Pervasive and Mobile Computing 77 (2021) 101474

Contents lists available at ScienceDirect

Pervasive and Mobile Computing

journal homepage: www.elsevier.com/locate/pmc

Feature selection for proximity estimation in COVID-19 contact tracing apps based on Bluetooth Low Energy (BLE)

Pablo G. Madoery [a,c], Ramiro Detke [a], Lucas Blanco [a], Sandro Comerci [a], Juan Fraire [a], Aldana Gonzalez Montoro [b], Juan Carlos Bellassai [b], Grisel Britos [b], Silvia Ojeda [b], Jorge M. Finochietto [a,c,*]

[a] *Facultad de Ciencias Exactas, Físicas y Naturales - Universidad Nacional de Córdoba, Av. Velez Sarsfield 1611, Córdoba, Argentina*
[b] *Facultad de Matemática, Astronomía, Física y Computación - Universidad Nacional de Córdoba, Av. Medina Allende 2144, Córdoba, Argentina*
[c] *Instituto de Estudios Avanzados en Ingeniería y Tecnología (IDIT) - CONICET, Av. Velez Sarsfield 1611, Córdoba, Argentina*

ARTICLE INFO

Article history:
Received 21 November 2020
Received in revised form 28 August 2021
Accepted 8 September 2021
Available online 24 September 2021

Keywords:
COVID-19
Bluetooth
Contact tracing
Proximity estimation
Machine learning
Feature selection

ABSTRACT

During the COVID-19 pandemic, contact tracing apps based on the Bluetooth Low Energy (BLE) technology found in smartphones have been deployed by multiple countries despite BLE's debatable performance for determining close contacts among users. Current solutions estimate proximity based on a single feature: the mean attenuation of the BLE signal. In this context, a new generation of these apps which better exploits data from the BLE signal and other sensors available on phones can be fostered. Collected data can be used to extract multiple features that feed machine learning models which can potentially improve the accuracy of today's solutions. In this work, we consider the use of machine learning models to evaluate different feature sets that can be extracted from the received BLE signal, and assess the performance gain as more features are introduced in these models. Since indoor conditions have a strong impact in assessing the risk of being exposed to the SARS-CoV-2, we analyze the environment (indoor or outdoor) role in these models, aiming at understanding the need for apps that could increase proximity accuracy if aware of its environment. Results show that a better accuracy can be obtained in outdoor locations with respect to indoor ones, and that indoor proximity estimation can benefit more from the introduction of more features with respect to the outdoor estimation case. Accuracy can be increased about 10% when multiple features are considered if the device is aware of its environment, reaching a performance of up to 83% in indoor spaces and up to 91% in outdoor ones. These results encourage future contact tracing apps to integrate this awareness not only to better assess the associated risk of a given environment but also to improve the proximity accuracy for detecting close contacts.

1. Introduction

The COVID-19 pandemic has attracted lot of attention on the use of smartphones for contact tracing [1], which consists in identifying close contacts that can be at risk of being infected with the virus. While contact tracing existed as a procedure in public health much before the pandemic, it has never been automated or partly automated by relying on

[*] Correspondence to: Av. Velez Sarsfield 1611, Córdoba, Argentina.
E-mail address: jorge.finochietto@unc.edu.ar (J.M. Finochietto).

https://doi.org/10.1016/j.pmcj.2021.101474
1574-1192/

data collected by smartphones rather than people [2]. The widespread use of smartphones worldwide enables a unique opportunity to implement contact tracing apps as a supplementary means to control the outbreak. Besides the advantage to systematically collect data related to close contacts, these apps can reduce the spread of the infection by alerting potential infected individuals much earlier, even before symptoms onset.

Smartphones are enabling a new dimension of the traditional manual contact tracing [2], now bringing unprecedented tracing speed and coverage [3–6]. The concept has been widely implemented throughout the world [7–10]. Both the public and private sectors have engaged in investigating the approach. Notably, Google and Apple have released tracing services in their operating systems, enabling a true global potential of the approach [11]. However, contact tracing apps have generated much discussion around their key attributes, including system architecture, data management, privacy, security, attack vulnerability, and proximity estimation [12–16].

In this work, we consider one of the most speculative attributes which relates to the capability of these apps to estimate proximity among devices. Most contact tracing apps use Bluetooth Low Energy (BLE) signal attenuation to estimate proximity [17]. However, due to reflection, obstacles, noise and antenna orientation, the relationship between signal attenuation and proximity becomes extremely complex and challenging to model [18].

Current contact tracing apps are unaware of the environment, even if the level of exposure to the virus can be much higher in closed (indoor) spaces with respect to open (outdoor) ones. Hence, it is expected that more evolved apps could also gather auxiliary information to determine the type of location (indoor or outdoor) where a close contact occurred. Current phones have several sensors that can be used for this purpose, among them all have an ambient light sensor (used for adjusting the screen brightness) which can be a good estimator of indoor and outdoor conditions. Besides, awareness of the environment can potentially be used to better estimate proximity among phones. In open spaces, BLE signals mostly follow a free space attenuation model where no obstacles nor reflections are present; however, in closed environments, the observed attenuation of a BLE signal can be lower than expected due to contributions of reflections from walls and obstacles nearby [17]. In this context, machine learning models can also be designed and fitted to exploit the specific features of each scenario.

This work explores the use of machine learning models for proximity estimation and focus on the feature selection process from observed BLE signals. These features are extracted from the observed BLE signals, not limiting to only average or minimum attenuation values but also considering others like standard deviation, range, skewness, etc. Different feature counts and selections are evaluated for both indoor and outdoor environments to better understand the dependency of proximity accuracy on the type of location and its sensitivity to multiple features. Different machine learning models are fitted with an increasing number of features to better understand not only the most relevant features but also to determine their actual number and diversity.

To this end, we designed and developed a custom app targeted to generate and scan BLE beacons, and conducted a series of measurements campaigns under different scenarios comprising different environments (i.e., outdoor or indoor) and multiple distances between the devices (from 0 to 4 m). Collected data was organized into a variety of datasets to support the following contributions of this paper:

- We assess the gain of introducing multiple features for each environment to better understand which one is more sensitive to feature selection, and the required number of features in each case.
- We analyze the influence of the environment (indoor or outdoor) on proximity estimation accuracy in order to determine its role in these models, and to identify the need of contact tracing apps that can be context-aware.

The article is structured as follows. In Section 2, we review recent literature related to contact tracing apps, and focus on recent proposals and its open challenges. Section 3 discusses different models used for proximity estimation, their main parameters, and the motivation behind using machine learning models to improve accuracy on determining close contacts. Since these models are fitted with data, we describe the experimental setup used to collect data in diverse environments in Section 4. Next, in Section 5, we introduce a methodology for selecting features that feed machine learning models. Detailed results to assess the value of feature count in both indoor and outdoor scenarios are presented in Section 6 and a benchmark is provided in Section 7 to compare our best models' results with reference models. Finally, Section 8 summarizes the main findings of this work and provides insights for future works.

