# Cellular-Assisted COVID-19 Contact Tracing

Fan Yi, Yaxiong Xie, Kyle Jamieson
Department of Computer Science, Princeton University
{fanyi,yaxiongx,kylej}@cs.princeton.edu

## Abstract

The coronavirus disease (COVID-19) pandemic has caused social and economic upheaval around the world. Contact tracing is a proven effective way that health authorities may contain the spread of COVID-19, but is challenging for airborne disease. In this paper, we propose **LTESafe**, a cellular-assisted privacy-preserving COVID-19 contact tracing system. LTESafe leverages a deep neural network based feature extractor to map the cellular CSI to a high-dimensional feature space, within which the Euclidean distance between points indicates the proximity of devices. By doing so, we preserve user privacy by hiding the physical locations of smartphones and at the same time achieve high accuracy. Our preliminary experimental results demonstrate that LTESafe achieves an overall accuracy of 92.79% in determining whether two devices are within six feet proximity or not, and only misses 1.35% of close contacts.

## CCS Concepts

• **Human-centered computing** → *Ubiquitous and mobile computing*.

## Keywords

COVID-19, Proximity estimation, Contact tracing, Cellular network, LTE, Neural networks

**ACM Reference Format:**
Fan Yi, Yaxiong Xie, Kyle Jamieson. 2021. Cellular-Assisted COVID-19 Contact Tracing. In *2nd Workshop on Deep Learning for Wellbeing Applications Leveraging Mobile Devices and Edge Computing (HealthDL'21), June 24, 2021, Virtual, WI, USA.* ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/3469258.3469848


*HealthDL'21, June 24, 2021, Virtual, WI, USA*

ACM ISBN 978-1-4503-8598-5/21/06…$15.00
https://doi.org/10.1145/3469258.3469848

## 1 Introduction

The ongoing COVID-19 pandemic has already resulted in millions of deaths all around the world, causing unprecedented social and economic crisis. Except vaccines, a proven effective way of containing the spread of COVID-19 is accurate, complete and timely *contact tracing*. According to the world health organization (WHO), a *close contact* is a person who has been within six feet to someone that is COVID-19 positive for more than 15 minutes. Contact tracing is the process of identifying, assessing, and managing people who are close contacts with the contagious positive cases. Currently, contact tracing is accomplished through manually interviewing each COVID-19 positive cases by the health authorities, which, however, is extremely labor intensive, time consuming, and unscalable, motivating techniques that can accelerate and automate the process of contact tracing.

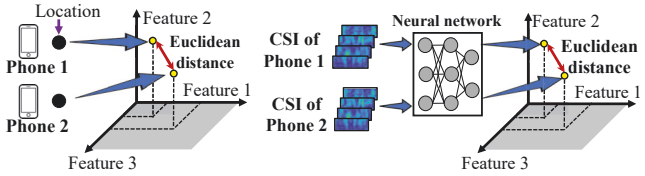
The core task of contact tracing is to identify the close contacts of the positive cases, which requires comparing the locations of two citizens. The ubiquity of smartphones, *e.g.*, more than 81% of Americans own a smartphone in 2019, and their capability of performing device localization and proximity estimation make them ideal devices for building an automatic and fast contact tracing system that scales.

Identifying close contacts by comparing the exact locations of smartphones is a straightforward solution. There exists many mature techniques we can leverage to accurately localize the smartphones, including GPS, Wi-Fi based [18–20], cellular based [6] and Bluetooth based [2, 12]. Exposing the location to any third party, however, hinders the privacy of both healthy citizens and the people who have been infected with the virus, making the location based solutions impractical to implement.

Essentially, identifying close contacts only requires the distance between two people, so localizing the smartphones people carry is overkill. Knowing the proximity of devices is enough for contact tracing, which also preserves user privacy. Proximity estimation using RSSI of Bluetooth has drawn a significant amount of attention from the research community [5, 16], the industry [1, 8], and the government [4].

RSSI based proximity estimation, however, suffers from errors [21], as there are many factors other than the distance that can affect the received signal strength, including hardware imperfections, interference from other signals that share the ISM band and multipath effect. Furthermore, these systems require the device to frequently transmit beacons

to detect each other, which consumes a large amount of energy and also makes the device trackable by malicious third parties.




