A CrowdSensing-Based Approach for Proximity Detection in Indoor Museums with Bluetooth Tags

Michele Girolami[1a], Davide La Rosa[a], Paolo Barsocchi[a]

[a] *Institute of Information Science and Technologies, National Research Council (ISTI-CNR), Pisa, Italy*
{*name.surname*}*@isti.cnr.it*

Abstract

In this work, we investigate the performance of a proximity detection system for visitors in an indoor museum exploiting data collected from the crowd. More specifically, we propose a CrowdSensing-based technique for proximity detection. Users' smartphones can collect and upload RSS (Received Signal Strength) values of nearby Bluetooth tags to a backend server, together with some context-information. In turn, the collected data are elaborated with the goal of calibrating two proximity detection algorithms: a range-based and a learning-based algorithm. We embed the algorithms with R-app, a visiting museum application tested in the Monumental Cemetery's museum located in Piazza dei Miracoli, Pisa (IT). We detail in this work an experimental campaign to measure the performance improvements of the CrowdSensing approach with respect to state-of-the-art algorithms widely adopted in the field of proximity detection. Experimental results show a clear improvement of the performance when data from the crowd are exploited with the proposed architecture.

Keywords: CrowdSensing, Bluetooth, Indoor Localization, Proximity, Cultural Heritage

1. Introduction

The last ten years have been showing an increasing attention to location-based services (LBS), exploiting the current user's location to deliver an enriched user experience. This is the case of many popular mobile applications exploiting the user's position to deliver location-aware services. The performance of such systems is rapidly increasing also when considering indoor environments, thanks also to the combined use of heterogeneous sensing units and learning-based algorithms [1, 2, 3].

[1]Michele Girolami is the corresponding author, National Council of Research ISTI-CNR Italy, michele.girolami@isti.cnr.it

Nevertheless, an increasing number of scenarios do not require accurate knowledge of the user's position, rather it is sufficient determining the *proximity* of a person with respect to a point of interest (POI). This is the case of contact tracing applications which are designed to track proximity between people [4]. Many technologies have been successfully tested in this context. Among them, we refer to the family of RF (Radio-Frequency) technologies such as Wi-Fi, UltraWide Band (UWB), Infrared (IR) and Bluetooth. The latter, in particular, has demonstrated successful in many contexts. Two main reasons drive the diffusion of Bluetooth for proximity detection. On the one hand, Bluetooth is widely diffused with commercial devices, e.g. smartphones, wristbands and smart watches. On the other hand, the Bluetooth Low Energy specification and the recent Bluetooth 5.x specification, reduce the energy consumption of such communication protocol. However, the accuracy of proximity detection algorithms based on Bluetooth is still affected by environmental and hardware constraints as studied in [5].

In this work, we investigate a promising approach to improve the accuracy of proximity detection based on Bluetooth. We propose a CrowdSensing-based technique according to which the crowd can collect data through commercial devices useful to (re)calibrate an existing algorithm. To the best of our knowledge, this approach has been never adopted to detect proximity between people and POI. We first propose a software architecture according to which user's devices can collect RSS values of received Bluetooth tags, together with some context-information. Such data are required to determine the Ground Truth. In turn, the collected data can be uploaded to a backend server, where data are processed to calibrate two state-of-the-art algorithms: a threshold-based and a learning-based algorithm. It is worth to notice that our goal is showing how a CrowdSensing approach can improve the performance of state-of-the-art algorithms, commonly adopted to solve the problem described in this work. We test the proposed approach in a challenging environment: the Monumental Cemetery located in the famous Piazza dei Miracoli of Pisa (IT). To this end, we design and implement a mobile application, named R-app, built to automatically detect proximity of visitors with respect some artworks. The application offers to visitors some multi-media contents, such as artworks' description, an image gallery and info-graphics. During our tests, we monitor a number of artworks and we execute several museum's visits with commercial smartphones. In the paper we detail the followed methodology to organize a data collection campaign, whose goal is the performance assessment of the two algorithms with and without the use of the CrowdSensing approach. All the collected data are available to the community as a public repository [6].

Our experimental tests include three settings of increasing complexity: i) we vary the adopted smartphone, ii) we modify the visiting order of the monitored artworks and, iii) we combine the two previous settings together. From our results, it is clearly possible to observe the performance improvement of the algorithms when data collected from the crowd are exploited, with a net improvement of the Accuracy metric up to 30% with respect to the state-of-the-art. In summary, the novel contributions of this paper are the following:

- we propose a proximity detection architecture based on the CrowdSensing approach;
- we compare the performance of two state-of-the-art algorithms with and without the CrowdSensing approach;
- we test our solution in a realistic indoor museum. Moreover, We conduct a data collection campaign with commercial Bluetooth tags and smartphones;
- we release the collected dataset to the community for further experimental testing, [6].

The remainder of this work is organized as follows: Section 2 describes the state-of-the-art of proximity solutions based on the Bluetooth technology with a summarizing table. Such section also provides the background information for proximity detection based on RSS analysis. The CrowdSensing-based architecture is described in Section 3 as well as the proposed algorithms. Section 4.3 describes the data collection campaigns and the architectural information of the indoor museum. Results are finally discussed in Section 5 in which we compare the proposed solution with a baseline proximity algorithm. The paper concludes with a discussion about new research lines.

2. Background and Related Work

2.1. Proximity Detection in Indoor Museums

In this section, we report the most relevant works addressing the proximity detection problem. We restrict the analysis to museum's environments and to the Bluetooth technology, so that to highlight the novel contribution of our work with respect to current literature.

The BLE (Bluetooth Low Energy) technology implements features suitable for proximity detection with commercial devices [7, 8]. Under this respect, it is important to mention the the Google's Exposure Notification APIs [2]. Such framework allows detecting devices in proximity by exploiting RSS values, modified with an *attenuation* factor. Such factor is designed to compensate the differences of RSS values between the transmitting and the receiving devices. Although the attenuation factors offer cross-compatibility between Android and iOS devices, such APIs rely on a static list of attenuation factors estimated with a number of devices [3]. Differently from the static list proposed with the Exposure Notification APIs, our approach also includes a reference architecture to collect, store and elaborate data provided by the crowd.

[2] https://developers.google.com/android/exposure-notifications/ble-attenuation-overview?hl=en

[3] The CSV list of attenuation factors, last version available at the time of writing this paper is Dec. 2020: https://developers.google.com/static/android/exposure-notifications/files/en-calibration-2020-08-12.csv

In [9], the BLE beacons are used to develop an efficient localization system inside the Expo Museum at Postojna, Slovenia. The improvements are implemented by storing the signal measurement's history to mitigate the RSS fluctuations. Such work proposes a method to estimate the nearest transmitter, by analyzing the strongest RSS value. However, due to the significant RSS fluctuations, false positives and negatives may occur. In order to overcome such inaccuracies, authors propose storing the RSS signals collected in a dynamically set interval from every transmitter in a manually-fixed-length array, where the average RSS value is calculated. The choice of the strongest signal is made by comparing all averages of all detected transmitters in a specific area. Moreover, the user's position is estimated by also fusing data obtained from inertial sensors, such as the accelerometer, and by adopting a threshold-based mechanism.

Authors of [8] analyze the Bluetooth signals with fingerprinting and machine-learning approaches in order to improve the accuracy in a mixed indoor-outdoor environment. The paper describes a new BLE RSS database composed of raw measurements from different smartphones.

In [10], authors propose an approach based on neural networks, namely MLP NN - Multilayer Perceptron Neural Network. This approach allows authors to re-build with high accuracy the visitor's paths. In [11], authors assume that visitors bring a BLE devices during the museum's visits. The device emit beacons collected by BLE receivers with a known position. In turn, the collected beacons are forwarded to a server computing the visitors' location. The location is estimated running a pre-trained neural network.

The "Ghosts!" project [12] is focused on the development of a game application based on BLE technology for the Cambridge museums. To overcome RSS fluctuations, authors exploit a map reporting the positions of BLE tags deployed in the museum. Through such map, visitors are guided to reach their destination.

In Trowulan Museum [13](Indonesia) visitor's smartphones are used to collect information extracted from the Bluetooth tags. In particular, smartphones upload the tag's identifier and the coordinates of the tag. In turn, such information are elaborated by a backend server and the visitor's position is estimated with a trilateration-based technique. Authors of [14] study the visitor's behavior in the Louvre Museum. More specifically, museum's rooms are equipped with Bluetooth tags to track the time spent by visitors in rooms. Through this process, it is possible to construct the visiting path of visitors. Note that the deployed tags are not linked to artworks, rather they are deployed in the environment with the objective of reconstructing the followed paths of visitors.