2. Background

In order to preemptively break the SARS-CoV-2 chain of transmission, contact tracing has been considered and successfully implemented in several countries [2]. Leveraging smartphones could, in principle, enhance the efficiency of the approach by providing a *quicker* [3,4] and *broader* [5,6] contact tracing process. Moreover, individuals situated two hops away from a confirmed case (so-called ''recursive'' contact tracing) can be easily tracked via smartphones for an even broader coverage [19,20], which proves to be of concrete practical value [21]. Automated contact tracing applications have been reported in Singapore [7], Taiwan [8], India [9], China [10], and many others. In general, the benefits of using contact tracing apps have prevailed over controversial cases where technical, legal, ethical, or privacy issues hindered an adequate implementation [13–16].

On April 2020, Apple and Google launched the Privacy-Preserving Contact Tracing Project, with the aim of developing an interoperable contact tracing technology [11]. Indeed, the combined global reach of the two companies' operating systems made this announcement a promising initiative with the potential to succeed at a global scale. Inspired by the DP-3T [22] and TCN [23] protocols, the companies designed and integrated a specific BLE service (or application programming interface, API) to enable proximity sensing with user privacy and security as the central element [24]. The first specification was released as v1.0 [25], currently known as the (Google/Apple) Exposure Notification (GAEN) system, and currently defined in its v1.7 version of the API [26].

The GAEN specification defines that each device periodically advertises itself using BLE beacons and also scans for other devices' presence. During a scan, a device may detect several identical BLE beacons (up to once every 250 ms), each with their own attenuation. All observed attenuation values during a scan window are grouped together and represented by the average (typical) attenuation. This attenuation value is then used to estimate a level of exposure to a specific device over a period of up to 30 min. This level of exposure is actually inversely proportional to the attenuation values. The lower the attenuation, the higher the level of exposure. In practice, attenuation values are accumulated into 4 different buckets, where each bucket represents a proximity level which could be translated into too-close, close, far, and too-far indications. Each bucket is assigned with a specific weight in order to compute a weighted duration metric for the exposure period, where time spent during a lower attenuation condition counts more than that related to higher attenuation. However, the GAEN specification does not define values for these weights nor for the thresholds of the buckets, which need to be setup through the provided API interface.

During the pandemic, many countries have gravitated toward contact tracing BLE apps, most of them based on the GAEN specification, as they generally represent a more privacy-preserving option compared to using location-based technologies such as GPS. While location is kept private using BLE, user identification might not. In order to estimate infection risk, contact tracing applications need to map received beacons to actual users who were diagnosed with COVID-19. Centralized approaches (i.e., TraceTogether app used in Singapore, or the CovidSafe app used in Australia) rely on a centralized server under control of the healthcare authority which can associate beacon identifiers to users. On the other hand, decentralized contact tracing (i.e., GAEN, DP-3T) improves privacy further by computing infection risk locally on user's devices; thus, only diagnosed identifiers are distributed by the health authority. In both cases, users can be timely warned about exposure risk, which can indeed mitigate COVID-19 spread or other airborne transmission diseases.

Unfortunately, there exists too much optimism about the accuracy for determining close contacts using BLE. This technology was not originally designed to accurately estimate the distance between cellphones. Current generation of contact tracing apps rely on either theoretical or statistical models to translate smoothed attenuation values to distances [27] and determine close contacts. To better characterize the effectiveness of range and time estimation using the BLE signal, many researchers around the world are collecting BLE data as well as other phone sensor data (e.g., accelerometer, gyroscope, proximity) between various types of phones with simulated real-world variability [18].

From May to August 2020, the U. S. National Institute of Standards and Technology (NIST), in coordination with the MIT PACT (Private Automated Contact Tracing) project, organized an online challenge[1] that gathered research groups around the world to design and evaluate solutions for detecting if two people have been too close for too long (TC4TL). The basic task of the challenge consisted in estimating the distance and time between two cellphones given a series of attenuation values along with other cellphone sensor data. Each data series had a contact label associated, i.e., TC4TL or not-TC4TL. These labels were generated using true distance and time from contact events where data were collected. Most authors of this article participated in the challenge which concluded that a more accurate generation of contact tracing apps could be developed based on machine learning models instead of parametric models as mostly used today. However, no clear insights on how to feed these models were obtained from this effort and remained an open question.

Digital contact tracing has emerged as a promising approach to improve the management of the COVID-19 crisis. However, and despite a notable effort of the public and private communities, challenges remain in properly exploiting BLE signals to accurately determine and declare close contacts. Besides *quick* and *broad*, *accuracy* arises as a third fundamental condition for a fully successful smartphone-based contact tracing. Specifically, accuracy is aimed at reducing false negatives and positives, which could otherwise undermine the benefits of the technique [28]. Thus, ensuring accurate contact detection would provide the means to exploit the true potential of contact tracing apps not only with COVID-19, but with any other future air-borne diseases. In this context, the main goal of this work is to provide insights on the features that can be used to feed machine learning models for next generation of contact tracing apps. It is worth mentioning that our work does not aim at finding the best model to estimate proximity but instead identify those features that can provide best results. To this end, we investigate the selection of features derived from the received BLE signal and how the awareness of the environment can be valuable information in the process of proximity estimation.

3. Proximity estimation

Distance is the most straightforward target variable used to estimate proximity between two objects. Proximity is then represented by a numerical variable typically expressed in terms of meters. When two BLE devices communicate, at a given point in time, one is acting as an advertiser and the other as a listener. The former emits its signal at a certain power level, sometimes referred to as *TxPower*, measured in dBm, while the latter observes this signal at an attenuated power level known as received signal strength indicator (RSSI), also expressed in dBm. Since attenuation increases with the

[1] NIST Pilot TC4TL Challenge: https://tc4tlchallenge.nist.gov/.

square of distance (radio waves spread out by the inverse square law), distance D between both devices can be estimated considering the well-known path loss model [29] as follows:

$$D = 10^{\frac{TxPower - RSSI}{10\ n}} \tag{1}$$

where n is the path loss exponent that corresponds to the environment, whose value is normally in the range of 2 to 4 (where 2 is for propagation in free space, 4 is for relatively lossy environments), but can also be slightly lower than 2 (1.6–1.8) for indoor environments where reflections can increase the actual received power. The $TxPower$ value can be carried inside the BLE message payload so that the receiving device can locally calculate the resulting attenuation as $TxPower - RSSI$ to finally estimate the distance D using Eq. (1).

Since smartphones can have different BLE chipsets and/or antenna configurations, observed RSSI values can also depend on specific device models. Hence, a more accurate predictor of distance based on RSSI can be obtained by using a power regression against a known table of distance/RSSI values for a specific receiving device.

$$D = C_1 \left(\frac{RSSI}{TxPower} \right)^{C_2} + C_3 \tag{2}$$

where C_1, C_2 and C_3 are constants derived for specific device models. The *Android Beacon Library*[2] implements this scheme and provides some use cases for these constants.

However, since RSSI values typically fluctuate in time or are influenced by other factors such obstacles and reflections, filtering techniques can be used to smooth its values. Bayesian filtering techniques such Kalman, particle and non-parametric filters have been proposed to mitigate these effects and deliver a better distance estimation [30,31]. These methods compute a single feature, the RSSI value, from a list of noisy measurements, to estimate distance. In this work we consider the impact of using multiple features instead. Even if these features could be extracted from smoothed data generated by filtering techniques, we consider raw data instead to compute them as our main goal is to explore as many different features as possible to better assess their value.

