Real Time Autonomous Point of Interest Mining through Ambient Smartphone Sensing

Tanusri Bhattacharya, Lars Kulik, James Bailey
The University of Melbourne
Department of Computing and Information System, Parkville campus
Melbourne, Victoria 3010, Australia
tanusri.bhattacharya@alumni.unimelb.edu.au,{lkulik, baileyj}@unimelb.edu.au

ABSTRACT

The advancement of sensor equipped smartphones provides tremendous opportunities for fine-grained monitoring of user Points of Interest (POIs) for a range of mobile applications such as place-based advertisement, personalised healthcare services, location based social networks. Existing systems, however, cannot infer both indoor and outdoor POIs using a single approach and typically require a mix of technologies for localization such as GPS, GSM or Wi-Fi. The accuracy of these techniques depend on the availability of local infrastructure and normally can retrieve POIs only at a coarse level, for example at the level of a building or region. We develop a novel algorithm to automatically detect user POIs in near real time at room level accuracy using only lightweight ambient environment sensors. Our method can infer both indoor and outdoor POIs at a fine granularity without depending on local infrastructure or without using GPS or Wi-Fi. It works in an unsupervised manner using covariances of ambient sensor data to detect user visits to POIs. An experimental study with real-world data shows that our system can achieve an F1 score of approximately 80% for the top 3 retrieved locations and outperforms the existing approaches such as *Google place search* and *Foursquare venue search*.

CCS Concepts

•**Information systems** → **Location based services; Mobile information processing systems;** *Data mining;*

Keywords

Point of interests; indoor location based services

1. INTRODUCTION

Places play an important role in people's daily life. People spend most of their time at significant indoor or outdoor places such as home, office, park, restaurant, shopping-mall. In previous literature, a significant place or POI has been defined as a region where a user spends significant amount of time [34, 10]. According to the coarseness of space and time dimensions, a POI can refer at different levels of granularity. For example, a user's POI can refer at a coarse level, to a university, at a finer level, to an office building at the university or at the finest level to an office room.

The rise of sensor equipped smartphones with GPS, Wi-Fi, magnetometer or barometer on board has created exciting opportunities for on- and offline monitoring of POIs visited by users in their daily life.These acquired places form the building block for many current and future mobile applications such as place *check-in* for location based social networks (LBSNs), place recommender services, predicting the next place to visit, place based personalized services and advertisements, ubiquitous healthcare and aged-care services for elderly peopleFor example, a number of location based social networking applications such as Foursquare, Google+ and Facebook Places allow users to *check-in* to a place, such as a restaurant or cinema upon arrival. Other applications, such as Instagram enable users to select a location from a list, when they share a photo. Predicting a user's true place or POI from the current geographic location and displaying them within the top few positions of a list on the device's screen, is highly desirable in such applications to improve the user's experience. Moreover, accurate prediction of the POI at the top position in the list can assist with automatic check-ins to a place [11].

Existing applications, however, can suffer from poor accuracy in retrieving and ranking the POIs due to unreliability or unavailability of the underlying localization techniques such as GPS, Wi-Fi or cell-tower based localization. Although satellite-based GPS positioning provides good coverage outdoors, it does not work well for indoor places or in urban canyons, due to poor line-of-sight transmission between the receiver and the satellites. The accuracy of cell-tower based localization is rather limited, often in the range of a few hundred meters to a few kilometers. Wi-Fi or RFID based techniques can infer indoor POIs at a higher granularity [21, 14, 17], however, the opportunity to use such techniques is dependent on the local infrastructure and may not be always available. Typically, there is also additional cost for implementation [11]. Furthermore, GPS or Wi-Fi based localization consumes high energy from a mobile device [29] and is thus unsuitable for continuous real time POI monitoring.

Many approaches have been proposed to improve the accuracy of POI ranking in LBSN scenarios, based on POI popularity, user's *check-in* histories, other user behaviour towards those POIs or closest distance from the user's geographic location [25, 26, 31]. However, these algorithms

MOBIQUITOUS '16, November 28-December 01, 2016, Hiroshima, Japan
 ISBN 978-1-4503-4750-1/16/11…$15.00
DOI: http://dx.doi.org/10.1145/2994374.2994380

still rely on accurate GPS or Wi-Fi based localization. Recently, an infrastructure-independent indoor navigation system has been developed in [33] using only geomagnetic fingerprints which exploits crowdsensing for floor map construction. However, unlike our proposed system, this system is designed for accurate navigation only at indoor environments.

In this paper, we propose a novel algorithm, *Real Time POI Identification* or RTPOI to automatically detect and predict a user's POI in real time at room level accuracy using only lightweight ambient environment sensors. The novelty of our approach is that it can infer both indoor and outdoor POIs with high accuracy, without relying on local infrastructure or without using any additional knowledge. Our method also does not require any energy hungry localization sensors and enables real time continuous POI monitoring. RTPOI is generally useful for personalized location based applications such as place-based advertisements or reminder services. It can be used for automatic retrieval of POIs in collaborative applications such as LBSN. RTPOI also caters for the growing demand of *self-quantification* and *personal analytics* [30] for place-based health monitoring and aged-care services: the visited POIs can be continuously monitored at room level to answer questions like *"how much time do elderly people spend in a kitchen or bathroom?"*.

In previous literature, a user's POI has been characterized with location context (latitude, longitude) and time dimension [19, 34, 13, 10, 11]. Most existing techniques do not work in real time and detect a place typically only at a relatively coarse level, for example at the level of a region or building. We argue that contextual information such as a user's ambient environment can help to further refine the granularity of place detection. For example, the geographical location in terms of latitude and longitude of two places located at different floors might be the same, however the barometric pressures are likely to be different due to the height difference. Some approaches have considered the use of contextual data such as ambient light, sound or magnetic field strength [9, 14]. However, they still rely on GPS/Wi-Fi based localization to recognize the POIs.

One of our key contributions is to provide a new direction of identifying POIs in terms of their ambient environments captured through previously unexplored smartphone sensors. We hypothesise that a POI is an indoor or outdoor region where a user's ambient environment remains largely unchanged (below a threshold) for a certain amount of time. We assume that each POI has a unique fingerprint of its ambient environment in terms of magnetic field strength, pressure, temperature, relative humidity and sound intensity. For example, different floors in a building typically result in different atmospheric pressures; magnetic field strength in a computer lab is likely to be different to a home environment. The sound level in a food court or cafeteria is usually louder compared to a library or office room. Some POIs might be more humid than others such as aquatic centres; even the temperature at different POIs might differ depending on a user's personal preference. In real life, however, the environment of a POI may change depending on the visit time. For example, the sound intensity of a restaurant may differ at morning and noon; the temperature of a place may vary at day and night time. We may also observe that users normally visit POIs at certain times of a day in their daily life - they stay at their offices at certain times of a day; attend seminars in other rooms at other times of a day, eat lunch at restaurants at other times, and so forth. Therefore, we also capture time of visits in addition to ambient environments to recognize POIs.