In [15], authors implement a localization system in MUST museum, Lecce (IT). Authors exploit user's wearable devices to collect RSS values from BLE devices deployed in the museum. The localization system is deployed on board of wearable devices and it estimates the user's location with a path-loss model. The adopted model, combined with an image-based technique, allows authors to detect the proximity between visitors and artworks. The same path-loss model has been also adopted in [16]. In this case, authors use such model to compute the distance between tags and visitors' smartphones. In turn, the obtained

Experimental Scenario	Adopted Devices	Crowd exploitation	Algorithms	Reference
Postojna museum Slovenia	Samsung S7	✗	Strongest RSS signal	[9]
Uni. of Extremadura campus, Badajoz	Honor 8	✗	Fingerprinting and ML-based techniques	[8]
Galleria Borghese, Rome, Italy	iBeancon (Tx) RaspberryPi (Rx)	✗	Neural Network	[10]
Unspecified museum	BlueUp mini(Tx) ESP32-WROOM-32D and ESP32-WROOM-32U(RX)	✗	Trilateration	[11]
Uni. of Cambridge Museums	Visitor's smartphone, BLE beacons	✗	Strongest RSS signal	[12]
Unspecified museums	Visitor's smartphone	✗	Trilateration	[13]
Louvre Museum	Visitor smartphone, BLE sensors	✗	"If I see you, here you are"	[14]
MUST museum, Lecce, Italy	Wearable device(Rx), Raspberry PI (TX)	✗	Strongest RSS signal and image-based recognition	[15]
Unspecified museum	Visitor's smartphone, (iBeacon protocol)	✗	Trilateration and Kalman filter	[16]
Portsmouth City Museum	Visitor's smartphone, EddyStone-URL	✗	path-loss model	[17]
Unspecified museum	Raspberry PI	✗	Strongest RSS signal and image-based recognition	[18]
Monumental Cemetery's museum, Pisa, IT	Visitor's smartphone, BLE beacons	✓	Neural Network, Strongest RSS signal and Threshold	our solution

Table 1: Overview of Bluetooth-based Proximity Detection solutions for indoor museums.

distances are combined with a trilateration technique to estimate the visitors' positions.

Similarly to the two previous works, also in [17], authors adopt a path-loss model. This work describes how the transmitting power of tags affects the proposed method. In [18], authors adopt a Raspberry PI unit to localize museum's visitors. The unit collects and analyses beacons emitted by tags deployed nearby the artworks. The implemented localization algorithm relies on RSS values and in particular, authors adopt an indoor propagation model to correlate RSS with distance from the artworks.

We report in Table 1 a survey of works referring to our scenario and based on the Bluetooth technology. Works have been compared across four criteria: (i) Experimental Scenario: the scenario considered for the tests; (ii) Adopted Device: the hardware adopted for the experiments; (iii) Crowd Exploitation: if the surveyed work considers exploiting data from the crowd; (iv) Adopted Algorithms: a short description of the adopted algorithm. As reported in the table and to the best of our knowledge, it emerges that none of the selected works exploits the crowd for proximity detection. Differently, this paper moves towards the idea of collecting and re-using retrievable data from end-users, to improve the performance of proximity detection algorithms tested in an indoor museum.

2.2. Proximity Detection based on RSS Analysis

With the term proximity we refer to the ability of estimating the *closeness* of a subject with respect to a POI, e.g. museum's artworks. In our scenario, POIs are equipped with an emitting device while the visitor brings the receiving device. i.e. a smartphone. We adopt the Bluetooth wireless protocol and, more specifically, the Bluetooth Low Energy (BLE) specification, widely supported by most commercial devices. The BLE specification offers the possibility for a device to emit beacons, small packets containing broadcasted information. Generally, beacons adhere to one of the three main payload formats: iBeacon, AltBeacon and EddyStone. We adopt the iBeacon format as it guarantees compatibility with Android and iOS devices.

Some recent works exploit RSS values to estimate distance and to determine the proximity. Authors of [19] introduces a crowd-sourcing localization system combining Wi-Fi and Bluetooth tags. In particular, the mobile device not only sends Wi-Fi fingerprint data to a map server, but it also activates Bluetooth tags to share its location and to determine the fingerprint information. Such approach allows authors to rapidly build the signal map. As a result, information from Bluetooth tags is used to acquire room-level location information without the need for unnecessary user prompts.

Authors of [20] address the distance estimation problem with a classification approach known as the d-Classifier. The classifier combines the capabilities of the Kalman filter (KM) and it supports vector machine (SVM) for precise distance categorization between beacons and mobile devices. Authors state that while KF and SVM are commonly employed in proximity and distance-related applications, the primary innovation of d-Classifier lies in the creation of feature vectors that take into account diverse hardware types, deployment locations, and beacon configurations, ultimately leading to enhanced accuracy. The experimental findings underscore the unreliability of RSS as a standalone metric for distance estimation, even after undergoing a filtering process, given the varied internal and external configurations in which tags are deployed. Consequently, the d-Classifier advocates the inclusion of additional information beyond RSS for more accurate estimations. Authors validate the d-Classifier using a substantial dataset encompassing over 200.000 samples, featuring RSS measurements acquired from diverse hardware setups and configurations with 25% error reduction over existing methods.

In [21] authors present a system that harnesses the willingness of smartphone's users to detect Bluetooth devices. Such CrowdSensing approach enables the analysis of crowd dynamics in urban settings. The proposed solutions extends a simple device counting by incorporating a range of sophisticated and resilient features. The adopted methods have been evaluated using a substantial dataset comprising nearly 200,000 discoveries collected from nearly 1,000 scanning devices over a three-day city-wide festival held in Zurich. Authors validate the proposed solution with a GPS-based dataset from almost 30,000 users, serving as ground truth reference.

In [22] authors study how to proximity between people with an RSS-based distance estimation. The addressed scenario is referred to as Too Close For Too

Long (TC4TL). The paper presents a systematic investigation into the application of Machine Learning techniques, utilizing available datasets. Authors extract a set of 20 statistical features from accelerometer and gyroscope sensor signals, as well as some statistical features from Bluetooth signals. These features are used for classification tasks to determine whether individuals are within a six-foot proximity and to infer the context of the subjects. Among the 19 classification and regression methods explored by authors, ensemble methods yield the best performance when applied to accelerometer and gyroscope data. Results reported in [22] show that proximity can be classified with an accuracy ranging from 72% to 90% when utilizing accelerometer data, 78% to 84% with gyroscope sensor data, and a remarkable 76% to 92% accuracy when using Bluetooth data.

2.3. RSS Variability

All the aforementioned works exploit the RSS value estimated by a receiving device. RSS measures the strength of the receiving signal and it is generally expressed in decibel dBm unit. According to the Bluetooth chipset installed on a receiving device, the RSS value can vary significantly. In the following, we will refer to *device heterogeneity* to denote RSS variations due to the different hardware adopted to estimate RSS values. As firstly mentioned in Section 2.1, the Google's Exposure Notification implements a static-based approach to mitigate the device heterogeneity. In particular, such APIs allow to add an attenuation factor to the RSS values collected by a device.

As a general consideration, the closer the emitter to a receiver, the stronger the signal and the higher the RSS value estimated by a receiver. Such relationship has been widely investigated in the last years, giving rise to a number of propagation models for indoor and outdoor scenarios [23, 24]. The path loss model represents a reference model for wireless propagation, as also discussed in [25]. According to such model, the relationship between RSS and distance is given by:

$$RSS = RSS_0 - 10n \log_{10}(d/d_0), \quad d > d_0 \tag{1}$$

where d_0 is the reference distance, such that the emitter and the receiver are always in line of sight (typically 1 m distance), RSS_0 is the RSS at a reference distance d_0, and n is the path loss exponent that regulates how severe is the attenuation in a given environment. Nevertheless, in indoor environments, a number of factors affect signal propagation such as obstacles, the presence of people acting as barriers, and existing wireless interference on 2.4 GHz band. As a result, determining the distance of a Bluetooth emitter with respect to the receiver still represents a challenging task.

RSS variability is generated by several reasons. In our previous studies, we investigated some of these aspects, such as the effect of body attenuation [26], the impact of environmental obstacles [5], but also the influence of the adopted Bluetooth channel for advertising beacons, as also studied in [27]. Under this respect, it is worth noticing that beacons are propagated on three channels: 2.402 GHz (channel 37), 2.426 GHz (channel 38) and 2.480 GHz (channel 39) each of

which has a slightly different effect to the estimated RSS value. However, the Android OS does not allow filtering beacons based on the Bluetooth channel, causing a significant RSS variability [28, 29]. In order to show the RSS variability, we conduct a preliminary test in which we deploy five Bluetooth tags at 1.5 meters (-23 dBm power of emission and 2 Hz advertisement frequency) distance from the receiving device (Google Pixel 4a). The test involves ten minutes of data collection, and Figure 1 shows the results. From the figure, it is clear that even if Bluetooth tags are at the same distance (same model, same distance, same receiving device), RSS' distributions remarkably differ.

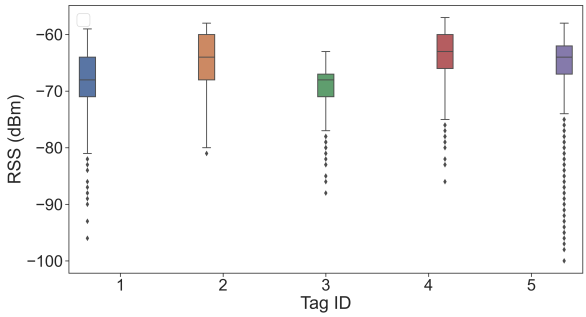


Figure 1: RSS variability of five tags deployed at 1.5m distance from the receiving device at stationary conditions.

We replicate the test reported in Figure 1 by using two receiving devices: Google Pixel 4a and Honor 9. In this case, the goal is to show how different receiving devices estimate beacons broadcasted by five tags at 1.5 distance for ten minutes. Figure 2 compares the two distributions: on the left side we report the Pixel's distribution, while on the right side the Honor 9's distribution. Also in this case, it is clear the impact of device heterogeneity to the RSS distributions. The previously mentioned considerations make proximity detection a complex task at realistic conditions.

