset, so it was important to mitigate as much noise as possible. Inspired by work in [16], we also selected a fixed length of 150 samples per window of 4-seconds to minimise the noise and also to transform the train set into time-series data. Samples are transformed into a fixed-length feature vector representing the most recent values obtained from each sensor. In this method, we use the gyroscope, accelerometer, magnetometer, and RSSI-BLE readings. In addition to such features, we added the distance estimate and experiment-level metadata such as TXDevice, RXDevice, TXPower, RXPower, Device carriage, and activity in our experiments.

Work in [16] conducted a comparison of various deep learning algorithms on MITRE and NIST datasets, and Conv1D has the most favorable results. As a result, we chose to utilise Conv1D as a classification algorithm in our research. We implemented three different 1D convolutional neural networks with different regularizations and dropouts. Instead of a separate training model for fine and coarse grain, we trained on a single model for both. Fine-grain and coarse grain are fed into the same model. Our aim was to test the hypothesis that a single model trained on both fine grain and coarse grain can reduce the average error compared to current state-of-the-art methods. The experiment is optimized using the Adam optimizer [7]. The Conv1D is built using PyTorch with Kaggle environment.

The classification result of the CNN is shown in Table III.

TABLE III
OUTPUT RESULTS WITH THE CNN MODEL

Subset	D	P_miss	P_fa	nDCF
fine grain	1.2	0.57	0.22	0.79
fine grain	1.8	0.68	0.15	0.75
fine grain	3.0	0.60	0.11	0.79
coarse grain	1.8	0.68	0.10	0.70
Average		0.61	0.15	**0.76**

IV. MODEL DEPLOYMENT

To ensure the usability of the model, we built an inference system for model deployment. We built the user interface using Streamlit library. The interface requires an event file containing sensor information that has the same format as the event file from the training dataset. The output is the prediction computed by feeding the input file to the deployed model. The interface is shown in Figure 4.

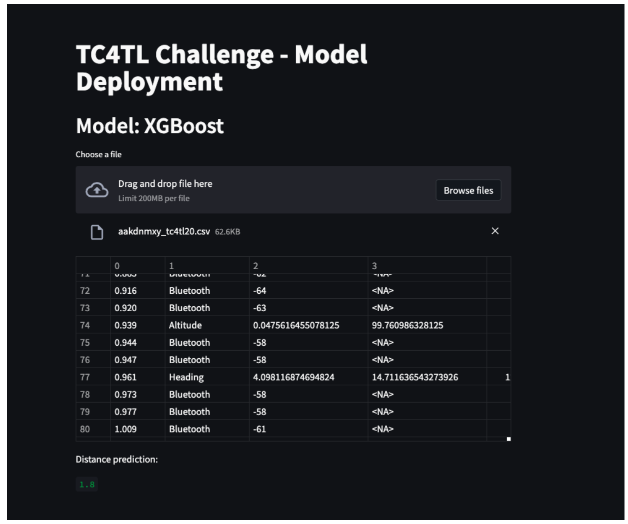

Fig. 4. The model deployment interface with a browse box for uploading file and a distance prediction

V. RESULTS AND DISCUSSION

In this project, we have developed three models based on the CatBoost, XGBoost and CNN methodologies. Among the three methods, our CatBoost achieves the best performance with an average nDCF of 0.66. Though we have not achieved state-of-the-art results, we have successfully built a model deployment system where we can deploy a model of our choice for contact tracing. Once a file is uploaded, the system returns a distance estimate within seconds. Although our models did not achieve the best possible results, state-of-the art results have been achieved using similar modelling methodologies to our own. This indicates that state-of-the-art models should also translate well to our model deployment method. It's important to keep the end goal of this modelling exercise in mind - we need to be able to use the models for effective, large scale contact tracing. These models need to have reasonable run times when creating predictions for new data points. Otherwise, using the models on a nation-wide scale would not be feasible. Our deployment framework seems to indicate that this should be the case for current state-of-the-art modelling techniques.

VI. FUTURE WORK

Future work could go towards advancing the state-of-the-art in the predicting power of these models. This could involve exploring new features that can be extracted from the event file, or exploring new network architectures for the classification tasks. However, we think it would be very interesting to see work that further investigates the large scale deployment of these state-of-the-art models. Related further work could also investigate if users could aggregate data similar to the data provided using only their phone, and use this data to upload files to a model deployment site. This could allow for users to perform contact tracing using their own devices, which could be an effective tool in combating COVID-19 or other future pandemics.

ACKNOWLEDGMENT

This work was funded by Science Foundation Ireland through the SFI Centre for Research Training in Machine Learning (18/CRT/6183). We would also like to thank Devanshu Anand for mentoring us throughout this project.

