# Proximity Inference with Wifi-Colocation during the COVID-19 Pandemic

**Mikhail Dmitrienko**
PathCheck Foundation
Cambridge, MA

**Abhishek Singh**
MIT Media Lab
Cambridge, MA

**Patrick Erichsen**
PathCheck Foundation
Cambridge, MA

## Abstract

In this work we propose using WiFi signals recorded on the phone for performing digital contact tracing. The approach works by scanning the access point information on the device and storing it for future purposes of privacy preserving digital contact tracing. We make our approach resilient to different practical scenarios by configuring a device to turn into hotspot if the access points are unavailable. This makes our proposed approach to be feasible in both dense urban areas as well as sparse rural places. We compare and discuss various shortcomings and advantages of this work over other conventional ways of doing digital contact tracing. Preliminaries results indicate the feasibility and efficacy of our approach for the task of proximity sensing which could be relevant and accurate for its relevance to contact tracing and exposure notifications.

## 1 Introduction

Privacy-preserving proximity inference is crucial to the success of a digital contact tracing solution. The majority of mainstream solutions use either Bluetooth or GPS based colocation to achieve this. However, a third option that is not as widely discussed is wifi-colocation. In this paper, we investigate a hybrid implementation of three wifi-colocation approaches described in Sapiezynski et. al, Das et al., and Carreras et. al. The hybrid implementation functions as follows: when a user is actively using WiFi, a simple if/else classifier infers proximity using features extracted from wifi scans of nearby wifi access points (APs). In the case that a user is not actively using WiFi, a duty-cycle that rotates the device between acting as a WiFi hotspot and a WiFi receiver will allow for proximity inference without interfacing with APs. We conduct an experiment to see whether three of the proximity features mentioned in Sapiezynski and Das could be used as reliable proxies for distance, and we assess the performance of a simple classifier using these features. Lastly, we assess potential privacy issues associated with wifi-based colocation.

## 2 Related Work

A number of groups have developed methods to perform localization and co-localization using WiFi signals. Ren et al [7] proposed a scheme that uses a Log-based Differential method to predict location from RSSI values. Musa et al [5] developed a passive WiFi tracking system that estimates the trajectory of vehicles based on WiFi transmission from devices inside them. Sapiezynski et al [8] trained a gradient boosting classifier to infer proximity with features extracted from wifi scans. Nakatani et al [6] used a neural network to estimate distance for indoor navigation and Wi-Fi geofencing. Das et al [3] developed an unsupervised learning algorithm to detect the formation of passively encountering groups (PECs) that outperforms the supervised approach from Sapiezynski et al [8]. Sen et al [9] proposed a group-monitoring scheme that combines Bluetooth Low Energy, WiFi,

33rd Conference on Neural Information Processing Systems (NeurIPS 2019), Vancouver, Canada.

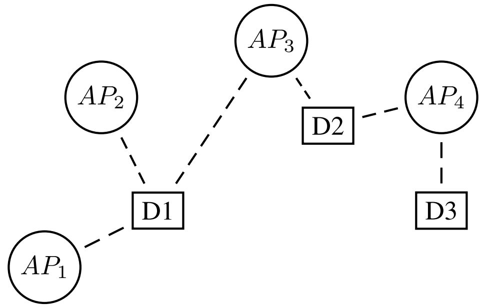


Figure 1: Devices scan for the signal strengths and MAC addresses from nearby access points. Engineered features based on this data are then used to predict proximity between users.

and other sensor data to identify groups in dense indoor environments. Carreras et al [2] proposed a novel approach that cycles devices between acting as hotspots and signal receivers to work around privacy issues associated with access-point based colocation. Trivedi et al [10] proposed a network-centric approach that uses maintenance logs to detect proximity between users connected to an enterprise network. A prototype of their approach has already been implemented on on two college campuses. In the recent cases of re-opening the universities, Harvard has also started using network centric approach for WiFi based contact tracing [1]. VContact [4] proposes access point information similar to our method and then allows sharing this information by infected individuals which can be downloaded by healthy users to query this data locally.

# 3 Methods

## 3.1 Proximity Classifier

We propose a simple if/else classifier that uses the following three features:

- Pearson correlation of signal strengths from overlapping APs between two devices
- Jaccard similarity between the lists of APs
- "Proximity feature" described in [3] which we will refer to as "Das proximity"

The classifier makes its prediction according to the following logic:

$scan_{ij} = \{Jaccard_{ij}, Pearson_{ij}, Das_{ij}\}$ represents the vector of the Jaccard Similarity, Pearson correlation, and Das proximity for scan recorded by subject $i$ at distance $j$ from access point

$avgMetrics_k = \{avgJaccard_k, avgPearson_k, avgDas_k\}$ represents the vector of the average Jaccard, Pearson, and Das at the distance threshold $k$ for proximity.