Distance estimation is mainly used in location or tracking applications, which typically assume that BLE messages are received periodically and that the current distance can also be estimated periodically. Instead, in contact tracing apps BLE messages are received during short observation windows spaced in time. Typically, these windows last a few seconds (e.g., 4 s) and are spaced in the order of minutes (e.g., 5 min). Since the actual goal of contact tracing apps is to determine close contact rather than the exact distance among devices, proximity can instead be treated as a classification problem to be addressed for each observation window independently. Thus, we consider proximity estimation as a binary classification problem where the goal is to determine whether an encounter between two devices is a close contact or not. Given the World Health Organization (WHO) defines a close contact as a distance within almost 2 meters (6 ft) for a total of 15 min or more, we focus on determining if the users are within 2 meters or not at a given observation window. Even if the exact distance of a close contact is still under discussion, proximity estimation can also be used to determine the duration of an encounter which, together with the environment, seems to be the most relevant input to assess the risk of exposure to the virus. In other words, we only consider the distance of 2 meters as a reference value for our analysis as has been proposed in other works [32,33].

Previous works related to BLE have proposed the use of machine learning models for determining social interaction among users [34–37]. In this work, we use these models to assess the importance of features that can be extracted from the observed RSSI values collected from different experiments carried out in either outdoor or indoor locations. The process is guided by the goal of exploring the benefit of using multiple features in order to improve the accuracy on determining close contacts.

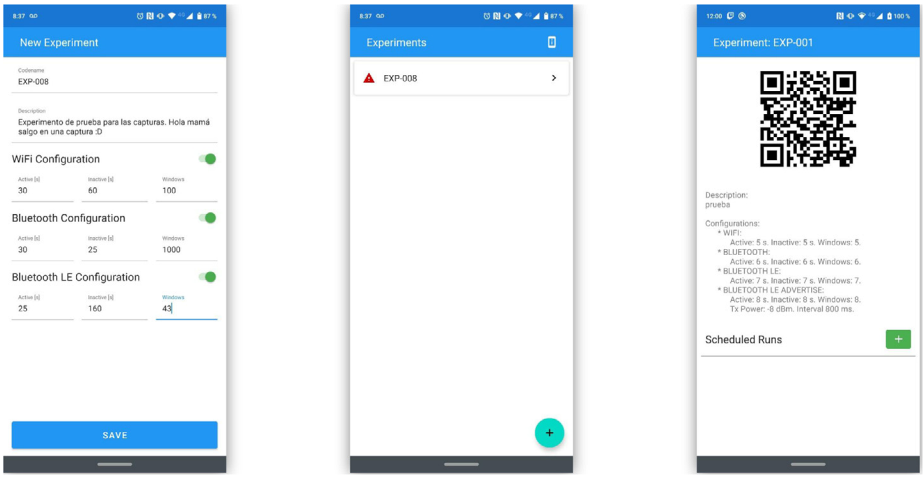
4. Experimental setup

In order to assess the performance of each potential model when multiple features are used, an experimental setup was designed and deployed to carry out experiments. Each experiment consisted of at least two smartphones on the same environment which periodically emitted and scanned BLE beacons during an observation window.

To this end, a custom Android application was developed whose user interface is shown in Fig. 1. This app is available online in a public repository,[3] and can be freely used by the research community to collect data from smartphones. Besides handling BLE beacons, the app is able to log data from multiple sensors on each device. Collected data was then exported to specific files that contain metadata describing the experiment. Even if a few apps that can emit and collect BLE messages exist, they are not oriented to program and run multiple experiments, typically requiring manual operation to configure them and to start and stop observation windows. This strongly limits carrying out large experiment campaigns to collect enough data and may also introduce undesirable variability in experiments due to manual intervention. Hence, we designed and developed an app oriented to running multiple experiments without any intervention (experiments are scheduled and run for specific times and duration).

[2] Android beacon library. https://altbeacon.github.io/android-beacon-library/index.html.

[3] Contactar public repository: https://lcd-unc.github.io/.



Codename: EXP-008

Description: Experimento de prueba para las capturas. Hola mamá salgo en una captura :D

WiFi Configuration — Active [s]: 30, Inactive [s]: 60, Windows: 100

Bluetooth Configuration — Active [s]: 30, Inactive [s]: 25, Windows: 1000

Bluetooth LE Configuration — Active [s]: 25, Inactive [s]: 160, Windows: 43

EXP-008

Description:
prueba

Configurations:
* WIFI:
Active: 5 s. Inactive: 5 s. Windows: 5.
* BLUETOOTH:
Active: 6 s. Inactive: 6 s. Windows: 6.
* BLUETOOTH LE:
Active: 7 s. Inactive: 7 s. Windows: 7.
* BLUETOOTH LE ADVERTISE:
Active: 8 s. Inactive: 8 s. Windows: 8.
Tx Power: -8 dBm. Interval 800 ms.

Scheduled Runs

Fig. 1. App User Interface for scheduling an experiment. **Left**: Setup Experiment **Middle**: List/Add Experiments **Right**: Run Experiment.

Each device is assigned a universally unique identifier (UUID) of 16 bytes, which is sent on the payload of each BLE beacon together with the *TxPower* value. BLE data are collected on each device by logging the arrival time (timestamp), the UUID, the *TxPower*, and the observed *RSSI* value for each received BLE beacon. As discussed in [38], the *TxPower* can be integrated in the estimation model to improve its accuracy since each device can transmit its beacons with a different power value. In [38] the *TxPower* is used as a feature to feed the models, but in this work we use it in combination with the observed *RSSI* values to compute attenuation values, which are then used to feed the models. To this end, for simplicity, all phones used in the experiments were configured to send beacons with the same power level, which was set to 0 dBm.

All experiments lasted at least 5 min and some where extended up to 20 min to better understand the stationarity of the collected data. Observation windows of 5 s were used to capture BLE beacons. These windows were also spaced by default for another 5 s to represent a 50% duty cycle. In real apps, this duty cycle could be further reduced to save power consumption.

Besides the environment (i.e., indoor or outdoor), each experiment was conducted while carefully measuring distances among participating devices (ranges from 0 to 4 meters were considered). Experiments were carried out by placing smartphones horizontally on a flat surface. The rationale was to mimic social gatherings, which have been identified as one of the most critical scenarios for virus infection. Even if other situations could have been considered as having phones at hand or even inside a bag or a pocket, we focused on social gatherings where devices are left on flat surface (i.e., a table).

Different outdoor and indoor spaces were used to conduct measurement campaigns during several weeks. On each space, a balanced number of experiments considering close and far contact distances were carried out. Indoor experiments were conducted at different rooms, comprising both houses and apartments from authors. Outdoor spaces consisted of open parking lots, parks and large building terraces.

Devices from different vendors and models were first analyzed, in particular two different models from Motorola (G8 Play and One Zoom) and two others from Samsung (S20 and A01). This selection was aimed to better understand potential differences between high-end and lower-end technology. Early experiments demonstrated some differences up to 10 dB on the received RSSI values (from the same source) on two different phones; however, this difference showed up to be model specific rather than vendor dependent. Since most significant difference was seen among the two different Motorola models, the final testbed was made up of 2 phone pairs: one Motorola G8 Play pair, and one Motorola One Zoom pair.