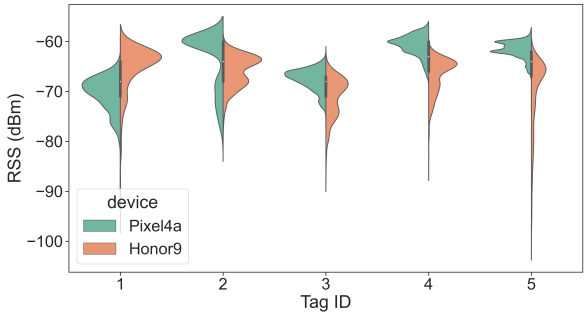


Figure 2: Effect of device heterogeneity of five tags deployed at 1.5m from the two receiving devices at stationary conditions.

3. Proximity Detection with a CrowdSensing Architecture

3.1. The Reference Architecture

In this section, we first detail our reference scenario and then we detail how we exploit data collected from the crowd to increase the performance of two state-of-the-art algorithms.

We refer to an indoor museum in which visitors are free to move and to visit a number of POIs, i.e. artworks, information points, galleries etc. All visitors carry their smartphones, provisioned with a Bluetooth network interface. The existing POIs are equipped with Bluetooth tags advertising beacons at regular intervals. Our objective is enabling users' smartphones to automatically detect the proximity with existing POIs to offer them an enriching user-experience. As for example, an application installed on the smartphone can provide artwork's details as soon as a visitor moves close to the corresponding POI. We also assume that smartphones have wireless connectivity for two purposes. On the one hand, smartphones download some *configuration settings*, useful to (re)configure the proximity detection algorithms. On the other hand, smartphones collect and upload sensing information, that will be used to improve the accuracy of the algorithms. We refer to $[R, P, GT]$ as the data that visitors' smartphones can upload to the backend:

- R: RSS values of the beacons advertised by POIs. This information can be easily collected through APIs provided by Android and iOS operating systems. In particular, such APIs offer the possibility of logging RSS values of Bluetooth tags (expressed in dbm unit);
- P: the POI's identifiers. This information generally corresponds to the MAC addresses of the Bluetooth tags associated with a POI. Also in this case, APIs generally provide information about the MAC address of the Bluetooth emitting devices, as well as the UUID (Universally unique identifier);
- GT: the POI identifier that a visitor is in proximity with. This information is also referred to as *Ground Truth (GT)*. We discuss in Section 4.1 a possible approach to infer the visitor's GT with the use of a mobile application.

The previously mentioned scenario might correspond to a very common situation in which visitors first download the museum's application, and then they use it to get access to contents provided by the museum. We start from this experience to design a CrowdSensing-based architecture exploiting the data collected by each of the smartphone: $[R, P, GT]$. The underlying idea is to design a system able to iterate through three steps, as reported in Figure 3:

1. Setting Provider: providing the algorithm's settings $[s]$ to visitor's smartphones to calibrate the algorithms;
2. Data Collector: storing data provided by visitors: $[R, P, GT]$;

3. Re-calibration: updating settings $[s]$ with data provided by smartphones and elaborated by the backend system.

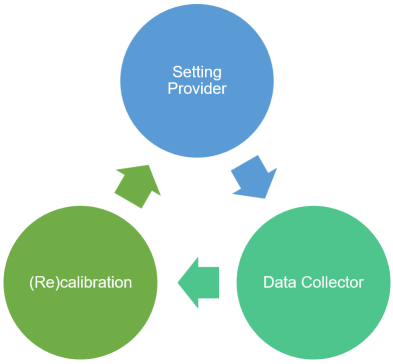


Figure 3: Life-cycle of the proposed CrowdSensing architecture for proximity detection.

The reference architecture we propose is composed by a backend server in charge of providing settings to new visitor's smartphones through wireless connectivity. The configuration settings allow the proximity detection algorithms to detect nearby artworks and to access the relative contents. The backend also acts as data collector. In particular, the newly received data $[R, P, GT]_i$ from visitor i are processed to refine the configuration settings $[s]$. Figure 4 shows our reference CrowdSensing architecture.

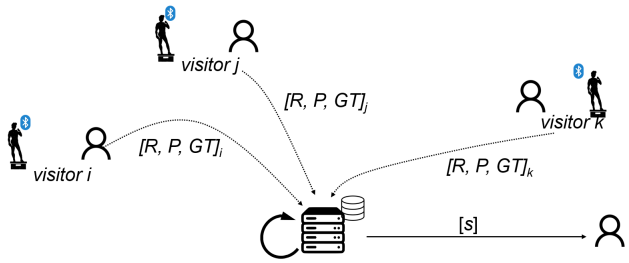


Figure 4: The reference CrowdSensing proximity architecture.

3.2. Proximity Detection Algorithms

We now detail the algorithms that we test to evaluate the proposed architecture based on a CrowdSensing approach. For each of the algorithms, we describe two versions: a static version and a crowd-based version.

Threshold-based Algorithm: the first algorithm estimates the proximity from a set of POIs by comparing RSS values with respect to a reference threshold τ. In the rest of this paper, we refer to this algorithm as `Threshold`. This approach is relatively simple and widely adopted in literature [14, 17]. The

static version of `Threshold` considers one only threshold τ for all the POIs in the environment. More formally, the algorithm performs two operations: filtering the sampled RSS values and comparing them against τ. The adopted filtering technique is based on a moving window that keeps into account only the most recently received RSS. Filtering RSS values allows us to stabilize the RSS fluctuations, discussed in Section 2.2. In particular, we adopt the p-percentile filter, e.g. a median or a percentile of the RSS values, which is often adopted in computer graphics to remove outliers without degrading the resulting image (as proposed in [30]). For the purpose of filtering RSS samples, we use such a filter to minimize the variance of RSS values, to obtain stables values during the visiting path. More specifically, given a set of collected beacons from POI_j: $B_j = \{b_1, b_2, ..., b_i\}_j$ and the corresponding sequence of noisy RSS values $RSS_j = \{rss_1, rss_2, ..., rss_i\}_j$, the p-percentile filter is given as:

$$RSS_j = P(RSS_j) \quad \forall POI_j \tag{2}$$

where $P(RSS_j)$ corresponds to the p-percentile of RSS_j. After the filtering process, `Threshold` identifies the nearest POI by comparing RSS values of POI_j with respect to a *single* reference threshold τ as follows: $RSS_j \geq \tau$. This comparison is performed for all the detected POIs, namely POI_i, POI_j, ..., POI_k. The algorithm ranks the list of nearby POIs according to RSS values in an increasing order, e.g. $[POI_i, POI_j, POI_k]$.

The **crowd** version of `Threshold` implements a multi-threshold mechanism. The underlying idea is still comparing the filtered RSS values with respect to a reference threshold, but differently from the previous version, now the algorithm adopts k distinct thresholds $[\tau_i, \tau_j, \cdots \tau_k]$, one for each POI. The set of thresholds is obtained exploiting data collected by the crowd. Indeed, we can exploit the CrowdSensing architecture previously described to assume that smartphones upload $[R, P, GT]$ data containing respectively:

- R: the list of RSS values;
- P: the POI identifier for every RSS value in R;
- GT: the POI to which the visitor is in proximity with, namely the Ground Truth.

In order to clarify this aspect, let's suppose three visitors visit the museum according to the pattern reported in Table 2. Each visitor contributes to collect $[R, P, GT]$ data, from which it is possible to extract thresholds for POI_i, POI_j and POI_k . Let's suppose visitor a provides the following matrix:

$$\begin{bmatrix} R & P & GT \\ -76 & i & i \\ -77 & k & i \\ -75 & k & i \\ -77 & i & i \\ \vdots & \vdots & \vdots \end{bmatrix} \tag{3}$$

Table 2: `Threshold` Algorithm: Pattern of visits of three visitors with three POIs and the corresponding elaborated data.

	Visited POI	User-Collected data
Visitor a	POI_i	$\begin{bmatrix} R & P & GT \\ -76 & i & i \\ -77 & k & i \\ \vdots & \vdots & \vdots \end{bmatrix}$
Visitor b	POI_j	$\begin{bmatrix} R & P & GT \\ -44 & i & j \\ -48 & j & j \\ \vdots & \vdots & \vdots \end{bmatrix}$
Visitor c	POI_k	$\begin{bmatrix} R & P & GT \\ -33 & i & k \\ -30 & j & k \\ \vdots & \vdots & \vdots \end{bmatrix}$
Crowd-Elaborated data		$[\tau_i, \tau_j, \tau_k]$

Table 3: RSS features of `DeepProximity` Algorithm

Metric	Description
Average	average RSS value of n samples: $\frac{1}{n}\sum_{i=i}^{n} x_i$
Std. deviation	standard deviation of the n values: $\sqrt{\frac{1}{n}\sum_{i=i}^{n}(x_i - \bar{x})^2}$
Maximum	maximum observed RSS value;
Minimum	minimum observed RSS value;
Max-Min	difference between the maximum and minimum RSS observed values;
Percentile	$75th$ percentile of the RSS values;
Skewness	$\frac{1}{n}\sum_{i=1}^{n}\frac{(x_i-\bar{x})^3}{\sigma^3}$;
Kurtosis	$\frac{1}{n}\sum_{i=1}^{n}\frac{(x_i-\bar{x})^4}{\sigma^4}$;
Inter-quantile	inter-quantile range of the RSS values obtained as the difference between the $75th$ and $25th$ percentile of the RSS values, offering a much more robust metric against outliers.

then the CrowdSensing architecture can compute the threshold for POI_i by selecting from the matrix only RSS values with GT $= POI_i, [-76, -77, -75, \cdots]$. Given the set of RSS values, the architecture can now compute the POI_i's threshold as the p-percentile $\tau_i = P[-76, -77, ...] = -76.25$ dBm. Similar computations can be executed with POI_j, POI_k of visitors b, c. As a result, a multi-threshold list is obtained: $[\tau_i, \tau_j, \tau_k]$.