**Algorithm 1:** Proximity classifier logic

Inputs: $scans_{ij}, i, j \in I, J, avgMetrics_k$
Outputs: $predictions$
  $predictions \leftarrow \varnothing$
  **for** $scan_{ij}, \forall i, j \in I, J$ **do**
    if $scan_{ij}[0] > avgMetrics_k[0]$ **or** $scan_{ij}[1] > avgMetrics_k[1]$ **or**
    $scan_{ij}[2] > avgMetrics_k[2]$ **then**
      $predictions_{ij} \leftarrow true$
    **else**
      $predictions_{ij} \leftarrow false$
    **end if**
  **end for**

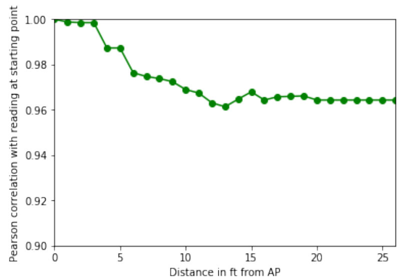


Figure 2: Average Pearson over all scans

### 3.2 Hotspot Duty Cycle

In settings where an individual is not connected to a WiFi access point, we propose an implementation of a hotspot duty cycle described in [2]. An immediate limitation of this approach is that it is only possible on Android devices - the necessary APIs are not exposed on iOS.

For individuals with an Android device, [2] describe four distinct advantages to an access point based approach to colocation:

i) does not require any additional access points to be present aside from the users device
ii) only requires information from pairwise interactions
iii) users do not need to actively connect to any access points
iv) provides accuracy in the range of 0.5m - 1.0m across a range of settings

Native APIs are available on Android to programmatically create and destroy a hotspot on a users device for the duty cycle logic. In order to easily integrate this logic into a React Native app, such as SafePaths, we developed a set of React Native bindings for the underlying Android code.[1]

With these bindings available, future work would include an implementation of this duty cycle in a npm package. This package would have the ability to detect when a user is connected to a WiFi access point, and if not, it would then begin to perform a hotspot duty cycle as described in [2]. The package could be included in a React Native application, such as SafePaths, to perform proximity inference with WiFi c-location while providing the least disruptive experience to an end user.

#### 3.2.1 Distance proxy experiment

We launched a study to see whether the features from 3.1 could be used as reliable proxies for distance. We recruited six subjects and instructed them to collect wifi sensor log data using an Android app developed by PathCheck called PrivateKit. Subjects recorded their first scan right next to their Access Point in their indoor living environment. Then, they took scans at 1 ft intervals away from the AP until they were 25ft away or as high of a distance as space permitted.

By comparing the wifi logs of the scan at each distance interval to those of the initial scan right next to the access point, we were able to capture how the three features changed with distance. These calculated features were then averaged over all the scans.

### 3.3 Results

We first assess the viability of Pearson correlation, Jaccard similarity, and Das proximity as proxies for distance based on the scans we collected from the experiment. The change in these features averaged over all the scans can be seen in figures 2-4.

Then, we evaluate the performance of the classifier using standard metrics for binary classification: recall, precision, and F-score. We also found the corresponding metrics for different distance thresholds. The change in the metrics with increasing distance thresholds is captured in figures 5-7.

[1]The code for this project can be found at `https://github.com/Patrick-Erichsen/react-native-wifi-hotspot/tree/pe-local-only-hotspot`.