A total of 517 experiments were carried out to feed the datasets used in this work. In particular, 248 corresponds to indoor and 269 to the outdoor scenarios. The following distances were used on these experiments: 0, 0.5, 0.8, 1.0, 1.5, 1.8, 2.0, 3.0 and 4.0 m. All data and scripts used in this work are available on the same public repository where the Android app is hosted, and they can be freely used by the research community. As discussed before, distances less than 2 meters were used to label data as *close contact* (302 experiments), and those from 2 meters on, as *far contact* (215 experiments).

5. Feature selection

Data from the aforementioned experiments was used to investigate on the potential sets of features to feed machine learning models. For each experiment, 15 features were derived from the series of normalized RSSI values. These features were: mean, trimmed mean, median, first and third quartiles, minimum, maximum, standard deviation, range,

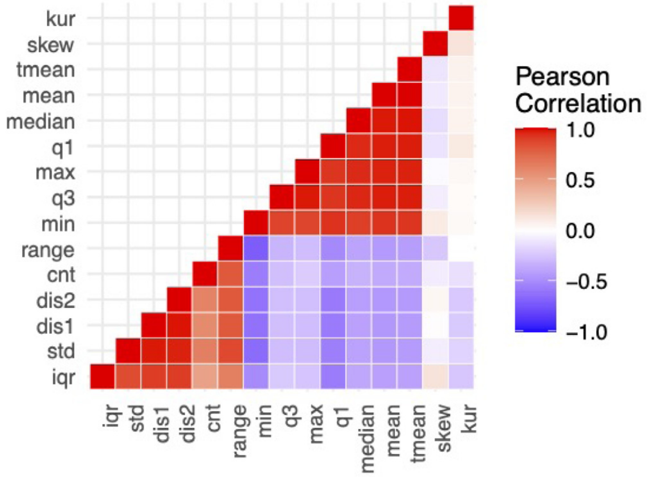


Fig. 2. Correlation between all different features.

Table 1
Feature groups.

Group	Group	Group
Position	Dispersion	Shape
Mean (mean)	Standard deviation (std)	Skewness (skew)
Trimmed mean (tmean)	Range (range)	Kurtosis (kur)
Median (median)	Interquartile range (iqr)	Count (cnt)
First quartile (q1)	L1 distance to the mean (dis1)	
Third quartile (q3)	Mean L1 distance to the median (dis2)	
Minimum (min)		
Maximum (max)		

interquartile range, L1 distance to the mean ($\frac{1}{n}\sum_{i=1}^{n}|x_i-\bar{x}|$), L1 distance to the median ($\frac{1}{n}\sum_{i=1}^{n}|x_i-\tilde{x}|$), kurtosis, skewness, and the count of different values describing the series.

Due to the high dependency among many of these explanatory variables, it may not be worth considering all possible combinations but only a subset of them. Therefore, we performed an analysis by observing the Pearson correlation between each of these features plotted in Fig. 2 heatmap. From this figure, it can be seen that 3 different groups can be clearly identified. Strong correlated features are grouped together. As summarized in Table 1, the 3 groups are the following: measures of position (Group 1), measures of dispersion (Group 2), and measures of shape (Group 3). Within-groups correlations for groups 1 and 2 are larger than between-groups correlations. For instance, all within-group correlations for Group 1 are larger than 0.9 (except for $corr(min-q3)=0.88$). Features in Group 3 are the less correlated.

Even if all features could be considered to feed models, we limit them up to 3 seeking to preserve the principle of parsimony with a more simple model with great explanatory predictive power [39]. Also, as some features are very correlated with each other, including all of them in the model can introduce multicollinearity, which can mask the importance of certain features.

Once the groups were determined, feature selection was performed by choosing variables from different groups. In this way we avoided multicollinearity and reduced the number of models to explore. We evaluated combinations of 1, 2 and 3 features in the following way: when using a unique feature (single feature) it was possible to choose any feature of the position group. When 2 features were used (double feature), it was possible to choose 1 position feature and 1 dispersion feature or 1 position feature and 1 shape feature. When 3 features were evaluated (triple feature) a feature from each group was selected.

Taking this criterion into account, all combinations of 1, 2 and 3 features were evaluated for the 3 different models: Logistic Regression (LR), Support Vector Machine (SVM) and Random Forest (RF). The accuracy was considered as the figure of merit. It can be defined as the percentage of correctly classified instances: $(TP+TN)/(TP+TN+FP+FN)$, where TP, FN, FP and TN represent the number of true positives, false negatives, false positives and true negatives, respectively.

In order to obtain correct estimates of the models' accuracy, classes were balanced, outlier values were removed and each model was trained and evaluated using grid search with cross validation with KFold=5 and with 5 repetitions. Therefore, each accuracy score (condensed into a box plot) was computed in the validation sets as the average of 25

Table 2
Hyperparameters optimization.

	Hyperparameters
LR	[{’penalty’ : [’l2’], ’C’ : [1, 5, 10, 100]}]
SVM	[{’kernel’ : [’rbf’], ’C’ : [1, 10, 100]}, {’kernel’ : [’poly’], ’degree’ : [2], ’C’ : [1, 10, 100]}]
RF	[{’n_estimators’ : [50, 100], ’max_depth’ : [4, 6, 10, 14], ’criterion’ : [’entropy’]}]

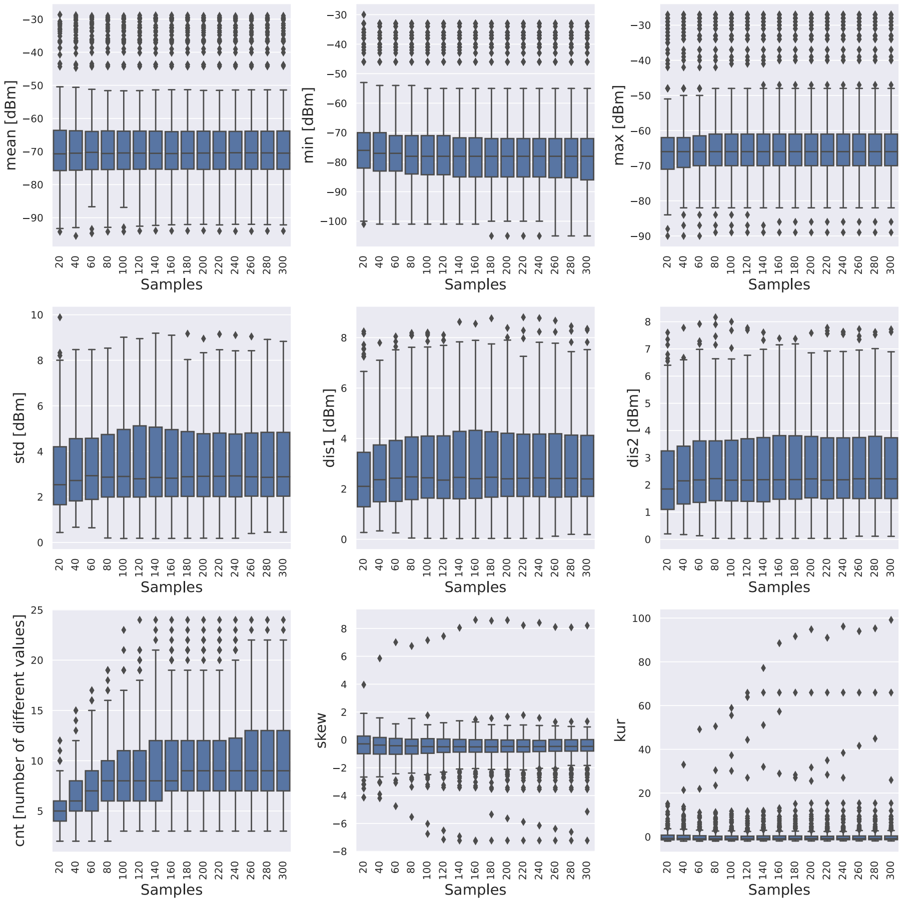


Fig. 3. Feature variability for different RSSI samples length.

values. The Python scikit-learn library and the hyperparameters values defined in Table 2 were used in the grid search with cross validation process. Furthermore, 20% of the dataset was kept out as a test set to evaluate final models’ metrics.

