Finding Dory in the Crowd: Detecting Social Interactions using Multi-Modal Mobile Sensing

Kleomenis Katevas†⋄, Katrin Hänsel⋄, Richard Clegg⋄, Ilias Leontiadis♮, Hamed Haddadi†, Laurissa Tokarchuk⋄
† Imperial College London, ⋄ Queen Mary University of London, ♮ Telefónica Research

ABSTRACT

Remembering our day-to-day social interactions is challenging even if you aren't a blue memory challenged fish. The ability to automatically detect and remember these types of interactions is not only beneficial for individuals interested in their behavior in crowded situations, but also of interest to those who analyze crowd behavior. Currently, detecting social interactions is often performed using a variety of methods including ethnographic studies, computer vision techniques and manual annotation-based data analysis. However, mobile phones offer easier means for data collection that is easy to analyze and can preserve the user's privacy. In this work, we present a system for detecting stationary social interactions inside crowds, leveraging multi-modal mobile sensing data such as Bluetooth Smart (BLE), accelerometer and gyroscope. To inform the development of such system, we conducted a study with 24 participants, where we asked them to socialize with each other for 45 minutes. We built a machine learning system based on gradient-boosted trees that predicts both 1:1 and group interactions with 77.8% precision and 86.5% recall, a 30.2% performance increase compared to a proximity-based approach. By utilizing a community detection based method, we further detected the various group formation that exist within the crowd. Using mobile phone sensors already carried by the majority of people in a crowd makes our approach particularly well suited to real-life analysis of crowd behaviour and influence strategies.

Keywords

Mobile Sensing; Crowd Sensing; Social Interactions

Categories and Subject Descriptors

Human-centered computing [**Ubiquitous and mobile computing**]: Empirical studies in ubiquitous and mobile computing

1. INTRODUCTION

The ability to automatically detect social interactions in unorchestrated scenarios is highly sought after in many areas including social and behavioral science, crowd management, and targeted advertising. This ability would facilitate a wide range of technologies, for example: (i) crowd reconfiguration in evacuation management, providing instructions strategically to groups is more efficient than to individuals and avoids different members of a group being sent conflicting instructions; (ii) networking analytics, allowing individuals to trace their interactions in networking events (instead of exchanging business cards) and providing analytics to event organizers to optimize and monetize events; (iii) targeting advertisements to groups.

There have been many attempts for detecting social interactions automatically, primarily from video analysis. Most of the initial works are either based on manual annotated videos [15, 9] or use resource-hungry computer vision techniques [32, 3]. Other approaches use custom wearable hardware [8, 25] with advanced sensors (*e.g.* infrared light). These works report reasonable accuracy, but are expensive and problematic to scale in larger environments.

Smartphones and their wide range of embedded sensors enable researchers to explore social interactions in an automated way that depends entirely on the use of mobile sensing technology [22, 27], without the need for additional wearable equipment or computer vision systems that can also have implications with the user's privacy. Mobile sensing based solutions are also easier and more cost efficient to deploy in unknown or new spaces as they only rely on the users' own hardware. Early systems that use mobile sensing report accurate results, but primarily focus on detecting one-to-one social interactions. Furthermore they are restricted to controlled only environments, a situation that only covers a subset of the formations that occur in a natural setting. Finally, they rely on pre-trained models that only work with specific mobile devices.

In this paper, we investigate an approach for detecting stationary social interactions in a natural, non-artificial social setting. We built a machine learning system based on gradient-boosted trees to detect both 1:1 and group interactions. We then use a community detection algorithm based on graph theory to detect the various group formation that exist within the crowd.

We evaluate our system in a case study with 24 participants interacting together for 45 minutes. We tested two different approaches for inferring whether a group of people are close enough that a social interaction is feasible: (i) using high-performance, long-range beacons installed in the ceiling of the room, and (ii) without any fixed infrastructure. Notice that due to software limitations, the phones were not able to transmit beacons when the device is locked. Therefore, we ended up using coin-shaped beacons as a wearable device that simulates the smartphone's Bluetooth broadcasting function.

The main contributions of this work are:

- A machine-learning-based approach that predicts social interactions relying on data collected via the mobile phone, achieving a 77.8% precision and 86.5% recall, a 30.2% performance increase in terms of Average Precision compared to a proximity-based approach.
- A graph theoretical solution capable of detecting the different types of group formations that exist within the crowd.
- A proper evaluation in a natural (not artificially created) environment in three ways: (a) *link-level*, when an interaction between a pair of participants exist, (b) *node-level*, where a participant belongs to the correct interactive group, and (c) *group-level*, where a group of people is detected to include all participants correctly.

We further contribute and share a freely available dataset with unconstrained natural one-to-one and group interaction in varying sizes. To our knowledge, we are the first to present a system that automatically detects the social interactions in a natural environment, using mobile sensor data.

2. RELATED WORK

Hung & Krose [15] proposed a solution to detect face-to-face social interactions. In their study they used information about the proximity between people, as well as the body orientation to identify interactions with an accuracy of 92%. In a similar research, Cristani *et al.* [9] suggested a system that also detects interactions using information about people's position and head orientation. They also reported comparable results of 89% accuracy. Both works assume that the information of related proximity between participants or absolute position in the space, as well as the body or head orientation is known, either using manual annotations or computer vision techniques.

One of the first attempts to identify stationary, face-to-face interactions in an automated way is the *Sociometer* by Choudhury and Pentland [8], a wearable device that can be placed on each person's shoulder and identify other people wearing the same device using Infrared (IR) sensors. In addition, it is equipped with an accelerometer sensor to capture motion as well as a microphone to capture speech information. During the system evaluation, Sociometer was able to identify social interaction with an accuracy of 63.5% overall and 87.5% for conversations lasted for more than one minute.

Montanari *et al.* [25] created a wearable device named *Protractor* that uses near-infrared light to monitor the user proximity with a mean error of 2.3 – 4.9 cm, and relative body-orientation with a 6° error 95% of the time. The device was evaluated in a group-collaborative task where 64 participants split into groups of four were asked to collaborate with each other to build a construction made of spaghetti and plastic tape. Using a supervised machine learning approach based on Random Forest, they achieved 84.9% accuracy when detecting the individual's task role (*i.e.* the verbal role of each participant) and 93.2% accuracy when identifying the task timeline (*i.e.* the building phases of the collaborative task). Note that even though the evaluation of this work is focused on the social behavior of an existing group that is interacting, Protractor could also be used to detect stationary interactions within crowds by using the estimated proximity and relative orientation between participants.

Matic *et al.* [22] presented a solution based on the RSSI of the WiFi sensor as a way of estimating the proximity between people, and the embedded magnetometer to extract the standard deviation of the relative body orientation, as an indication of the position stability between participants. By also placing an external accelerometer device on each user's chest they analyzed the vibrations produced by the user's vocal chords and detected speech activity. They evaluated their approach in a office-located study with four participants for seven working days, achieving a performance of 89% true positives and 11% false negatives in detecting the social interactions taking place in the office.

Palaghias *et al.* [27] presented a real-time system for recognizing social interactions in real-world scenarios. Using the RSSI of Bluetooth Classic radios and a 2-layer machine learning model, they classified the proximity between two devices into three interaction zones, based on the theory of Proxemics [12]: *public*, *social* and *personal*. In addition, they used an improved version of *uDirect* research [13] that utilizes a combination of accelerometer and magnetometer sensors to estimate the user's facing direction with respect to the earth's coordinates. This work reported results of 81.40% accuracy for detecting social interactions, with no previous knowledge of the device's orientation inside the user's pocket. However, this work has been evaluated in a limited dataset with eight participants while an ob-

server was keeping notes that were later used as ground truth. Moreover, it is only capable of detecting one-to-one social interactions using a specific device model (HTC One S) and has not been evaluated in scenarios of interactions with dynamic sizes.