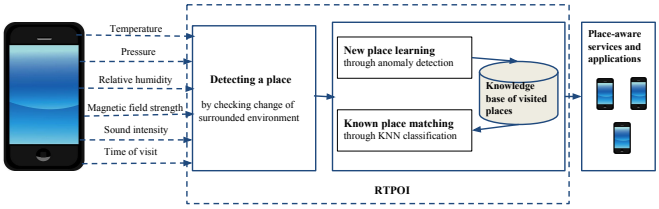


Figure 1: The system diagram of RTPOI.

Figure 1 describes the system diagram of our proposed method. Our algorithm, RTPOI, has three phases. In the first phase, the algorithm detects whether a user is at a POI through checking the variances of ambient smartphone sensor readings. The key idea is that if a user is at a POI, the variance of sensor reading should be lower due to the unchanged surrounding environment. In the second phase, the algorithm evaluates whether the detected POI is newly visited or a place that has been visited earlier. This is performed through *isolation forest* anomaly detection technique [27] by considering whether the newly visited POI is an anomaly with respect to the previously visited POIs stored in the knowledge base. If the POI is detected as newly visited, the algorithm asks for the user's feedback to provide the logical name of the POI. Otherwise, in the third phase, RTPOI recognises the POI by finding the best possible match in its knowledge base of visited POIs. A *K-nearest neighbour (KNN)* algorithm has been used to obtain the best match in terms of the ambient environment and the time of visit.

Our evaluation shows that for offline learning, the proposed algorithm can detect a user's true POIs in near real time (within 1 minute) at room level accuracy for the top 3 retrieved locations with an F1 score of approximately 80%. Using online learning with a cold-start, where the newly visited POIs are gradually added to the knowledge base, our algorithm achieves an F1 score of approximately 66.3% in identifying the true POIs at room level accuracy within the top 3 positions. The precision@1 of RTPOI at building level, is approximately 17% and 33% greater than the GPS-based solutions using *Google place search* [4] and *Foursquare venue search* [1], respectively.

Our main contributions can be summarized as follows:

- We develop an algorithm that can automatically recognize both indoor and outdoor POIs in near real time at room level accuracy using only lightweight ambient smartphone sensors.
- Our proposed algorithm RTPOI can detect indoor and outdoor user POIs in an unsupervised manner checking the spread in the ambient environment sensor data.
- RTPOI does not use any energy demanding localization techniques such as GPS or Wi-Fi, which enables continuous real time POI monitoring.
- RTPOI is infrastructure independent as it does not require any local infrastructure.
- Without any additional knowledge such as user *check-in* histories or POI popularities, RTPOI can predict

a user's true POIs in near real time (approximately within 1 minute) with high precision and recall.

2. SENSOR SELECTION

In this paper, we provide a novel method of identifying user POIs in real time in terms of ambient environments and time of visits captured through modern smartphone sensors. We selected sensors which have the potential to characterise user POIs at room level, rather than detecting the phone's motion. Thus, we avoided accelerometer, gyroscope and orientation sensors whose readings greatly vary with the phone's movement or orientation.

We use the ambient sensors (pressure, magnetometer, temperature, relative humidity and sound) available in modern smartphones except the light sensor, since in real-life the light intensity readings can greatly vary depending on how the phone is carried (in pocket, bag or by hand), or whether it is covered by some other objects such as phone cover. We compute the magnitude of the magnetic field strength (s_m) using the equation $s_m = \sqrt{m_x^2 + m_y^2 + m_z^2}$; where m_x, m_y and m_z are the magnetic field readings in x, y and z directions, respectively, as measured by the magnetometer. In order to understand which sensors can better identify a POI, we capture sensor readings at six different POIs (A - F) at room level. Four of the POIs (A - D) are located at two adjacent buildings in our university in the urban area and two POIs (E, F) are located in the outer suburbs of the city. The sensor records are manually labelled with the POI names according to the ground truth. We compute the importance of the sensors to classify the POIs in terms of *Information Gain Ratio* which is a widely used data mining technique to assess feature importance [18]. The pressure and magnetic field sensors show the highest importance (*gain ratio* of 0.773 and 0.646, respectively) and the sound sensor has the lowest value (*gain ratio* of 0.192). All these sensors together consume much less energy compared to GPS or Wi-Fi [29]. The efficiency of a sensor is judged based on the following two characteristics: 1) ability to distinguish different POIs and 2) stability of readings for a particular POI across different days.

Ability to distinguish POIs Figure 2 shows the box plots of different sensors for the six different POIs. The bottom and top of a box represent the first and third quartile of the data and the bar inside a box represents the 2nd quartile (median). The data points outside a box represent the outliers. In general, the smaller the overlap between the boxes, the higher is the ability of the sensor to distinguish the POIs. The pressure sensor has the lowest overlaps with low variance for a particular POI while the sound sensor has largest overlaps between the boxes with high variances for the POIs and thus is less effective.

Stability of sensor readings across days To analyse the stability of the sensors across multiple days for a particular POI, we capture sensor readings at a POI on two consecutive days: Day 1 and Day 2. The sensors should have similar medians for both the days in order to be considered as stable for POI recognition. Figure 3 describes the box plots of different sensors for these two days. The pressure sensor can distinguish the POIs most effectively; however, it does not have stable medians for the same POI across the two days. The relative humidity and the sound intensity sensors also have large variations in their median values for day 1 and 2. On the contrary, the magnetic field strength and temperature sensors demonstrate much more stable behaviour for the POI across the two days.

Table 1 summarises our overall findings of the sensor behaviours for identifying POIs. According to the initial experiments, none of the sensors is individually, efficient enough to 1) distinguish the POIs and also to 2) show stable readings for a POI across multiple days. Hence instead of using the sensors individually, we decide to use the collective (multimodal) sensor readings along with the *time of visit* as a basis for POI discrimination and identification.

3. REAL TIME PLACE LEARNING

In this section we describe our methodology to identify a user's POIs in real time using ambient smartphone sensors. We propose a three phase algorithm, RTPOI, to detect and predict a user's POI. A sliding window of width w is run over the observations of time-annotated sensor data to process the data in real time. In the first phase, RTPOI determines whether a user is at a POI. In the second phase, the algorithm verifies whether the current POI is newly visited or has been earlier visited by the user. If the current POI is a known place, in the third phase, RTPOI recognizes the POI name by finding the best match from its knowledge base of visited POIs. First, we provide the terminology used in this paper. This is followed by the descriptions of the three phases of the proposed algorithm.

3.1 Definitions

Ambient Sensor Record: An ambient sensor record $s = (s_m, s_p, s_t, s_h, s_s, t)$ describes the measured sensor readings at the time stamp t where s_m, s_p, s_t, s_h, and s_s are the ambient magnetic field strength, pressure, temperature, relative humidity, and sound intensity, respectively.

Place Data: A *Place Datum* L can be defined as $L = \{(s_1, t_1), (s_2, t_2) \ldots, (s_n, t_n)\}$, corresponding to a user's stay at a POI such that $\forall i \in [1, n]$, $t_i < t_{i+1}$; where s_i is the measured ambient environment record at time stamp t_i.