Another important analysis that was taken into account is the number of RSSI samples that are necessary to obtain features with low variability. Fig. 3 shows 3 features of each group in each row. It can be seen that position features like mean and max get stabilized with 60 to 80 samples of RSSI values, while dispersion and shape features require between 120 and 140 samples. It is noteworthy that, in stationary conditions, if the time between samples is in the order of 100 ms to 200 ms, the data collection time required to make a prediction is in the order of 15 to 30 s. If we consider a 50% duty cycle with the objective of saving cellphone battery energy, it is reasonable to make predictions in less than 1 min.

6. Main results

Proximity estimation was analyzed under 3 different scenarios. The first two of them consider that contact tracing apps can be aware of its environment for determining whether a contact is a close one or not. In these scenarios, features are selected to optimize accuracy for a specific dataset related to one specific environment. Hence, selected features can be different for each environment, which means that these apps can have two different models: one for indoor estimation, and another for outdoor. Since the app is aware of its location conditions, it chooses the proper model to estimate proximity. In this way, we can assess whether this awareness is worth to be integrated in these apps to improve its accuracy.

Indoor–Outdoor (IO) detection using a smartphone has been well-investigated and several techniques proposed [40,41], which primarily leverage lightweight sensing resources, such as light sensors, magnetism sensors, cell tower signals and GPS satellites' visibility. Even if it is out of the scope of this work to propose and investigate how awareness can be integrated, we have analyzed its feasibility using only the light sensor available on phones. In our experiments, we were able to predict the type of environment (either indoor or outdoor) where the experiment was carried out with an accuracy of 87%. We claim that this could be further improved by other data potentially available on the phone such as the amount of GPS satellites seen as proposed in [41], which in general is large in outdoor environments with respect to indoor ones. Besides, voting schemes could be implemented if phones advertise in their BLE messages (using a flag field) the type of environment they estimate based on local data. Hence, we claim that awareness integration is feasible on these devices. In this paper, we always assume the smartphone can detect the environment properly as our contribution relates to the analysis of the set of features that can be used for each environment rather than how to detect the environment.

Mixed models (one phone indoor while the other outdoor) are not considered as most likely there is no need to estimate proximity as these conditions could be associated to a low risk exposure (i.e., far contact). However, it is worth mentioning that devices would need to be aware of its context and send in the beacon payload its environment. Hence, each phone could check if both phones belong to the same environment before assessing proximity. If not, then proximity estimation may be not relevant at all.

Having two different proximity estimation models can be justified if there is a performance gain with respect to the case where a single model is used instead. To evaluate this condition, Section 6.3 considers the context-unaware scenario where apps cannot distinguish between indoor and outdoor conditions.

For each environment, we also assess the gain of introducing multiple features in the models. To this end, we evaluate the performance gain of adding up to three features, one from each group as discussed in Section 5. Even if this enables to compare the relative gain of introducing a new feature, it does not provide an absolute comparison to assess its actual performance. To this end, we introduce as a benchmark, the accuracy results derived from using 3 different single-feature models proposed in the literature.

6.1. Indoor aware analysis

First, we analyze proximity estimation in indoor environments. We consider different scenarios defined by the machine learning model and by the number of considered features (from 1 to 3). For each scenario, the best 5 feature sets, out of all possible options, are selected.

Fig. 4 summarizes performance for all 9 different scenarios resulting from the 3 model types and the 3 feature counts. Accuracy for each feature selection is illustrated using box plots ordered by its mean value from left to right. For all cases, results confirm that accuracy improves as more features are fed to the models. However, most significant gain is obtained when considering 2 features instead of only one; while a lower gain is obtained when using 3 features with respect to only 2. Furthermore, accuracy becomes less feature dependent as the number of features is increased. Indeed, accuracy variability is higher for a single feature, but significantly reduced for 3 features.

Among the three model types, LR models show the lowest performance, while RF ones slightly outperform SVM models. Even if we do not aim at selecting one model type, it can be observed that best performing models are less sensitive to feature count. Indeed, the accuracy gain as the number of features is increased is lower for RF with respect to SVM, and also SVM shows a lower gain with respect to LR. Despite these gain differences, we claim that using models with 3 features seems to be a good design criteria if accuracy needs to be maximized.

6.2. Outdoor aware analysis

Next, we consider proximity estimation in outdoor environments. Similar scenarios as for the indoor case are shown in Fig. 5. If compared to Fig. 4, it becomes evident that proximity estimation in outdoor can be much more accurate than for indoor spaces. Besides, accuracy seems to be less sensitive to the model types as all of them have quite similar performance. However, the same trend as for our previous indoor analysis is still present: RF models slightly outperform SVM, while SVM models perform better than LR. Performance gain as the feature count is increased, has a much lower improvement when compared to the indoor case. Indeed, a small but reasonable gain is achieved when considering 2 features with respect to only one, but almost negligible gains are obtained for 3 features. Therefore, outdoor models could rely on 2 features instead of 3 as required for indoor ones. This also provides evidence that the main gain from introducing multiple features is when considering proximity estimation in indoor environments, which actually represent the riskiest location for virus exposure.

6.3. Context unaware analysis

Finally, we compare accuracy results assuming that a single model is used for proximity estimation in both environments. To this end, we compute the average accuracy obtained for models where feature selection is optimized for each specific environment. Hence, we compare the accuracy of indoor models when feature selection is optimized for