**(a)** Location mapping. **(b)** CSI mapping.

**Figure 1:** Mapping physical locations of smartphones (**a**) and measured CSI (**b**) to a high-dimensional feature space.

This paper proposes ***LTESafe***, a location based proximity estimation algorithm that is accurate in estimating proximity and at the same time preserves user privacy by hiding the exact user location. We plot the intuition of our algorithm in Figure 1(a), from which we see that, instead of directly comparing the physical locations to get the proximity of two devices, we propose to map the locations to a point inside a high-dimensional feature space and then calculate the Euclidean distance inside the feature space to derive the proximity. To achieve this goal, we have two requirements for such mapping. First, we require that the Euclidean distance of two points in the feature space indicates the proximity of their corresponding physical locations. Second, to preserve user privacy, the mapping should be irreversible so any third party cannot reconstruct the exact user location using the exposed location inside the feature space.

To realize our idea in Figure 1(a), two tasks remain unsolved: localizing the smartphone and finding a mapping that that satisfies the aforementioned two requirements.

For the localization task, we rely on the channel state information (CSI), which fully characterizes the signal propagation between the transceivers, including both the parameters of line-of-sight path and surrounding objects' reflections. To fully make use of the information conveyed by CSI, we directly map the CSI to a point in the feature space, as shown in Figure 1(b). In our implementation, we choose the cellular CSI because of its wide coverage, and the availability of fine-grained cellular reference signals.

For the mapping task, we propose to leverage a deep learning based feature extractor to find the mapping that satisfies our two requirements, as shown in Figure 1(b). Recent advances in deep learning has proven that convolutional deep neural network (CNN), is powerful in selecting representative features for diverse tasks. Therefore, after training, a CNN based feature extractor would automatically select the set of features that forms the desired high-dimensional feature space within which the Euclidean distance indicates the proximity. Furthermore, the extracted features become incomprehensible when the neural network goes deep.

We implement a prototype of LTESafe using USRP as frontend to collect CSI from commercial cell towers. Our experiments show that LTESafe achieves an overall accuracy of 92.79% in determining whether two devices are within six feet or not, and only misses 1.35% of close contacts.

## 2 LTE Primer

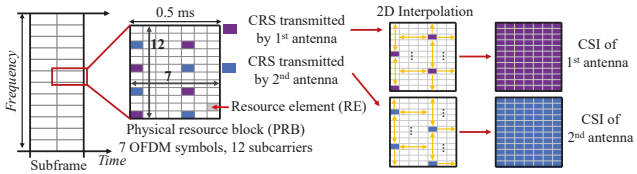



**Figure 2:** Location of LTE CRS and the mobile client side two-dimensional interpolation.

LTE adopts OFDM in the physical layer, so the smallest time-frequency unit is one subcarrier in frequency and one OFDM symbol in time, which is also denoted as one *resource element* (RE). LTE groups all REs inside a block spanning seven OFDM symbols and 12 subcarriers into a *physical resource block* (PRB), as shown in Figure 2. To support multiple access, LTE divides the time into one millisecond length *subframes* and allocates the PRBs inside each subframe to one or multiple mobile users for data transmission.

**Channel estimation in LTE network.** To facilitate channel estimation, the base station transmits predefined *cell specific reference signal* (CRS), inside several specific REs of a PRB, as shown in Figure 2. The CRSs transmitted by multiple antennas of one base station are non-overlapping with each other, so a mobile client can separate them and estimate the channel between each transmitting antennas and its receiving antenna independently. Since the sequence of CRS is known, a mobile client is able to estimate the channel of all REs that carry CRS. To obtain the channel estimation of every RE, the mobile client performs a two-dimensional interpolation, *i.e.*, over time and frequency, as shown in Figure 2. We note that the base station always broadcast the CRS no matter it has data to transmit or not, so a mobile user can estimate the downlink channel at any time point.

## 3 LTESafe Design

In this paper, we propose LTESafe, a cellular-assisted, privacy-preserving contact tracing system, whose architecture is plotted in Figure 3. Generally, LTESafe consists of a mobile client data collection module and server side close contact identifier, which are detailed in the following sections.