Knowledge base of Place Data: A knowledge base K of place data is a set of m previously visited places such that $K = (\{L_1, PN_1\}, \{L_2, PN_2\}, \ldots, \{L_m, PN_m\})$, where $\forall i \in [1, m], L_i$ and PN_i are the ith place datum and its user-given place name, respectively.

3.2 Phase 1: Is it a place/POI?

POI detection is performed by using the *sensor variance* over the current time window of size w. The key idea is that when a user is at a POI, the surrounding environment is likely to remain largely the same compared to movements between POIs. We consider the magnetic field, pressure, temperature and relative humidity data of the current ambient environment. The sound intensity is not used for POI detection since it appears to be less informative to differentiate a user's *at-a-place* versus *not-at-a-place* behaviour. For example, the sound intensity of a crowded path and a coffee shop can be quite similar. Figure 4a, 4b, 4c and 4d describe the real life sensor readings for a user's stay at different nearby POIs during a day. The ground truth of the user's stay has been described in Table 2: the variance of sensor data is much smaller when the user is staying at a POI compared to moving from one POI to another.

Let $S = s_{(k+1)}, s_{(k+2)}, \ldots s_{(k+w)}$ represent the ambient sensor records (see Subsection 3.1) over the current window of w observations such that $\forall i \in [1, w], s_{(k+i)}$ is the mea-

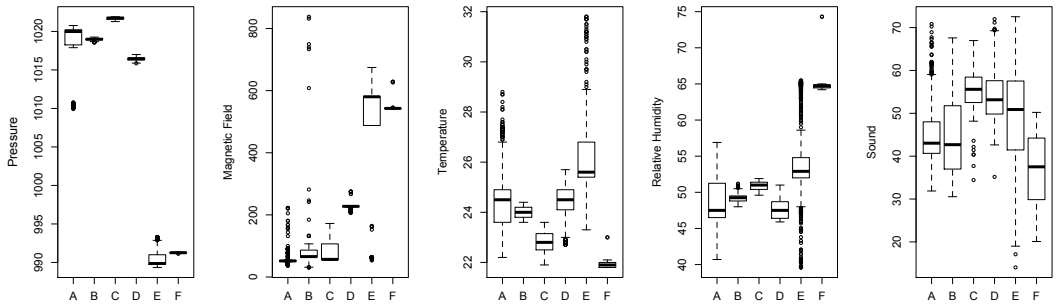


Figure 2: The behaviour of the sensors at six different POIs: A, B, C, D, E and F. The lesser the overlaps between the boxes, the higher is the ability of the sensor to distinguish the POIs. The pressure sensor has the lowest and sound sensor has the highest overlaps.

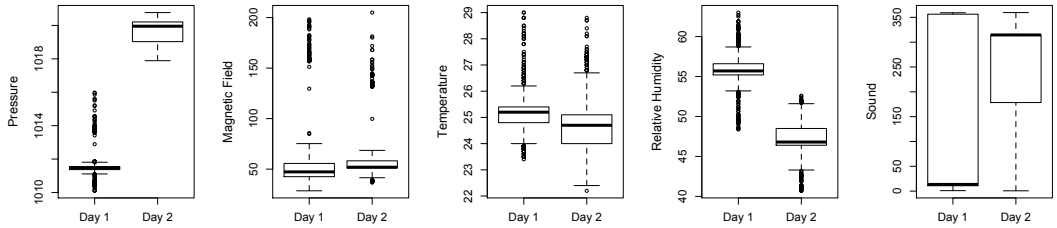


Figure 3: The behaviour of the sensors at a POI for two different days. The magnetic field strength shows the most stable behaviour over the days, with a low variance for each day. The temperature is quite stable over the days. However, the barometric pressure, relative humidity and the sound intensity vary largely from Day 1 and Day 2.

sured ambient sensor record at time-stamp $t_{(k+i)}$. Ignoring the time dimension, the sensor data set S can be viewed as a $w \times 4$ matrix representing the sensor records over the current window. To detect whether a user is at a POI, we compute the 4×4 covariance matrix of S that captures the spread of the multivariate data in 4 dimensions using the formula:

$$Cov(S) = (1/w - 1)S'S \tag{1}$$

where S' is the transpose of matrix S. We compute the determinant of $Cov(S)$ representing the magnitude of the spread in the multivariate sensor data within the current window. Therefore, RTPOI continuously checks whether the det $Cov(S)$ for the current window is below a threshold, S_{th}, to detect if the user is at a POI. The performance of the place detection phase depends on the parameters w and S_{th} and their choice. The following section details the selection procedure.

Table 2: A user's ground truth: each record describes the arrival time and the name of the POI

Arrival time	Name of a user's POI
10:50 am	My office x at building A
1:02 pm	Going out for lunch in cafeteria
1:17 pm	Back to my office
1:37 pm	Meeting room y at building A
2:59 pm	Computer lab at a nearby building B

Selecting the parameters, w and S_{th}: The *place detection* (phase 1) performance of RTPOI depends on two parameters: 1) width of the sliding window (w) and 2) the threshold value of the determinant of covariance of sensor matrix (S_{th}). To select proper parameter values, we perform an experiment varying these two parameters on 5 days of training data collected for a user, who was visiting different POIs within our university campus. The optimal parameter values have been selected based on the performance (precision and recall as described in the section 4.2) and have been used for all other experiments. Figure 5 (left hand side) describes the place detection performance of RTPOI with varying width of the sliding window. In this experiment, the S_{th} value has been kept as 2.0 and the window width has been varied from 5 to 30 observations. Since the data has been collected at a sampling rate of every 10 seconds, the window widths of 5 and 30 observations correspond to approximately 1 and 5 minutes, respectively. After $w = 20$ observations, precision and recall do not improve significantly. We select w as 20 observations (approximately 3 minutes), since increasing the window size further may cause missing of POI visits of smaller duration such as cafeteria visits.

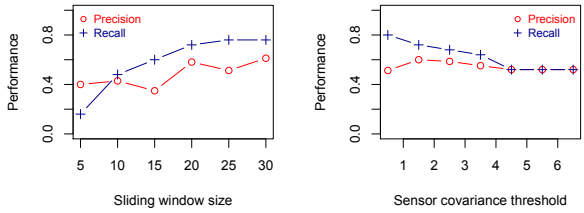
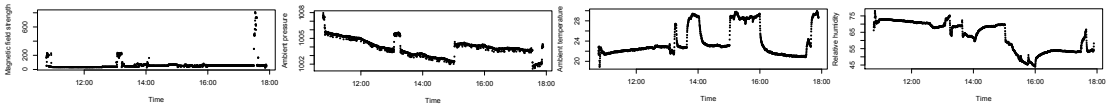


Figure 5: The performance of *place detection* phase with varying window size and covariance threshold.