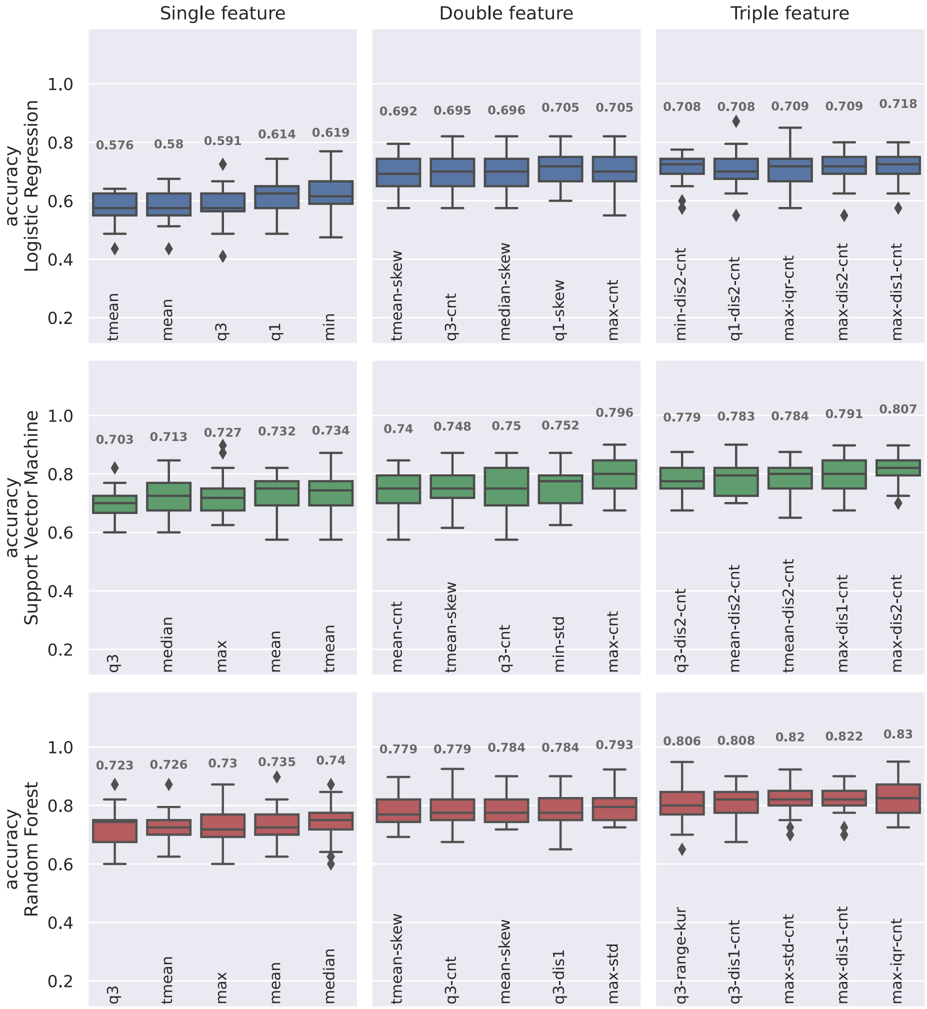


Fig. 4. Indoor accuracy with respect to the number of considered features for different models. **Top**: Logistic Regression Models **Middle**: Support Vector Machine Models **Bottom**: Random Forest Models.

the indoor environment with respect to the accuracy of these models when optimized for an outdoor environment, and vice-versa.

Fig. 6 shows accuracy values for each model type (in different colors). If a single feature is used for these models, then proximity can be estimated using an unaware model since accuracy difference between indoor and outdoor feature selections is negligible. Indeed, from Figs. 4 and 5, it can be observed that the best features for all models and environments are almost the same (i.e. from the same group). However, as the feature count is increased, awareness becomes more relevant as the accuracy for an aware model is much higher than that of an unaware one.

Awareness seems also more relevant for improving proximity estimation in indoor environments. Up to 5% accuracy increase can be obtained for the RF models, and up to 7%, for SVM ones. Instead, for outdoor models the increase is about 3%.

7. Benchmark analysis

Even if our main goal is to assess the performance gain of introducing multiple features in predictive models, comparing these models to others can also provide insights of this performance gain. To this end, we provide as a benchmark the accuracy results derived from using 3 different single-feature models proposed in the literature: the Path Loss Model and the Android Beacon Library Model that were described in Section 3, and an Alternative Model that was described and used in a related recent work [42]. Each of their coefficients/parameters were fitted (data adjusted) using the same data that was used to feed the machine learning models. To perform the fit, the sum of squared errors was minimized using the

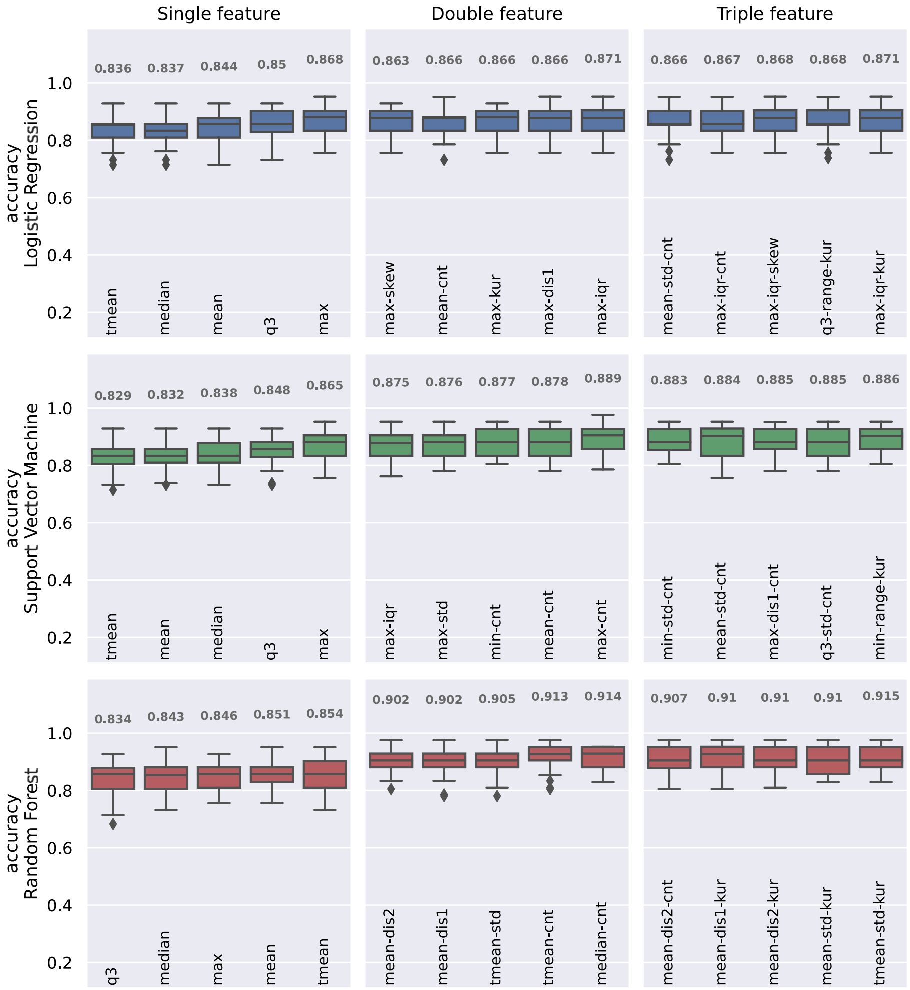


Fig. 5. Outdoor accuracy with respect to the number of considered features for different models. **Top**: Logistic Regression Models **Middle**: Support Vector Machine Models **Bottom**: Random Forest Models.