In summary, we modify the `Threshold` algorithm with a crowd-based version which compares the filtered RSS values of POI_j with respect to a specific threshold τ_j automatically estimated by the crowd.

DeepProximity Algorithm The second algorithm we design is a learning-based algorithm, referred to as `DeepProximity`. Its design follows a feed-forward fully connected network to detect proximity between visitors and artworks. The implemented learning model extracts a set of RSS features from collected Bluetooth beacons during a time window T_k of k seconds. The extracted features are reported in Table 3.

In particular, STD measures the dispersion of RSS values, SKW measures the symmetry of the distribution, while KURT measures if RSS values are heavy-tailed or light-tailed with respect to normal distribution. RSS values are re-

ported as dBm unit, we apply a normalization process in the range $[0, 1]$. The normalization process is important as it allows the neural network to quickly converge and to avoid triggering large gradient updates that sparse and double digit numbers. The structure of the implemented neural network consists of three Dense layers separated by two inner Dropout layers, to avoid overfitting, and an L2 regularizer applied with a coefficient of 0.001. The first two Dense layers have $ReLU$ as activation function and they have 1024 neurons each. The last Dense layer implements a *Softmax* activation function, with ten neurons in order to provide, for each of the artwork, the probability of the user's proximity to it. As done with the Threshold algorithm, we implement two versions of DeepProximity: a static and a crowd version as detailed in the next. The **static** version of `DeepProximity` follows the well-known learning pipeline, according to which the model is trained once with the training set, it is validated with the validation set, and lastly it is tested with the test set. The training set is used only once, and it is never updated. Differently, with the **crowd** version of this algorithm, we propose to exploit data collected by visitors to continuously re-train the model as soon as new data are uploaded. A graphical representation of the neural network is reported in Figure 5. Visitors upload $[R, P, GT]$ and whenever new data is received, the system extracts the nine features shown in Table 3 from the raw RSS values. In turn, such features can be feeded the neural network to estimate, for each of the artwork, the probability of being in its proximity. After this step, the original data received from the visitor, together with the GT information, is added to the training data and a new version of the neural network model is generated in order to be used with future visitors. This approach can improve the performance of the neural network, as over time, the training dataset keeps growing. We report in Table 4 an example with three visitors and the corresponding user-collected and crowd-elaborated data used to re-calibrate the neural network.

Table 4: `DeepProximity` Algorithm: Pattern of visits of three visitors with three POIs and the corresponding elaborated data.

	Visited POI	User-Collected data
Visitor a	POI_i	$\begin{bmatrix} R & P & GT \\ -50 & i & i \\ -51 & j & i \\ \vdots & \vdots & \vdots \end{bmatrix}$
Visitor b	POI_j	$\begin{bmatrix} R & P & GT \\ -44 & i & j \\ -45 & j & j \\ \vdots & \vdots & \vdots \end{bmatrix}$
Visitor c	POI_k	$\begin{bmatrix} R & P & GT \\ -33 & i & k \\ -34 & j & k \\ \vdots & \vdots & \vdots \end{bmatrix}$
Crowd-Elaborated data		$\begin{bmatrix} RSS & ID \\ -50 & i \\ -51 & j \\ -44 & i \\ -45 & j \\ -33 & i \\ -34 & i \\ \vdots & \vdots \end{bmatrix} \begin{bmatrix} GT \\ i \\ i \\ j \\ j \\ k \\ k \\ \vdots \end{bmatrix}$

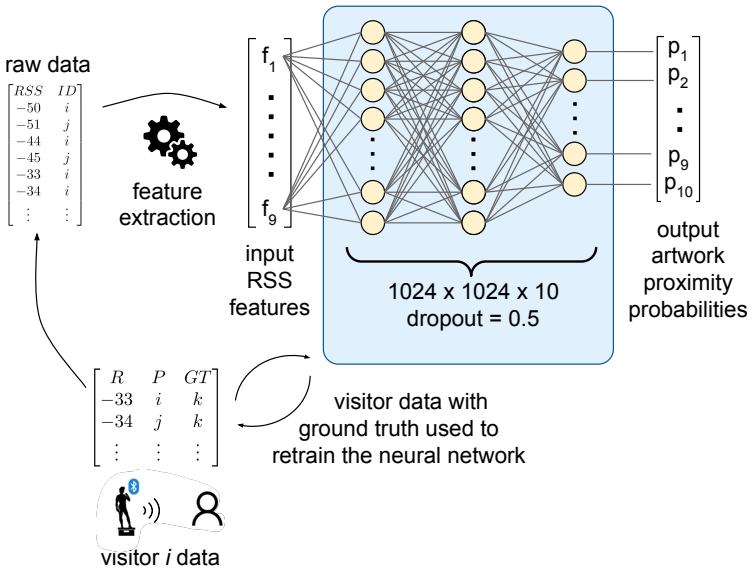


Figure 5: `DeepProximity` algorithm model with the implementation of the crowd version.

4. The Case Study of Monumental Cemetery of Pisa

Experiments has been conducted in the context of the RE.S.I.STO project [5]. The project targets to visitors of Pisa medieval city, with the goal of providing high-quality digital contents accessible via smart devices. The goal of RE.S.I.STO is twofold. On the one hand, the project aims at recovering a database with historical contents related to several artworks of Pisa medieval city. The recovery process required refreshing contents from an obsolete digital support to maintainable technological framework. On the other hand, the project aims at exploring the potentialities of market-ready technologies to improve the user-experience of museum visitors. In this respect, we design and test the RE.S.I.STO mobile application, R-app (as presented in [5]), which offers to visitors a digital guide based on the recovered contents such as: artwork's description, details, multi-media contents. The application implements several proximity detection algorithms and it provides to visitors the appropriate content as soon as they move in proximity to a specific artwork. Section 4.1 details the design principles of the application. The case study is located in Monumental Cemetery's museum of Piazza dei Miracoli Pisa (IT), details are reported in Section 4.2, while in Section 4.3, we describe the different data collection campaign.

4.1. The Design of R-app for Museum Visits

R-app is a multi-platform application based on the React Native Framework. The application implements three core operations:

- Logging beacon's information, such as the corresponding RSS values, beacon's identifier (Bluetooth MAC address);
- running the proximity detection algorithms to determine the nearest artworks;
- showing the artwork's contents to visitors.

The three operations are implemented with three software modules: Beacon Logging Service (BL), Proximity Detection Service (PD) and UI Content Viewer Service (UI). BL is designed to collect only beacons emitted by Bluetooth tags included in a *white-list*. This filter allows us to avoid collecting data from existing Bluetooth devices, not relevant for the proximity detection process. The PD module, implements the proximity detection algorithms. More specifically, the BL listens for beacons in a moving time window Δt. The collected information is then processed in the next time window βt, applying the RSS filter, namely the p-percentile. Filtered data are provided as input to the PD module in charge of identifying the three nearest artwork identifiers according to a specific algorithm. Indeed, we decided to rank the top-three artworks so that to mitigate any possible error while detecting the proximity, but without affecting the final user-experience. Lastly, the UI module shows to the visitor the thumbnails of the top-3 artworks.

The UI module implements three screens, optimized for mobile devices: a splash page, the main page and the artwork's detail page, as reported in Italian language in Figure 6. The Main page shows the top-three list of nearby artworks with the corresponding thumbnail, clicking on a thumbnail it is possible to access to the artwork's details.

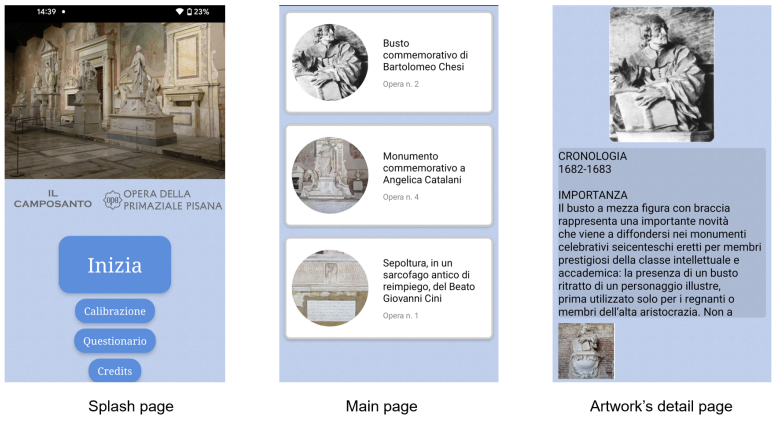


Figure 6: Screenshots of the three main pages of R-app.