**Mobile client side.** Each cellular-connected mobile client first measures the CSI of the channels between itself with

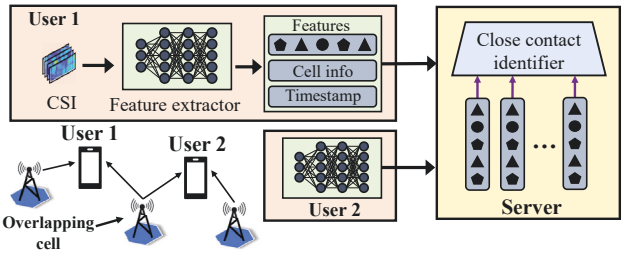



**Figure 3:** The system architecture of LTESafe. The mobile clients extract features from CSIs, and then send packed feature vectors to the server. The server stores feature vectors reported by all mobile clients, and run close contact identifier to find all exposed users.

one or multiple cell towers, then maps the CSI to a vector of contact features via a deep-learning based feature extractor, and at last tags each feature vector with timestamp and information about the cell from which the CSI is measured, including the cell ID, antenna number, and bandwidth. The mobile client regularly uploads the extracted feature vectors together with the tagged information to the server.

**Server side.** The server stores feature vectors reported by all mobile clients, and runs the proximity estimation process to find all potentially exposed users. In the proximity estimation process, the server first identify all the users who appear at the same cell coverage area at the same time with positive COVID-19 users, by checking the timestamp and cell information list associated with positive COVID-19 users against the list of all other users. The server then runs close contact identifier on feature vectors of potentially exposed users and that of positive COVID-19 users to further find out users who have close contacts with positive cases.

## 3.1 Data Preprocessing

The extracted CSI has a granularity of 14 samples per millisecond, which has information redundancy in time domain. To reduce the number of CSI being processed, we downsample the CSI to one sample per 32ms, if the extracted CSI is finer than this granularity.

The measured CSI is noisy because of hardware imperfections. For CSI amplitude, we run Hampel filter to identify and remove outlier CSIs. To eliminate the phase error introduced by frequency offsets and timing offset, we feed the phase difference across antennas to the deep neural network. Phase difference captures the relative relationship between phase across antennas but loses the absolute value. We, therefore, input the sanitized phase of the first antenna [11], to compensate for the information loss.

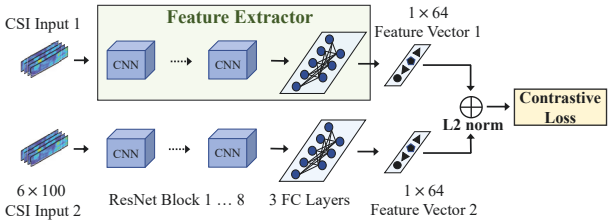



**Figure 4:** The architecture of Siamese network, which runs two identical neural networks on two CSIs.

## 3.2 CNN based Contact Feature Extractor

LTESafe leverages a deep neural network to automatically select a set of features of the CSI. The requirement of the selected features is that the Euclidean distance between the feature vectors of any two CSIs represent the physical proximity of two mobile clients from which the CSIs are measured. We plot the structure of LTESafe's deep learning based feature extractor in Figure 4, which is modified from ResNet [10]. The neural network converts the CSI inputs into the final feature vector.

We adopt the Siamese neural network (sometimes called the Twin network) to train our feature extractor [3], which consists of two identical neural networks that run in parallel, as shown in Figure 4. These two neural networks share weights, and thus extract the same set of features of the input CSI. The Siamese network outputs the Euclidean distance of two feature vectors, which are extracted by the two parallel neural networks.

The goal of training is to minimize the loss function over the training dataset. We use *contrastive loss* [9] as our loss function, which is defined as:

$$\mathcal{L}_c = Y \cdot D^2 + (1 - Y) \cdot [\max(0, m - D)]^2 \quad (1)$$

where $\mathcal{L}_c$ represents the contrastive loss; $D$ is the Euclidean distance of the two feature vectors generated from the two feature extractors; and $Y$ is the proximity ground truth of two CSI inputs, which is a binary value representing whether the two devices are close or not. The $m$ is a configurable hyperparameter whose value is set to 2 in our training.