Katevas *et al.* [18] presented a simple approach for detecting stationary interactions in planned events, using the interpersonal proximity estimated by the device's Bluetooth Smart sensor. They evaluated the social interactions that took place in a controlled environment with six participants for four minutes, reporting a performance of 90.9% precision and 92.4% recall. This work was evaluated in a limited dataset with artificially created interactions instructed by the designer of the study and the algorithm used is similar to the baseline used in the this work.

3. INTERPERSONAL PROXIMITY ESTIMATION

In this section we present two experiments that evaluate the use of BLE-based beacon technology to estimate whether two participants are close enough for a social interaction to be feasible. The aim is to investigate whether the beacon's RSSI on a custom *Broadcasting Power* configuration can be a good predictor for a supervised machine learning classifier.

There have been several ways of estimating the distance between devices using wireless sensors such as *Time of Arrival*, *Time Difference of Arrival*, *Angle of Arrival* and using the *RSSI*. Currently, the only method that is applicable in smartphones is the RSSI of either the Bluetooth or the WiFi sensor.

In the past, researchers have used the RSSI of Bluetooth [21, 14, 27], WiFi [23] or even a combination of them [2] by measuring the RSSI of every wireless sensor available in range and comparing it with a *Measured Power* constant that indicates the signal strength (in dBm) at a known distance (usually $1m$). In 2010, the Bluetooth Special Interest Group released Bluetooth v4.0 with a Low Energy feature (BLE) that was branded as Bluetooth Smart. Bluetooth Smart is low cost for consumers, has low latency in communications ($6ms$) and is power efficient. Moreover, it supports a low energy advertising mode where the device periodically broadcasts specially formatted advertising packets to all devices in range with a customizable sample rate of approximately $3Hz$. This packet can include 31 bytes of information, such as a unique ID for each user, but also the measured power constant that was mentioned above. The advantage of using this technology for proximity estimation is that each manufacturer can configure the device to use its own pre-calibrated measured power constant, making the proximity estimation more accurate and device-type independent. In addition, devices do not need to maintain a connection with each other in order to measure the RSSI, having a minimum impact on the device's battery life. In its latest version, marketed as Bluetooth 5 [5], the sensor provides additional benefits including longer range (x4) and longer capacity in the advertising packet (x8).

Apple developed a proprietary protocol based on Bluetooth Smart, branded as *iBeacon*™and supported it as of iOS 7 (June 2013) in all mobile devices with Bluetooth 4.0 or greater (iPhone 4 or newer) [1]. The specification of iBeacon advertising packet includes: a 16 byte *Universally Unique Identifier (UUID)* used to separate beacon applications, two 2 byte unsigned integer identifiers named *Major*, which separates beacon groups (e.g. on the same venue or floor), and *Minor*, which separates individual beacons within the group, and an 1 byte *Measured Power* value used as an RSSI reference at $1m$ distance. The remaining available bytes are used as a static prefix and cannot be customized by the developer. An iOS application can register to monitor for beacons of specific UUIDs, and estimate its proximity whenever a beacon exists within range. The app can also advertise iBeacon™packets, however, only while the device is unlocked (*i.e.* in-use while the screen is on) and the app remains in the foreground, a restriction applied by the mobile operating system. Furthermore, it is not possible to customize the *Broadcasting Power* of the Bluetooth sensor, using the maximum power by default. Note that even though iBeacon™is an Apple product for iOS devices, it is possible to scan or broadcast as an iBeacon™from Android platform using third-party libraries[1]. Similar restrictions have been added in Android platformm with the release of Android v8.0 (Oreo), restricting apps to execute long-running services in the background[2].

To overcome these limitations, we use wearable beacons (*i.e.* RadBeacon Dot from Radius Networks[3]) to broadcast a beacon signal while a sensor data collection app is running as a background process on each user's smartphone. This also allows the customization of the broadcasting power of each beacon, achieving better accuracy in estimating the social space of each participant as shown below.

While previous evaluations report RSSI results from setups with beacons and phones mounted on tripods [18], or water bottles [27] to simulate the effect of body water on the beacon's RSSI, neither captured the affect of human posture or blockage by body parts. Moreover, the signal is not only affected by the body's water, but also the electric properties of human tissues (muscle, fat and skin) [29]. In this experiment, actual participants were used as a more realistic environmental setting.

[1]https://github.com/AltBeacon/android-beacon-library
[2]https://developer.android.com/about/versions/oreo/background
[3]https://www.radiusnetworks.com

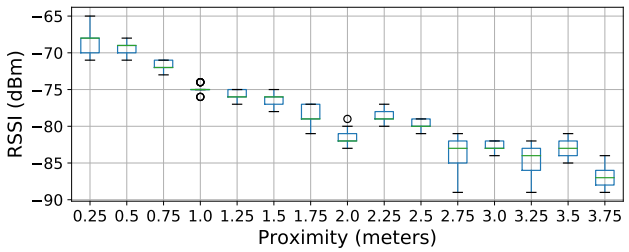

Figure 1: iBeacon™RSSI at varying proximity at minimum broadcasting power ($-18dBm$).

3.1 Effect of Broadcasting Power in the RSSI

In order to evaluate the optimal broadcasting power setting for detecting if a pair is within a social enabled zone, we conducted a short experiment. Two participants were recruited: P1, male with height $1.79m$ and weight $73kg$, and P2, male with height $1.83m$ and weight $87kg$. P1 served as the broadcaster and was equipped with eight coin beacons of the same type. Each beacon was configured to a different broadcasting power setting. P2 had the role of the receiver and had an iPhone SE device placed in one pocket. A mobile sensing app was used to collect iBeacon™Proximity data from all eight beacons, for 30 seconds at 15 distances from 0.25 to 4.00, every $0.25m$.

Our results show that each beacon, due to its configuration, has a different RSSI range and a unique pattern. For example, for the highest $+3dBm$ power, it was challenging to differentiate between distances 0.75 and 1.25, or 1.00 and 1.50. Most signals greatly fluctuate, especially the longer distances ($>1.75m$). We chose the minimum broadcast power (*i.e.* $-18dBm$) as it clearly separates the RSSI in distances until $1.5m$ (see Figure 1). Moreover, the signal looks relatively smooth compared to the others, which should aid classification in the distances of interest. Similar choice was made in [22] where the device's WiFi sensor in lowest power was used for the detecting social interactions.

3.2 Effect of Body Orientation in the RSSI

The low frequency of Bluetooth results in RSSI measurements that are highly affected by the human body. A second experiment was conducted to report how the RSSI is affected by relative body orientation and whether there are distinctive patterns that a machine learning classifier could benefit from. P1 and P2 were asked to stand facing each other at $1m$ distance and engage in a conversation. P1 was the broadcaster having two coin beacons configured to $-18dBm$ broadcasting power, one placed in each of his pockets. P2 was the receiver having two iPhone SE phones, one in each of his pockets. An app collected data for $30sec$ on all 64 combinations of different orientations, with a resolution of 45°. For each $30sec$ window, data was collected for all four combinations of device placement (left/right pocket).