Concerning the implementation of proximity algorithms, they are included with the R-app code and packaged as a mobile application for Android OS. In particular, concerning `DeepProximity`, the neural network models can be designed and tested by means of the Keras library for Python. The Keras model consists of: (i) the structure of the network such as the number of layers and the connections among them, (ii) the set of weights associated to the nodes, (iii) the optimizer obtained by compiling the model, and (iv) a set of losses and metrics. In order to use the implemented Python model in the Android app, it is necessary to export the model structure and the associated weights by saving it to a serialized JSON file format and subsequently load the pre-trained model in the R-app exploiting the Tensorflow.js[4] library for React-Native (Figure 7).

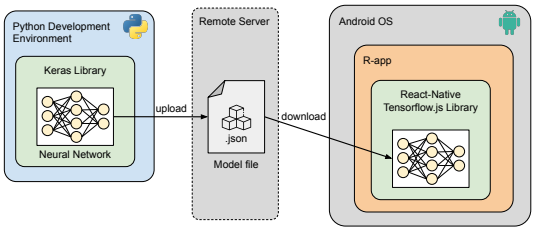


Figure 7: Conversion and deployment of the neural network model from the Keras format to Tensorflow.js for React-Native.

[4]https://www.tensorflow.org/js

The model file is stored in a public repository and R-app downloads it when needed. Although in the static version of the application the model is downloaded only once during the startup, with the crowd-based version of the application an additional task is required to periodically update the model from a remote server.

A last consideration refers to the collection of the Ground Truth. As reported in Section 3, we assume that visitor's smartphones upload $[R, P, GT]$ data. GT identifies the nearest artwork for a visitor, and this information is crucial to exploit R, P data. We now discuss a possible strategy to infer the GT. Our approach consists of logging user's gestures during the usage of R-app, such as navigating through the pages. More specifically, it is possible to log the timestamp associated with the `click` event of the Artwork's detail page. This event tells us that a visitor is interested to details of a specific artwork and hence is presumably in front of it. The events we track are: `<timestamp, select_artwork, artwork_ID>` and `<timestamp, back_from_artwork, artwork_ID>`, from which it is possible to extract the GT for the $[R, P]$ data to upload.

4.2. The Monumental Cemetery's Museum

The Monumental Cemetery's museum of Pisa, located in Piazza dei Miracoli [5] is a semi-indoor space with a rectangular shape. The museum is composed of an indoor section composed by long and short corridors where artworks are positioned. The covered area of the museum covers approximately $5{,}000m^2$, the long corridors are 130x9m, while the short corridors are 42x11m. Statues, frescos, sarcophagi, tombstones are all placed inside the inner corridors at different distances from each other, as well as along the walls, as reported in Figure 8a. In the short corridors, artworks are close to each other (about 1 m - 5 m distance). Differently, in the long corridors, artworks are more distant. Artworks are placed so that visitors can observe them frontally; tombstones, instead, are vertically-placed becoming tiles of the floor, visitors cans step over them. This last aspect is the distinguishing feature of the Monumental Cemetery's museum, as visitors step over historical tombstones.

4.3. Experimental Settings

We conduct a data collection campaign whose goal is measuring the improvement of a CrowdSensing approach for two proximity algorithms. Data are collected by varying: i) the visiting paths and ii) the adopted smartphones, as shown in Figure 9 and as summarized in Table 5. The collected data are available to the community as a public repository [6].

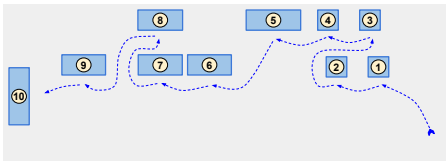
Artworks are monitored with Bluetooth tags produced by GlobalTag. Tags advertise beacons at 2 Hz and they are powered with CC2032-type battery and 0 dBm power of emission, as shown in Figure 8b. Visitors stop about two minutes in front of each artwork, according to the four different visiting layouts.

[5] GPS coordinates: 43.724028582, 10.394915035

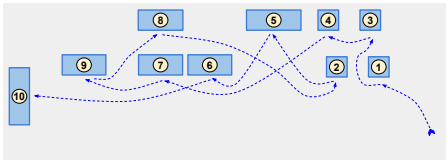
(a) Layout of the Monumental Cemetery of Pisa.

(b) Bluetooth tag used for the data collection.

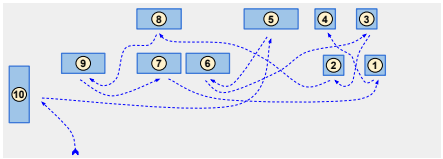
Figure 8: The Monumental Cemetery of Pisa and the deployed Bluetooth tags.



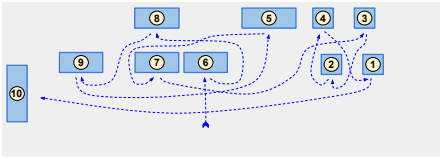
(a) Visiting layout 1.



(b) Visiting layout 2.



(c) Visiting layout 3.



(d) Visiting layout 4.

Figure 9: The four visiting layouts, the blue arrow denotes the path.

Table 5: Details of the tests executed. The table reports the test ID, the visiting layout, the corresponding artworks' orders and the adopted device.

Test ID	Visiting Layout	Artwork's order	Adopted Device
R1	Layout 1	1-2-3-4-5-6-7-8-9-10	Pixel 4
R2			Redmi 8
R3			Honor 8-2
R4			Honor 9
R5			Huawei P30
R6			Honor 8-1
R7			Honor 8 Pro
R8			Huawei P8
R9	Layout 2	1-3-4-7-9-8-2-5-6-10	Pixel 4
R10			Redmi 8
R11			Honor 8-2
R12			Honor 9
R13			Huawei P30
R14			Honor 8-1
R15			Honor 8 Pro
R16			Huawei P8
R17	Layout 3	10-5-6-3-2-8-9-7-1-4	Pixel 4
R18			Redmi 8
R19			Honor 8-2
R20			Honor 9
R21			Huawei P30
R22			Honor 8-1
R23			Honor 8 Pro
R24			Huawei P8
R25	Layout 4	6-8-9-5-7-3-2-4-1-10	Pixel 4
R26			Redmi 8
R27			Honor 8-2
R28			Honor 9
R29			Huawei P30
R30			Honor 8-1
R31			Honor 8 Pro
R32			Huawei P8

5. Experimental Results

The collected data have been analyzed to measure the performance by considering the static and the crowd versions of the algorithms presented in Section 3. We compute the performance by considering three metrics: Accuracy, Precision and Recall, as typically done with a classification task. Accuracy measures the fraction of predictions an algorithm got right. Precision corresponds to the number of true correct answers in a given class, divided by the number of observations of that class. The recall metric measures the number of correct answers in each class divided by the actual number of objects in that class. With precision, we can make sure that what we identify as proximity is actually a proximity event, while with recall, we can make sure to not miss out other positive observations. We compare the performance of `Threshold` and `DeepProximity` algorithms against a reference baseline, namely the `Max` algorithm. `Max` adopts a path-loss propagation model to convert the RSS values into a distance measure. The resulting nearest artwork is the one with minimum found distance [9, 12, 15, 18]. Such algorithm is often-adopted to estimate the proximity, hence it is used to compare the results of the static and crowd algorithm fairly. More specifically, given the path-loss model described with Equation 1, the distance function used with `Max` from a given artwork j can be estimated as:

$$f_j(d) = e^{-\frac{RSS_j - RSS_0}{10n}} \quad \forall j \tag{4}$$

The nearest artwork $px(k)$ at time k is given by $px = \underset{d}{\operatorname{argmin}}\, f_j(d)$ while a ranking of the "most near" artworks could be easily generated by sorting Equation (4).

5.1. Results

Experimental results are obtained by studying the performance of two algorithms: `Threshold` and `DeepProximity` compared against a baseline algorithm referred to as `Max`. We adopt three settings for the proposed algorithms reported in Table 6, namely: Device, Visiting Layout and by All-Runs. Each setting requires a specific configuration for the two algorithms, as described in the next. Concerning the Device setting, the goal is testing the algorithms without considering the device heterogeneity that we introduced in Section 2.2. More specifically, we compare the algorithms when using the same device model for training and testing purposes. The static version of `Threshold` is executed using one single threshold τ obtained from the first test. For example, tests executed with Google's Pixel 4 are: $[R_1, R_9, R_{17}, R_{25}]$ and in this case, Threshold is calibrated using R1. The performance of the algorithm is then computed with the three remaining tests: $[R_9, R_{17}, R_{25}]$, we denote this setting with the notation: $[R1 \rightarrow R9, R17, R25]_\tau$ as reported in Table 6. Concerning the crowd version of `Threshold`, a similar approach is followed but, differently from the previous case, we exploit the crowd mechanism. In particular, we use ten thresholds $[\tau_1 \cdots \tau_{10}]$ one for each artwork, such thresholds are obtained by using data