The combination of Siamese network and contrastive loss turn the training process into a process of finding the weights that guarantee the Euclidean distance is minimized for any CSI pairs that are measured from two mobile devices that are within six feet, *i.e.* the close contacts; and are maximized for CSI pairs that are measured from two far away mobile devices. Specifically, for CSI pairs with ground truth $Y = 1$, *i.e.* two devices are close contact with each other, the loss equals to $\mathcal{L}_c = D^2$, so the training goal becomes minimizing the Euclidean distance $D$ of feature vectors of the CSI pair. On the other hand, for CSI pairs with ground truth $Y = 0$, *i.e.* two far away devices, the loss equals to $\mathcal{L}_c = [\max(0, m - D)]^2$,

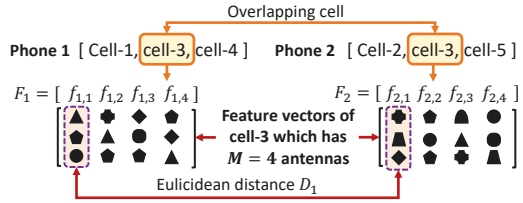



**Figure 5:** Each smartphone reports the feature vectors of a group of neighbouring cells. Two smartphones may have one or multiple overlapping cells in their reported data.

where the training goal becomes maximizing the Euclidean distance $D$. The contrastive loss also ignores the CSI pairs whose distance $D$ are already large enough.

*3.2.1 Diverse physical layer configurations.* The configurations of the cellular physical layer determine the size of CSI matrices that are fed into the contact feature extractor. To be more specific, the size of the CSI matrix is represented as $M \times N \times S$, where $M$ and $N$ is the number of antennas in array of the base station and the mobile phone, respectively, and $S$ is the number of subcarriers, which is determined by the channel bandwidth of the base station [1].

The physical layer configuration varies across base stations and mobile devices, so the size of CSI varies accordingly. We train for the combinations of one base station antenna ($M = 1$), two mobile phone antennas ($N = 2$) and all available bandwidth. We separate the CSI with $M > 1$ into $M$ CSIs with antenna number $M = 1$, so that we can reuse the single antenna model.

## 3.3 Close Contact Identifier

When a positive case is reported, the server runs the close contact identifier to find all users that are potentially exposed to the COVID-19 positive user, which involves two steps. First, the identifier reduces the search space by finding users who has overlapping cells in their reported the feature vectors, as shown in Figure 5. Second, the identifier scans all users found in step one and identifies all possible close contacts that has been exposed to the positive case.

In the rest of this section, we introduce our algorithm to identify close contacts using reported feature vectors. Since each user reports feature vectors of a group of neighbouring base stations, the number of overlapping cells between two users may vary. We, therefore, begin with the introduction of close contact identification under the scenario of one overlapping cell and then generalize to multi-cell cases.

*3.3.1 Single overlapping cell.* Supposing two smart phones share one base station that has $M$ antennas in its array, the close contact identifier first calculates the Euclidean distance $D_j$ between the feature vector of $f_{1,j}$ and $f_{2,j}$, as shown in Figure 5, and then derive the average Euclidean distance between the feature vectors $F_i$ of two users as:

$$D = \frac{1}{M}\sum_{1}^{M} D_j, \tag{2}$$

based on which, the identifier makes a preliminary identification $P_{pre}$:

$$P_{pre} = \begin{cases} -1, & \text{if } D \le D_{threh} \\ 1, & \text{if } D > D_{threh}, \end{cases} \tag{3}$$

where $D_{threh}$ is the prediction threshold. We set $D_{threh}$ to 0.9 in our experiments. A preliminary identification of $P_{pre} = -1$ means the two devices, that report the two feature vectors, are within 6 feet with each other, and vice versa.