The right hand side of Figure 5 describes the performance of RTPOI with varying S_{th} parameter. In this case, the width of the sliding window has been kept as $w = 20$ observations and the value of the S_{th} parameter has been varied from 0.5 to 6.5. We find that the precision and recall of place detection do not change after $S_{th} = 4.5$. The recall is the

Table 1: Overall capability of the ambient smartphone sensors to identify POIs.

Sensor	Drift behaviour	Ability to distinguish POIs	Stability for a POI across days
Pressure	Very small gradual drift, no sudden change	Good	Poor
Magnetometer	No gradual drift, can suffer with sudden drift due to presence of strong nearby magnetic objects	Moderate - Good	Good
Temperature	Small - medium gradual drift	Moderate	Moderate
Relative humidity	Small - medium gradual drift	Moderate	Low
Sound	Sudden drift due large sudden noise	Low	Moderate - Low



(a) Magnetic field strength (b) Ambient pressure (c) Ambient temperature (d) Relative humidity

Figure 4: Raw ambient sensor readings over time for the ground truth described in Table 2.

highest at $S_{th} = 0.5$, however the precision is very low at this threshold. This is due to the fact that with a low covariance threshold, the opportunity of wrongly detecting the change of the surrounding environment as a POI, increases. This results in the decrease of the precision value. The best overall performance is achieved for $1.5 \leq S_{th} < 2.5$, after that both precision and recall decrease. Hence, we select $S_{th} = 2.0$ for all other experiments.

3.3 Phase 2: Is it a newly visited place/POI?

In the second phase, RTPOI determines whether the detected POI in the first phase is newly visited or has been visited by the user earlier. One of the novelties of our algorithm is that it treats the newly visited places as anomalies with respect to the *place data* stored in the knowledge base. We use an existing anomaly detection technique, called *isolation forest* (see below), since it is efficient for high dimension data and has linear time complexity and low memory requirement [27]. The *isolation forest* technique and the procedure for selecting the features and detecting the anomalies are described in the following paragraphs.

Isolation forest anomaly detection.

Most existing anomaly detection technique detect anomalies by creating profiles for normal data points. However, *Isolation Forest* or *iForest* explicitly isolate the anomalies rather than profiling the normal data points. As described in [27], it works on the principle that the anomalies are 'few and different' compared to the normal data points and are more susceptible to isolation. Thus, if a tree structure is created for every single data instance, the anomalies will be isolated closer to the root of the tree, thus having smaller path lengths compared to the normal instances. *iForest* creates a set of such trees called *Isolation Tree* or *iTree* for the entire data set. The anomalies are those which have small average path lengths across the set of *iTrees*. The algorithm has two parameters: i) the number the trees and ii) the subsampling size. As demonstrated in [27], *iForest's* anomaly detection performance converges very quickly with a very small number of trees.

Feature selection, learning and anomaly detection.

Note that *isolation forests* do not consider the time dependencies of the data points which naturally exists in our sensor data. For example, if a sensor record at time stamp t is detected by the algorithm as an anomaly (i.e. a *NEWPLACE*) with respect to the training data in the knowledge base, the sensor record at the next time stamp $(t+1)$ should also be an anomaly or *NEWPLACE*, assuming the sampling frequency of collecting the sensor data is not very large. To incorporate this time dependency, we select the sensor record at time t and time $(t+1)$ as features to train the isolation forest model. Formally, each training record (r) in the knowledge base is defined in the following way:

$$r =< s_t, s_{t-1} > \quad (2)$$

where s_t and s_{t-1} are the multimodal *ambient sensor records* (as defined in section 3.1) at time t and time $t-1$, respectively. Our initial experiments showed that the performance of the algorithm does not improve by increasing the lags further, for example considering each record r as $r =< s_t, s_{t-1}, s_{t-2}, ...s_{t-m} >$ with m lags, where m is a positive integer.

During the learning or training phase (both offline and online training), the *isolation forest* model is trained using the training POI data stored in the knowledge base. This produces anomaly scores for all the sensor records stored in the knowledge base. The sensor records with anomaly scores greater than a_{th} are then considered as anomalies.

To select a_{th}, we compute the mean and standard deviation of the anomaly scores of the training records and plot the corresponding Gaussian curve. We use quantile function which is the inverse cumulative distribution function that returns the minimum value of anomaly score among all the anomaly scores whose c.d.f exceeds the probability $p > 0.999$. We chose a very high probability region to cover almost all the data points since we know that all the sensor records stored in the knowledge base are collected at the known POIs and they should be regarded as normal instances rather than anomalies or new POIs.

During online detection of POIs (testing phase), the sensor records in the current window are arranged as per equation 2 and input to the *isolation forest* anomaly detection technique which computes an anomaly score for each of the sensor records. If the anomaly score of a sensor record is greater than a_{th}, RTPOI labels the record with the constant *NEWPLACE* ,otherwise it goes to *Phase 3* to label the record with a name of the previously visited POIs stored in the knowledge base.

3.4 Phase 3: Labelling a known POI

In *Phase 2* if a sensor record in the current window has not

been labelled as *NEWPLACE*, we assume that the record must have been obtained from one of the existing POIs in the knowledge base K, previously visited by the user. In that case, we implement a *kNN* algorithm to find the best possible match of the current *sensor record* with the labelled place records in K. We use 10-nearest neighbour since RTPOI performs effectively with that value as seen in our initial experiments. Euclidean distance has been used to compute the similarity between the current sensor record and the place records in the knowledge base. The smaller the distance is, the more similar are the two records. We scale both the training and test data between 0 to 1 before implementing the *kNN* algorithm to normalise the ranges of the sensor features so that the distance computation is not affected by features with varying ranges. The final POI name for the sensor record is obtained according to a majority vote of the 10-nearest neighbour algorithm.

3.5 Computing the final probability of POIs

At the end of phase 3, each sensor record S_j in the current window of size w has a POI name either as *NEWPLACE* or as obtained from the 10-nearest neighbor algorithm. The final probability of a POI named 'L' is given by counting the number of records in the current window labelled as 'L', divided by the number of observations (w) in that window. For the current window, the labelled places are ranked in descending order of their probabilities and ranked. For online or cold-start learning, the sensor data for a *NEWPLACE* will be automatically added to the knowledge base with the user given place name so that the system can recognise the POI subsequently.

4. EXPERIMENTAL EVALUATION

We have evaluated our algorithm on a real user's data collected for a selection of times during a three month period. Publicly available datasets such as [2, 6] generally do not contain ground-truth of user POIs. In addition, ambient sensors such as pressure, temperature and relative humidity sensors are only recently available for specific smartphone models such as Samsung Galaxy S4. These limitations led us to collect our own data using an Android application [5] on an Android mobile phone (Samsung Galaxy S4). The collected data was sampled every 10 seconds for 20 days (consisting of 44,303 records) at our university campus and other places such as tram stops, train stations and food courts. The university is located near the city's centre and has densely populated nearby buildings. The user carried the smartphone naturally while visiting different indoor or outdoor places during the day such as user's office room, meeting room, computer lab, seminar rooms, outdoor cafeteria, food-court, tram stops and train stations. We evaluated the performance of our algorithm for both offline and online learning.