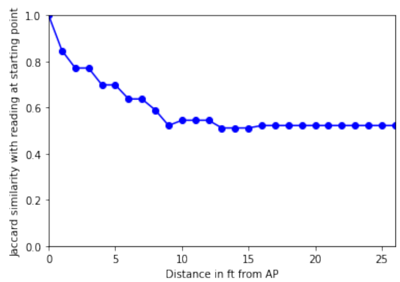


Figure 3: Average Jaccard over all scans

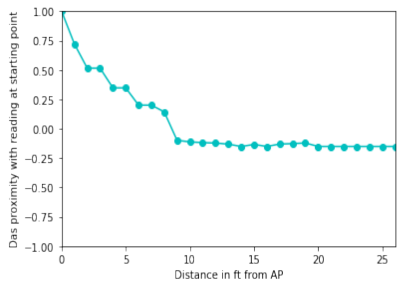


Figure 4: Average Das over all scans

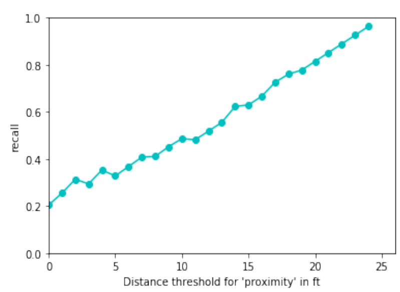


Figure 5: Change in recall with changing threshold

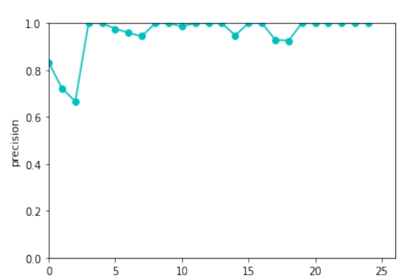


Figure 6: Change in precision with changing threshold

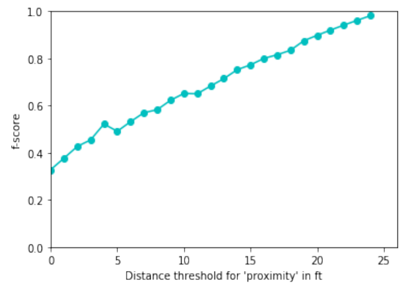


Figure 7: Change in F score with changing distance threshold

# 4 Discussion

## 4.1 Privacy Considerations

Privacy is one of the most critical piece in the digital contact tracing ecosystem. Having a high level of privacy not only alleviates a majority of the ethical concerns but it also drives up the adoption rate among the masses which is a mandatory requirement for the success of these contact tracing infrastructure. While alternate solutions like bluetooth, GPS, and ultrasound have witnessed privacy aware solutions for contact tracing in light of covid-19, WiFi has not seen anything substantial so far. There are a few important differentiating factors when it comes to the privacy of a WiFi based contact tracing solution compared to other co-location technologies. The first big difference is the lack of direct interaction between participating parties. Similar to GPS and unlike bluetooth and ultrasound, WiFi logs its own data without exchanging any information with the other participating parties. This puts a restriction on the entropy of information which can be obtained through the third party providing information, in this case wifi router. Therefore the only private and common information held by the two proximate parties is the MAC address. Each MAC address can be represented by a 48 bit and hence it is not sufficient to safeguard against brute force attacks. However, we can increase the entropy by making assumptions about the additional number of hotspots/access points available which can add to the total entropy. Nevertheless, adding extra access points only leads to minimal improvements because an attacker can create a map of access points close to each other hence reducing the overall entropy of the available data. The attack can be further enhanced by a dictionary mapping available through websites such as wigle.net.

# 5 Conclusion

In this proposal we outlined a two pronged approach to performing wifi-colocation during COVID-19. We demonstrated how a simple if/else classifier could be used in a limited capacity to infer proximity using features extracted from wifi scan data, as well as how a hotspot duty cycle could be implemented in a consumer application. Finally, we outlined some potential privacy concerns including low information entropy and limited safeguards against brute force attacks.

We believe future efforts would best be directed towards conducting large scale experiments with wifi scan data as well as further analyzing the privacy pitfalls inherent in wifi-based co-location implementations.

## References


[1] Harvard to Track Affiliates Wi-Fi Signals as Part of Contact Tracing Pilot. `https://www.thecrimson.com/article/2020/8/2/tracefi-wifi-contract-tracing-coronavirus/`, 2020.

[2] I. Carreras, A. Matic, P. Saar, and V. Osmani. Comm2sense: Detecting proximity through smartphones. In *2012 IEEE International Conference on Pervasive Computing and Communications Workshops*, pages 253–258, 2012.

[3] S. Das, S. Chatterjee, S. Chakraborty, and B. Mitra. An unsupervised model for detecting passively encountering groups from wifi signals. In *2018 IEEE Global Communications Conference (GLOBECOM)*, pages 1–7, 2018.

[4] Guanyao Li, Siyan Hu, Shuhan Zhong, and S-H Gary Chan. vcontact: Private wifi-based contact tracing with virus lifespan. *arXiv preprint arXiv:2009.05944*, 2020.

[5] A. B. M. Musa and Jakob Eriksson. Tracking unmodified smartphones using wi-fi monitors. In *Proceedings of the 10th ACM Conference on Embedded Network Sensor Systems*, SenSys '12, page 281294, New York, NY, USA, 2012. Association for Computing Machinery.

[6] Tomoya Nakatani, Takuya Maekawa, Masumi Shirakawa, and Takahiro Hara. Estimating the physical distance between two locations with wi-fi received signal strength information using obstacle-aware approach. 2(3), 2018.

[7] Yongli Ren, Flora Salim, Martin Tomko, Brian Bai, Jeffrey Chan, Kyle Kai Qin, and Mark Sanderson. D-log: A wifi log-based differential scheme for enhanced indoor localization with single rssi source and infrequent sampling rate. *Pervasive and Mobile Computing*, 09 2016.

[8] Piotr Sapiezynski, Arkadiusz Stopczynski, David Kofoed Wind, Jure Leskovec, and Sune Lehmann. Inferring person-to-person proximity using wifi signals, 2016.

[9] Rijurekha Sen, Youngki Lee, Kasthuri Jayarajah, Archan Misra, and Rajesh Krishna Balan. Grumon: Fast and accurate group monitoring for heterogeneous urban spaces. In *Proceedings of the 12th ACM Conference on Embedded Network Sensor Systems*, SenSys '14, page 4660, New York, NY, USA, 2014. Association for Computing Machinery.

[10] Amee Trivedi, Camellia Zakaria, Rajesh Balan, and Prashant Shenoy. Wifitrace: Network-based contact tracing for infectious diseases using passive wifi sensing, 2020.