We note that this preliminary estimation is made on a single pair of CSI, which covers only one millisecond in time. Due to the fine granularity of downlink reference signal, we have dense preliminary estimations within a short period, where the specific number of estimations depends on CSI sampling rates. To mitigate the influence of sudden environmental changes or unpredictable interference, we propose to add another voting layer on top of the preliminary estimations. The voting result $\mathcal{P}_v$ over a series of feature vector pairs is defined as follows:

$$\mathcal{P}_v = \sum_{t=1}^{k} P_{pre,t} \tag{4}$$

where $k$ is the number of feature vector pairs in a voting, $P_{pre,t}$ is the preliminary estimation on the $t$-th feature vector pair. If the resulting voting decision $\mathcal{P}_s \le 0$, the final estimation is that these two devices has a close contact in the voting time span, and vice versa.

*3.3.2 Multiple overlapping cells.* We modify the voting scheme to handle multiple overlapping cells. Supposing two mobile users share $n_c$ overlapping cells in their reported feature vectors, the voting results is given by:

$$\mathcal{P}_m = \sum_{t=1}^{k}\left(\sum_{i=1}^{n_c} w_i P_{t,i}\right), \tag{5}$$

where $P_{t,i}$ is the preliminary estimation results obtained from $t$-th feature vector of the $i$-th base station, and the weight $w_i$ is used to adjust the impact of $i$-th base station on the final voting results. Different base station have different bandwidth and transmitting antennas, so feature vectors originated from different bandwidths contain different amount information. Higher weight should be given to base station with larger bandwidth and more antennas. In our experiments, we set $w_i$ to 1 when the $i$-th base station has 20 MHz bandwidth and four antennas in its array, and then decrease

[1] A mobile device measures the CSI of the entire frequency band of the channel (§2), regardless of the detailed bandwidth allocation.

the value of $w_i$ proportional to the bandwidth and array size when the $i$-th base station adopts other configurations.

## 4 Implementation

**Mobile client side.** As a proof of concept, we implement the CRS decoding and CSI extraction parts on USRP X310 and B210 radios by modifying srsLTE [7], an open-source LTE library. We use two laptops, each connecting with two USRPs, to emulate two mobile clients. Each mobile client extracts the feature vectors from the measured CSI, tags the feature vector with timestamps and cell information, and at last uploads the feature vectors together with the tag to the server.

**Server side.** We use PyTorch [17] to train our feature vector on a server, where the CPU is Intel i7-9700, and the GPU is Geforce RTX 2060. We set the initial learning rate to 0.01, and decrease the learning rate by a factor of 0.7 every 5 epochs to stabilize the training. We set the hyperparameters $m = 2$ during the training. The server distributes the CNN based feature extractor to mobile clients, after finishing the training. To identify close contacts, the server sets the voting time span to 18s and the prediction threshold $D_{threh}$ to 0.9.

## 5 Evaluation

In this section, we evaluate the performance of LTESafe. We first introduce our evaluation methodology, and then give the end-to-end performance under diverse physical layer configurations, followed by a micro-benchmark evaluating the accuracy gains arising from each component of LTESafe.

### 5.1 Methodology

**Data collection.** We collect a diverse dataset of CSI pairs by moving the two mobile clients together with two people around each mobile client to emulate the impact of human body on wireless signal propagation. To get the ground truth of proximity between clients, we fix the distance between two mobile clients to smaller than six feet and larger than six feet.

In total we collect data at ten different locations over a period of 16 days. and the signals are from 6 nearby cell towers owned by three mobile network operators, Verizon, AT&T and T-mobile The operating frequencies range from 1805MHz up to 2355MHz. These 6 cell towers cover all antenna configurations, *i.e.*, one, two and four transmitting antennas, and three commonly used bandwidths, 5MHz, 10MHz and 20MHz. Each mobile client has two antennas in its array. In total, we collect 3,959,442 CSI pairs. We train our feature extractor using the data we collected from the first six days and then evaluate the system performance using data from the later ten days.

| Ground truth | | Close | Far |
|---|---|---|---|
| Predictions | Close | TP = 48.86% | FP = 5.85% |
| | Far | FN = 1.35% | TN = 43.93% |