The user manually recorded the ground truth time by entering a POI and its name. The user's annotations created a ground truth dataset where each record contains a POI ID (or name), address and the time entered into that POI. While validating the result, for each record in the ground truth data set we check if the ground truth time lies within the time windows of any record in the result set. Since the user manually recorded the time, there is a small possibility of entry errors in the time annotations. We allow an error time of 3 mins to handle these situations.

4.1 Baseline techniques

We compare the performance RTPOI against against two POI search services, *Foursquare venue search* [1] and the *Google near-by place search*, widely used in LBSN applications. These baseline techniques, however, do not support automatic POI detection and require manual initiation. For automatic POI detection, we use the first phase of RTPOI to detect a POI in real time. Once a place has been detected, the baseline techniques were then applied to the mean of the GPS observations in the current window to retrieve and rank the POIs. The POIs stored in the RTPOI's knowledge base (as shown in Table 3) were also found in the Foursquare or Google's place databases, but only at the building level. Hence we could compare their performance against RTPOI only at building level accuracy. All other POIs retrieved by calling Foursquare venue search API or Google near-by place search API, which were not in RTPOI's knowledge base, were treated as *NEWPLACE*.

The *Google near-by place search* provides two types of near-by POI search. One version retrieves and ranks the POIs based on the distance from the current geographic location. Some additional parameters, such as POI types can be provided to this API to refine the search. The other version considers parameters such as POI popularity, number of other users' visits, along with the user's current geographic location to rank the near-by POIs. Using the second version, no meaningful POIs were retrieved that corresponded to the user's routinely visited places in RTPOI's knowledge base. Hence we use the first version of the place search API with the Google place types as *university* or *cafe* or *food* as parameters; since only those place types were available.

We also compare our algorithm against our own constructed baseline techniques - a *random ranking* algorithm and a *Distance-based ranking* algorithm. In case of *random ranking* method, once the user is detected to be in a place, the algorithm makes three consecutive random guesses about the name of the POI from the set of the POIs stored the knowledge base (K) or as *NEWPLACE*. Therefore, if there are m number of POIs stored in K, then the probability of selecting a particular POI at rank 1 is $\frac{1}{m+1}$.

Similar to the Google's rank-by-distance place search [4], the *Distance-based ranking* algorithm ranks the POIs based on their distance from the user's current geographic location, the mean latitude-longitude of the observations in the current window. However, here we create our own POI database at building levels using Google Maps [3] from the POIs stored in the knowledge base. This is to ensure that all the visited POIs in RTPOI's knowledge base are also in the POI database.

4.2 Evaluation Metrics

We evaluate the performance of the proposed algorithm and the baseline techniques in retrieving the user's true POIs within the top 1, top 2 and top 3 retrieved places. The performance has been measured in terms of precision, recall and F1 score which are as follows: $Precision = TP/(TP + FP)$, $Recall = TP/(TP + FN)$ and $F1 = 2 * \frac{Precision*Recall}{Precision+Recall}$. In this context, TP (True Positive) means the number of retrieved POIs which are in the ground truth, FP (False Positive) means the number of retrieved POIs which are not in the ground truth and FN (False Negative) means the number of ground truth POIs which are not retrieved.

4.3 Discussion

RTPOI has three phases: 1) detecting a place, 2) recognizing a new place, and 3) labelling or identifying an old place. We compute the precision, recall and F1 score for each phase to understand their performance. Below we discuss the results for both offline and online (cold-start) learning of our algorithm. The algorithm runs over a sliding window of size $w = 20$ observations and S_{th} has been kept as 2.0.

4.3.1 Offline learning

In case of offline learning, the initial 5 days of data consisting of 8,636 sensor records have been stored in the knowledge base acting as the training set. The remaining 15 days of data (35,667 records) have been used for testing. Each day of test data contains different *place data*, as defined in the section 3.1, as well as data reflecting the user's transition path from one POI to another. The POIs in the test data are either known places which are in the knowledge base or newly visited places by the user. Table 3 describes the seven different POIs stored in the knowledge base. Note that, three different POIs are located at the same building (*Building 1*) at different rooms at the university. Among them, the user's *office* and *meeting room 1* are located on the sixth and seventh floors whereas the seminar room is located at the tenth floor. The *cafeteria* is an outdoor POI in the ground floor of *Building 2* which is adjacent (within 20 meters) to the user's office building. The POIs E and F are located in different buildings within the university campus. They are nearby (within 300 meters) to *Building 1*, but not adjacent. POI G is a train station located in the central business district which is approximately 1.5 km away from the campus. The ground-truth dataset contains 79 records corresponding to the user's visits at different known and newly visited POIs. Among these, 22 visits are at some new places which are not in the knowledge base and the rest are at the known places, with a probability of visiting a new POI of 27.8%.

Table 3: User's POIs stored in the knowledge base for offline learning. The user spent of the time in office room *POI A*.

POI	Purpose	Floor	Building	# of visits
A	User's office	Level 6	Building 1	13
B	Meeting room	level 7	Building 1	2
C	Cafeteria	Ground floor	Building 2	4
D	Seminar room	level 10	Building 1	1
E	Admin office	level 4	Building 3	1
F	Admin office	Ground floor	Building 4	1
G	Train station	Ground floor	Building 5	1

Table 4 describes the performance of RTPOI to identify the user's true POIs at room level accuracy at different rank cut-offs of the retrieved lists of POIs. Note that, the place detection (*Phase 1*) performance is independent of the rank cut-offs. The place detection stage of our algorithm achieves a high recall of 94.9%, which means it can detect 94.9% of the POIs which are in the ground truth dataset. However, it has a low precision as it detects many other POIs which are not in the ground truth. This is because, the test dataset contains not only the data representing the visited POIs, but also the sensor records representing the transition paths from one POI to another. This includes the user's data collected in tram, train, or car. Many additional POIs have been detected by RTPOI, such as waiting at traffic signals or at tram-stops, which are not included in the user's ground truth. However, with an ideal ground truth annotation, we can expect a much improved precision for this stage.

Table 4: Performance of the three phases of *RTPOI* to identify true POIs at room level accuracy at different rank cut-offs (offline learning).

Phase	Rank cut-off	Precision	Recall	F1 score
Phase 1: Is it a POI?		0.610	**0.949**	0.743
Phase 2: Is it a new POI?	top 1	0.429	0.353	0.387
	top 2	0.714	0.714	0.714
	top 3	**0.792**	**0.864**	**0.826**
Phase 3: Which POI?	top 1	0.689	0.544	0.608
	top 2	0.809	0.667	0.731
	top 3	**0.875**	**0.737**	**0.800**
Overall performance	top 3	**0.813**	**0.772**	**0.792**

Table 4 describes the performance of RTPOI to identify new POIs (*Phase 2*) and recognizing the known POIs (*Phase 3*) in our data set. The proposed algorithm achieves high precision (79.2%), recall (86.4%) and F1 score (82.6%) in terms of detecting a new POI within the top 3 positions. Recognizing the known POIs within the top 2 and top 3 positions, the algorithm achieves high precision (above 80%) with F1 scores of 73.1% and 80%, respectively.