Table 6: Experimental settings: Device, Visiting Layout and All-Runs

	Device (ex. Pixel 4)	
	static	**crowd**
`Threshold`	$[R_1 \rightarrow R_9, R_{17}, R_{25}]_{\tau}$	$[R_1 \rightarrow R_9, R_{17}, R_{25}]_{[\tau_1 \cdots \tau_{10}]}$
`DeepProximity`	$[R_1 \rightarrow R_9, R_{17}, R_{25}]$	$[R_1 + R_9 + R_{17} \rightarrow R_{25}]$
	Visiting Layout (ex. Layout 1)	
`Threshold`	$[R_1 \rightarrow R_2, R_3, \cdots R_8]_{\tau}$	$[R_1 \rightarrow R_2, R_3, \cdots R_8]_{[\tau_1 \cdots \tau_{10}]}$
`DeepProximity`	$[R_1 \rightarrow R_2, R_3, \cdots R_8]$	$[R_1 + R_2 \cdots + R_7 \rightarrow R_8]$
	All-Runs ($R_1 \rightarrow R_{32}$)	
`Threshold`	$[R_i \rightarrow R_1, R_{i-1}, R_{i+2} \cdots R_{32}]_{\tau}$	$[R_i \rightarrow R_1, R_{i-1}, R_{i+2} \cdots R_{32}]_{[\tau_1 \cdots \tau_{10}]}$
`DeepProximity`	$[R_i \rightarrow R_1, R_{i-1}, R_{i+2} \cdots R_{32}]$	$[R_1 + \cdots + R_{20} \rightarrow R_{21} \cdots R_{32}]$

collected with test R1 and the performance are computed with three remaining tests, we denote to this setting as $[R_1 \rightarrow R_9, R_{17}, R_{25}]_{[\tau_1 \cdots \tau_{10}]}$ in Table 6. The static version of `DeepProximity`, it is obtained by training the neural network with one out of four tests, and we test the algorithm with the three remaining tests, we denote this setting with the notation: $[R_1 \rightarrow R_9, R_{17}, R_{25}]$ in Table 6. While, concerning the crowd version of `DeepProximity`, the algorithm is trained with three out of four runs (R_1, R_9 and R_{17}), and it is tested on the last test (R25). It is important to note that the training set can be further extended as soon as the visitors provide new data; this is the strength of the proposed crowd solution.

We report in Figure 10 the results of the Device setting for the three algorithms: `Max` (baseline), `Threshold` and `DeepProximity` as a bar plot. Each row of the figure shows a specific metric: Accuracy, Precision and Recall, each column shows results for a different algorithm and bars show the performance comparing the static and crowd version (blue vs orange bars). As a general trend, we observe from Figure 10 an increase of the performance when adopting the crowd version of an algorithm. In particular, the `DeepProximity` algorithm provides the best results both on the static and crowd versions, with 72% and 81% of Accuracy. An interesting observation is related to the Threshold algorithm, whose performance remarkably increases from the static to the crowd version, with an average increase of the Accuracy score of 30% (from 32% of the static version to 62% of the crowd version). Despite its simplicity, the `Threshold` algorithm exhibits lower accuracy when compared to `Max` and `DeepProximity`, as evidenced in our comparisons. However, the introduction of the crowd version, which incorporates multiple thresholds for various artworks, significantly enhances overall performance. This aspect underscores the effectiveness of the CrowdSensing approach outlined in our work.

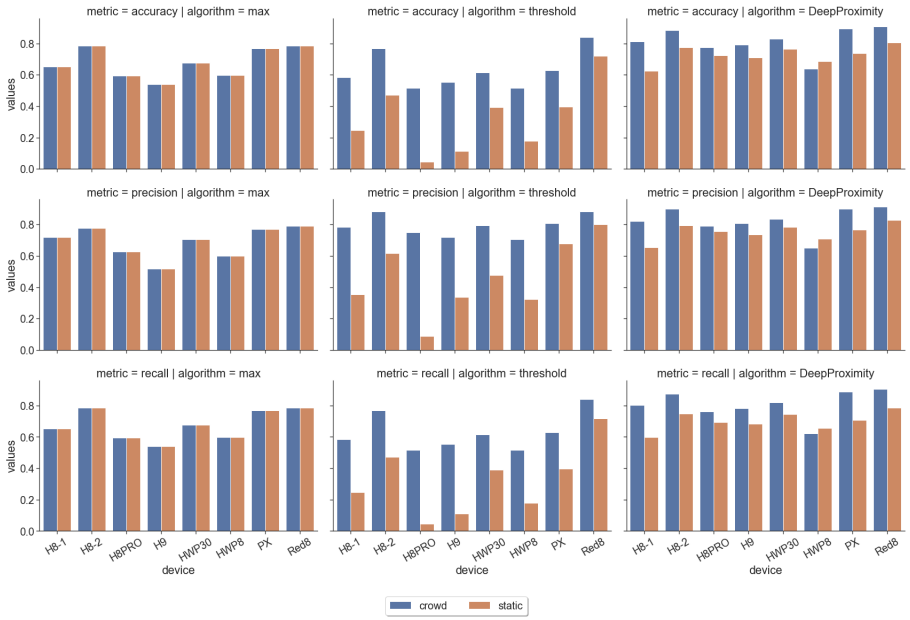


Figure 10: Experimental results of the Device setting.

The second setting is referred to as Visiting Layout setting (see Table 6). In this case, we increase the complexity by also varying the visiting layout. More specifically, we test the three algorithms with different smartphones and with different visiting layouts (see Figure 9). Given Layout 1, the following tests are executed: $[R_1 \cdots R_8]$. In this case, the static version of `Threshold` adopts R_1 as calibration test, and the remaining seven runs to test the performance, we denote this setting as: $[R_1 \rightarrow R_2, R_3, \cdots R_8]_\tau$ in Table 6. Concerning the crowd version of `Threshold`, we use ten different thresholds obtained from test R_1, while the performance are computed on the remaining seven tests: $[R_1 \rightarrow R_2, R_3, \cdots R_8]_{[\tau_1 \cdots \tau_{10}]}$. For what concerns `DeepProximity`, we vary the test set according to the static or crowd versions. In particular, when using the static version, we use R_1 as a test set and $[R_2 \cdots R_8]$ as validation set: $[R_1 \rightarrow R_2, R_3 \cdots R_8]$. Differently, when using the crowd version of `DeepProximity`, we use R_1 to R_7 as test test and R_8 as validation test: $[R_1, R_2 \cdots, R_7 \rightarrow R_8]$. Results of the Layout setting are reported in Figure 11. We observe a general increase of the performance when switching to the crowd version of the algorithms. We measure an average Accuracy score for the `DeepProximity` of 63% with the static version and 80% with the crowd version, a net increase of 16%. Concerning the `Threshold` algorithm, the increase of the performance is also evident: from 32% to 47% and a net improvement of 15%. It is worth to notice that the overall performance are slightly lower than that of the Device setting. This decrease depends from the device heterogeneity which makes the proximity detection a more complex task. This is particularly evident for the `Threshold` algorithm. Indeed, we calibrate the static and the

crowd version of the `Threshold` algorithm by using a specific smartphone model, such as the Pixel 4, but we test `Threshold` algorithm with different smartphone equipped with a different Bluetooth chipsets (and with a different antenna's sensitivity). As a result, the calibration obtained with a specific device model might no longer be valid. We report in Appendix Appendix A an in-depth analysis of the device's offsets.

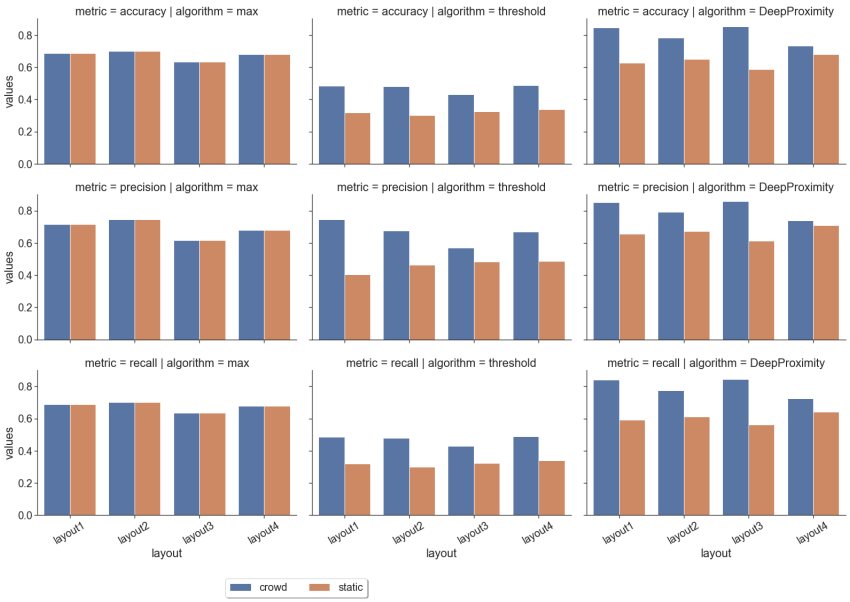


Figure 11: Experimental results of the Layout setting.

Finally, we adopt more challenging testing conditions, namely the All-Runs setting, see Table 6. With this setting, we combine results from all the 32 testing runs, mixing two orthogonal complexities: device heterogeneity and different visiting paths. Concerning the static version of `Threshold`, we calibrate the algorithm with one run, e.g. $R_i, i \in [R_1 \cdots R_{32}]$, and we test it with the remaining 31 runs, and then we average the results: $[R_i \rightarrow R_1, R_{i-1}, R_{i+2} \cdots R_{32}]_\tau$. The crowd version follows the same approach of the Device and Layout settings. In this case, we use a multi-threshold set extracted from a calibration run, $[R_i \rightarrow R_1, R_{i-1}, R_{i+2} \cdots R_{32}]_{[\tau_1 \cdots \tau_{10}]}$. For what concerns the `DeepProximity` algorithm, for the static version we train the model with one only run, and we test on the remaining 31 runs, after which we average the results: $[R_i \rightarrow R_1, R_{i-1}, R_{i+2} \cdots R_{32}]$. While, for the crowd version we train on 60% of the runs, and test on the remaining 40% of the runs: $[R_1 + \cdots + R_{20} \rightarrow R_{21} \cdots R_{32}]$. Results of the All-Runs setting are reported in Figure 12.