**Table 1:** The confusion matrix result of LTESafe

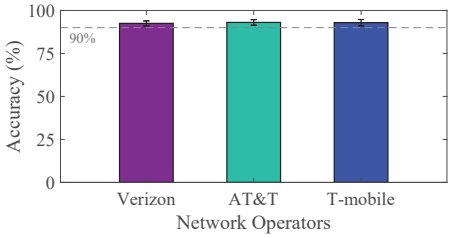
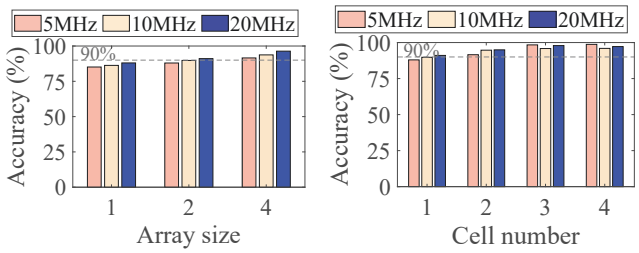



**Figure 6:** Overall accuracy of LTESafe working with data collected from different network operators.




**(a)** Varying antenna, 1 cell. **(b)** Varying cells, 2 antenna.

**Figure 7:** LTESafe's performance under diverse physical layer configurations.

### 5.2 Close Contacts Identification Accuracy

We evaluate LTESafe's accuracy in identifying close contacts. We compare the identification results with the ground truth to calculate the accuracy. Except the overall accuracy, we also calculate the true positive (TP), *i.e.*, the correct close contact discovery rate, the true negative (TN), *i.e.*, the correct far contact identification rate, the false negative (FN), *i.e.*, the close contact missing rate, and the false positive (FP), *i.e.*, the false alarm rate.

*5.2.1 End-to-end accuracy.* We run LTESafe on all testing CSI pairs we collected and calculate the accuracy. We give the confusion matrix of LTESafe's identification accuracy in Table 1, from which we see that the overall accuracy of LTESafe, is $TP + TN = 92.79\%$. In addition, in the context of Covid-19 contact tracing, missing close contacts is a much more severe problem than giving false alarms. From Table 1, we see that even though LTESafe gives false alarms to 5.85% of CSI pairs, it only misses 1.35% of close contacts in the dataset. We also provide LTESafe's identification accuracy with CSI pairs measured from cell towers of different network operators in Figure 6. We observe that LTESafe achieves a

high average accuracy working with data collected from different network operators.

*5.2.2 Impact of diverse physical layer configurations.* In this section, we evaluate the impact of diverse physical layer configurations on the end-to-end accuracy. The configurations we investigate include channel bandwidth, array size of the cell tower and the number of overlapping cells.

**Impact of bandwidth.** We plot the accuracy of close contact identification in Figure 7(a) and 7(b). We can observe from these two figures that LTESafe achieves higher accuracy when the bandwidth becomes larger and the highest achieved accuracy under one overlapping cell is 96.36% with 20 MHz bandwidth and four antennas. Even in the worst case, where two devices share one overlapping cell that has 5 MHz bandwidth and one antenna in its array, LTESafe can still achieve an accuracy of 85.14%.

**Impact of array size.** To evaluate the impact of array size of the cell tower on the accuracy, we fix the number of overlapping cells to one, and plot the accuracy of close contact identification in Figure 7(a). We see that increasing the array size significantly improve the accuracy. When the two devices are in connected mode with a configuration of 20MHz, one overlapping cell, two antennas, which is the most commonly used configuration in our collected data, LTESafe can achieve a accuracy of 91.07% on our testing dataset.

**Impact of overlapping cells.** To evaluate the impact of the number of overlapping cells on the identification accuracy, we fix the array size to two antenna, and plot the accuracy of close contact identification in Figure 7(b). We note that, since each mobile user in our implementation has only two radio chains (two USRPs due to limited available hardware), so we emulate the four cell cases by concatenating two traces. We observe higher accuracy when two devices have more overlapping cells. Specifically, the highest accuracy is 98.74% when two devices share four cells.