4.3.2 Online or cold-start learning

We also perform experiments to anaylse the performance of RTPOI when POIs are learned online with a cold-start approach. In this case, we assume that initially no data is stored in the knowledge base. As the user visits different POIs, the sensor records for those POIs are gradually added to the knowledge base, if they are newly visited. Otherwise, the algorithm recognises the POIs based on the data stored in the knowledge base so far. Therefore, for online learning, the entire 20 days of data consisting of 44,303 sensor records are used for testing.

Table 5 describes the overall performance of the three phases of RTPOI for online learning. The performance is computed to identify the true POIs at room level accuracy within the top 3 positions. The POI detection phase (*Phase 1*) achieves a high recall of 89.4%, but a moderate precision of 60.4%, since many stops in the user's transition path (e.g waiting at traffic signals, trams stops) which are not in the ground truth dataset, are also retrieved by our algorithm. The *new POI detection phase (Phase 2)* achieves a very high recall of 92.3%, which means most of the new POIs in the ground truth dataset are detected by our algorithm. However, it achieves a low precision of 50% with cold-start approach compared to a high precision of approximately 80% when learning offline. This is because, initially, the algorithm does not have enough training data in the knowledge base to obtain accurate clusters of data points for the known POIs. Therefore, some known (previously visited) POIs have been recognized as new POIs (or anomalies) producing high false positives and hence low precision. For the same reason, comparing Table 4 and 5, the performance of *Phase 3* to recognise the known POIs is also poor for cold-start approach than when learned offline. However, the performance can be expected to improve gradually as more POIs are visited and added to the knowledge base.

To analyse the gradual performance of RTPOI, we compute the individual day's performance in terms of accuracy starting from day 1 to day 20. For a single day, the precision, recall or F1 score may have unrealistic values if there is no new (or known) POI visited by the user for that day. For example, initially, on day 1, as there is no ground truth

Table 5: Performance of the three phases of RTPOI to identify true POIs at room level within the top 3 positions for online learning.

Phase	Precision	Recall	F1 score
Phase 1: Is it a POI?	0.604	**0.894**	0.721
Phase 2: Is it a new POI?	0.500	0.923	0.649
Phase 3: Which POI?	0.791	0.609	0.688

records for the known POIs stored in the knowledge base, the known POI recognition phase (*Phase 3*) will produce unrealistic values for those metrics. Similarly, for the days when the user visits only the known POIs, the new POI detection phase (*Phase 2*) will produce unrealistic precision, recall or F1 score values since the number of *True Negatives* are not considered in those metrics. On the contrary, accuracy is measured, considering both *True Positives* and *True Negatives*, as $(TP + TN)/(TP + FP + TN + FN)$; where TP, TN, FP, FN are the number of *True Positives*, *False Positives*, *True Negatives* and *False Negatives*, respectively. Figure 6 describes the accuracy of the algorithm for each day for the newly visited and known POI recognition. The blue straight lines are the computed regression lines for those accuracy values showing the general trends in the results. As expected, the algorithm's accuracy progressively improves over the days for new POI detection (*Phase 2*) as more data are added to the knowledge base. For known POI recognition (*Phase 3*), the accuracy is slightly expected to be improved over days, but not as much as for new POI detection. Specifically, the accuracy values are surprisingly low for known POI recognition on day 19 and day 20. Analysing the reason behind this, we find that the user's office room produced a significantly large magnetic field strength (more than 700 μT for day 19 and approximately 150 μT for day 20) compared to the median value of 54.17 μT for those days. This could be due to placement of the phone adjacent to the user's key or some other materials which suddenly raised the surrounding magnetic field. As a result, RTPOI was not able to correctly recognise the POI, thus obtaining lower accuracy. Such sudden drifts might be handled by imposing a weighted confidence on the outcomes of individual sensors. For example, although the magnetic field strength at user's office room suddenly increases on those days, the temperature and barometric pressure remain the same. Therefore, putting more weight on these sensors might help to eliminate the drifts in magnetic field strength. We plan to investigate this in future work.

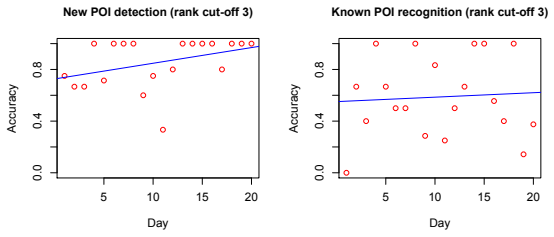


Figure 6: The gradual performance of RTPOI over days with a cold-start approach. The blue straight lines are the regression lines showing the general improvements in accuracy.

4.3.3 RTPOI vs. the baseline techniques

We compare the overall performance of RTPOI, using offline learning, against the *Random Ranking* algorithm as a baseline to recognize the POIs accurately at room level accuracy. The performance of *Random Ranking* algorithm is computed as the average of 10 runs. The Foursquare, Google's Place and our own constructed POI database for *Distance-based ranking* algorithm contain the POIs of our university only at the building levels. Hence, we could compare only the building level performance of RTPOI against these baseline techniques. Note that since there are only five buildings stored in the knowledge base, hence *Random Ranking* algorithm can easily guess the correct POI at building level and may look artificially better than other techniques such as *Distance-based ranking*. However, as more new buildings will be visited and added to the knowledge base, the performance of *random algorithm* will suffer. Hence we did not compare the building level performance of this algorithm against the other techniques.

Table 6 describes the performance of RTPOI against the *Random Ranking* algorithm in recognizing the user's POIs at room level accuracy. RTPOI clearly outperforms the baseline technique in terms of all the performance metrics for all rank cut-offs. Our algorithm achieves a room level improvement, in terms of F1 score, of approximately 36% and 40% in recognizing the user's true POIs at the top 1 and within the top 3 positions, respectively. The room level performance of RTPOI and the *Random Ranking* algorithm in terms of precision@1 are 52.9% and (13.6 ± 3)%, respectively.

Table 6: Overall performance of *RTPOI* with offline learning, vs. *Random Rank* baseline technique to identify true POIs at room level accuracy at different rank cut-offs

Algorithm	Rank cut-off	Precision	Recall	F1 score
Random Ranking	top 1	0.136 ± 0.03	0.125 ± 0.03	0.136 ± 0.03
	top 2	0.295 ± 0.06	0.279 ± 0.06	0.282 ± 0.06
	top 3	0.408 ± 0.04	0.387 ± 0.04	0.397 ± 0.04
RTPOI	top 1	0.529	0.468	0.497
	top 2	0.716	0.671	0.693
	top 3	**0.813**	**0.772**	**0.792**

Figure 7 summarises the overall performance of RTPOI (with offline learning) and the baseline techniques in recognizing the user's POIs at building level accuracy. The *Foursquare venue search* has poor performance compared to *Google Place Search* and RTPOI in recognizing and retrieving user's POIs at the first position of the ranked places. The precision@1 and precision @3 of RTPOI are 60% whereas the same for *Foursquare venue search* is only 27.4% and *Google Place Search* is 42.6%. Out algorithm achieves a precision @3 of 85.3% and recall @3 of 81%. This result suggests that RTPOI can be a better option compared to the existing techniques for automatic place *check-in*s in LBSNs. It is interesting to see that *Google place search* has better performance than *Foursquare venue search* with our dataset, although all the old places visited by the user also exist in the Foursquare database. This could be due to the consideration of additional parameters such as place types by *Google rank by distance place search* technique in retrieving and ranking the near-by POIs. The baseline *Distance based ranking* technique has the poorest performance compared to all other techniques at rank cut-off 1 with a precision @1 as 17.3%. The performance, however, improves for rank cut-off 2 and 3 since there are only five buildings in the POI database. However, with the increase of the number POIs, we can expect much lower performance of this technique compared to

the proposed algorithm.