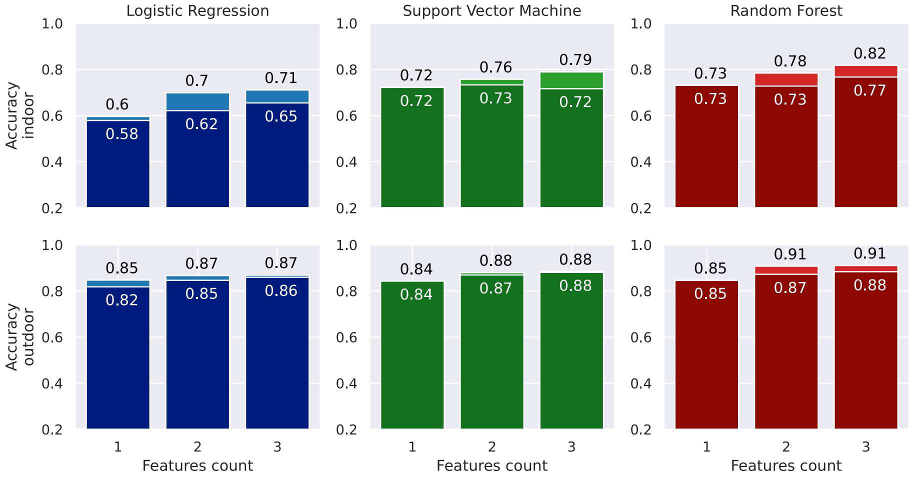


Fig. 6. Accuracy when apps are unaware of the environment. **Top**: Performance of indoor models when feature selection is aware of the environment (light color) and when is not (dark color) **Bottom**: Performance of outdoor models when feature selection is aware of the environment (light color) and when is not (dark color).

Table 3
Comparison of accuracy obtained in indoor and outdoor environments with single-feature reference models: Path Loss Model, Android Beacon Library Model, Alternative Model, and models that can incorporate more features: Logistic Regression, Support Vector Machine and Random Forest. Best case for each model is highlighted in bold font.

Model	Indoor accuracy	Outdoor accuracy
Path Loss Model (data adjusted coefs)	**61.7%**	**80.8%**
Android Beacon Lib (Nexus 5 coefs)	48.4%	82.7%
Android Beacon Lib (Moto X Pro coefs)	49.6%	72.7%
Android Beacon Lib (XT1092 coefs)	49.2%	66.9%
Android Beacon Lib (data adjusted coefs)	**60.9%**	**83.5%**
Alternative Model (data adjusted coefs)	**68.9%**	**87.7%**
Logistic Regression 1 Feature	61.9%	86.8%
Logistic Regression 2 Features	70.5%	87.1%
Logistic Regression 3 Features	**71.8%**	**87.1%**
Support Vector Machine 1 Feature	73.4%	86.5%
Support Vector Machine 2 Features	79.6%	88.9%
Support Vector Machine 3 Features	**80.7%**	**88.6%**
Random Forest 1 Feature	74.0%	83.4%
Random Forest 2 Features	79.3%	91.4%
Random Forest 3 Features	**83.0%**	**91.5%**

Levenberg–Marquardt algorithm. In addition, in the case of the Android Beacon Library Model, the coefficients proposed for 3 different cellphones were also considered (Nexus 5, Moto X Pro, XT1092).

As shown in Table 3, the Path Loss Model and the Android Beacon Library Model have a lower accuracy than the Logistic Regression with a single feature, while the Alternative Model performs better and approaches the performance of the Logistic Regression with 2 features. In general, our models provide a considerably higher accuracy than the 3 reference models used as benchmarks and the difference is up to 14% with respect to the best reference model when the Random Forest is used in indoor environments which are the places with the highest risk of infection. Based on this analysis, it is clear that all considered machine learning models provide greater performance when at least 2 features are used.

8. Conclusions

Contact tracing apps can become a valuable means to help control virus spread. Even if its potential has been recognized worldwide during the COVID-19 pandemic (with an increasing number of deployments in many countries during 2020), its value has still to be proven in the field. Besides privacy concerns, its actual accuracy in determining close contacts is questionable mainly due to the nature of BLE signals. Current apps make use of simple models to estimate proximity, with little evidence of its resulting performance. In this work, we provided insights to foster the development of a next generation of these apps which could exploit both the awareness of its surrounding environment as well as the use of multiple features to feed models based on machine learning. In particular, we focused on the feature selection process aiming to understand which data could be used to improve accuracy.

Results obtained from experimental campaigns demonstrated that increasing the feature count contributes to enhancing proximity estimation, in particular for indoor environments. Environmental awareness has also been identified as a potential improvement to these apps, not only to better assess exposure risks but also because a better model can be used for each space, increasing the resulting accuracy.

We envision as future work the analysis of other scenarios that can consider greater variability, hence not limiting to social gatherings as mainly assumed in this work. This will enable to validate if our findings can be generalized to other situations that are relevant for addressing this pandemic.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgment

This work was supported by the National Agency of Research, Development and Innovation of Argentina [IP-121 COVID-19, 2020].

References

[1] N. Ahmed, R. Michelin, W. Xue, S. Ruj, R. Malaney, S. Kanhere, A. Seneviratne, W. Hu, H. Janicke, S. Jha, A survey of COVID-19 contact tracing apps, IEEE Access 8 (2020) 134577–134601, http://dx.doi.org/10.1109/ACCESS.2020.3010226.

[2] I. Braithwaite, T. Callender, M. Bullock, R.W. Aldridge, Automated and partly automated contact tracing: a systematic review to inform the control of COVID-19, Lancet Digit. Health (2020).

[3] M.E. Kretzschmar, G. Rozhnova, M.C. Bootsma, M. van Boven, J.H. van de Wijgert, M.J. Bonten, Impact of delays on effectiveness of contact tracing strategies for COVID-19: a modelling study, Lancet Publ. Health 5 (8) (2020) e452–e459.

[4] M.E. Kretzschmar, G. Rozhnova, M. Bootsma, M.E. van Boven, J. van de Wijgert, M. Bonten, Time is of the essence: impact of delays on effectiveness of contact tracing for COVID-19, MedRxiv (2020).

[5] J. Hellewell, S. Abbott, A. Gimma, N.I. Bosse, C.I. Jarvis, T.W. Russell, J.D. Munday, A.J. Kucharski, W.J. Edmunds, F. Sun, et al., Feasibility of controlling COVID-19 outbreaks by isolation of cases and contacts, The Lancet Global Health (2020).

[6] M.J. Keeling, T.D. Hollingsworth, J.M. Read, The efficacy of contact tracing for the containment of the 2019 novel Coronavirus (COVID-19), MedRxiv (2020).

[7] V.J. Lee, C.J. Chiew, W.X. Khong, Interrupting transmission of COVID-19: lessons from containment efforts in Singapore, J. Travel Med. 27 (3) (2020) taaa039.

[8] C. Lin, W.E. Braund, J. Auerbach, J.-H. Chou, J.-H. Teng, P. Tu, J. Mullen, Policy decisions and use of information technology to fight Coronavirus disease, Taiwan, Emerg. Infect. Diseases 26 (7) (2020) 1506.

[9] S. Garg, N. Bhatnagar, N. Gangadharan, A case for participatory disease surveillance of the COVID-19 pandemic in India, JMIR Publ. Health Surveill. 6 (2) (2020) e18795.

[10] P. Mozur, R. Zhong, A. Krolik, In coronavirus fight, China gives citizens a color code, with red flags, New York Times 1 (2020).

[11] D.J. Leith, S. Farrell, GAEN due diligence: Verifying the Google/Apple exposure notification API, in: CoronaDef21, Proceedings of NDSS ‘21, 2021.

[12] L. Maccari, V. Cagno, Do we need a contact tracing app? Comput. Commun. (ISSN: 0140-3664) (2020) http://dx.doi.org/10.1016/j.comcom.2020.11.007, URL: http://www.sciencedirect.com/science/article/pii/S0140366420319873.

[13] M. Burgess, Why the NHS Covid-19 Contact Tracing App Failed, Wired, 2020.