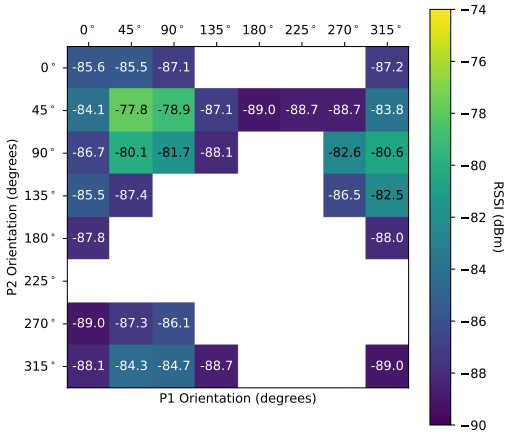

Figure 2: Heat map visualization of the mean RSSI of over different relative orientations of the two participants, having the devices placed at their left pocket. Blank entries indicates no data due to the low *Broadcasting Power* configuration used.

Figure 2 shows that the RSSI varies based on the relative orientation of the two participants. There are orientations that the device could not receive any signal from the beacon as the body effectively blocked the broadcaster's signal. This is a desirable result as social interactions are not possible at such orientations. Another important discovery is that the RSSI differs based on the configuration of in which pocket each device was placed (left or right). This suggests that knowing this configuration (*i.e.* in which pocket the user placed his/her phone) would result in better accuracy.

4. EXPERIMENTAL SETUP

In order to identify and evaluate the sensors needed for detecting stationary interactions in a natural setting, data was collected from participants during a social networking event. This section includes a description of the participants (Section 4.1), the procedure followed (Section 4.2), as well as the sensor data collected (Section 4.3).

4.1 Participants

37 potential participants were recruited via email and flyers; 24 of those took part in the actual study of which 9 were male and 15 female, with average weight $63.75kg$ (± 18.02), and average height $167.21cm$ (± 9.11). Participants were selected based on mobile phone model (iPhone 4 or higher) and operating system version (iOS 7 or higher) and availability of the iBeacon™sensor. Two devices experienced errors during the study (*i.e.* Bluetooth Smart sensor reported an internal error and did

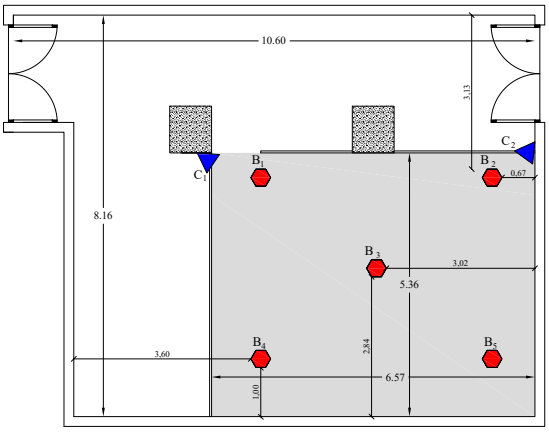

Figure 3: Floor plan of experimental location. The space contains two cameras (C_i) in blue, five ceiling beacons (B_i) in red and the interaction space is highlighted in grey.

not collect data) and were excluded from the data analysis, resulting into 22 valid participants.

4.2 Procedure

Participants installed a sensor data collection app, based on SensingKit for iOS v0.5 continuous sensing framework [17]. The app automated the sensor calibration, participant registration and data collection. Participants were invited to an indoor location with the floor plan given in Figure 3. The space is 10.60×8.16 meters, with $3.90m$ height; it is suitable for such type of experiments not only due to its isolation from outside noise and environmental factors, but also due to it being a natural space often used for social events and performances. It provides a DMX lighting rig installed in $3.27m$ height which was used to fix two HD cameras (C_i) recording video (but not audio) in $25fps$, covering an area of 6.57×5.36 meters (highlighted in grey in Figure 3). These videos were annotated to provide the ground truth for social interaction (see Section 5.1). This area was restricted using plastic dividers of $1.94m$ height to make sure that all interactions would be recorded by the cameras. Additionally, five Estimote Location Beacons[4] (B_i) were installed into the lighting rig, configured into the device default $300ms$ *Advertising Interval* and $-12dBm$ *Broadcasting Power*.

Before the study began, participants were asked to read the information sheet and sign the consent form. Participants were equipped with a Radius Networks RadBeacon Dot[5] each (coin shaped Bluetooth 4 based low energy beacons), to place in one of their pockets. All coin beacons were pre-configured to $10ms$ advertising interval (highest) and $-18dBm$ broadcasting power (lowest), based on the results reported in Section 3. Half of the participants were instructed to place the beacons in the left pocket and the other half in the right pocket. The phone was always placed in the other pocket as the beacon to avoid signal interference between the two devices.

During the setup process, participants were guided through the mobile app configuration. This process included a facial photograph used to enable the later ground truth video annotations and completion of the demographic collection form for the gender, weight and height of the participant. Finally, participants were asked to synchronously perform a wave-movement in front of the cameras. The recorded sensor data of each participant was later synced with the $25fps$ video feed, achieving a sync accuracy of $\pm$40ms.

Participants were then instructed to socially network for a total of 45 minutes. Snacks and beverages were served before and after the end of the experiment. The discussion topic was intentionally left open, trying to simulate a realistic networking scenario. After the session, participants returned the beacons, submitted the collected data and were reimbursed with £20 for their time.

In total, 99 one-to-one interactions were observed with a mean duration of $254.9sec$ ($\pm$161.7) and 22 group interactions (*i.e.* interactions that include more than two participants) with a mean duration of $117.2sec$ ($\pm$139.4). A separate interaction begins when the members of a group change. If the group configuration consisted less than $5sec$, then the interaction is not counted.

All data collection and analysis was made with informed consent and approved by the ethics committee of our institution.

4.3 Sensor Data Set

The dataset collected for each participant contains the following sensor data:

- **iBeacon™Proximity:** The RSSI from the mobile device with all beacons in range. This includes 24 coin beacon carried by the participants and also the high performance ceiling beacons.
- **Linear Acceleration:** The device measured acceleration changes in three-dimensional space. This excludes the 1g acceleration produced by gravity.
- **Gravity:** The orientation of the device relative to the ground, by measuring the 1g acceleration produced by gravity.
- **Rotation Rate:** The device's rate of rotation around each of the three spatial axes.

The sampling rate was set to the maximum supported (100Hz) for all motion and orientation sensors. iBeacon™Proximity sensor has a fixed (non-customizable) sample rate of 1Hz.

[4]https://estimote.com
[5]https://radiusnetworks.com

5. DETECTING SOCIAL INTERACTIONS

5.1 Ground Truth

Video recorded from two different angles was annotated by two independent annotators using ELAN multimedia annotator software [34]. As the aim of the study is to detect stationary interaction only, the annotators logged the beginning and end of each stationary interaction for each participant separately, using a unique ID per interaction. The annotations were cross-validated afterwards and finally verified by a third person. The instructions that the annotators followed were based on Kendon's F-formation system [19]:

> *An interaction begins at the moment two or more stationary people cooperate together to maintain a space between them to which they all have direct and exclusive access.*

5.2 Target Variable

The dataset has a total of 645,895 labels for each combination of the 22 valid participants interacting for a total of 45 minutes in the case study. The target variable is binary, with the following two classes: 1 when a pair of participants is interacting together, and 0 when they are not. That resulted into 38,332 labels in class 1 (6.31%), and 607,563 labels in class 0 (93.69%).