We also compute the time performance of RTPOI in terms of latency in recognizing POIs. For each ground truth POI detected by the algorithm, the latency is computed as the time difference between the detected time and the corresponding ground truth arrival time. The mean and median value of latency for all the detected POIs are 1.16 and 0.733 minutes, respectively. This reflects RTPOI can detect most of the POIs in near real time, within 1 minute.

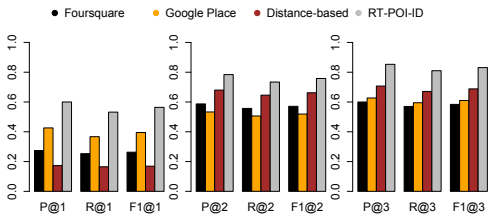


Figure 7: Precision (P@k), recall (R@k) and F1 score (F1@k) of RTPOI and the baseline techniques to accurately identify user's true POIs at building level accuracy within the top k retrieved places. RTPOI outperforms the baseline techniques for all rank cut-offs.

5. RELATED WORK

A set of existing works rely on GSM or GPS based localization to extract significant places from user trajectories [8, 7, 28, 34, 13, 10, 32, 11]. These methods typically work offline and can extract POIs at a much coarser level, for example at the level of a building or a region.

Another set of literature use fingerprinting based technique to learn places more accurately in real time. Traditionally these methods use cell towers, RFID or Wi-Fi access points. For example, Laasonen et al. [24] proposed to learn significant places based on the currently connected cell towers. They clustered the cell towers offline and identified the clusters as places with a time duration greater than a threshold value. In [23], Krumm et al. inferred the transition of states from "still" to "moving", by examining the variance of the strongest Wi-Fi access points and using an HMM based model. *BeaconPrint* [20] and *PlaceSense* [21] algorithms were proposed to automaically learn places in real time by continuously monitoring the radio environment to look for new *beacons* around a mobile device. Kim et al. used a combination of Wi-Fi, accelerometer and GPS data to predict a user's places, movements and paths, respectively [22]. In [12], Brouwers et al. performed a comparative study using GPS, Wi-Fi and Geolocation sensors to detect a mobile user's dwelling location in an urban canyon.

There are some recent works which also explore motion, sound, light or camera sensors available in modern smartphones to fingerprint a place. Our proposed method can be considered as a fingerprinting approach. However, unlike the existing approaches, our technique does not rely on GPS, GSM or Wi-Fi based localization. Some recent studies such as Chung et al. [15] have proposed magnetic field-based fingerprinting technique for accurate indoor localization, achieving $\approx$ 1 m accuracy without using GPS or Wi-Fi. However, unlike these methods, we use everyday smartphone sensors rather than dedicated sensors to fingerprint a place at room level accuracy. *SurroundSense* algorithm has been proposed in [9] to identify a user's place at room level accuracy using ambient fingerprinting. Light, color, sound, accelerometer and Wi-Fi data were used to fingerprint a POI. The LifeMap system was proposed by Chon et al[14] to identify a user's POI using inertial sensors, Wi-Fi access points and GPS or cell tower based locations at room level accuracy. Recently, Ficco et al. has proposed a hybrid system to infer a user's position by combining signal strength based fingerprinting technique for indoor positioning and opportunistically switching to GPS for outdoor environments [17]. However, this technique still relies on local indoor infrastructure to support Wi-Fi based fingerprinting, which may not be available everywhere. Our work is closely related to [9, 14] as we also discover and learn visited places in real time at room level accuracy. However, we use new ambient smartphone sensors such as temperature, pressure and relative humidity. In addition, our algorithm does not rely on GPS or Wi-Fi based localization to recognize POIs in real time.

Another set of existing works aim to map a user's geographic location (latitude and longitude) to a POI for place *check-ins* in LBSNs such as Foursquare. These techniques explored additional knowledge about the POIs (such as POI popularities) and other users' *check-in* behaviours to predict user POIs in real time [25, 26, 31]. Recently, a system called *CheckInside* was proposed to identify a user's POI by opportunistically crowd-sensing the mobile sensors data during users' *check-in* operations and using the knowledge already stored in current LBSNs such as popularities of POIs [16]. Our algorithm can be extended to use for personalized place *check-in*s in location based social networks without requiring additional knowledge about users or POIs.

6. CONCLUSION

In this paper, we have described a novel algorithm, RTPOI, to automatically detect and predict a user's POI in real time using lightweight ambient smartphone sensors. Without using any energy-hungry localization sensors such as GPS or Wi-Fi, our proposed algorithm can detect a user's visit to a POI in real time within approximately 1 minute. It can also identify a new POI or can recognise a known POI by finding the best match from its knowledge base of visited POIs. Our evaluation shows that our algorithm can recognize a user's true POI within the top 3 retrieved locations with an F1 score of 79.2% for room level prediction and more than 83% for building level prediction. It also outperforms existing techniques such as *Foursquare venue search* and *Google Place Search* and other competitive baseline techniques.

RTPOI recognizes previously visited POIs by finding the best match of the surrounding environment from its knowledge base of visited POIs. However, in real life, the ambient environment of a POI may change over time, thus causing a change in the concept that describes the POI. For example, the temperature of a POI can be different in summer and winter or the magnetic field strength of a POI can vary due to the addition of electronic instruments. Thus, the sensor data representing the POIs in the knowledge base might need to be changed periodically to work in a changed environment. RTPOI does not consider these changes yet and they are an interesting area for future research.

7. REFERENCES

[1] Foursquare for developers: Search venues. https://developer.foursquare.com/docs/venues/search. Accessed: 07/11/2014.

[2] Geolife GPS trajectories. http://research.microsoft.com/en-us/downloads/b16d359d-d164-469e-9fd4-daa38f2b2e13/. Accessed: 03/10/2013.

[3] Google maps api v2 geocoder tool. http://gmaps-samples.googlecode.com/svn/trunk/geocoder/v2-geocoder-tool.html. Accessed: 03/10/2013.

[4] Google places api: Place search - near-by search request. https://developers.google.com/places/documentation/search\#PlaceSearchRequests. Accessed: 07/11/2014.