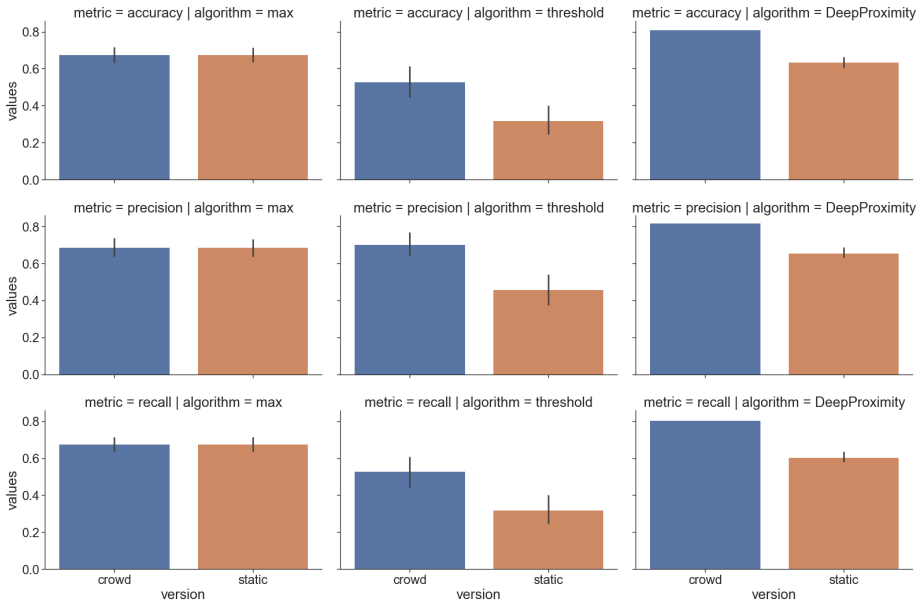


Figure 12: Experimental results of the All-Runs setting.

The All-Runs setting interestingly provides the best improvement. We measure an average Accuracy of `DeepProximity` algorithm of 63% with the static version and of 81% with the crowd version, with a net improvement of 17%. While, for the `Threshold` algorithm, the Accuracy metric varies from 32% to 52%, with 20% of net improvement. As previously introduced, the All-Runs setting well reproduces a realistic condition in which visitors own different smartphones and they follow an arbitrary path. Under these challenging conditions the `DeepProximity` correctly classifies 81% of proximity events with the existing artworks for any adopted smartphone and visiting layout. As a last consideration, we observe that `DeepProximity` always increases the performance with the crowd version.

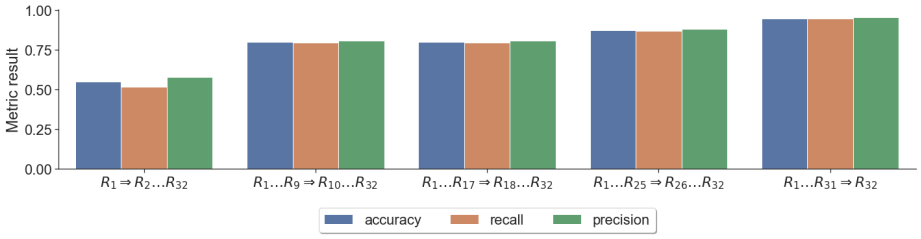


Figure 13: Experimental results obtained with the `DeepProximity` crowd algorithm as the training data increases.

To highlight the enhancement achievable with the crowd version, we tested

this approach by subsequently increasing the number of runs used in the training phase. As shown in Figure 13, as we train the neural network with more runs, we observe an improvement on all the metrics. The accuracy quickly increases from 55% when using just one run (R_1) as training, to 75% when using nine runs ($R_1 \dots R_9$) as training set. From there the accuracy slowly but constantly increases as additional runs are used for training, reaching the value of 95% when employing 31 runs ($R_1 \dots R_{31}$). The other two metrics, recall and precision, closely match the corresponding accuracy values. This is an expected behavior since, as a general trend, the more data are used for training a network, the higher the expected performance.

From the obtained results, we can summarize the main findings as follows:

- Except for the Max algorithm used as a base line, the crowd version of Threshold and DeepProximity algorithms almost always improve the overall performance with respect to the static versions in all the three settings (Device, Layout and All-Runs);
- The crowd version of the DeepProximity achieves the best performance in terms of accuracy, precision and recall in all the tested settings;
- The crowd approach effectively addresses the device heterogeneity issue allowing the algorithms to adapt to different devices even after the deployment phase;

Therefore, we argue that combining a CrowdSensing approach with a learning-based algorithm for proximity detection represents a promising approach.

6. Conclusions

The massive diffusion of wearable devices equipped with sensing units and wireless connectivity has opened to a new computational paradigm, referred to as CrowdSensing. The underlying idea is designing systems able to collect and exploit data provided by user's devices, by unleashing their potentiality as active data providers. This approach is often used by many successful mobile applications, such as navigation apps and sport tracker applications. In these cases, contents are continuously updated as new data from the crowd are collected and elaborated. We apply this principle in a cultural heritage context. In particular, in this work, we face with the problem of designing a full-stack architecture able to detect the proximity between visitors and artworks. To this purpose, we describe the design of two algorithms, implementing a static and a crowd version and comparing them with a baseline proximity algorithm. In the first case, crowd data are not used, while in the second case, algorithms are continuously (re)calibrated with data provided by the crowd. We detail in the paper the data collection campaign we organized in the Monumental Cemetery's museum located in Piazza dei Miracoli, Pisa (IT). Tests have been done using eight different smartphones and following several visiting paths. From our test, we are able to clearly observe an increase of the performance when switching to

the crowd-based version of the tested algorithms. It is worth to notice that the collected dataset is available to the community [6].

The proposed idea only covers one of the many improvements areas of the CrowdSensing parading. We also investigate the combined use of RSS and direction sensors to estimate the relative orientation of a visitor with respect to a POI. More specifically, the recent Bluetooth 5.1 Direction Finding specification introduces the possibility for a Bluetooth device to estimate the Angle of Arrival (AoA). This feature can potentially be exploited to determine the relative orientation of a visitor with respect to an artwork. The combined adoption of RSS and AoA allows to implement a navigation service, guiding a visitor along a pre-defined visiting path. A preliminary study about the potentialities of AoA in indoor environments is reported in [31] and a Bluetooth 5.1 dataset is available in [32].

Acknowledgment

This work is partially funded by RE.S.I.STO project: "Recupero di Sistemi Informativi STOrico-artistici per una rinnovata comunicazione del patrimonio", funded by Regione Toscana, progetti POR FSE 2014-2020. Authors would like to also thank "Opera della Primaziale Pisana" (OPA) for their valuable support during the data collection campaigns. This work is also partially funded by European Union - Next Generation EU, in the context of The National Recovery and Resilience Plan, Investment 1.5 Ecosystems of Innovation, Project Tuscany Health Ecosystem (THE), CUP: B83C22003920001.

CRediT author statement

Michele Girolami: Conceptualization, Methodology, Software, Writing, Supervision, Investigation, Visualization. Davide La Rosa: Conceptualization, Methodology, Software, Writing, Investigation, Visualization, Data Preparation. Paolo Barsocchi: Conceptualization, Methodology, Writing, Supervision, Investigation.

Appendix A. Analysis of the Device's Offset

We detail in this appendix how thresholds obtained with a reference device differ with respect to other commercial devices. Figure A.14 shows for every tested smartphone, the threshold's offsets obtained the remaining smartphones. As for example, given the Pixel 4a as calibration smartphone (with a threshold at 1.5 m from an artwork of $\tau = -67$ dBm), we report in Figure A.14 the threshold's offsets as a bar plot. The ideal condition is a perfect threshold match (0 offset) between calibration and testing smartphone, but at real conditions, we observe remarkable differences. Those smartphones with negative offsets are those estimating stronger RSS values with respect to the calibration smartphone, e.g. -67 dBm vs -57 dBm. Conversely, those smartphones

with positive offsets are those estimating lower values than that of the calibration smartphone. The case of Honor 8 PRO (H8PRO in Figure A.14) clearly shows that all the tested smartphones provide lower RSS values than that of the calibration smartphone.