The dataset is naturally imbalanced since it includes one label for all combinations of the participants interacting with each-other per second. The level of this imbalance obviously depends on the number of people interacting, but also on the type of interaction (*e.g.* one-to-one, groups of three etc.). For instance, a small event of six people will include $C(6,2) = 15$ pairs, and thus 15 labels per second. If only three participants (A, B and C) interact together in a group of three, the dataset will include three labels of 1, one for each combination of them (AB, AC and BC), while the remaining 12 will belong to the class 0. In this small example, it is feasible to observe 15 labels of 1, however as the number of participants increases this becomes impossible. Thus lowering the feasible proportion of observed interactions from 100% (15/15) to $\sim 33\%$ ((2 x group of 9 + group of 4)/236).

5.3 Sensor Data Pre-processing

The data and video feed were synchronized based on the synchronous wave-movement in front of the cameras as mentioned in Section 4.2. For all iBeacon™Proximity data, all data reporting *Unknown* values (where RSSI is -1) were excluded. This usually occurs at the beginning of iBeacon™ranging process due to insufficient measurements to determine the state of the other device [1], or for a few seconds after the device gets out of the beacon's broadcasting range. All measurements from each user's beacon (*i.e.* from a participant's phone to their beacon) were also excluded.

Since mobile devices are not real-time systems, setting a sample rate is only a suggestion to the operating system, the actual rate varies second to second. Thus, the signal for the *Device Motion* sensor was resampled and interpolated to 100Hz. Finally, the magnitude was computed from the three axis of all motion data (*i.e.* User Acceleration, Gravity and Rotation Rate) available in the dataset, a process required since each user had their device in a different physical alignment and individual axis reading would not have provided useful information.

The iBeacon™sensor was the only sensor that reported missing values. Since most machine learning algorithms (*e.g.* Logistic Regression, Random Forest) do not accept features with unknown values, a data imputation process was required. Thus, missing values for external beacon data were imputed using linear interpolation [24] on the estimated distance from the device to the ceiling beacons. Since users are changing their state less frequent, it should be possible to estimate any possible missing value reliably using this approach. For the interpersonal distances inferred from the coin beacon, missing values were replaced with the maximum available distance, as in these cases, due to the low *Broadcast Power* that was used, the reason for missing data was that the device was out of range from the broadcaster.

5.4 Proximity Estimation

The Path Loss Model (PLM) was applied in order to estimate the proximity (d) between each device and all beacons in range using the RSSI ($P(d)$), as shown in the following formula:

$$d = 10^{\frac{P(d_0) - P(d) - X}{10 \times n}}, \quad (1)$$

where $P(d_0)$ is the Measured Power (in dBm) at 1 meter distance, n the path loss exponent, d the distance in which the the RSSI is estimated and X a component that describes the path loss by possible obstacles between the transmitter and the receiver. The value $n = 1.5$ was set as a default constant for indoor environments [22]. The value $X = 0$ was also chosen as it was required to measure a direct contact where no obstacles (*e.g.* other participants) between the two devices exist. In the situation that another participant exists in between, PLM would report a longer distance due to the decreased RSSI, and consequently, the accuracy of the distance estimation will decrease. This is a desired effect as, in the case of the coin beacons, it is only wanted to cluster whether the two users are within a range that a social interaction can be achieved. According to Hall [12], personal social interactions are achievable between 0.5 and 1.5 meters distance. Moreover, since all five long-range beacons were installed in

the ceiling of the room, a clear path between the phone and most of the ceiling beacons is expected.

5.5 Normalized Proximity

The *Normalized Proximity* (NP) is suggested by this work as an easy to compute approach for detecting social interactions using proximity based information. More specifically, the distance of two participants is used (computed using the Path Loss Model discussed in Section 5.4) with all unknown values (*i.e.* when the pair is out of beacon range) being replaced with the max of all distance estimations. A proximity value x is normalized into the range $[0, 1]$ as follows:

$$\hat{y} = \frac{\max(x) - x}{\max(x) - \min(x)}, \tag{2}$$

where $\hat{y}$ is an estimate as to whether the pair is interacting, and x is the estimated proximity between the pair and the min and max are taken over all observed values of x for all pairs. Because $\hat{y}$ is in the range $[0, 1]$ it can be compared to probability estimates.

The advantage of this baseline compared with other works in this area (*e.g.* [18]) is that it is comparable with other probabilistic performance metrics such as Precision-Recall (PR) and Receiver Operating Characteristic (ROC) plots. Probabilistic predictions have the advantage that the designer can choose a cut-off threshold that maximizes precision or recall, depending on the use case. For example, it might be desired to choose a high precision over recall so that the model only makes a positive prediction when the probability of an interaction is very high, resulting in an accurate result with the disadvantage of loosing some interactions that took place. The Normalized Proximity is also based on the estimated proximity between two people and is expected to report similar results.

5.6 Feature Engineering

A series of common features were computed for all $C(22, 2) = 231$ combinations of the participant pairs. Features reflecting the current moment were initially computed, in a static window of $1sec$, following with features reflecting past information. A set of features that are commonly included in mobile sensing problems were used, such as features extracted from motion and orientation sensors. Based on the results from the validation experiments reported in Section 3, additional features were explored that provide more precise information for detecting the social interactions (*i.e.* interpersonal space, device position and indoor positioning features). Table 1 lists the extracted features used in the data analysis of this work. The rest of this section reports on all 74 produced features and the selection strategy that was followed.

Table 1: List of the features used in the data analysis.

Table of Features	
Interpersonal Space Features	$f_{prox\_rssi\_mean}$ $f_{prox\_rssi\_diff}$
Device Position Features	$f_{device\_position\_LL}$ $f_{device\_position\_LR}$ $f_{device\_position\_RL}$ $f_{device\_position\_RR}$
Indoor Positioning Features	$f_{external\_beacon\_1\_diff}$ $f_{external\_beacon\_2\_diff}$ $f_{external\_beacon\_3\_diff}$ $f_{external\_beacon\_4\_diff}$ $f_{external\_beacon\_5\_diff}$
Motion and Orientation Features	$f_{time\_since\_moving\_diff}$ $f_{device\_linear\_acc\_ccf\_lag}$ $f_{device\_linear\_acc\_ccf\_max}$ $f_{device\_gravity\_ccf\_lag}$ $f_{device\_gravity\_ccf\_max}$ $f_{device\_rotation\_rate\_ccf\_lag}$ $f_{device\_rotation\_rate\_ccf\_max}$
Example of Past Information Features	$f_{external\_beacon\_1\_diff\_min}$ $f_{external\_beacon\_1\_diff\_max}$ $f_{external\_beacon\_1\_diff\_mean}$ $f_{external\_beacon\_1\_diff\_std}$ $f_{time\_since\_moving\_diff\_min}$ $f_{time\_since\_moving\_diff\_max}$ $f_{time\_since\_moving\_diff\_mean}$ $f_{time\_since\_moving\_diff\_std}$

Interpersonal Space Features

iBeacon™Proximity sensor data of a pair includes two measurements: Let $rssi_{ij}$ be the RSSI between the two participants as measured from the device of user i and $rssi_{ji}$ be the RSSI from the same distance as measured from the device of user j. The mean of the two measurements was computed as an indication of how close the two participants are in space:

$$f_{prox\_rssi\_mean} = (rssi_{ij} + rssi_{ji})/2 \tag{3}$$

In addition, a feature that represents the absolute difference between the two measurements was computed:

$$f_{prox\_rssi\_diff} = |rssi_{ij} - rssi_{ji}| \tag{4}$$

Note that in this case, the raw RSSI was used as the same hardware was used for broadcasting a beacon signal across all participants, and thus, a *Measured Power* constant is not required. In the case of multiple devices being used, then a feature that estimates the interpersonal distance based on a calibrated *Measured Power* constant would be required, using the PLM equation mentioned in Section 3.