[5] Google play: Androsensor. https://play.google.com/store/apps/details?id=com.fivasim.androsensor\&hl=en. Accessed: 07/11/2014.

[6] GPS share. http://www.gpsshare.com. Accessed: 03/10/2013.

[7] L. O. Alvares, V. Bogorny, B. Kuijpers, J. A. F. de Macedo, B. Moelans, and A. Vaisman. A model for enriching trajectories with semantic geographical information. In *Proceedings of the 15th Annual ACM International Symposium on Advances in Geographic Information Systems*, GIS '07, New York, NY, USA, 2007. ACM Press.

[8] D. Ashbrook and T. Starner. Using GPS to learn significant locations and predict movement across multiple users. *Personal Ubiquitous Comput.*, 7(5):275–286, October 2003.

[9] M. Azizyan, I. Constandache, and R. Roy Choudhury. Surroundsense: mobile phone localization via ambience fingerprinting. In *Proceedings of the 15th annual international conference on Mobile computing and networking*, pages 261–272. ACM, 2009.

[10] T. Bhattacharya, L. Kulik, and J. Bailey. Extracting significant places from mobile user GPS trajectories: A bearing change based approach. In *Proceedings of the 20th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, GIS '12, November 2012.

[11] T. Bhattacharya, L. Kulik, and J. Bailey. Automatically recognizing places of interest from unreliable GPS data using spatio-temporal density estimation and line intersections. *Pervasive and Mobile Computing*, 19:86–107, 2015.

[12] N. Brouwers and M. Woehrle. Dwelling in the canyons: Dwelling detection in urban environments using GPS, wi-fi, and geolocation. *Pervasive and Mobile Computing (Special issue on Pervasive Urban Applications)*, 9(5):665–680, October 2013.

[13] X. Cao, G. Cong, and C. S. Jensen. Mining significant semantic locations from GPS data. *Proceedings of the VLDB Endowment*, 3(1-2):1009–1020, September 2010.

[14] Y. Chon, E. Talipov, and H. Cha. Autonomous management of everyday places for a personalized location provider. *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on*, 42(4):518–531, 2012.

[15] J. Chung, M. Donahoe, C. Schmandt, I.-J. Kim, P. Razavai, and M. Wiseman. Indoor location sensing using geo-magnetism. In *Proceedings of the 9th international conference on Mobile systems, applications, and services*, pages 141–154. ACM, 2011.

[16] M. Elhamshary and M. Youssef. Checkinside: a fine-grained indoor location-based social network. In *Proceedings of the 2014 ACM International Joint Conference on Pervasive and Ubiquitous Computing*, pages 607–618. ACM, 2014.

[17] M. Ficco, F. Palmieri, and A. Castiglione. Hybrid indoor and outdoor location services for new generation mobile terminals. *Personal and ubiquitous computing*, 18(2):271–285, 2014.

[18] J. Han, M. Kamber, and J. Pei. *Data mining: concepts and techniques: concepts and techniques*. Elsevier, 2011.

[19] R. Hariharan and K. Toyama. Project lachesis: parsing and modeling location histories. In *Geographic Information Science*, pages 106–124. Springer, 2004.

[20] J. Hightower, S. Consolvo, A. LaMarca, I. Smith, and J. Hughes. Learning and recognizing the places we go. In M. Beigl, S. Intille, J. Rekimoto, and H. Tokuda, editors, *UbiComp 2005: Ubiquitous Computing*, volume 3660 of *Lecture Notes in Computer Science*, pages 159–176. Springer Berlin Heidelberg, 2005.

[21] D. H. Kim, J. Hightower, R. Govindan, and D. Estrin. Discovering semantically meaningful places from pervasive RF-beacons. In *Proceedings of the 11th International Conference on Ubiquitous Computing*, Ubicomp '09, pages 21–30, New York, NY, USA, 2009. ACM.

[22] D. H. Kim, Y. Kim, D. Estrin, and M. B. Srivastava. Sensloc: sensing everyday places and paths using less energy. In *Proceedings of the 8th ACM Conference on Embedded Networked Sensor Systems*, pages 43–56. ACM, 2010.

[23] J. Krumm and E. Horvitz. Locadio: Inferring motion and location from wi-fi signal strengths. In *in First Annual International Conference on Mobile and Ubiquitous Systems: Networking and Services*, Mobiquitous, 2004.

[24] K. Laasonen, M. Raento, and H. Toivonen. Adaptive on-device location recognition. In A. Ferscha and F. Mattern, editors, *Pervasive Computing*, volume 3001 of *Lecture Notes in Computer Science*, pages 287–304. Springer Berlin Heidelberg, 2004.

[25] N. D. Lane, D. Lymberopoulos, F. Zhao, and A. T. Campbell. Hapori: context-based local search for mobile phones using community behavioral modeling and similarity. In *Proceedings of the 12th ACM international conference on Ubiquitous computing*, pages 109–118. ACM, 2010.

[26] D. Lian and X. Xie. Learning location naming from user check-in histories. In *Proceedings of the 19th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, pages 112–121. ACM, 2011.

[27] F. T. Liu, K. M. Ting, and Z.-H. Zhou. Isolation forest. In *Data Mining, 2008. ICDM'08. Eighth IEEE International Conference on*, pages 413–422. IEEE, 2008.

[28] A. T. Palma, V. Bogorny, B. Kuijpers, and L. O. Alvares. A clustering-based approach for discovering interesting places in trajectories. In *Proceedings of the 2008 ACM Symposium on Applied Computing*, SAC '08, pages 863–868, New York, NY, USA, 2008. ACM Press.

[29] V. Radu, P. Katsikouli, R. Sarkar, and M. K. Marina. A semi-supervised learning approach for robust indoor-outdoor detection with smartphones. In *to appear) Proceeding of The 12th ACM Conference on Embedded Networked Sensor Systems (SenSys)*, 2014.

[30] A. Regalado. A business report on big data gets personal. *MIT Technology Review*, 2013. Accessed: 10/08/2015.

[31] B. Shaw, J. Shea, S. Sinha, and A. Hogue. Learning to rank for spatiotemporal search. In *Proceedings of the 6th ACM International Conference on Web Search and Data Mining*, WSDM'13, pages 717–726. ACM, 2013.

[32] X. Xiao, Y. Zheng, Q. Luo, and X. Xie. Finding similar users using category-based location history. In *Proceedings of the 18th SIGSPATIAL International Conference on Advances in Geographic Information Systems*, GIS '10, pages 442–445, New York, NY, USA, 2010. ACM.

[33] C. Zhang, K. P. Subbu, J. Luo, and J. Wu. Groping: Geomagnetism and crowdsensing powered indoor navigation. *Mobile Computing, IEEE Transactions on*, 14(2):387–400, 2015.

[34] Y. Zheng, L. Zhang, X. Xie, and W.-Y. Ma. Mining interesting locations and travel sequences from GPS trajectories. In *Proceedings of the 18th International Conference on World Wide Web*, WWW '09, pages 791–800. ACM Press, 2009.