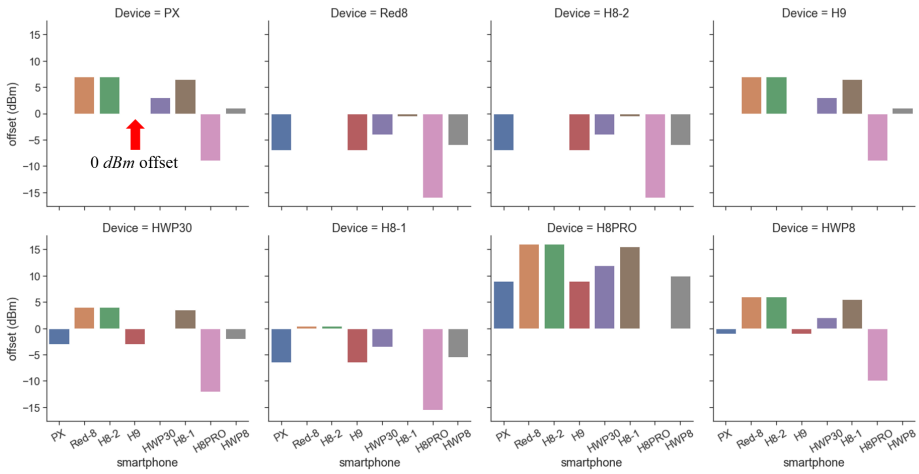


Figure A.14: Offsets of thresholds obtained with a reference smartphone and compared against the remaining ones.

References

[1] F. Potortì, A. Crivello, F. Palumbo, M. Girolami, P. Barsocchi, Trends in smartphone-based indoor localisation, in: 2021 International Conference on Indoor Positioning and Indoor Navigation (IPIN), IEEE, 2021, pp. 1–7.

[2] F. Zafari, A. Gkelias, K. K. Leung, A survey of indoor localization systems and technologies, IEEE Communications Surveys & Tutorials 21 (3) (2019) 2568–2599. doi:10.1109/COMST.2019.2911558.

[3] A. Kocian, M.-A. Badiu, B. H. Fleury, F. Martelli, P. Santi, A unified message-passing algorithm for mimo-sdma in software-defined radio, EURASIP Journal on Wireless Communications and Networking 2017 (1) (2017) 4. doi:10.1186/s13638-016-0786-y.
URL https://doi.org/10.1186/s13638-016-0786-y

[4] Z. Su, K. Pahlavan, E. Agu, Performance evaluation of covid-19 proximity detection using bluetooth le signal, IEEE Access 9 (2021) 38891–38906. doi:10.1109/ACCESS.2021.3064323.

[5] P. Barsocchi, M. Girolami, D. La Rosa, Detecting proximity with bluetooth low energy beacons for cultural heritage, Sensors 21 (21) (2021). doi:10.3390/s21217089.
URL https://www.mdpi.com/1424-8220/21/21/7089

[6] M. Girolami, D. La Rosa, P. Barsocchi, A bluetooth dataset for proximity detection in indoor environments collected with smartphones, Data in Brief co-submitted with this paper. doi:10.17632/sbhch9pxwh.1.

[7] A. Mackey, P. Spachos, L. Song, K. N. Plataniotis, Improving ble beacon proximity estimation accuracy through bayesian filtering, IEEE Internet of Things Journal 7 (4) (2020) 3160–3169. doi:10.1109/JIOT.2020.2965583.

[8] F. J. Aranda, F. Parralejo, F. J. Álvarez, J. Torres-Sospedra, Multi-slot ble raw database for accurate positioning in mixed indoor/outdoor environments, Data 5 (3) (2020). doi:10.3390/data5030067.
URL https://www.mdpi.com/2306-5729/5/3/67

[9] M. Pušnik, M. Galun, B. Šumak, Improved bluetooth low energy sensor detection for indoor localization services, Sensors 20 (8) (2020). doi:10.3390/s20082336.
URL https://www.mdpi.com/1424-8220/20/8/2336

[10] P. Centorrino, A. Corbetta, E. Cristiani, E. Onofri, Managing crowded museums: Visitors flow measurement, analysis, modeling, and optimization, Journal of Computational Science 53 (2021) 101357. doi:https://doi.org/10.1016/j.jocs.2021.101357.
URL https://www.sciencedirect.com/science/article/pii/S1877750321000521

[11] R. Giuliano, G. C. Cardarilli, C. Cesarini, L. Di Nunzio, F. Fallucchi, R. Fazzolari, F. Mazzenga, M. Re, A. Vizzarri, Indoor localization system based on bluetooth low energy for museum applications, Electronics 9 (6) (2020). doi:10.3390/electronics9061055.

[12] T. Nilsson, A. Blackwell, C. Hogsden, D. Scruton, Ghosts! a location-based bluetooth le mobile game for museum exploration (07 2016).

[13] A. Handojo, R. Lim, T. Octavia, J. K. Anggita, Museum interactive information broadcasting using indoor positioning system and bluetooth low energy: A pilot project on trowulan museum indonesia, in: 2018 3rd Technology Innovation Management and Engineering Science International Conference (TIMES-iCON), 2018, pp. 1–5. doi:10.1109/TIMES-iCON.2018.8621815.

[14] Y. Yoshimura, A. Krebs, C. Ratti, Noninvasive bluetooth monitoring of visitors' length of stay at the louvre, IEEE Pervasive Computing 16 (2) (2017) 26–34. doi:10.1109/MPRV.2017.33.

[15] S. Alletto, R. Cucchiara, G. Del Fiore, L. Mainetti, V. Mighali, L. Patrono, G. Serra, An indoor location-aware system for an iot-based smart museum, IEEE Internet of Things Journal 3 (2) (2016) 244–253. doi:10.1109/JIOT.2015.2506258.

[16] P. Spachos, K. N. Plataniotis, Ble beacons for indoor positioning at an interactive iot-based smart museum, IEEE Systems Journal 14 (3) (2020) 3483–3493. doi:10.1109/JSYST.2020.2969088.

[17] J. Allen, Bluetooth low energy: Museum connectivity application (05 2017).

[18] K. Sornalatha, V. R. Kavitha, Iot based smart museum using bluetooth low energy, in: 2017 Third International Conference on Advances in Electrical, Electronics, Information, Communication and Bio-Informatics (AEEICB), 2017, pp. 520–523. doi:10.1109/AEEICB.2017.7972368.

[19] J. Zhu, K. Zeng, K.-H. Kim, P. Mohapatra, Improving crowd-sourced wi-fi localization systems using bluetooth beacons, in: 2012 9th Annual IEEE Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks (SECON), 2012, pp. 290–298. doi:10.1109/SECON.2012.6275790.

[20] C. H. Lam, K. E. Jeon, S. Wong, J. She, Distance estimation using ble beacon on stationary and mobile objects, IEEE Internet of Things Journal 9 (7) (2022) 4928–4939. doi:10.1109/JIOT.2021.3120921.

[21] J. Weppner, P. Lukowicz, U. Blanke, G. Tröster, Participatory bluetooth scans serving as urban crowd probes, IEEE Sensors Journal 14 (12) (2014) 4196–4206. doi:10.1109/JSEN.2014.2360123.

[22] O. Semenov, E. Agu, K. Pahlavan, Z. Su, Covid-19 social distance proximity estimation using machine learning analyses of smartphone sensor data, IEEE Sensors Journal 22 (10) (2022) 9568–9579. doi:10.1109/JSEN.2022.3162605.

[23] A. Madhavapeddy, A. Tse, A study of bluetooth propagation using accurate indoor location mapping, UbiComp'05, Springer-Verlag, Berlin, Heidelberg, 2005, p. 105–122. doi:10.1007/11551201_7.

[24] S. Zhou, J. Pollard, Position measurement using bluetooth, IEEE Transactions on Consumer Electronics 52 (2) (2006) 555–558. doi:10.1109/TCE.2006.1649679.

[25] P. Barsocchi, S. Lenzi, S. Chessa, G. Giunta, Virtual calibration for rssi-based indoor localization with ieee 802.15.4, in: 2009 IEEE International Conference on Communications, 2009, pp. 1–5. doi:10.1109/ICC.2009.5199566.

[26] P. Baronti, M. Girolami, F. Mavilia, F. Palumbo, G. Luisetto, On the analysis of human posture for detecting social interactions with wearable devices, in: 2020 IEEE International Conference on Human-Machine Systems (ICHMS), 2020, pp. 1–6. doi:10.1109/ICHMS49158.2020.9209510.

[27] R. Faragher, R. Harle, Location fingerprinting with bluetooth low energy beacons, IEEE Journal on Selected Areas in Communications 33 (11) (2015) 2418–2428. doi:10.1109/JSAC.2015.2430281.

[28] C. Gentner, D. Günther, P. H. Kindt, Identifying the ble advertising channel for reliable distance estimation on smartphones, IEEE Access 10 (2022) 9563–9575. doi:10.1109/ACCESS.2022.3140803.

[29] J. Wilson, N. Patwari, Radio tomographic imaging with wireless networks, IEEE Transactions on Mobile Computing 9 (5) (2010) 621–632. doi:10.1109/TMC.2009.174.

[30] O. Appiah, M. Asante, J. B. Hayfron-Acquah, Improved approximated median filter algorithm for real-time computer vision applications, Journal of King Saud University - Computer and Information Sciences (2020).

[31] M. Girolami, P. Barsocchi, F. Furfari, D. La Rosa, F. Mavilia, Evaluation of angle of arrival in indoor environments with bluetooth 5.1 direction finding, in: 2022 18th International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob), 2022, pp. 284–289. doi:10.1109/WiMob55322.2022.9941619.

[32] M. Girolami, F. Furfari, P. Barsocchi, F. Mavilia, A bluetooth 5.1 dataset based on angle of arrival and rss for indoor localization, IEEE Access 11 (2023) 81763–81776. doi:10.1109/ACCESS.2023.3301126.