Device Position Features

As mentioned in Section 3, information about the device position is important as it highly influences the RSSI signal between the two devices. For that reason, four features have been developed that includes the information of the device position (left vs. right per participant) using one-hot encoding:

- $f_{device\_position\_LL}$: Both P1 and P2 placed the device on the left pocket.
- $f_{device\_position\_LR}$: P1 placed the device on the left pocket, P2 on the right.

- $f_{device\_position\_RL}$: P1 placed the device on the right pocket, P2 on the left.
- $f_{device\_position\_RR}$: Both P1 and P2 placed the device on the right pocket.

Indoor Positioning Features

The absolute difference of each participant from the five ceiling beacons was computed using the following formula:

$$f_{external\_beacon\_k\_diff} = |D_{ik} - D_{jk}|, \quad (5)$$

where D_{ik} is the distance reported from user i's, and D_{jk} is the distance reported from user j's mobile device to the fixed installation k. It is expected that users close together would result in similar distances from the ceiling beacons and the feature will be close to zero. Note that estimated distance using PLM was used in this case instead of the raw RSSI as the long-range beacons that were used were installed in the room ceiling.

Motion and Orientation Features

By using the measurements of the linear acceleration sensor, a feature that indicates the time since the participant has moved (in seconds) was added. A threshold of $0.15g$ was empirically chosen, indicating whether a user is moving or not, and computed the absolute difference between the pair. It is expected that if two users are moving, they will stop at the same moment and engage into a conversation, and thus, the value of that feature will be close to zero. When both users had the status 'in motion', the feature was set to NaN (Unknown).

For all motion sensor data (*i.e.* Linear Acceleration, Gravity, Rotation Rate), a cross correlation function was applied on an overlapping window of 10 seconds and extracted the maximum correlation, as well as the distance (in seconds) from the max correlation, as an indication of how similar a pair is behaving on those windows. The 10 seconds constant was chosen as indicated by [22], but further investigation in the range of 2 to 60 also verified it as the most optimal constant. An alternative to the cross correlation function was also tested based on the Dynamic Time Warping (DTW) [31] method. However, due to its high computational complexity as well as its low predictive power in this context, it was excluded from the final feature list.

Past Information Features

In order to take advantage of past information available in the data set, the *min*, *max*, *mean* and *std* was computed on all time-series features (*i.e.* excluding the one-hot encoded device positioning features), in an overlapping window of 10 seconds. Note that due to the length of those features, only some representative examples are listed in Table 1.

5.7 Evaluation Procedure

For evaluating the performance of the model, a standard 10-fold cross-validation schema was used. The dataset was initially split over time, however, due to the time-series nature of our study, a significant overfitting was reported. More particularly, since participants were changing their interactive state at any given moment, the model was memorizing the features per split and inferring them back with very high performance, due to information leakage. Thus, the data was split per participant combination (*i.e.* 23 samples out of 231 due to the 10-fold schema) rather than over time.

In the context of this work, *Precision* is: from the detected interactions, how many of them did the model detect correctly, whereas *Recall* is: from all interactions taking place, how many of them did the model detect. Depending on the use case, applications can emphasize one measure over the other. The evaluation metrics that will be used in the rest of this report is *Precision-Recall (PR)* curve. Although ROC curves are heavily used when reporting performance in classification problems, due to the nature of our dataset being unbalanced, PR plots as suggested for this case by [30] and [10] were used.

5.8 Model Choice

As a learning model we use XGBoost [7]. XGBoost is a state-of-the-art gradient boosting regression tree algorithm that has emerged as one of the most successful feature-based learning models in recent machine learning competitions. We empirically found XGBoost consistently outperformed other well-established classifiers, such as Logistic Regression [26], Support Vector Machines [11], or Random Forests [6]. We used XGBoost v0.7.2.1 as part of the Python library scikit-learn [28] v0.19.1 and its wrapper for the XGBoost Python package.

A parameter tuning was performed on a 20% subset of the dataset (*i.e.* 46 samples out of 231). This subset was only used for the model tuning task and was never used in the training/validation procedure. The aim was to discover the model's configuration that maximizes the Average Precision (AP) performance. More specifically, a grid search algorithm over all possible combinations of the most influential parameters was followed, based on the following strategy:

- The total number of trees was set to 50.
- The balance of positive and negative weights was set to:
  $sum(negative\_cases)/sum(positive\_cases)$,
  as suggested by XGBoost documentation[6]. This was required due to the imbalanced nature of the dataset.

[6]http://xgboost.readthedocs.io/en/latest/parameter.html

- The maximum depth of the tree was tested with values [4, 6, 8, 10].
- The number of features to consider when looking for the best split was tested with values [0.2, 0.4, 0.6, 0.8, 1].
- The sub-sample ratio of the training instance was tested with values [0.5, 0.75, 1].
- The model's learning rate was tested with values [0.01, 0.05, 0.1].
- All other parameters used the library default values.

The configuration with the best performance of AP 80.4% (*i.e.* performance using the 20% subset) had the parameters `max_depth=4`, `colsample_bytree=0.2`, `subsample=0.5` and `learning_rate=0.05`. This configuration is used in the rest of this section for training and validating the model with the remaining 80% of the data set.

5.9 Detecting Group Formations

Detecting communities is important for a variety of applications including mobile social networks, recommender systems, security applications, and crowd management. One of our objectives is to automatically detect such group formations and classify the formed communities. Our concept for detecting group formation is based on graph theory. Each moment (in seconds) is represented as a undirected weighed graph $G = (V, E, w)$, with a set of vertices V and weighed edges $E(w)$. Each vertex corresponds to a participant, and each weighted edge corresponds to the probability of a pair that is interacting, as detected using the XGBoost classifier.

We use the modularity optimization approach developed by Blondel *et al.* [4], relying on the time-based stability of the network conditions at short time intervals [20], also known as *resolution parameter*. We used the Louvain Community Detection library[7] v0.11 using NetworkX[8] v2.1 to handle graph operations.

We applied the community detection on the network per second and considered a group formation when a connected component exists within the graph. We evaluate the performance of our approach in three ways: (a) *link-level*, where a link represents an interaction between a pair of participants, (b) *node-level*, where a node represents a participant that belongs to the correct interactive group, and (c) *group-level*, where a group is detected to include the correct participants.

6. RESULTS

[7]https://github.com/taynaud/python-louvain
[8]https://networkx.github.io

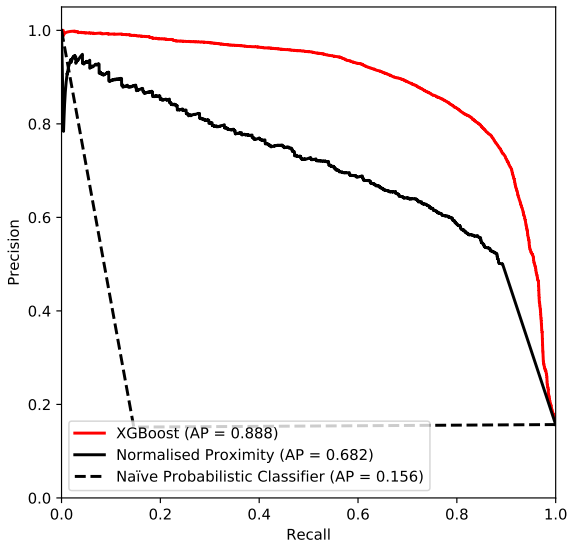

Figure 4: General performance of XGBoost classifier using a Precision-Recall (PR) Curve. The figure also includes the performance of the Naïve Probabilistic Classifier (NPC) and the Normalized Proximity (NP) for easy comparison.