## 6 Related Work

Contact tracing using device proximity hides the exact user location and thus preserves user privacy, which makes it a promising solution. Diverse techniques have been proposed to estimate the proximity of mobile devices. WiFi proximity using direct signal transmissions between two device has been proposed [15], which, however, only works when the devices are in close proximity, *i.e.*, centimeters apart. Bluetooth based proximity estimation has been well studied [13, 14], and its application to contact tracing has been explored by both the research community [5, 16] and commercial companies like Google and Apple [8]. Most of these contact tracing systems estimates the proximity using Bluetooth RSSI, whose value is affected by several factors other than distance, including the hardware, interference from signals that share the 2.4 GHz ISM band and the multipath effect.

## 7 Conclusion

We propose a cellular-assisted, privacy-preserving contact tracing system for containing the spread of COVID-19. We leverage a deep learning based feature extractor to map CSI into a point in a feature space, which preserves user privacy and achieves high accuracy in identifying close contacts.

## Acknowledgments

This work is supported by an award from the Office of the Princeton University Dean for Research. This material is based upon work supported by the RAPID program of the National Science Foundation under Grant No. CNS-2027647.

## References

[1] N. Ahmed, R. A. Michelin, *et al.* A survey of covid-19 contact tracing apps. *IEEE Access*, 2020.
[2] R. Ayyalasomayajula, D. Vasisht, D. Bharadia. Bloc: Csi-based accurate localization for ble tags. *ACM CoNEXT*, 2018.
[3] J. Bromley, I. Guyon, Y. LeCun, E. Säckinger, R. Shah. Signature verification using a "siamese" time delay neural network. *NeurIPS*, 1993.
[4] CDC. Digital Contact Tracing Tools. [cdc.gov].
[5] J. Chan, S. Gollakota, *et al.* Pact: Privacy sensitive protocols and mechanisms for mobile contact tracing. *arXiv:2004.03544*, 2020.
[6] J. A. del Peral-Rosado, *et al.* Survey of cellular mobile radio localization methods: From 1g to 5g. *IEEE Communications Surveys Tutorials*, 2018.
[7] I. Gomez-Miguelez, A. Garcia-Saavedra, *et al.* srslte: an open-source platform for lte evolution and experimentation. *ACM WiNTECH*, 2016.
[8] Google, Apple. Exposure Notification Bluetooth Specification. [2020].
[9] R. Hadsell, S. Chopra, Y. LeCun. Dimensionality reduction by learning an invariant mapping. *IEEE CVPR*, 2006.
[10] K. He, X. Zhang, S. Ren, J. Sun. Deep residual learning for image recognition. *IEEE CVPR*, 2016.
[11] M. Kotaru, K. Joshi, D. Bharadia, S. Katti. Spotfi: Decimeter level localization using wifi. *ACM SIGCOMM*, 2015.
[12] P. Lazik, N. Rajagopal, *et al.* Alps: A bluetooth and ultrasound platform for mapping and localization. *ACM SenSys*, 2015.
[13] R. Momose, T. Nitta, M. Yanagisawa, N. Togawa. An accurate indoor positioning algorithm using particle filter based on the proximity of bluetooth beacons. *IEEE GCCE*, 2017.
[14] A. Montanari, *et al.* A study of bluetooth low energy performance for human proximity detection in the workplace. *IEEE PerCom*, 2017.
[15] T. J. Pierson, T. Peters, R. Peterson, D. Kotz. Proximity detection with single-antenna iot devices. *ACM MobiCom*, 2019.
[16] R. L. Rivest, J. Callas, *et al.* The pact protocol specification. *Private Automated Contact Tracing Team.*, 2020.
[17] P. Team. PyTorch. https://pytorch.org/.
[18] Y. Xie, Z. Li, M. Li. Precise power delay profiling with commodity Wi-Fi. *ACM MobiCom*, 2018.
[19] Y. Xie, J. Xiong, M. Li, *et al.* mD-Track: Leveraging multi-dimensionality for passive indoor Wi-Fi tracking. *ACM MobiCom*, 2019.
[20] Y. Xie, Y. Zhang, J. C. Liando, M. Li. SWAN: Stitched wi-fi antennas. *ACM MobiCom*, 2018.
[21] Q. Zhao, H. Wen, *et al.* On the accuracy of measured proximity of bluetooth-based contact tracing apps. *Springer SECURECOMM*, 2020.