[14] C. Kind, Exit through the app store? Patterns 1 (3) (2020) 100054.

[15] M. Klenk, H. Duijf, C. Engels, Ethics of digital contact tracing and COVID-19: Who is (not) free to go? 2020, Available At SSRN 3595394.

[16] H. Cho, D. Ippolito, Y.W. Yu, Contact tracing mobile apps for COVID-19: Privacy considerations and related trade-offs, 2020, arXiv preprint arXiv:2003.11511.

[17] S. Liu, Y. Jiang, A. Striegel, Face-to-face proximity estimation using bluetooth on smartphones, IEEE Trans. Mob. Comput. 13 (4) (2014) 811–823, http://dx.doi.org/10.1109/TMC.2013.44.

[18] D.J. Leith, S. Farrell, Coronavirus contact tracing: Evaluating the potential of using bluetooth received signal strength for proximity detection, SIGCOMM Comput. Commun. Rev. (ISSN: 0146-4833) 50 (4) (2020) 66–74, http://dx.doi.org/10.1145/3431832.3431840.

[19] W.J. Bradshaw, E.C. Alley, J.H. Huggins, A.L. Lloyd, K.M. Esvelt, Bidirectional contact tracing is required for reliable COVID-19 control, MedRxiv (2020).

[20] S. Kojaku, L. Hébert-Dufresne, Y.-Y. Ahn, The effectiveness of contact tracing in heterogeneous networks, 2020, arXiv preprint arXiv:2005.02362.

[21] A. Anglemyer, T.H. Moore, L. Parker, T. Chambers, A. Grady, K. Chiu, M. Parry, M. Wilczynska, E. Flemyng, L. Bero, Digital contact tracing technologies in epidemics: a rapid review, in: Cochrane Database of Systematic Reviews, vol. 8, John Wiley & Sons, Ltd, 2020.

[22] Decentralized privacy-preserving proximity tracing, 2020, https://github.com/DP-3T/documents/blob/master/DP3T%20White%20Paper.pdf. (Accessed 21-11-2020).

[23] Temporary contact numbers protocol, 2020, URL: https://github.com/TCNCoalition/TCN. (Accessed 21-11-2020).

[24] Y. Gvili, Security analysis of the COVID-19 contact tracing specifications by Apple inc. and Google inc., IACR Cryptol. EPrint Arch. 2020 (2020) 428.

[25] Apple Inc. and Google Inc. Contact Tracing Bluetooth Specification, (v1.1), 2020, URL: https://www.blog.google/documents/58/Contact_Tracing_-_Bluetooth_Specification_v1.1_RYGZbKW.pdf. (Accessed 21-11-2020).

[26] Apple Inc. and Google Inc. Exposure Notifications Bluetooth Specification, (v1.7), 2020, URL: https://developers.google.com/android/exposure-notifications..

[27] Q. Zhao, H. Wen, Z. Lin, D. Xuan, N. Shroff, On the accuracy of measured proximity of Bluetooth-based contact tracing apps, in: International Conference on Security and Privacy in Communication Networks, 2020.

[28] L. Ferretti, C. Wymant, M. Kendall, L. Zhao, A. Nurtay, L. Abeler-Dörner, M. Parker, D. Bonsall, C. Fraser, Quantifying SARS-CoV-2 transmission suggests epidemic control with digital contact tracing, Science 368 (6491) (2020).

[29] T. Rappaport, Wireless Communications: Principles and Practice, second ed., Prentice Hall PTR, USA, ISBN: 0130422320, 2001.

[30] C. Lam, P. Ng, J. She, Improved distance estimation with BLE beacon using Kalman filter and SVM, in: 2018 IEEE International Conference on Communications (ICC), 2018, pp. 1–6, http://dx.doi.org/10.1109/ICC.2018.8423010.

[31] A. Mackey, P. Spachos, L. Song, K. Plataniotis, Improving BLE beacon proximity estimation accuracy through Bayesian filtering, IEEE Internet Things J. 7 (4) (2020) 3160–3169, http://dx.doi.org/10.1109/JIOT.2020.2965583.

[32] D. Yang, E. Yurtsever, V. Renganathan, K.A. Redmill, U. Özgüner, A vision-based social distancing and critical density detection system for COVID-19, Sensors 21 (13) (2021) 4608.

[33] C.J. Wang, Contact-tracing app curbed the spread of COVID in England and Wales, 2021.

[34] N. Palaghias, S. Hoseinitabatabaei, M. Nati, A. Gluhak, K. Moessner, Accurate detection of real-world social interactions with smartphones, in: 2015 IEEE International Conference on Communications (ICC), 2015, pp. 579–585, http://dx.doi.org/10.1109/ICC.2015.7248384.

[35] A. Montanari, S. Nawaz, C. Mascolo, K. Sailer, A study of bluetooth low energy performance for human proximity detection in the workplace, in: 2017 IEEE International Conference on Pervasive Computing and Communications (PerCom), 2017, pp. 90–99, http://dx.doi.org/10.1109/PERCOM.2017.7917855.

[36] M. Girolami, F. Mavilia, F. Delmastro, E. Distefano, Detecting social interactions through commercial mobile devices, in: 2018 IEEE International Conference on Pervasive Computing and Communications Workshops (PerCom Workshops), 2018, pp. 125–130, http://dx.doi.org/10.1109/PERCOMW.2018.8480397.

[37] K. Katevas, K. Hänsel, R. Clegg, I. Leontiadis, H. Haddadi, L. Tokarchuk, Finding dory in the crowd: Detecting social interactions using multi-modal mobile sensing, in: Proceedings of the 1st Workshop on Machine Learning on Edge in Sensor Systems, SenSys-ML 2019, Association for Computing Machinery, New York, NY, USA, ISBN: 9781450370110, 2019, pp. 37–42, http://dx.doi.org/10.1145/3362743.3362959.

[38] M. Al Qathrady, A. Helmy, Improving BLE distance estimation and classification using TX power and machine learning: A comparative analysis, in: Proceedings of the 20th ACM International Conference on Modelling, Analysis and Simulation of Wireless and Mobile Systems, MSWiM ’17, Association for Computing Machinery, New York, NY, USA, ISBN: 9781450351621, 2017, pp. 79–83, http://dx.doi.org/10.1145/3127540.3127577.

[39] J. Vandekerckhove, D. Matzke, E. jan Wagenmakers, Model comparison and the principle of parsimony, in: The Oxford Handbook of Computational and Mathematical Psychology, Oxford University Press, 2015, pp. 300–319.

[40] M. Li, P. Zhou, Y. Zheng, Z. Li, G. Shen, IODetector: A generic service for indoor/outdoor detection, ACM Trans. Sen. Netw. (ISSN: 1550-4859) 11 (2) (2014) http://dx.doi.org/10.1145/2659466.

[41] K. Chen, G. Tan, SatProbe: Low-energy and fast indoor/outdoor detection based on raw GPS processing, in: IEEE INFOCOM 2017 - IEEE Conference on Computer Communications, 2017, pp. 1–9, http://dx.doi.org/10.1109/INFOCOM.2017.8057095.

[42] Z. Su, K. Pahlavan, E. Agu, Performance evaluation of COVID-19 proximity detection using bluetooth LE signal, IEEE Access 9 (2021) 38891–38906, http://dx.doi.org/10.1109/ACCESS.2021.3064323.