In this section we present the results from the analysis reported in Section 5. As mentioned earlier, we report the performance of our approach in three ways: (a) *link-level*, (b) *node-level*, and (c) *group-level*.

6.1 Link-level Prediction

We report the performance of the XGBoost classifier predicting the pair interactions between participants using a standard 10-fold cross-validation on the remaining 80% samples of the dataset (*i.e.* excluding the 46 samples used for model tuning).

Indoor Positioning Features – We tested the general performance of the model, using features based on the ceiling beacons. Our aim was to detect interactions depending entirely on external infrastructure. Results reported a low performance of 18.2% AP, suggesting that it is not possible to achieve this with the current configuration of external beacons.

Interpersonal Distance Features – We further tested the performance using the features related to the coin-shaped beacon. Results report a performance of 88.8% AP (*i.e.* 30.2% increase from NP and 469.2% increase from NPC baseline). Figure 4 shows the performance using a Precision-Recall (PR) curve plot.

Further investigation that includes both types of features (*i.e.* Indoor Positioning and Interpersonal Distance) reported a lower performance of 86.3% AP. In the rest of this work, indoor positioning features will be excluded from the analysis.

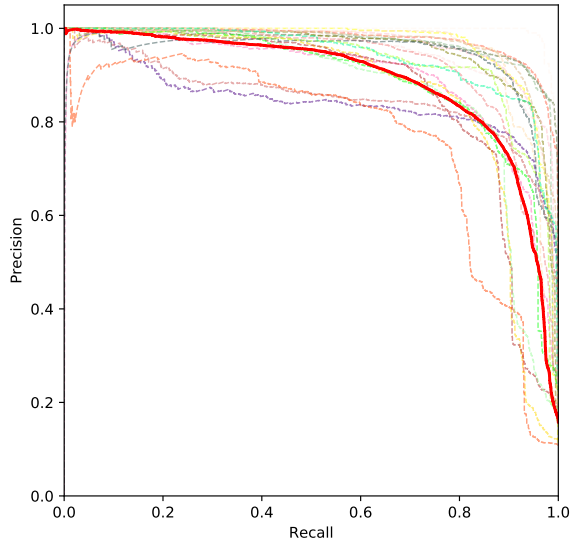

Figure 5: Performance of XGBoost classifier per participant. The coloured lines correspond to the performance of each participant, and the thick red line corresponds to the overall average across all participants.

Figure 5 provides a more fine-grained analysis of the results. It depicts the user-averaged PR curve for every participant, as well as the overall performance across all participants. It is clear that the model performance varies per participant, with some of them reporting almost perfect scores, while in some others perform lower than average (red thick line).

6.2 Sensor Importance

Accessing mobile sensor data has a significant effect on the battery life of the device, with some sensors such as the *Device Motion* being one of the most power expensive sensor of all others [17]. In this section we explore the sensor contribution of this approach, aiming to understand which sensors produce the features that are the worst predictors and how the model's performance will be affected when they are excluded.

For measuring the sensor contribution, a leave-one-sensor-out technique was used. The model was tuned (using the same approach discussed in Section 5.8) and then validated with all features except the ones produced by the excluded sensor. The following sensors (or external information in the case of device position) were manipulated:

- Interpersonal Space Features (*i.e.* features related to the coin beacons).
- Device Position Features (*i.e.* the one-hot encoded information of the smartphone position).
- Motion and Orientation Features (*i.e.* features related to the Device Motion sensor).

In the case of *Motion and Orientation Features*, the three sensor-fused data (*i.e.* Linear Acceleration, Gravity and Rotation Rate) are explored together, but separately as well. Figure 6 shows the results from this analysis. It is evident that by excluding the interpersonal space related features the model reports random performance, similar to the NPC classifier (*i.e.* AP 18.2%). The remaining sensors have a less significant effect to the model performance, with the removal of *Device Position Features* reporting AP 86.0% and *Motion and Orientation Features* reporting AP 83.0%.

These findings suggest that a model that depends entirely on the interpersonal space features would achieve a reasonable performance, considering the fact that the contribution from the other sensors with the current engineered features is very small and might not be significant if you take into account the battery consumption of such sensors.

6.3 Probabilistic Threshold Choice

Until now, reporting the performance of the model was made through metrics or plots that depend on probabilities (*i.e.* Precision-Recall as well as a numerical representations of these plots such as AP). In a real-world implementation of this model, a binary prediction will be required instead of a probability. The designer of such system can choose a threshold (also called probability cut-off) of which probabilities greater or equal to this threshold would be classified as 1, and 0 in all other cases. Choosing such threshold would lead into reporting the performance of the model in terms of *precision* and *recall*. Applications can emphasize one measure over the other. For example, in a use case of a mobile app that users install in order to log their interactions at a social event, the designer could emphasize on recall if the requirement is not to loose people they've interacted with, even if that results into increased false positives. If the requirement is a sticker model that captures interactions only when the probability is high, the designer could emphasize on precision.

For computing the most optimal threshold, the F1 score was used as a harmonic mean between precision and recall. Other measures such as F_2 score could be used that weights recall higher than precision, or $F_{0.5}$ which puts more emphasis on precision rather than recall. F1 was maximized in the same set used for model tuning. The value $p = 0.61$ for the XGBoost model and $p = 0.48$ for the NP was found to be the most optimal that maximizes the F1 score.

Table 2 shows the confusion matrix of the XGBoost classifier using a cut-off $p = 0.61$. It reports a precision of 77.8% and recall of 86.5%. Table 3 shows the confusion matrix of NP using a cut-off $p = 0.48$. It reports a

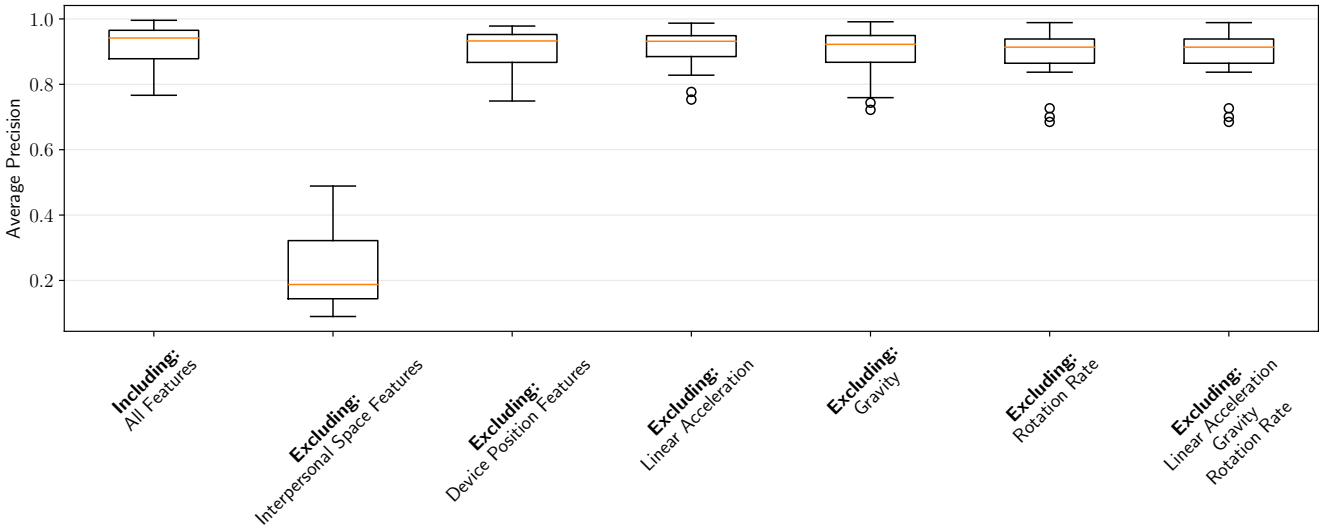

Figure 6: Sensor importance using leave-one-sensor-out as reported by the XGBoost classifier. A model that includes all features is also listed for easy comparison. The variability in the boxes corresponds to the considered participants.