REFERENCES

[1] Johannes Abeler, Matthias Bäcker, Ulf Buermeyer, Hannah Zillessen, et al. Covid-19 contact tracing and data protection can go together. *JMIR mHealth and uHealth*, 8(4):e19359, 2020.

[2] Hao-Yuan Cheng, Shu-Wan Jian, Ding-Ping Liu, Ta-Chou Ng, Wan-Ting Huang, Hsien-Ho Lin, et al. Contact tracing assessment of covid-19 transmission dynamics in taiwan and risk at different exposure periods before and after symptom onset. *JAMA internal medicine*, 180(9):1156–1163, 2020.

[3] Mikhail Dmitrienko, Abhishek Singh, Patrick Erichsen, and Ramesh Raskar. Proximity inference with wifi-colocation during the covid-19 pandemic. *arXiv preprint arXiv:2009.12699*, 2020.

[4] Luca Ferretti, Chris Wymant, Michelle Kendall, Lele Zhao, Anel Nurtay, Lucie Abeler-Dörner, Michael Parker, David Bonsall, and Christophe Fraser. Quantifying sars-cov-2 transmission suggests epidemic control with digital contact tracing. *Science*, 368(6491), 2020.

[5] Carlos Gómez, Niamh Belton, Boi Quach, Jack Nicholls, and Devanshu Anand. A simplistic machine learning approach to contact tracing. *arXiv preprint arXiv:2012.05940*, 2020.

[6] Gary F Hatke, Monica Montanari, Swaroop Appadwedula, Michael Wentz, John Meklenburg, Louise Ivers, Jennifer Watson, and Paul Fiore. Using bluetooth low energy (ble) signal strength estimation to facilitate contact tracing for covid-19. *arXiv preprint arXiv:2006.15711*, 2020.

[7] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[8] Ching Hong Lam, Pai Chet Ng, and James She. Improved distance estimation with ble beacon using kalman filter and svm. In *2018 IEEE International Conference on Communications (ICC)*, pages 1–6. IEEE, 2018.

[9] Jinfeng Li and Xinyi Guo. Global deployment mappings and challenges of contact-tracing apps for covid-19. *Available at SSRN 3609516*, 2020.

[10] Po-Shen Loh. Flipping the perspective in contact tracing. *arXiv preprint arXiv:2010.03806*, 2020.

[11] Andrew Mackey, Petros Spachos, Liang Song, and Konstantinos N Plataniotis. Improving ble beacon proximity estimation accuracy through bayesian filtering. *IEEE Internet of Things Journal*, 7(4):3160–3169, 2020.

[12] Ichiro Nakamoto, Sheng Wang, Yan Guo, and Weiqing Zhuang. A qr code–based contact tracing framework for sustainable containment of covid-19: Evaluation of an approach to assist the return to normal activity. *JMIR mHealth and uHealth*, 8(9):e22321, 2020.

[13] Ramesh Raskar, Abhishek Singh, Sam Zimmerman, and Shrikant Kanaparti. Adding location and global context to the google/apple exposure notification bluetooth api. *arXiv preprint arXiv:2007.02317*, 2020.

[14] Seyed Omid Sadjadi, Craig S Greenberg, Douglas A Reynolds, et al. Nist pilot too close for too long (tc4tl) challenge evaluation plan. 2020.

[15] Piotr Sapiezynski, Arkadiusz Stopczynski, David Kofoed Wind, Jure Leskovec, and Sune Lehmann. Inferring person-to-person proximity using wifi signals. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, 1(2):1–20, 2017.

[16] Sheshank Shankar, Rishank Kanaparti, Ayush Chopra, Rohan Sukumaran, Parth Patwa, Myungsun Kang, Abhishek Singh, Kevin P McPherson, and Ramesh Raskar. Proximity sensing: Modeling and understanding noisy rssi-ble signals and other mobile sensor data for digital contact tracing. *arXiv preprint arXiv:2009.04991*, 2020.

[17] Fazli Subhan, Asfandyar Khan, Sajid Saleem, Shakeel Ahmed, Muhammad Imran, Zubair Asghar, and Javed Iqbal Bangash. Experimental analysis of received signals strength in bluetooth low energy (ble) and its effect on distance and position estimation. *Transactions on Emerging Telecommunications Technologies*, page e3793, 2019.

[18] Tyler M Yasaka, Brandon M Lehrich, and Ronald Sahyouni. Peer-to-peer contact tracing: development of a privacy-preserving smartphone app. *JMIR mHealth and uHealth*, 8(4):e18936, 2020.