Table 2: Confusion matrix for XGBoost classifier with cut-off $p = 0.61$

	Predicted Class		
	Positive	Negative	Total
Actual Positive	**123762 (TP)**	5940 (FN)	129702
Actual Negative	3269 (FP)	**20870 (TN)**	24139
Total	127031	26810	153841

Table 3: Confusion matrix for Normalized Proximity (NP) with cut-off $p = 0.48$.

	Predicted Class		
	Positive	Negative	Total
Actual Positive	**118589 (TP)**	11113 (FN)	129702
Actual Negative	6052 (FP)	**18087 (TN)**	24139
Total	124641	29200	153841

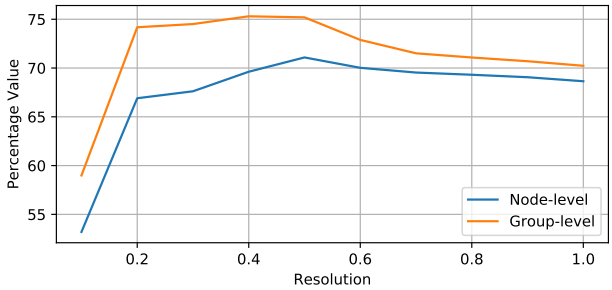

Figure 7: Performance of group formation detection at node- and group-level, using different resolution constants.

lower precision of 61.9% and lower recall of 74.9%.

6.4 Group Formation Detection

Figure 7 shows the performance of the group detection as described in Section 5.9. It displays the group detection accuracy on node- and group-level, using different community detection resolution constants within the range of 0.1 and 1.0.

The optimal resolution value in this case, shown in Figure 7, is 0.5, achieving a node-level performance of 71.09%, and group-level performance of 75.19%. Applying the same method on the NP baseline with the optimal resolution of 0.2 gives node-level performance of 48.65%, and group-level performance at 50.90%.

7. DISCUSSION AND IMPLICATIONS

Our results provide evidence that it is possible to detect interactive groups of various sizes relying on data collected from mobile devices, with a reasonable performance (77.8% precision and 86.5% recall). That means that 77.8% of the participants that the model discovered as interacting were correctly detected, and 86.5% of all interactions that actually took place during the event were detected by the model. Our approach is capable of detecting group formations at node-level performance of 71.09%, and group-level performance of 75.19%.

The dataset that has been analyzed, even though extended compared to other similar studies [27, 22, 18], only represents a subset of what is expected in similar social gatherings, such as conferences or other networking events. One technical challenge that arises is whether a real-time system that would continuously receive data from larger number of participants would be possible. Such system could be implemented as a cloud-based solution that either relies on a reliable internet connection, or save the data temporarily into the device and only submit when the event is completed. At the moment, such implementation is only possible using wearable beacons that simulate the beacon broadcast-

ing of each device, due to the restrictions of current mobile operating systems mentioned earlier. Although these limitations are software restrictions from the mobile operating systems and can change in future updates of the systems.

Importantly, to analyze the data, features from all combinations of participants have been extracted. This is possible with a reasonable number of participants, but does not scale to larger crowds. The use of external beacons installed in the room, even though did not improve the performance of the final model, could be a possible solution to this limitation. By using the proximity of each external beacon as as a pre-filter, clustering the crowd into smaller groups and only applying the model on each cluster. Such implementation could also benefit from parallel processing, assigning clusters to analyze each region in parallel.

Another challenge for our work is the real-world application of the model and user adoption. Requesting access to individuals' phone can be cumbersome as it relies on their willingness to share their data. Concerns about the privacy of the data collection, or the battery life of their devices might discourage users from participating on this experiment. However according to Wirz *et al.* [33], users are open to share their data as long as they receive some benefits from it or if they realize that sharing such information is for their own good and safety. In the next two sections, the battery consumption implications and privacy implications are discussed separately.

7.1 Battery Consumption Implications

As evaluated by Katevas *et al.* [16], collecting sensor data from mobile devices can have a noticeable impact in the device's battery. Some sensors, such as the sensor fused *Device Motion* that reports the *Linear Acceleration*, *Gravity* and *Rotation Rate* are consuming lots of processing power. Moreover, using the internet connection to periodically submit data packets to the cloud service would have an additional impact, something that is not considered in our case.

In the current study, the battery consumption of each device was also collected through the *Battery* sensor support of SensingKit framework [16]. For the total duration of the study, the battery drop was 0.08% per hour (SD: 0.06). That included a sensor data collection from eight sensors (*i.e.* Accelerometer, Gyroscope, Magnetometer, Device Motion, Heading, iBeacon™ Proximity and Battery) stored into the device's memory in CSV format. Note that in the current analysis, only features from the *Device Motion* sensor were used from the motion and orientation sensors, sampled in the highest sampling rate of 100Hz. According to the results reported in Section 6.2, similar results can be achieved with only the iBeacon™Proximity sensor.

7.2 Privacy Implications

Even though the method depends on using anonymous IDs when broadcasting iBeacon™data and no other personal information is broadcast, there is always the danger that this anonymity can be compromised by tracking the openly available ID of a user. This is the reason that, according to Apple, an iOS device is only allowed to broadcast as an iBeacon™while the device is unlocked and the app is actively running in the foreground [1].

A possible solution for protecting the user's privacy could be the the use of encryption on the advertising packet, so that only authorized people or applications can make use of it. Google provides an official support of encryption in the latest Eddystone-EID frame type[9], released in April 2016. Even though not officially supported in iBeacon™specification, third-party companies provide alternative solutions by rotating the beacon's attributes (*i.e.* Major and Minor) so that the broadcaster's ID is unpredictable[10].

8. CONCLUSION

In this work, we introduced a supervised machine learning approach capable of detecting stationary social interactions of a variety of sizes inside crowds. As far as we are aware, this is the first model capable of detecting group interactions from mobile sensing data larger than two and the first to be able to detect various sizes. Furthermore, our work does this in a relatively large (as compared to other related works) study, achieving a performance of 77.8% precision and 86.5% recall, when evaluating the interactions of the participants on link-level. Our approach is capable of detecting group formations at a node-level performance of 71.09%, and group-level performance of 75.19%. We will share our dataset that includes natural one-to-one and group interaction in varying sizes in anonymized format.

We believe that our work will be particular useful to researchers and practitioners wishing to explore crowd dynamics in social gatherings, event organizers aiming to monetize their events by providing rich analytics about their attendees, or event attendees wishing to remember their contacts without the need for exchanging business card or social media details.

Future work includes the exploration of device orientation sensors (*e.g.* gyroscope or magnetometer) as a way to measure the relative orientation between users in order to improve the performance of the model. Additionally, we aim to apply a real-time version of this work in an large-scale social event and explore the ways in which the crowd are interacting in planned events.

[9]https://developers.google.com/beacons/eddystone-eid
[10]https://developer.estimote.com/ibeacon/secure-uuid/

9. REFERENCES

[1] Apple Inc. Getting Started with iBeacon v1.0. `https://developer.apple.com/ibeacon/Getting-Started-with-iBeacon.pdf`, 2014. [Online; accessed 07-April-2018].

[2] N. Banerjee, S. Agarwal, P. Bahl, R. Chandra, A. Wolman, and M. Corner. *Pervasive Computing: 8th International Conference, Pervasive 2010, Helsinki, Finland, May 17-20, 2010. Proceedings*, chapter Virtual Compass: Relative Positioning to Sense Mobile Social Interactions, pages 1–21. Springer Berlin Heidelberg, Berlin, Heidelberg, 2010.

[3] L. Bazzani, M. Cristani, and V. Murino. Decentralized particle filter for joint individual-group tracking. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pages 1886–1893, June 2012.

[4] V. D. Blondel, J.-L. Guillaume, R. Lambiotte, and E. Lefebvre. Fast unfolding of communities in large networks. *Journal of Statistical Mechanics: Theory and Experiment*, 2008(10):P10008, 2008.

[5] Bluetooth Special Interest Group (SIG). Bluetooth Core Specification v5.0. `https://www.bluetooth.com/specifications/bluetooth-core-specification`, 2016. [Online; accessed 07-April-2018].

[6] L. Breiman. Random forests. *Mach. Learn.*, 45(1):5–32, Oct. 2001.

[7] T. Chen and C. Guestrin. XGBoost: a scalable tree boosting system. In *Proc. of the ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining (KDD)*, pages 785–794, 2016.

[8] T. Choudhury and A. Pentland. Sensing and modeling human networks using the sociometer. In *Proceedings of the 7th IEEE International Symposium on Wearable Computers*, ISWC '03, pages 216–, Washington, DC, USA, 2003. IEEE Computer Society.

[9] M. Cristani, L. Bazzani, G. Paggetti, A. Fossati, D. Tosato, A. Del Bue, G. Menegaz, and V. Murino. Social interaction discovery by statistical analysis of f-formations. In *Proceedings of the British Machine Vision Conference*, pages 23.1–23.12. BMVA Press, 2011. http://dx.doi.org/10.5244/C.25.23.

[10] J. Davis and M. Goadrich. The relationship between precision-recall and roc curves. In *Proceedings of the 23rd international conference on Machine learning*, pages 233–240. ACM, 2006.

[11] N. Deng, Y. Tian, and C. Zhang. *Support Vector Machines: Optimization Based Theory, Algorithms, and Extensions*. Chapman & Hall/CRC, 1st edition, 2012.

[12] E. T. Hall. *The hidden dimension*. Doubleday & Co, 1966.

[13] S. A. Hoseinitabatabaei, A. Gluhak, and R. Tafazolli. udirect: A novel approach for pervasive observation of user direction with mobile phones. In *Pervasive Computing and Communications (PerCom), 2011 IEEE International Conference on*, pages 74–83, March 2011.

[14] W. Hu, G. Cao, S. V. Krishanamurthy, and P. Mohapatra. Mobility-assisted energy-aware user contact detection in mobile social networks. In *Distributed Computing Systems (ICDCS), 2013 IEEE 33rd International Conference on*, pages 155–164, July 2013.

[15] H. Hung and B. Kröse. Detecting f-formations as dominant sets. In *Proceedings of the 13th International Conference on Multimodal Interfaces*, ICMI '11, pages 231–238, New York, NY, USA, 2011. ACM.

[16] K. Katevas, H. Haddadi, and L. Tokarchuk. Poster: Sensingkit: A multi-platform mobile sensing framework for large-scale experiments. In *Proceedings of the 20th Annual International Conference on Mobile Computing and Networking*, MobiCom '14, pages 375–378, New York, NY, USA, 2014. ACM.

[17] K. Katevas, H. Haddadi, and L. Tokarchuk. Sensingkit: Evaluating the sensor power consumption in ios devices. In *Intelligent Environments (IE), 2016 12th International Conference on*, pages 222–225. IEEE, 2016.

[18] K. Katevas, H. Haddadi, L. Tokarchuk, and R. G. Clegg. Detecting group formations using iBeacon technology. In *Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing: Adjunct*, UbiComp '16, pages 742–752, New York, NY, USA, 2016. ACM.

[19] A. Kendon. *Conducting interaction: Patterns of behavior in focused encounters*, volume 7. CUP Archive, 1990.

[20] R. Lambiotte, J.-C. Delvenne, and M. Barahona. Laplacian dynamics and multiscale modular structure in networks. *arXiv preprint arXiv:0812.1770*, 1, 12 2008.

[21] S. Liu, Y. Jiang, and A. Striegel. Face-to-face proximity estimation using bluetooth on smartphones. *IEEE Transactions on Mobile Computing*, 13(4):811–823, April 2014.

[22] A. Matic, V. Osmani, A. Maxhuni, and O. Mayora. Multi-modal mobile sensing of social interactions. In *Pervasive Computing Technologies for Healthcare (PervasiveHealth), 2012 6th International Conference on*, pages 105–114, May 2012.

[23] A. Matic, A. Papliatseyeu, V. Osmani, and O. Mayora-Ibarra. Tuning to your position: Fm

radio based indoor localization with spontaneous recalibration. In *Pervasive Computing and Communications (PerCom), 2010 IEEE International Conference on*, pages 153–161, March 2010.

[24] E. Meijering. A chronology of interpolation: From ancient astronomy to modern signal and image processing. *Proceedings of the IEEE*, 90(3):319–342, 2002.

[25] A. Montanari, Z. Tian, E. Francu, B. Lucas, B. Jones, X. Zhou, and C. Mascolo. Measuring interaction proxemics with wearable light tags. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, 2(1):25, 2018.

[26] J. A. Nelder and R. J. Baker. *Generalized linear models*. Wiley Online Library, 1972.

[27] N. Palaghias, S. A. Hoseinitabatabaei, M. Nati, A. Gluhak, and K. Moessner. Accurate detection of real-world social interactions with smartphones. In *Communications (ICC), 2015 IEEE International Conference on*, pages 579–585, June 2015.

[28] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[29] J. Rapiński, D. Zinkiewicz, and T. Stanislawek. Influence of human body on radio signal strength indicator readings in indoor positioning systems. *Technical Sciences/University of Warmia and Mazury in Olsztyn*, nr 19(2)(19 (2)):117–127, 2016.

[30] T. Saito and M. Rehmsmeier. The precision-recall plot is more informative than the roc plot when evaluating binary classifiers on imbalanced datasets. *PloS one*, 10(3):e0118432, 2015.

[31] S. Salvador and P. Chan. Toward accurate dynamic time warping in linear time and space. *Intelligent Data Analysis*, 11(5):561–580, 2007.

[32] J. Sochman and D. C. Hogg. Who knows who - inverting the social force model for finding groups. In *Computer Vision Workshops (ICCV Workshops), 2011 IEEE International Conference on*, pages 830–837, Nov 2011.

[33] M. Wirz, T. Franke, E. Mitleton-Kelly, D. Roggen, P. Lukowicz, and G. Tröster. Coenosense: A framework for real-time detection and visualization of collective behaviors in human crowds by tracking mobile devices. In *Proceedings of the European Conference on Complex Systems 2012*, pages 353–361. Springer, 2013.

[34] P. Wittenburg, H. Brugman, A. Russel, A. Klassmann, and H. Sloetjes. Elan: a professional framework for multimodality research. In *Proceedings of LREC*, volume 2006, page 5th, 2006.